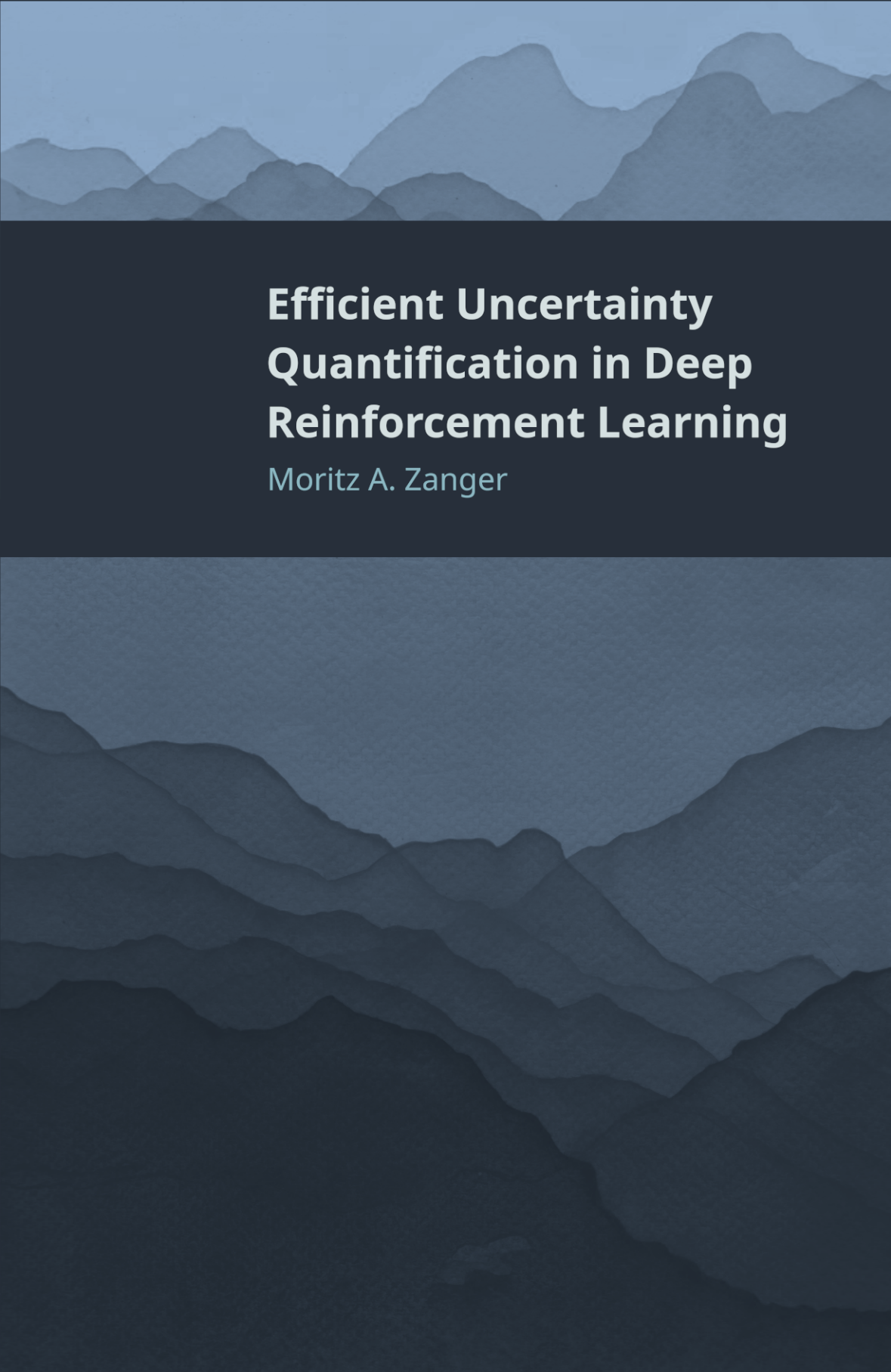


Efficient Uncertainty Quantification in Deep Reinforcement Learning

Moritz A. Zanger

Efficient Uncertainty Quantification in Deep Reinforcement Learning

Efficient Uncertainty Quantification in Deep Reinforcement Learning

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus,
Prof.dr.ir. H. Bijl,
Chair of the Board for Doctorates
to be defended publicly
on Monday 11th of May 2026, 10:00

by

Moritz Akiya ZANGER

This dissertation has been approved by the (co)promotors.

Composition of the doctoral committee:

Rector Magnificus,	Chairperson
Prof.dr. M.T.J. Spaan,	Delft University of Technology, promotor
Dr. J.W. Böhmer,	Delft University of Technology, copromotor
Independent Members:	
Prof.dr.ir. B. De Schutter,	Delft University of Technology
Prof.dr. A. Krause,	Eidgenössische Technische Hochschule Zürich, Switzerland
Prof.dr. A. Nowé,	Vrije Universiteit Brussel, Belgium
Dr. T.M. Moerland,	Universiteit Leiden
Prof.dr. F.A. Oliehoek	Delft University of Technology, *reserve member*

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 101016509 (EPISTEMIC AI).

Keywords:	Reinforcement Learning, Deep Learning, Uncertainty Quantification, Epistemic Uncertainty, Exploration
Printed by:	proefschriftmaken.nl
Cover:	Moritz A. Zanger
Style:	TU Delft House Style, with modifications by Moritz Beller and Moritz A. Zanger
ISBN:	978-94-6384-959-3

An electronic version of this dissertation is available at `http://repository.tudelft.nl/`.

For Leen and Bonnie.

Contents

Nomenclature **xi**
Acronyms and Abbreviations . . . xi
Latin Letters . . . xiii
Greek Letters . . . xv

Summary **xix**

Samenvatting **xxiii**

Zusammenfassung **xxvii**

要約 **xxxi**

1 Introduction **1**
1.1 Uncertainty in Artificial Intelligence . . . 2
1.2 Uncertainty in Reinforcement Learning . . . 5
1.3 Examples . . . 7
1.3.1 Efficient Exploration . . . 8
1.3.2 Reliable and Conservative Decision-Making . . . 10
1.3.3 Types of Uncertainty . . . 11
1.4 Towards Efficient and Principled Uncertainty Estimation in RL . . . 14
1.4.1 The State of Research. . . . 15
1.4.2 Research Mission . . . 16
1.5 Contents of This Thesis . . . 17
1.5.1 Research Questions. . . . 18
1.5.2 Contributions . . . 19
1.5.3 Thesis Outline . . . 20

2 Background **23**
2.1 Markov Decision Processes . . . 24
2.2 Reinforcement Learning. . . . 27
2.3 Deep Reinforcement Learning . . . 30
2.4 Aleatoric Uncertainty in Deep Reinforcement Learning. . . . 34
2.5 Epistemic Uncertainty in Deep Reinforcement Learning . . . 36
2.5.1 Bayesian Inference . . . 36
2.5.2 Ensemble Methods . . . 39
2.5.3 Other Approaches . . . 41

2.6 Deep Learning Theory and Learning Dynamics 42

3 Distributional Projection Ensembles 47
3.1 Introduction 48
3.2 Background. 51
3.2.1 Distributional Reinforcement Learning 51
3.2.2 Categorical and Quantile Distributional RL. 52
3.3 Exploration with Distributional Projection Ensembles 53
3.3.1 Optimistic Bounds from Distributions. 55
3.3.2 Propagation of Distributional Errors 56
3.4 Deep Distributional Projection Ensembles 57
3.5 Empirical Analysis 59
3.5.1 Distributional Projections and Generalization Behavior 59
3.5.2 The Behaviour Suite 60
3.5.3 The Deep Sea and Ablations 61
3.5.4 The VizDoom Environment 61
3.6 Related Work. 62
3.7 Conclusion 63
3.8 Proofs. 64
3.8.1 Contractivity of Projection Mixtures 64
3.8.2 Optimistic Bounds from Distributions. 66
3.8.3 Propagation of Distributional Errors 67
3.8.4 Additional Proofs 69

4 Contextual Similarity Distillation 75
4.1 Introduction 76
4.2 Background. 78
4.2.1 Exploration in Reinforcement Learning. 78
4.2.2 Neural Tangent Kernel Gaussian Processes. 79
4.3 Contextual Similarity Distillation. 81
4.3.1 Ensemble Variance Predictions for A Priori Queries . 81
4.3.2 Ensemble Variance Estimation for Arbitrary Query Points 82
4.3.3 Deep Contextualized Similarity Distillation. 83
4.4 Empirical Evaluation. 85
4.4.1 Distribution Shift Detection 85
4.4.2 Exploration in VizDoom 86
4.5 Related Work. 87
4.6 Limitations and Assumptions 88
4.7 Conclusion 89

4.8 Proofs . . . 90
4.8.1 Linearized Neural Network Learning Dynamics . . . 90
4.8.2 Distribution of Neural Network Functions . . . 91

5 An Analysis of Random Network Distillation 93
5.1 Introduction . . . 94
5.2 Background . . . 96
5.3 Equivalence of Random Network Distillation & Deep Ensembles . . . 97
5.3.1 Multi-Headed Random Network Distillation . . . 101
5.4 Equivalence of Random Network Distillation and Bayesian Posteriors . . . 103
5.5 Related Work . . . 109
5.6 Limitations and Assumptions . . . 110
5.7 Discussion . . . 110
5.8 Proofs . . . 111
5.8.1 Ensemble Equivalence . . . 112
5.8.2 Posterior Equivalence . . . 122

6 Universal Value-Function Uncertainties 127
6.1 Introduction . . . 129
6.2 Background . . . 130
6.2.1 Myopic Uncertainty and Neural Tangent Kernels . . . 131
6.2.2 Value Uncertainty . . . 133
6.3 Universal Value-Function Uncertainties . . . 133
6.3.1 Building Intuition by an Example . . . 134
6.4 What do Universal Value-Function Uncertainties Represent? . . . 136
6.5 Empirical Analysis . . . 139
6.5.1 Experimental Setup . . . 139
6.5.2 Results . . . 140
6.6 Related Work . . . 141
6.7 Limitations and Assumptions . . . 142
6.8 Discussion . . . 143
6.9 Proofs . . . 144
6.9.1 Infinite-Width Learning Dynamics . . . 144
6.9.2 Error Distribution with Multiheaded Architectures . . . 150

7 Discussion and Outlook 157
7.1 Answers to Research Questions . . . 158
7.2 Future Research . . . 160
7.2.1 Towards a Deep Reinforcement Learning Theory . . . 161

7.2.2 Towards Uncertainty-Driven Representation Learning . . . 162
7.2.3 Towards Truly Uncertainty-Aware Agents . . . 164
7.2.4 Towards Generative Discovery with Reinforcement Learning . . . 166
7.3 Conclusion . . . 168

References **171**

Curriculum Vitæ **193**

List of Publications **195**

Acknowledgments **197**

Appendices **199**

A Appendix Distributional Projection Ensembles **201**
A.1 Experimental Details . . . 201
A.1.1 Hyperparameter settings . . . 201
A.1.2 Implementation details . . . 203
A.1.3 Additional experimental results . . . 209
A.1.4 Full results of bsuite experiments . . . 212

B Appendix Contextual Similarity Distillation **215**
B.1 Experimental Details . . . 215
B.1.1 Hyperparameter Settings . . . 215
B.1.2 Implementation Details . . . 216
B.1.3 Additional Experimental Results . . . 221

C Appendix Universal Value-Function Uncertainties **225**
C.1 Experimental Details . . . 225
C.1.1 Implementation Details . . . 225
C.1.2 Hyperparameter Settings . . . 228
C.1.3 Additional Experimental Results . . . 228

Nomenclature

Page numbers refer only to key occurrences; many symbols and acronyms appear throughout.

Acronyms and Abbreviations

AI	artificial intelligence 2, 168
AUPR	area under the precision-recall curve 86, 222
AUROC	area under the receiver operating characteristic curve 86, 216
BAMDP	Bayes-adaptive Markov decision process 164
BDQN	bootstrapped deep Q-network 59
BDQNP	bootstrapped deep Q-network + priors 61, 206
BNN	Bayesian neural network 19, 85, 94, 159
C51	categorical Q-network 57, 59, 201
CDF	cumulative distribution function 52
CLT	central limit theorem 117, 152
CNN	convolutional neural network 88
CSD	contextual similarity distillation 19, 76, 159
DLTV	decaying left-truncated variance 59, 206
DNN	deep neural network 2, 32, 77
DP	dynamic programming 26
DQN	deep Q-network 33, 59, 87, 140, 220
e.g.	for example (exempli gratia) 3
GP	Gaussian process 77, 80, 81, 95, 132
h.c.	Hermitian conjugate 92, 99, 114, 132
i.e.	that is (id est) 9

i.i.d.	independent and identically distributed 79, 95, 138, 149
IDS	information-directed sampling 59, 61, 87, 206
JVP	Jacobian-vector product 108
KL	Kullback–Leibler (divergence) 58
l.h.s.	left-hand side 50
MCMC	Markov chain Monte Carlo 39, 85, 95
MDP	Markov decision process 24, 51, 57, 78, 130
NN	neural network 5, 57, 77, 98, 132, 203
NNGP	neural network Gaussian process 91, 97, 137
NTK	neural tangent kernel 16, 44, 76, 94, 98, 128, 160
OOD	out-of-distribution 41, 76, 221
PE-DQN	projection ensemble deep Q-network 57, 206
QR	quantile regression 53, 206
r.h.s.	right-hand side 50
RAG	retrieval-augmented generation 165
RND	random network distillation 19, 41, 85, 87, 94, 159
RQ	research question 2, 48, 94, 128, 158, 159
s.t.	such that 53
SARSA	state-action-reward-state-action 29
SR	successor representation 167
TD	temporal difference 33, 50, 129, 160
UCB	upper confidence bound 38, 55, 79, 208
UVFA	universal value function approximator 131
UVU	universal value-function uncertainty 20, 128, 160
VI	variational inference 15, 39, 95

w.r.t.	with respect to 26

Latin Letters

A	random action 51, 78, 131
$\mathcal{A}$	action space 24, 51, 78, 130
a	action 24, 78, 130
$\mathcal{B}(\cdot)$	Borel σ-algebra 67
b	bias vector in neural networks 97, 150
$b(\cdot)$	bonus function 55
$\mathcal{C}$	set of contexts 82
c	bounding moduli in projections 56
c	context variable 83, 165
$\mathcal{D}$	set of data points 37, 139
$\mathcal{D}(\cdot)$	distribution operator 51
d	differential operator 91, 98, 136
d	generic counter variable for dimensions 104, 131
$\mathbb{E}[\cdot]$	expected value 25, 51, 78, 97, 131
e	Euler's constant 91, 99, 145
$F(\cdot)$	cumulative distribution function 52
$\mathscr{F}$	set of probability distributions (representation) 52
$f(\cdot)$	generic function 37, 43, 67, 79, 96, 132
$g(\cdot)$	generic function 96, 134
$h(\cdot)$	hat function in categorical projections 73
I	identity matrix 91, 97
i	generic indexing or counter variable 31, 58, 84, 96, 150

$J(\cdot)$ total return function 78
j generic indexing or counter variable 26, 58, 84, 96, 150

k generic indexing or counter variable 26, 52, 119, 154

L last layer in neural networks 33, 84, 97, 152
l layer index in neural networks 33, 97, 150
$\mathcal{L}(\cdot)$ loss function 43, 84, 98, 132
$l()\cdot,\cdot$ Cramér distance 71

M generic counter variable 54, 138, 154

N generic counter variable 9, 32, 61, 79, 96
$\mathcal{N}(\cdot,\cdot)$ Gaussian distribution 79, 97, 132
n generic counter variable 80, 98, 136

$P(\cdot|\cdot,\cdot)$ transition kernel 24, 51, 78, 130
$\mathscr{P}(\cdot)$ set of probability distributions over a space 51, 78, 130

$Q(\cdot,\cdot)$ action value function 25, 51, 78, 130

R random immediate reward 51, 78
$\mathcal{R}(\cdot|\cdot,\cdot)$ immediate reward kernel 51
$\mathbb{R}$ set of real numbers 51, 78, 96, 130
$r(\cdot,\cdot)$ expected reward function 24, 129

S random state 35, 68, 78, 131
$\mathcal{S}$ state space 24, 51, 78, 130
s state 9, 24, 61, 78, 131

T Bellman operator 51
$\mathcal{T}$ distributional Bellman operator 51
t time 25, 51, 78, 130

$\mathcal{U}$ uniform distribution 108
$u(\cdot)$ generic function 96, 134

$V(\cdot)$ state value function 26, 165

$\mathbb{V}[\cdot]$	variance 83, 100, 133
W	weight matrix in neural networks 33
w	weight matrix in neural networks 97, 150
$w_p(\cdot,\cdot)$	p-Wasserstein metric 52
X	generic random variable 64
$\mathcal{X}$	set of inputs 79, 97, 132
x	generic input variable to a function 79, 97, 132
$x(\cdot)$	post-activation in neural networks 37, 97, 150
Y	generic random variable 64
$\mathcal{Y}$	set of labels 79, 97, 132
y	generic label variable for labeled datasets 43, 79, 97, 132
Z	random return 34, 55
$Z(\cdot,\cdot)$	random return function 35, 51
$\mathcal{Z}$	set of tasks or policy encodings 131
z	task or policy encoding 136
$z(\cdot)$	layer output in neural networks 33, 97, 150

Greek Letters

α	continuous-time learning rate 90, 112, 136, 207
β	optimism parameter 59, 204
γ	discount 25, 51, 58, 78, 130, 207
Γ	set of couplings 64
δ	generic variable for differences 113, 145
$\delta(\cdot)$	Dirac delta distribution 52, 203
$\Delta(\cdot,\cdot)$	temporal difference neural tangent kernel 137
ϵ	exploration parameter 29, 205
$\epsilon(\cdot)$	error function 96, 131, 228
η	learning rate 28

η return distribution 51, 203

θ parameter 31, 32, 53, 79, 96, 131, 132, 203, 227
ϑ alternative parameter 96, 131, 228
$\Theta(\cdot, \cdot)$ neural tangent kernel 44, 80, 98, 132, 221

$\kappa(\cdot, \cdot)$ neural network Gaussian process kernel 91, 97, 132

λ eigenvalue 146
$\Lambda(\cdot, \cdot)$ temporal difference neural network Gaussian process kernel 137, 148

μ start state distribution 26, 51, 78, 130
μ mean 92, 97
$\mu(\cdot)$ mean function 92, 97

ν generic distribution variable 52

∇ gradient 43, 80, 98, 132, 207

π policy 9, 25, 51, 78, 130, 165
Π projection operator 52

ρ error function 53, 58, 208

$\Sigma(\cdot, \cdot)$ covariance function 92, 96
σ standard deviation 97, 133, 208
$\sigma(\cdot)$ nonlinearity 32, 203
σ as in σ-algebra, closed under countable unions (and complements) 67

τ quantile 50
τ integration parameter 71

ϕ feature 31, 83, 104, 163, 207
$\phi(\cdot)$ nonlinearity 150
φ torque 31
φ alternative feature 84, 219
$\varphi(\cdot)$ characteristic function 122

χ^2	Chi-squared distribution 102
ψ	context feature 83, 219
ψ	alternative parameter 96, 131, 228
Ω	Projection mixture operator 54, 211

Summary

This dissertation concerns the efficient quantification of uncertainty in the field of deep reinforcement learning. At the time of this writing, artificial intelligence is being adopted rapidly into the critical pipelines of numerous scientific and societal domains — from autonomous driving and medical diagnostics to scientific discovery. A particular class of machine learning models, deep neural networks, has been pivotal in this recent development due to their extraordinary scalability and expressive power. Such models *learn* by optimizing vast sets of parameters to shape predictions according to previous measurements, captured in large datasets. When we deploy such learned models for practical applications, however, they are asked to make predictions for novel inputs not represented in their training data. Such predictions are the result of inductive generalization — deriving insights about future situations from past experience — and are inherently subject to uncertainty. For these predictions to be actionable, they must often be accompanied by a reliable measure of confidence. An autonomous vehicle must not only recognize a pedestrian but also know when its perception is too uncertain to proceed safely; a diagnostic model must not only classify a tumor but also know when to defer to a human expert. This need to *know what one does not know* is addressed by the quantification of *epistemic uncertainty*, which arises from the imperfection of a learned model, often due to a lack of sufficient relevant data. This stands in contrast to *aleatoric uncertainty* — the irreducible, inherent randomness in a process — and it is this reducible, model-centric epistemic uncertainty that forms the central object of inquiry for this dissertation.

The challenge of epistemic uncertainty estimation becomes especially tangible in the context of sequential decision-making problems. In such settings, an agent's actions can have long-term consequences that compound over time, shaping downstream outcomes and choices. Reinforcement learning, a paradigm in which agents learn such decision-making strategies through direct interaction with an environment, faces several fundamental challenges that hinge on reliable uncertainty estimation. An agent with a well-calibrated sense of its own ignorance can actively seek out novel situations to gain information and discover superior strategies. Conversely, some applications demand agents that are naturally averse to such situations: we do not seek robotic assistants in elderly care to explore unfamiliar behaviors for the sake of information gain, but rather ones that operate conservatively within the

bounds of their knowledge. Underpinning both *efficient exploration* and *safe decision-making* is a principled understanding of an agent's own epistemic uncertainty — the central topic of this thesis.

Examining the current research landscape of uncertainty quantification in deep learning, we observe a persistent tension between theoretically well-motivated yet computationally expensive techniques on one hand, and computationally efficient yet less understood methods on the other. Bayesian inference, widely regarded as the gold standard for reasoning about epistemic uncertainty, is generally intractable for modern, large-scale neural networks. This has led to a spectrum of approximate methods — including deep ensembles, advanced sampling techniques, and variational inference — that navigate this trade-off to varying degrees. More pragmatic solutions, meanwhile, often offer substantial computational savings but lack a deeper theoretical understanding of what their uncertainty estimates represent, or how they behave in practice. From this landscape, we derive the research mission for this dissertation: to engage directly with this trade-off by *developing and analyzing uncertainty quantification methods that are both computationally tractable and theoretically well-motivated.* To this end, this thesis aims to depart from a "black-box" treatment of neural networks, instead pursuing methods that are grounded in and seek to leverage their inherent generalization properties.

Our first line of inquiry, presented in Chapter 3, begins by investigating a de facto standard for epistemic uncertainty estimation in deep learning: deep neural network ensembles. We hypothesize that the efficacy of ensembles is constrained not merely by the number of constituent models but by the *quality* of their diversity. Focusing on distributional reinforcement learning, we observe that specific architectural components — namely, the projection operators used to approximate return distributions — can induce strong inductive biases that significantly shape generalization behavior. Building on this insight, we develop *diverse projection ensembles*, which induce diversity by construction through the use of members with architecturally distinct projection operators. We show empirically that this approach yields more robust uncertainty signals, enabling smaller ensembles to achieve superior exploration performance in challenging environments compared to larger, homogeneous ensembles.

Our second line of inquiry, spanning Chapters 4 and 5, pursues the more ambitious goal of emulating the uncertainty properties of an entire deep neural network ensemble within a single, efficient model. In Chapter 4, we develop a novel technique — *contextual similarity distillation* — that is amenable to epistemic uncertainty estimation with a single model trained with gradient descent. By analyzing the learning dynamics and generalization properties of wide neural networks through the lens of the neural tangent kernel, we reframe the intractable problem of computing analytical ensemble variances

as a tractable, contextualized kernel regression task — solvable with a single function approximator, such as a neural network. In Chapter 5, we take a complementary approach by establishing a missing theoretical foundation for an existing, widely used single-model uncertainty quantification method: *random network distillation*. Our analysis reveals that the uncertainty produced by random network distillation is not merely a heuristic signal but is, in the idealized infinite-width limit, formally equivalent to the predictive variance of a deep ensemble. Building on this insight, we devise a novel *Bayesian random network distillation* algorithm whose error signal can be shaped to exactly match the posterior predictive variance of an infinitely wide Bayesian neural network. This places the method on principled theoretical footing within the framework of Bayesian inference.

Our research concludes in Chapter 6, which synthesizes insights from the preceding work to address a central challenge of uncertainty quantification in reinforcement learning: the direct estimation of long-term, cumulative uncertainty. The methods developed previously, while efficient, primarily quantify immediate, one-step uncertainties. In contrast, this chapter develops a novel single-model method — *universal value-function uncertainties* — that directly quantifies the cumulative uncertainty of value functions, accounting for all future uncertainties encountered under a given policy. The method measures uncertainty as the error between an online value function, trained via temporal difference learning, and a fixed target function, from which a synthetic reward signal is derived. Our theoretical analysis, grounded in neural tangent kernel theory, proves that this procedure yields uncertainty estimates equivalent to the variance of a full ensemble of universal value functions. We furthermore demonstrate empirically that our approach serves as a reliable uncertainty estimator in challenging multi-task offline reinforcement learning settings, providing long-term value uncertainty with the efficiency of a single model.

In conclusion, this dissertation follows a coherent path of scientific inquiry, progressing from the enhancement of multi-model ensembles to the development of a suite of theoretically grounded and computationally efficient single-model alternatives. The contributions presented herein provide both a practical toolkit for practitioners and novel theoretical insights toward a more thorough understanding of uncertainty estimation in deep learning. The overarching goal of this work is to take a definitive step towards creating more reliable, uncertainty-aware autonomous agents. By equipping agents with a principled understanding of their own knowledge and its limitations, we lay the foundation not only for their safe and responsible deployment in real-world applications but also for more efficient exploration and autonomous discovery.

Samenvatting

Deze dissertatie betreft de efficiënte kwantificatie van onzekerheid binnen het domein van deep reinforcement learning. Op het moment van schrijven wordt kunstmatige intelligentie snel geïntegreerd in de kritieke processen van talloze wetenschappelijke en maatschappelijke domeinen — van autonoom rijden en medische diagnostiek tot wetenschappelijke ontdekkingen. Een specifieke klasse van machine learning-modellen, diepe neurale netwerken, is bepalend geweest voor deze recente ontwikkeling, dankzij hun buitengewone schaalbaarheid en expressieve kracht. Dergelijke modellen *leren* door het optimaliseren van een groot aantal parameters om voorspellingen te vormen op basis van eerdere metingen uit omvangrijke datasets. Wanneer we zulke geleerde modellen inzetten in praktische toepassingen, worden ze echter geconfronteerd met nieuwe invoer die niet voorkwam in hun trainingsgegevens. Dergelijke voorspellingen zijn het resultaat van inductieve generalisatie — het afleiden van inzichten over toekomstige situaties op basis van eerdere ervaring — en zijn van nature onderhevig aan onzekerheid. Opdat deze voorspellingen bruikbaar zijn, moeten ze vaak vergezeld gaan van een betrouwbare maat voor vertrouwen. Een autonoom voertuig moet niet alleen een voetganger herkennen, maar ook weten wanneer zijn waarneming te onzeker is om veilig door te gaan; een diagnostisch model moet niet alleen een tumor classificeren, maar ook weten wanneer het beter is de beslissing over te laten aan een menselijke expert. Deze noodzaak om *te weten wat men niet weet* wordt aangepakt via de kwantificatie van *epistemische onzekerheid*: epistemische onzekerheid ontstaat uit de imperfectie van een geleerd model, doorgaans als gevolg van een tekort aan relevante data. Het is belangrijk om dit type onzekerheid te onderscheiden van *aleatorische onzekerheid*, die voortkomt uit inherente willekeur in een proces, en niet kan worden verminderd door het verzamelen van meer data.

De uitdaging van het inschatten van epistemische onzekerheid wordt bijzonder tastbaar in de context van sequentiële besluitvormingsproblemen. In zulke situaties kunnen de acties van een agent langetermijngevolgen hebben die zich opstapelen in de tijd, en zo toekomstige uitkomsten en keuzes beïnvloeden. Reinforcement learning, een paradigma waarbij agenten dergelijke beslissingsstrategieën leren door directe interactie met een omgeving, kent diverse fundamentele uitdagingen die afhangen van betrouwbare onzekerheidsinschatting. Een agent met een goed gekalibreerd besef van zijn eigen onwetendheid kan actief op zoek gaan naar nieuwe situaties om informatie te ver-

garen en betere strategieën te ontdekken. Daarentegen zijn er toepassingen waarin men juist vraagt om agenten die zulke situaties vermijden: we willen geen robotassistenten in de ouderenzorg die onbekend gedrag gaan verkennen omwille van informatievergaring, maar eerder agenten die zich voorzichtig gedragen binnen de grenzen van hun kennis. Ten grondslag aan zowel *efficiënte exploratie* als *veilige besluitvorming* ligt een principieel begrip van epistemische onzekerheid — het centrale onderwerp van dit proefschrift.

Bij het verkennen van het huidige onderzoekslandschap rond onzekerheidskwantificatie in deep learning, zien we een aanhoudende spanning tussen theoretisch goed onderbouwde maar computationeel dure technieken enerzijds, en computationeel efficiënte maar minder begrepen methoden anderzijds. Bayesiaanse inferentie, algemeen beschouwd als de gouden standaard voor het redeneren over epistemische onzekerheid, is doorgaans onpraktisch voor moderne, grootschalige neurale netwerken. Dit heeft geleid tot een spectrum van benaderingen — waaronder diepe ensembles, geavanceerde sampling-technieken, en variationale inferentie — die elk in meer of mindere mate deze afruil proberen te navigeren. Meer pragmatische oplossingen bieden vaak aanzienlijke computationele voordelen, maar missen een diepere theoretische onderbouwing van wat hun onzekerheidsschattingen precies representeren of hoe ze zich in de praktijk gedragen. Uit dit landschap volgt de onderzoeksmissie van deze dissertatie: om deze afweging direct aan te pakken door *onzekerheidskwantificatiemethoden te ontwikkelen en analyseren die zowel computationeel tractabel als theoretisch goed gemotiveerd zijn.* Daartoe wil dit proefschrift afstappen van een "black-box"-benadering van neurale netwerken, en in plaats daarvan methoden ontwikkelen die gebaseerd zijn op — en gebruikmaken van — hun intrinsieke generalisatie-eigenschappen.

Onze eerste onderzoekslijn, gepresenteerd in Hoofdstuk 3, begint met het analyseren van een de facto standaard voor epistemische onzekerheidsinschatting in deep learning: diepe neurale netwerkensembles. We stellen de hypothese dat de effectiviteit van ensembles niet slechts wordt bepaald door het aantal modellen, maar door de *kwaliteit* van hun diversiteit. In het bijzonder richten we ons op distributionele reinforcement learning, waarin bepaalde architecturale componenten — namelijk de projectie-operatoren die gebruikt worden om retourverdelingen te benaderen — sterke inductieve vooroordelen kunnen opleveren die het generalisatiegedrag aanzienlijk beïnvloeden. Op basis van dit inzicht ontwikkelen we *diverse projection ensembles*, die diversiteit van nature afdwingen door leden met architectonisch verschillende projectie-operatoren te combineren. We tonen empirisch aan dat deze aanpak robuustere onzekerheidssignalen oplevert, waardoor kleinere ensembles betere exploratieprestaties behalen in uitdagende omgevingen dan grotere, homogene ensembles.

Onze tweede onderzoekslijn, in Hoofdstukken 4 en 5, streeft het ambitieuzere doel na om de onzekerheidseigenschappen van een volledig ensemble na te bootsen met één enkel, efficiënt model. In Hoofdstuk 4 introduceren we een nieuwe techniek — *contextual similarity distillation* — die geschikt is voor epistemische onzekerheidsinschatting met een enkel model dat getraind wordt via gradient descent. Door de leerdynamiek en generalisatie-eigenschappen van brede neurale netwerken te analyseren via de neural tangent kernel, herformuleren we het onoplosbare probleem van analytische ensemblevarianties als een oplosbare, contextuele kernelregressietaak — uitvoerbaar met één functiebenaderaar, zoals een neuraal netwerk. In Hoofdstuk 5 hanteren we een complementaire aanpak door een ontbrekende theoretische basis te leveren voor een bestaande, wijdverbreide single-model-methode voor onzekerheidskwantificatie: *random network distillation*. Onze analyse toont aan dat de onzekerheidsschatting van random network distillation niet slechts een heuristiek is, maar in de geïdealiseerde oneindig-brede limiet formeel gelijkwaardig is aan de voorspellende variantie van een diep ensemble. Op basis van dit inzicht ontwikkelen we een nieuwe *Bayesian random network distillation*-algoritme waarvan het foutsignaal zodanig kan worden gevormd dat het exact overeenkomt met de posterior voorspellende variantie van een oneindig breed Bayesiaans neuraal netwerk. Hiermee plaatsen we de methode op solide theoretische grond binnen het kader van Bayesiaanse inferentie.

Ons onderzoek wordt afgesloten in Hoofdstuk 6, waarin we inzichten uit voorgaand werk synthetiseren om een centrale uitdaging in onzekerheidskwantificatie binnen reinforcement learning aan te pakken: de directe inschatting van langetermijn- of cumulatieve onzekerheid. De eerder ontwikkelde methoden zijn weliswaar efficiënt, maar richten zich voornamelijk op onmiddellijke, één-stap-onzekerheden. In contrast daarmee ontwikkelen we in dit hoofdstuk een nieuwe single-model methode — *universal value-function uncertainties* — die direct de cumulatieve onzekerheid in waarde-functies inschat, inclusief alle toekomstige onzekerheden onder een gegeven beleid. De methode meet onzekerheid als het verschil tussen een online waarde-functie, getraind via temporal difference learning, en een vaste doelfunctie, waaruit een synthetisch beloningssignaal wordt afgeleid. Onze theoretische analyse, gebaseerd op neural tangent kernel-theorie, bewijst dat deze procedure onzekerheidsschattingen oplevert die equivalent zijn aan de variantie van een volledig ensemble van universele waarde-functies. Daarnaast tonen we empirisch aan dat onze benadering zich gedraagt als een betrouwbare onzekerheidsschatting in veeleisende multi-taak offline reinforcement learning-settings, waarbij langetermijnonzekerheid wordt geleverd met de efficiëntie van een enkel model.

Concluderend volgt deze dissertatie een samenhangend pad van wetenschappelijke verkenning, gaande van het verbeteren van multi-model ensem-

bles tot het ontwikkelen van een reeks theoretisch gefundeerde en computationeel efficiënte single-model-alternatieven. De bijdragen die hier gepresenteerd worden, bieden zowel een praktische gereedschapskist voor gebruikers als nieuwe theoretische inzichten in onzekerheidsinschatting binnen deep learning. Het overkoepelende doel van dit werk is om een beslissende stap te zetten richting betrouwbare, onzekerheidsbewuste autonome agenten. Door deze agenten uit te rusten met een principieel begrip van hun eigen kennis en de grenzen daarvan, leggen we het fundament niet alleen voor veilige en verantwoorde inzet in de echte wereld, maar ook voor efficiëntere exploratie en autonome ontdekking.

Zusammenfassung

Diese Dissertation befasst sich mit der effizienten Quantifizierung von Unsicherheit im Bereich des Deep Reinforcement Learning. Zum Zeitpunkt des Verfassens wird künstliche Intelligenz zunehmend in die kritischen Prozesse zahlreicher wissenschaftlicher und gesellschaftlicher Bereiche integriert — von autonomen Fahrsystemen und medizinischer Diagnostik bis hin zu wissenschaftlichen Entdeckungen. Eine bestimmte Klasse von maschinellen Lernmodellen, tiefe neuronale Netzwerke, war in dieser Entwicklung besonders prägend, da sie eine außergewöhnliche Skalierbarkeit und Ausdrucksstärke besitzen. Solche Modelle *lernen*, indem sie große Mengen an Parametern optimieren, um Vorhersagen an frühere Messungen aus umfangreichen Datensätzen anzupassen. Wenn wir diese gelernten Modelle in praktischen Anwendungen einsetzen, werden sie jedoch mit neuen Eingaben konfrontiert, die nicht in den Trainingsdaten enthalten waren. Solche Vorhersagen beruhen auf induktiver Generalisierung — also dem Ableiten von Erkenntnissen über zukünftige Situationen auf Grundlage vergangener Erfahrungen — und sind von Natur aus mit Unsicherheit behaftet. Damit diese Vorhersagen handlungsrelevant sind, müssen sie häufig mit einer verlässlichen Einschätzung ihrer Vertrauenswürdigkeit versehen sein. Ein autonomes Fahrzeug muss nicht nur einen Fußgänger erkennen, sondern auch wissen, wann seine Wahrnehmung zu unsicher ist, um sicher weiterzufahren; ein diagnostisches Modell muss nicht nur einen Tumor klassifizieren, sondern auch erkennen, wann es besser ist, eine Entscheidung an einen menschlichen Experten zu übergeben. Dieses Bedürfnis, *zu wissen, was man nicht weiß*, wird durch die Quantifizierung von *epistemischer Unsicherheit* adressiert: Epistemische Unsicherheit entsteht durch die Unvollkommenheit eines gelernten Modells, typischerweise aufgrund eines Mangels an relevanten Daten. Es ist wichtig, diese Art der Unsicherheit von *aleatorischer Unsicherheit* zu unterscheiden, die durch inhärente Zufälligkeit in einem Prozess verursacht wird und nicht durch das Sammeln zusätzlicher Daten reduziert werden kann.

Die Herausforderung der Einschätzung epistemischer Unsicherheit wird besonders deutlich im Kontext sequentieller Entscheidungsfindung. In solchen Szenarien können die Handlungen eines Agenten langfristige Folgen haben, die sich im Zeitverlauf aufbauen und künftige Ergebnisse sowie Entscheidungen maßgeblich beeinflussen. Reinforcement Learning (RL), ein Paradigma, bei dem Agenten Entscheidungsstrategien durch direkte Interaktion mit einer

Umgebung erlernen, steht vor mehreren grundlegenden Herausforderungen, die von verlässlicher Unsicherheitsabschätzung abhängen. Ein Agent mit einem gut kalibrierten Bewusstsein für sein eigenes Nichtwissen kann gezielt neue Situationen aufsuchen, um Informationen zu sammeln und bessere Strategien zu entdecken. Umgekehrt verlangen manche Anwendungen nach Agenten, die solchen Situationen aus dem Weg gehen: Wir wünschen uns keine robotischen Assistenten in der Altenpflege, die unbekanntes Verhalten erkunden, sondern solche, die sich vorsichtig innerhalb der Grenzen ihres Wissens bewegen. Sowohl *effiziente Exploration* als auch *sichere Entscheidungsfindung* beruhen auf einem fundierten Verständnis epistemischer Unsicherheit — dem zentralen Thema dieser Dissertation.

Bei der Betrachtung der aktuellen Forschung zur Unsicherheitsquantifizierung im Deep Learning zeigt sich ein anhaltendes Spannungsfeld zwischen theoretisch gut begründeten, aber rechnerisch aufwendigen Verfahren auf der einen Seite und recheneffizienten, aber weniger verstandenen Methoden auf der anderen. Die Bayessche Inferenz, weithin als Goldstandard für das Schließen über epistemische Unsicherheit anerkannt, ist für moderne, großskalige neuronale Netzwerke im Allgemeinen nicht praktikabel. Dies hat zur Entwicklung eines Spektrums an Näherungsverfahren geführt — darunter tiefe Ensembles, fortgeschrittene Sampling-Methoden und Variationsinferenz — die in unterschiedlichem Maße diesen Zielkonflikt adressieren. Pragmatischere Lösungen bieten oft erhebliche rechnerische Vorteile, entbehren jedoch einer tiefergehenden theoretischen Fundierung dessen, was derartige Unsicherheitsabschätzungen tatsächlich ausdrücken und wie sie sich in der Praxis verhalten. Aus diesem Spannungsfeld ergibt sich die Forschungsmission dieser Dissertation: *die Entwicklung und Analyse von Unsicherheitsquantifizierungsverfahren, die sowohl rechnerisch effizient als auch theoretisch fundiert sind.* Zu diesem Zweck verfolgt diese Arbeit einen Ansatz, der sich von einer "Black-Box"-Betrachtung neuronaler Netzwerke entfernt und stattdessen Verfahren entwickelt, die auf den inhärenten Generalisierungseigenschaften dieser Modelle basieren und diese gezielt ausnutzen.

Unser erster Forschungsstrang, vorgestellt in Kapitel 3, beginnt mit der Untersuchung eines de-facto-Standards für die Schätzung epistemischer Unsicherheit im Deep Learning: Ensembles tiefer neuronaler Netzwerke. Wir stellen die Hypothese auf, dass die Effektivität von Ensembles nicht allein durch die Anzahl der enthaltenen Modelle bestimmt wird, sondern durch die *Qualität* ihrer Diversität. Im Fokus steht dabei das distributionale Reinforcement Learning, bei dem bestimmte architektonische Komponenten — insbesondere die Projektionsoperatoren zur Approximation von Rückgabeverteilungen — starke induktive Verzerrungen erzeugen können, die das Generalisierungsverhalten maßgeblich prägen. Aufbauend auf dieser Erkenntnis entwickeln wir

diverse projection ensembles, die durch architektonisch unterschiedliche Projektionsoperatoren gezielt Diversität in der Modellfamilie erzeugen. Unsere empirischen Ergebnisse zeigen, dass dieser Ansatz robustere Unsicherheitsabschätzungen liefert, sodass kleinere Ensembles in herausfordernden Umgebungen eine bessere Explorationsleistung erzielen als größere, homogene Ensembles.

Unser zweiter Forschungsstrang, dargestellt in den Kapiteln 4 und 5, verfolgt das ambitionierte Ziel, die Unsicherheitseigenschaften eines gesamten Ensembles in einem einzigen, effizienten Modell nachzubilden. In Kapitel 4 entwickeln wir eine neue Methode — *contextual similarity distillation* — die für die Schätzung epistemischer Unsicherheit mit einem einzelnen Modell geeignet ist, das mittels Gradientenabstieg trainiert wird. Durch die Analyse der Lernmechanismen und Generalisierungseigenschaften breiter neuronaler Netzwerke mithilfe des Neural Tangent Kernel reformulieren wir das ursprünglich unlösbare Problem der analytischen Berechnung von Ensemble-Varianzen als ein lösbares, kontextabhängiges Kernelregressionsproblem — lösbar durch einen einzigen Funktionsapproximator, wie etwa ein neuronales Netzwerk. In Kapitel 5 verfolgen wir einen ergänzenden Ansatz, indem wir eine bislang fehlende theoretische Grundlage für eine weit verbreitete Methode zur Unsicherheitsquantifizierung mit Einzelmodellen schaffen: *random network distillation*. Unsere Analyse zeigt, dass die durch random network distillation erzeugte Unsicherheit nicht nur ein heuristisches Signal ist, sondern im idealisierten Grenzfall unendlich breiter Netzwerke formal äquivalent zur Vorhersagevarianz eines tiefen Ensembles ist. Aufbauend auf dieser Erkenntnis entwickeln wir einen neuen Algorithmus — *Bayesian random network distillation* — dessen Fehlermaß so gestaltet werden kann, dass es exakt der posterioren Vorhersagevarianz eines unendlich breiten Bayesschen neuronalen Netzwerks entspricht. Damit wird die Methode auf eine solide theoretische Grundlage innerhalb des Rahmens Bayesscher Inferenz gestellt.

Unsere Forschung kulminiert in Kapitel 6, das die Erkenntnisse der vorangegangenen Arbeiten zusammenführt, um eine zentrale Herausforderung der Unsicherheitsquantifizierung im Reinforcement Learning anzugehen: die direkte Schätzung langfristiger, kumulativer Unsicherheit. Die bisher entwickelten Methoden sind zwar effizient, quantifizieren jedoch in erster Linie unmittelbare Ein-Schritt-Unsicherheiten. Im Gegensatz dazu entwickeln wir in diesem Kapitel eine neue Einzelmodell-Methode — *universal value-function uncertainties* — die die kumulative Unsicherheit von Wertfunktionen direkt quantifiziert und alle zukünftigen Unsicherheiten unter einer gegebenen Strategie berücksichtigt. Die Methode misst Unsicherheit als den Fehler zwischen einer online trainierten Wertfunktion, die mittels Temporal-Difference-Lernen aktualisiert wird, und einer festen Zielfunktion, aus der ein synthetisches Belohnungssignal abgeleitet wird. Unsere theoretische Analyse, gestützt auf die Theorie

des Neural Tangent Kernel, beweist, dass dieses Verfahren Unsicherheitsabschätzungen erzeugt, die äquivalent zur Varianz eines vollständigen Ensembles von universellen Wertfunktionen sind. Darüber hinaus zeigen wir empirisch, dass unser Ansatz zuverlässige Unsicherheitsabschätzungen in anspruchsvollen Multi-Task Offline Reinforcement-Learning Szenarien liefert — und das mit der Effizienz eines einzigen Modells.

Zusammenfassend folgt diese Dissertation einem kohärenten Weg wissenschaftlicher Untersuchung: von der Verbesserung von Multi-Modell-Ensembles bis hin zur Entwicklung einer Reihe theoretisch fundierter und rechnerisch effizienter einzelmodell Alternativen. Die hier vorgestellten Beiträge bieten sowohl ein praktisches Werkzeug für Anwender als auch neue theoretische Erkenntnisse für ein tieferes Verständnis der Unsicherheitsabschätzung im Deep Learning. Das übergeordnete Ziel dieser Arbeit ist es, einen entscheidenden Schritt in Richtung zuverlässiger, unsicherheitsbewusster autonomer Agenten zu gehen. Indem wir Agenten mit einem prinzipiellen Verständnis ihres eigenen Wissens und dessen Grenzen ausstatten, schaffen wir die Grundlage nicht nur für ihren sicheren und verantwortungsvollen Einsatz in realen Anwendungen, sondern auch für effizientere Exploration und autonome Entdeckung.

要約

本論文は、深層強化学習[1]における不確実性の効率的な定量化に関する研究である。執筆時点において、人工知能は自動運転、医療診断、科学的発見に至るまで、多くの科学的および社会的分野の重要なプロセスに急速に導入されつつある。機械学習モデルの一種である深層ニューラルネットワーク[2]は、その卓越したスケーラビリティと表現力により、この発展において重要な役割を果たしてきた。これらのモデルは、大規模データセットに記録された過去の測定値に基づいて予測を形成するために、多数のパラメータを最適化することで学習を行う。しかし、現実世界での応用においては、訓練データに含まれていない未知の入力に対して予測を行う必要が生じる。このような予測は、過去の経験から将来の状況に関する知見を導く帰納的一般化に基づくものであり、本質的に不確実性を伴う。高リスク環境において予測を活用可能にするためには、その信頼度を定量的に示す必要がある。自動運転車は歩行者を認識するだけでなく、認識が不確かで安全に進行できない場合を判断する必要がある。診断モデルは腫瘍を分類するだけでなく、判断を人間の専門家に委ねるべき場合を識別する必要がある。このような「知らないことを知る」ための必要性は、認識的不確実性[3]の定量化によって対処される。認識的不確実性は、通常は十分な関連データの欠如により生じる学習モデルの不完全性に起因する。一方で、プロセスに内在するランダム性によって発生し、データの追加取得によっては減少しない偶然的不確実性[4]とは区別されるべきである。

認識的不確実性推定の課題は、逐次的な意思決定問題において特に顕著となる。このような状況では、エージェントの行動が長期的な結果に影響を与え、それが将来の結果や選択に累積的に作用する。環境との直接的な相互作用を通じて意思決定戦略を学習するパラダイムである強化学習（RL）においては、不確実性を信頼性高く推定することが複数の根本的課題に直結する。認識的不確実性を正確に把握できるエージェントは、新しい状況を能動的に探索し、情報を取得してより優れた戦略を発見できる。一方で、状況によっては未知の行動を避けることが求められる応用も存在する。例えば、高齢者介護用ロボットは情報獲得のために未知の行動を試みるべきではなく、既知の知識範囲内で保守的

[1]Deep Reinforcement Learning
[2]Deep Neural Network
[3]Epistemic Uncertainty
[4]Aleatoric Uncertainty

に動作することが望ましい。効率的な探索と安全な意思決定の双方の基盤となるのは、エージェント自身の認識的不確実性に関する原理的理解であり、本論文の中心的課題である。

深層学習[5]における不確実性定量化の研究動向を概観すると、理論的に十分根拠があるが計算コストの高い手法と、計算効率は高いが理論的理解が不十分な手法との間に持続的な緊張関係が存在する。認識的不確実性推論のゴールドスタンダードとされるベイズ推論は、現代の大規模ニューラルネットワークに対しては一般に計算不能である。このため、深層アンサンブル[6]、先進的サンプリング手法、変分推論など、多様な近似手法がこのトレードオフに対処する形で提案されてきた。さらに実用的な手法は、計算効率の面で優れる一方、その不確実性推定が何を意味し、実際にどのように振る舞うかに関する深い理論的理解を欠く場合が多い。この状況から、本論文の研究目標は計算効率が高く、かつ理論的にも十分に根拠づけられた不確実性定量化手法の開発と解析と定まる。そのために、ニューラルネットワークを「ブラックボックス」として扱うことを避け、その本質的な一般化特性に基づき、かつそれを活用する手法を探求する。

第 3 章では、深層学習における認識的不確実性推定の事実上の標準手法である深層アンサンブルを対象とした研究を行う。アンサンブルの有効性は、単に構成モデルの数だけでなく、その多様性の質にも依存すると仮定する。特に分布型強化学習[7]において、リターン分布を近似するために用いられる射影演算子などの特定のアーキテクチャ要素が、強い帰納バイアスを生じさせ、一般化挙動に大きな影響を与えることを明らかにする。この知見に基づき、アーキテクチャ的に異なる射影演算子を組み込んだメンバーによって多様性を構造的に確保する「多様な射影アンサンブル[8]」を提案する。実験的評価により、この手法はより堅牢な不確実性信号を生成し、小規模なアンサンブルでも大規模で均質なアンサンブルを上回る探索性能を発揮することを示す。

第 4 章と第 5 章にわたる第 2 の研究課題では、深層ニューラルネットワーク・アンサンブル全体の不確実性特性を単一かつ効率的なモデルで再現するという、より野心的な目標に取り組む。第 4 章では、勾配降下法で学習可能な単一モデルによる認識的不確実性推定を可能にする新手法「コンテキスト類似度蒸留[9]」を開発する。幅広いニューラルネットワークの学習ダイナミクスと一般化特性をニューラルタンジェントカーネル[10]の観点から解析し、解析的アンサンブル分散の計算という非現実的な問題を、コンテキスト化されたカーネル回帰問題として

[5]Deep Learning
[6]Deep Ensemble
[7]Distributional RL
[8]Diverse Projection Ensembles
[9]Contextual Similarity Distillation
[10]Neural Tangent Kernel (NTK)

定式化することで、単一の関数近似器（例: ニューラルネットワーク）によって解くことを可能にする。第 5 章では、既存かつ広く用いられている単一モデルによる不確実性推定手法であるランダムネットワーク蒸留[11]に対し、欠落していた理論的基盤を確立する。理想化された無限幅の極限において、この手法の出力する不確実性は深層アンサンブルの予測分散と形式的に等価であることを示す。この結果に基づき、無限幅ベイズニューラルネットワークの事後予測分散と完全に一致する誤差信号を生成できる「ベイズ・ランダムネットワーク蒸留[12]」アルゴリズムを提案し、この広く用いられる手法をベイズ推論の原理的枠組みに位置づける。

第 6 章では、これまでの知見を総合し、強化学習における長期的かつ累積的な不確実性の直接推定という中心課題に取り組む。これまでに開発された手法は効率的ではあるが、主に即時的な 1 ステップの不確実性を定量化するものであった。これに対し、本章では、与えられた方策下で将来遭遇する全ての不確実性を含む、価値関数の累積的不確実性を直接推定する単一モデル手法「ユニバーサル価値関数不確実性[13]」を新たに開発する。この手法は、TD 学習[14]によりオンライン学習された価値関数と固定された目標関数との誤差を不確実性として測定し、そこから合成報酬信号を導出する。ニューラルタンジェントカーネル理論に基づく解析により、本手法が、全てのユニバーサル価値関数アンサンブルの分散と等価な不確実性推定を提供することを証明する。さらに、マルチタスク・オフライン強化学習という困難な設定においても、本手法が長期的価値不確実性の信頼できる推定値を単一モデルの効率で提供することを実証する。

結論として、本論文は、マルチモデル・アンサンブルの改善から始まり、理論的に裏付けられ、かつ計算効率の高い単一モデル代替手法の開発へと進む、一貫した科学的探究の道筋をたどる。本研究で示された貢献は、実務者にとっての実用的ツールキットであると同時に、深層学習における不確実性推定の理解を深める新たな理論的知見でもある。本研究の最終的な目標は、より信頼性の高い、不確実性を考慮した自律エージェントの実現に向けた決定的な一歩を踏み出すことである。エージェントに自らの知識とその限界を原理的に理解させることにより、現実世界での安全かつ責任ある運用だけでなく、効率的な探索や自律的発見の基盤を築く。

[11] Random Network Distillation (RND)
[12] Bayesian RND
[13] Universal Value-Function Uncertainties
[14] Temporal Difference Learning

1

Introduction

1

This introductory chapter outlines the scope and context for the research presented in this dissertation: *efficient uncertainty quantification in deep reinforcement learning*. To this end, we begin by outlining a broad perspective on the role of uncertainty quantification across the field of artificial intelligence and its various applications. We then focus on a particularly challenging class of problems, termed sequential decision making problems, which we intend to solve using data-driven reinforcement learning algorithms and principled uncertainty quantification techniques. The subsequent section aims to build a more nuanced understanding through illustrative examples, unified by a recurrent problem scenario and served to clarify the practical importance of uncertainty estimation and its key conceptual distinctions. Next, we provide an account of the current research landscape to evaluate existing methods and their respective limitations. From this assessment, a research direction guiding this thesis is formulated, that is, towards efficient and principled uncertainty quantification in deep reinforcement learning. The chapter concludes by presenting our central research questions (RQs) in this work, a summary of its main contributions, and a structural outline of the ensuing chapters.

1.1 Uncertainty in Artificial Intelligence

A dramatic increase in the adoption of artificial intelligence (AI) in day-to-day life, numerous industrial applications, and diverse scientific disciplines has established data-centric machine learning algorithms as foundational tools across a variety of fields (Kaddour et al., 2023; Nti et al., 2022; Thiyagalingam et al., 2022). Two significant propellants in this development are the continuous improvement of computational hardware and the expanding scale of available datasets (Kaplan et al., 2020). Within this landscape, deep neural networks (DNNs) have assumed a pivotal role, primarily due to their exceptional scalability, which permit the construction of immensely large and capable models (Bengio et al., 2013; LeCun et al., 2015). And increasingly, DNNs have become highly effective solutions to open challenges in diverse domains, including medical diagnostics, scientific computing, autonomous navigation and robotics, complex game environments, and natural language processing (Achiam et al., 2023; Grigorescu et al., 2020; Kalashnikov et al., 2018; Nti et al., 2022; Raghu and Schmidt, 2020; Suganyadevi et al., 2022).

Aleatoric and epistemic uncertainty. Yet, the translation of such *models* into robust real-world applications is frequently challenged by the reliability of their predictions (Amodei et al., 2016; Dulac-Arnold et al., 2021). This is because most significant applications demand models that can generalize effectively to novel, previously unobserved situations. Such generalization is fundamentally

a process of inductive inference — deriving insights about future or unseen instances from past observations — and is therefore inherently subject to uncertainty. For many practical purposes, and particularly in high-stakes scenarios, these predictions are only actionable if accompanied by a dependable measure of their confidence. Indeed, such *uncertainty estimates* — the central object of interest of this thesis — can prove critical: they may trigger human intervention in cases of high uncertainty, prompting an autonomous driving agent to yield driving control, soliciting further assessment by human medical workers, guiding decisions to gather more data, or simply leading to the refusal of a task the model is incapable of fulfilling.

At a closer look, uncertainty in this context manifests in distinct forms: as aleatoric or epistemic uncertainty (Der Kiureghian and Ditlevsen, 2009; Hüllermeier and Waegeman, 2020). *Aleatoric uncertainty* refers to inherent, irreducible randomness or stochasticity in the underlying process. Largely random events such as the outcome of a coin toss, trajectories of quantum particles, or short-term market movements exemplify this type; no amount of additional data regarding past events could eliminate the stochastic nature of these outcomes. For instance, while a predictive model, given access to sufficient data, might accurately learn the statistical properties governing radioactive decay (e.g., the half-life of an element, an average property of an ensemble of atoms), it cannot perfectly predict the precise moment of decay for an individual atom due to the inherent randomness of quantum mechanics.

In contrast, *epistemic uncertainty* stems from limitations in the model itself; typically arising from a lack of training data, relative to the complexity of the problem of interest (Hüllermeier and Waegeman, 2020; Kendall and Gal, 2017). This form of uncertainty reflects a lack of knowledge about the true underlying process of interest and is, in principle, reducible as more relevant data becomes available. To continue the radioactive decay example, consider a model trained to predict the half-life of elements based on their nuclear structure, using data from known terrestrial materials. When presented with a novel material exhibiting an unfamiliar nuclear configuration, perhaps discovered on an asteroid, the model's prediction for this new material's half-life carries a high chance of being inaccurate, due to the presence of epistemic uncertainty. Acquiring sufficient samples of this new material and updating the model with empirical measurements of its properties would serve to reduce this epistemic uncertainty, leading to a more refined statistical model. While both aleatoric and epistemic uncertainty relate to the likelihood of making erroneous predictions, their implications for decision-making and further action differ significantly:

- when aiming to predict the timing of an individual decay of an atom, endlessly collecting measurements of more individual atomic decays is

1

a futile endeavor.

- when aiming to predict the half-life of a material given atomic substructures, gathering data on novel materials is crucial for improving prediction quality, thereby reducing epistemic uncertainty.

Although both forms are of interest and often intermingled, *the quantification of epistemic uncertainty is typically the more challenging aspect when considering complex model classes and constitutes the primary focus of the research presented in this dissertation.*

To better understand the origin of this *epistemic uncertainty* in deep neural networks, we examine their operational mechanisms more closely. To this end, we examine the above example more closely and suppose it is our objective to infer from data a function mapping a material's nuclear structure (e.g., the arrangement of neutrons and protons) to its half-life, a presumably highly complex and nonlinear relationship. DNNs approximate such functions by composing a sequence of nested, simpler transformations organized in layers. Input data, representing the nuclear structure of a material, is passed through these intermediate layers, each of which learns to transform its input into a different, potentially more abstract, representation. These transformations are defined by relatively simple functions whose behavior is governed by a set of learnable parameters. The final layer then produces the network's output — in our example, the predicted half-life. The network's parameters are adjusted iteratively, typically via gradient-based optimization techniques, to minimize a loss function that measures the discrepancy between the network's predictions and the empirically measured half-lives in the training dataset. And indeed, it can be shown that DNN of this form are universal function approximators, that is, they can represent *any* function provided that they possess a sufficient number of layers and parameters (Cybenko, 1989; Hornik et al., 1989). Upon encountering a novel material (e.g., from the asteroid), this trained network can provide a prediction, but its correctness is not guaranteed. Due to the typically very high number of parameters, many different configurations of network parameters might explain the training data almost equally well, yet yield divergent predictions for this novel, out-of-distribution material. This plurality of plausible models consistent with the observed data is a key source of epistemic uncertainty.

Uncertainty at scale. A crucial property of DNNs is their remarkable scalability; modern large-scale models are constructed with ever-increasing depth and parameter count, sometimes reaching hundreds of billions of weights (Achiam et al., 2023). This development, while enabling unprecedented performance, also means that computational feasibility and scalability become

central requirements for any viable uncertainty quantification technique. One intuitive and theoretically well-motivated approach to quantifying epistemic uncertainty is to consider the diversity of predictions from a range of statistical models, all of which are compatible with the observed data. This concept underpins Bayesian inference, widely regarded as the gold standard framework for epistemic uncertainty quantification (Ghahramani, 2015; Jaynes, 2003). Bayesian methods aim to infer a posterior distribution over a hypothesis space of models (or model parameters), where each model is weighted by its consistency with the data and any prior beliefs. The variance or disagreement among predictions from models with high posterior probability can then serve as a measure of epistemic uncertainty. Applied to our half-life prediction example, different neural network (NN) configurations (or models) that adequately explain the terrestrial data might yield different predictions for the asteroid material. Assuming an appropriately chosen and sufficiently expressive hypothesis space, this predictive diversity reflects the epistemic uncertainty in the model's prediction. Translated to larger DNNs, however, this principled approach encounters significant computational hurdles. The hypothesis space, defined by all possible combinations of network weights, is extraordinarily large, rendering exact Bayesian inference computationally intractable (Neal, 1996) for models at scale. *A central objective of this thesis, therefore, is the development and analysis of approaches that are both principled in their quantification of uncertainty and computationally tractable for deep learning architectures.*

To summarize, we posit that reliable predictive uncertainty quantification is paramount for AI and machine learning models, particularly in applications where subsequent decisions carry significant consequences. Recent trends in the field of machine learning emphasize model scale, fueled by extensive data, as a driving factor for achieving high performance. Consequently, the scalability of uncertainty quantification methods themselves is central to their applicability. Furthermore, as AI systems increasingly function as decision-makers themselves, the integration of uncertainty awareness becomes an algorithmic cornerstone — not just a beneficial supplement. This setting introduces unique challenges but also offers novel perspectives and opportunities for designing uncertainty-aware algorithms, a theme we will explore in the context of reinforcement learning in the subsequent section.

1.2 Uncertainty in Reinforcement Learning

While the previous section aimed to highlight the broad importance of uncertainty quantification in AI, we now consider a specific (and still vast) class of problems centered on *sequential decision-making*. In this paradigm, an agent

must execute a series of decisions over time, where each action can influence the subsequent decision context, available choices, and potential outcomes. This interplay, characterized by temporal dependencies and evolving conditions, reflects a vast range of complex real-world problems and, arguably, more closely resembles biological learning processes. For example, the task of learning how to walk can be understood as a long sequence of decisions where each muscle contraction shapes the body's posture and, in turn, influences which muscles ought to contract (or relax) in the future. Often, our decision-making is guided by an intent to achieve desired consequences while averting undesirable ones and (mostly) improves as we move through life. This process of learning from experience, progressing from initially less informed decisions towards more accomplished strategies, is formalized *computationally* within the framework of reinforcement learning.

Reinforcement learning (RL) is a computational approach that emphasizes *learning from experience* as its main ingredient for addressing sequential decision-making problems under uncertainty (Kaelbling et al., 1996; Sutton et al., 1998). It operationalizes this learning process by defining an *agent* that interacts with an *environment* across a sequence of discrete time steps. At each step, the agent perceives the environment's *state*, selects an *action* based on its current knowledge, and as a consequence, the environment transitions to a new state. Concurrently, the agent receives a scalar *reward* signal, which quantifies the immediate desirability of executing said action in the given state. The overarching objective in RL is for the agent to learn a *policy* that prescribes how to act in any given state so as to maximize long-term *cumulative rewards*.

Many elements within RL are inherently stochastic: the agent's policy may itself be stochastic, environmental transitions may be random, and rewards can be noisy. Consequently, the cumulative sum of rewards over long horizons, often termed the *return*, is itself a random variable. Drawing a parallel to the discussion in Section 1.1, much like the precise decay time of an individual atom, precise future returns can be inherently unpredictable even if the underlying generative processes were perfectly known, due to aleatoric uncertainty. However, analogous to how the half-life of a radioactive material represents a predictable statistical average, RL typically seeks to estimate stable, predictable statistics of these random returns. The most common such statistic is the *expectation of returns*; an agent may aim to gauge how valuable the execution of an action in a certain state is, by estimating the expected subsequent returns. This quantity is also referred to as the *value* and lies at the core of numerous RL algorithms and serves as a crucial guide for *improving* policies. Much like the half-life prediction model discussed in Section 1.1, values can be learned from data through statistical inference in the form of *value functions* and are subject to various forms of uncertainty.

To illustrate how aleatoric and epistemic uncertainty manifest in the context of RL, consider an agent participating in a game of pure chance, such as a lottery. Here, we may interpret the played numbers as actions and the lottery payout as the reward. As the winning numbers are drawn uniformly at random, no amount of experience playing the lottery can make the agent a "formidable" player. In contrast, an agent learning a complex strategic game like chess primarily contends with *epistemic uncertainty*. Through extensive interaction and experience, the agent can reduce its lack of knowledge regarding the long-term consequences of its moves in various board configurations[1], thereby refining its internal model of optimal decision-making.

The above distinction highlights a central problem inherent to almost all of reinforcement learning: the *exploration-exploitation trade-off* (Thrun, 1992). To formulate effective decision-making strategies, particularly when assuming initial ignorance of the environment, the agent must sensibly balance actions that exploit known high-reward pathways against exploratory actions. Taking such exploratory actions allows the agent to probe unknown regions of its decision space (Auer, 2002; Thompson, 1933), potentially yielding information that is critical for discovering superior long-term strategies. This information gain, however, comes at the cost of sub optimality. In sequential decision-making problems, effective and targeted exploration may necessitate a more sophisticated form of uncertainty awareness than outlined in Section 1.1. It is then insufficient for an agent to merely recognize the novelty of a particular state or a given action; optimal behavior in sequential settings requires an understanding of how present actions might serve to encounter or avoid future uncertainties. The uncertainty about the optimal action in the present is therefore deeply interwoven with the agent's uncertainty about long-term consequences. *This notion of long-term uncertainty is innate to the RL framework and developing efficient and principled methods to account for these intricate epistemic uncertainties poses the defining challenge in the context of this thesis.*

1.3 Examples

To make the subtleties and applications of epistemic and aleatoric uncertainty in RL more tangible, this section introduces a recurrent illustrative scenario. Our chosen environment is derived from the popular deep sea scenario (Osband et al., 2016) and involves an autonomous submersible tasked with exploration and resource discovery missions under varying conditions. We will progressively augment this scenario with features designed to highlight how different forms of uncertainty arise and why their quantification is often critical

[1] To the agent, however, aleatoric uncertainty may still persist: a fixed, stochastic opponents' moves could indeed be considered a stochastic environment from the agent's perspective.

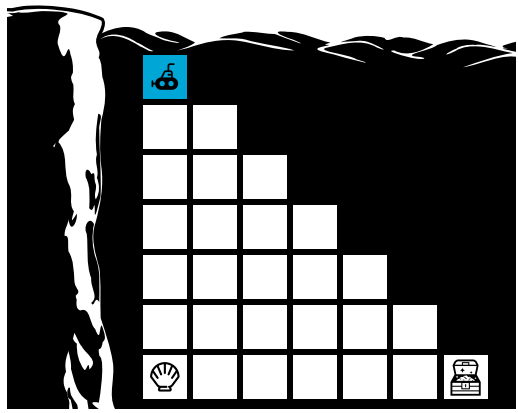


Figure 1.1: Deep Sea: Illustration of a sparse-reward environment. The agent (submersible at the top left grid cell) descends one row every timestep and can decide whether to descend to the left or right. A small reward is given for reaching sea shells (left cell on the sea floor), a large reward is given for reaching the treasure (rightmost cell on the sea floor).

for an agent's success. The first two subsections demonstrate specific applications where uncertainty awareness facilitates (1) efficient exploration and (2) conservative decision-making in safety-relevant and offline contexts. The last subsection will then, using this same illustrative framework, examine different categories of uncertainty in greater detail.

1.3.1 Efficient Exploration

As outlined in Section 1.2, a foundational challenge within RL is the exploration-exploitation tradeoff (Thrun, 1992). The difficulty of exploring an environment efficiently becomes particularly apparent in so-called *sparse-reward* environments. A characteristic of these environments is that they allot reward only to a few states and actions, typically requiring agents to perform a sequence of coordinated actions before they observe reward signals. Consider a deep sea exploration task as depicted schematically in Figure 1.1. Here, an autonomous submersible starts its mission from a predetermined entry point at the ocean surface (the top-left square in Fig. 1.1), navigating an underwater environment whose state space we model as a discrete grid. With each time step, the submersible descends one grid cell vertically and must simultaneously decide to move either one cell to the left or one cell to the right, effectively choosing between a diagonal left-downward or right-downward motion. A mission, or episode, ends when the agent arrives at the sea floor and the next episode begins at the top-left starting position. Only at the far-left rock wall can the agent descent to fields straight below it, by executing the `left` action while next to the wall; Located on the sea floor, directly beneath the submersible's

starting position and adjacent to this left rock wall, lies a bed of sea shells. Collecting these yields a consistent but modest reward. However, unbeknownst to the agent initially, a sunken treasure — offering a reward that substantially surpasses that of the sea shells — lies at the extreme rightmost reachable location on the sea floor. The objective of the agent is, as is standard, to discover the best possible policy that yields the maximum possible reward.

Let us assume the sea floor is at a depth of N cells, implying N directional decisions must be made during a single dive and it is the agent's goal to explore the sea floor for valuables. A naive exploration strategy, such as selecting `left` and `right` with equal probability (i.e., $\pi(\text{left}|s) = \pi(\text{right}|s) = 0.5$ for all states s above the sea floor), will likely lead to the discovery of the sea shells, as numerous distinct trajectories terminate at this location. However, reaching the sunken treasure requires a unique sequence of N consecutive `right` decisions. With a uniformly random policy, the probability of executing this specific sequence is 2^{-N}. Even for a moderate depth, such as $N = 25$, this probability is exceedingly small (approximately 3×10^{-8}), implying that, on average, around thirty million dives would be required to locate the treasure — a computationally prohibitive amount for such a structurally simple problem.

Conversely, an agent endowed with the capacity for uncertainty quantification can adopt a far more systematic and efficient exploration strategy. By maintaining a record of visited state-action pairs, the agent sustains estimates of its *epistemic uncertainty*, allowing it to identify actions that have remained unexplored. The uncertainty-aware agent can thus adopt a novelty-seeking strategy: It selects, at random, the log of one of its previous dives that passes a state in which there remain unexplored actions, that is, actions with high epistemic uncertainty; It repeats this dive up to said state, chooses the thus far unexplored action and follows a random policy thereafter; This simple strategy guarantees that the agent will gain information for at least one novel state-action pair with every dive such that, at $N = 25$, it should discover the treasure within a few hundred dives (for a total of $\frac{1}{2} \cdot 25 \cdot 26 = 325$ states with 2 possible actions each). This simple scenario thereby exemplifies the substantial improvements in sample efficiency and the accelerated discovery of optimal strategies attainable through proficient uncertainty quantification and its integration into the decision-making process. The above employed strategy, the prioritization of actions surrounded by high epistemic uncertainty, is sometimes referred to as the "optimism in the face of uncertainty" principle (Auer, 2002; Strehl et al., 2006) and underlies most approaches for efficient exploration.

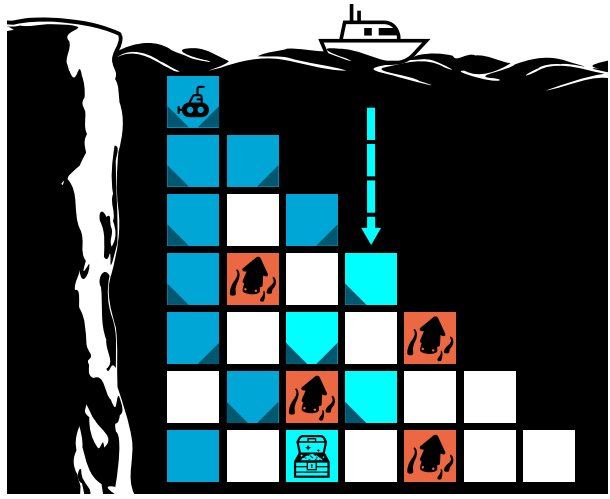


Figure 1.2: Deep Sea with Krakens: A modified version of the deep sea environment, where parts of the sea are occupied by dangerous krakens. Dark blue cells indicate trajectories contained in diving logbook and light blue cells indicate states recorded from a boat dive. The submersible must stich together logs of all dives and avoid visitation of unexplored regions to safely reach the treasure.

1.3.2 Reliable and Conservative Decision-Making

We now consider a modification to the above deep sea environment to illustrate the role of uncertainty quantification in safe and conservative decision-making. In this revised scenario, the primary treasure is presumed to be located in a more accessible region of the sea floor. However, the waters are significantly more perilous, populated by hostile *krakens* at various undisclosed locations, contact with which results in the loss of the submersible (see Fig. 1.2). Unconstrained exploration strategies, as might have been pivotal in the previous example (Section 1.3.1), are therefore deemed unacceptably risky. Instead, the agent's mission relies on a collection of logbooks from previous expeditions conducted by local divers. These logbooks document sequences of states and actions considered safe by those divers, with one log even reporting a treasure discovery. Critically, this particular treasure-finding dive originates from a different entry point from an expedition boat at sea, making it impossible for our agent to directly replicate the logged trajectory. The central challenge thus becomes to reach the rumored treasure while stringently avoiding unexplored, potentially hazardous state-action pairs (Levine et al., 2020).

In this context, epistemic uncertainty estimation derived from the logbook data becomes essential to safely navigate these waters. The submersible must infer which state-action pairs are well-supported by the combined experiences in the logbooks, exhibiting low uncertainty, and which are undocumented,

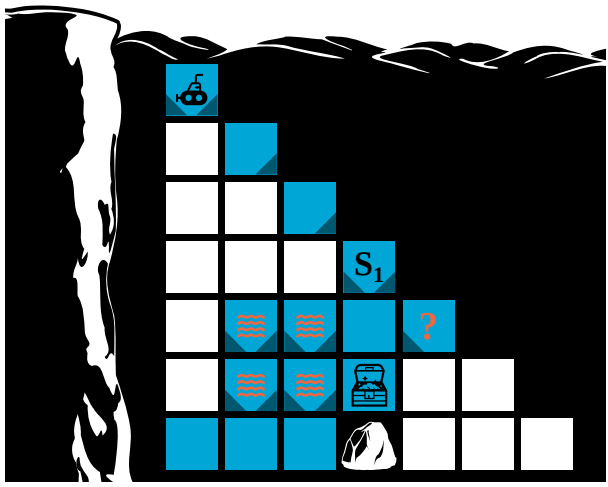


Figure 1.3: Deep Sea with Currents: A modified version of the deep sea environment illustrating aleatoric and epistemic uncertainty types. In state S_1, the agent can choose between a path through a stochastic high-current zone (action `left`) versus an unexplored path (action `right`).

thus possessing high uncertainty. A safe strategy involves constructing a path to the objective by exclusively selecting actions that have strong precedent in the logs. Actions absent from the logs for a given state are treated as having high epistemic uncertainty and are, by extension, presumed to carry unacceptable risk. Within this strategy, the agent effectively seeks to stitch together segments of known safe conduct to achieve its goal.

This principle — actively avoiding actions associated with high epistemic uncertainty — contrasts with the uncertainty-seeking behavior desirable for efficient exploration and may be interpreted as the *pessimistic* counterpart to the *optimism in the face of uncertainty* principle. Such conservative, uncertainty-aware decision-making is indispensable in many real-world, safety-critical applications; we do not seek autonomous driving agents or robotic assistants in elderly care who actively provoke novel experiences for exploration. The capacity to distinguish known, reliable strategies from unknown, potentially unsafe ones, grounded in effective uncertainty quantification, is therefore fundamental to the design and deployment of such agents (Amodei et al., 2016).

1.3.3 Types of Uncertainty

The previous examples have highlighted the functional utility of uncertainty quantification with a primary focus on the role of epistemic uncertainty. To develop a more nuanced understanding, we now further adapt our deep sea submersible scenario (illustrated in Fig. 1.3) to dissect different forms in which

1

uncertainty can manifest, how they may overlap and how these distinctions affect decision-making.

Aleatoric and epistemic uncertainty. Consider a situation where the submersible, having previously discovered the sunken treasure at a favorable location, has conducted numerous (e.g., 100) dives following an established policy, many of which successfully reach the desired treasure. This established policy, however, traverses a "high-current zone" entered by taking action `left` from a particular state S_1. Within this zone, the submersible's movements are subject to significant stochastic currents; against its own intention, the submersible may be randomly propelled to its left or right. The logs of previous dives indicate that in 90 of the 100 entries into this zone, the submersible reaches the treasure, while in the remaining 10 it is swept elsewhere. In addition to the 100 dives carried out under the established policy, the agent has performed 1 exploratory mission, in which the action `right` was executed in S_1. It is possible that this exploratory dive, too, enters a high-current zone but the single record of following this policy resulted in the desired treasure location. The choice at S_1 thus presents a clear distinction:

- Taking `left` involves primarily high *aleatoric uncertainty*. The agent has extensive data (100 dives) for this decision, so its knowledge of the outcome probabilities ($\frac{9}{10}$ treasure, $\frac{1}{10}$ elsewhere) is well-supported and thus subject to low epistemic uncertainty. However, the outcome of any single transit remains difficult to predict due to the inherent randomness of the ocean currents.

- Taking `right` as done in the single exploratory mission involves high *epistemic uncertainty*. The agent has little prior data (1 dive) for this option. While the single observed outcome for this action was successful, the true underlying outcome probabilities of this action are less well-known than for the alternative `left` action. Due to this lack of sufficient data, it is indeed possible that the true probability of reaching the treasure after choosing `right` is less favorable than the established policy.

In practice, aleatoric and epistemic uncertainty, while conceptually clearly distinct, are often present simultaneously and their interplay may complicate uncertainty quantification non-trivially.

Myopic and cumulative uncertainty. In addition to the *source* of uncertainty, as outlined previously, we can further distinguish between the relevant time horizon of the variable of interest. *Myopic uncertainty* refers to the uncertainty associated with the immediate outcome of a single action or event. *Cumulative*

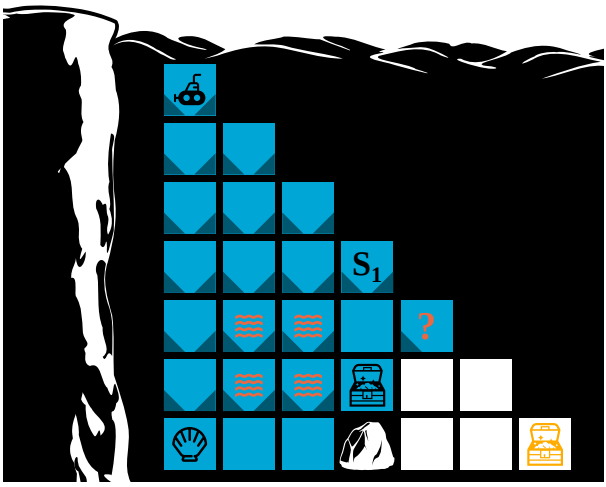


Figure 1.4: Deep Sea with Cumulative Uncerainty: A modified version of the deep sea environment, illustrating myopic and cumulative uncertainty. In the initial state (top left cell), the agent may choose between a path towards a well-explored region (action `left`) versus a path towards potentially vast, unexplored regions (action `right`) when seeking a new, greater treasure. Even though `right` bears lower myopic uncertainty, it bears higher cumulative uncertainty.

uncertainty, in contrast, pertains to the uncertainty aggregated over a sequence of decisions and environmental transitions.

To illustrate this, we modify the deep sea scenario further. Suppose, rumors emerge of an even *greater* treasure hidden elsewhere on the sea floor, and the submersible's mission is updated to find it. The existing logbook of the agent contains records of 151 dives from the initial starting state (top left cell): 50 entries involved taking the action `left` and stem from earlier missions where the agent primarily explored the region around the bed of sea shells near the cliff face; the other 101 dives involved taking `right` in the initial state and led to state S_1, through the subsequent high-current zone towards the original treasure. We focus on the *first* decision made in the initial state with the new objective of exploring the sea floor so as to find the *greater* treasure; Specifically, we aim to evaluate the choice between `left` and `right` in terms of *cumulative uncertainty*.

- Taking `left` leads to a region whose exploration utility for finding the new, greater treasure is largely exhausted. This is because all of the ensuing decisions are well-documented in our logbook indicating both *low myopic and cumulative epistemic uncertainty*.

- Taking `right`, while passing S_1 and the high-current zones in a large number of dives, can lead to regions of the open sea beyond the recorded

1

paths. This choice offers access to vast, unexplored sea regions and, provided the agent changes its policy compared to the logbook entries, exhibits *high cumulative epistemic uncertainty*.

Importantly, this situation differs from earlier scenarios where we often (and successfully) used counts to measure epistemic uncertainty. Where in these examples, the information-seeking agent was well-advised to choose actions with the least entries in a dataset, the correct decision for directed exploration (choosing `right` initially) now is documented by more samples than its alternative. This is because for directed, long-term exploration the agent ought to consider uncertainty not just about the immediate consequence of taking action `right` initially, but about the entire ensuing trajectory and its potential for information gain; even if the decision to take the action `right` initially is well-trodden, it is the only correct decision for an information-seeking agent who can subsequently encounter novel regions of the state-action space with an exploring policy. In this case, we thus speak of a decision that exhibits low *myopic epistemic uncertainty* but high *cumulative epistemic uncertainty*. Note that, while the above example treated cumulative epistemic uncertainty, we can equally distinguish between myopic and cumulative *aleatoric* uncertainties.

These categories of uncertainty — aleatoric versus epistemic, and myopic versus cumulative — can naturally intermingle and represent independent property dimensions. One might encounter myopic aleatoric uncertainty (randomness in the next step) or cumulative epistemic uncertainty (lack of knowledge about the value of a long sequence of actions). For the purposes of strategic, forward-looking decision-making, particularly in the context of exploration or ensuring long-term safety, the quantification of *cumulative epistemic uncertainty* is often the most challenging and impactful. It is precisely this form of uncertainty that is the central object of interest to many of the methods developed and analyzed in this dissertation.

1.4 Towards Efficient and Principled Uncertainty Estimation in RL

The preceding sections have highlighted the critical role of uncertainty quantification in enhancing the reliability and applicability of AI, with a particular emphasis on the unique demands presented by reinforcement learning. In the following, we first *review principles and the current state of relevant research* in this field in order to fully contextualize the research challenges addressed and the contributions made within this dissertation. This synopsis then serves to identify a research gap that the work herein aims to address. A more compre-

hensive treatment of foundational topics — including the formalism of Markov decision processes, reinforcement learning algorithms, and deep learning theory is provided in Chapter 2.

1.4.1 The State of Research

Deep reinforcement learning. The paradigm of reinforcement learning has undergone a transformative shift by the use of deep learning models, leading to landmark achievements such as mastering complex strategic games like Go, Atari, and StarCraft II (Mnih et al., 2015; Silver et al., 2016; Vinyals et al., 2019), learning intricate robotic manipulation skills from raw visual input (Kalashnikov et al., 2018; Levine et al., 2016), and fine-tuning the behavior of large-scale language models (Christiano et al., 2017; Ouyang et al., 2022). Despite this progress, deploying agents reliably and safely necessitates a robust handling of uncertainty. This is critical for guiding efficient exploration and overcoming the notorious sample inefficiency of RL (Burda et al., 2019b; Ecoffet et al., 2019; Guo et al., 2022), ensuring safety by avoiding high-risk, unobserved states in applications like autonomous driving (Hoel et al., 2023; Wu et al., 2022), and enabling stable, scalable learning from fixed, offline datasets by preventing the exploitation of actions which lack data (An et al., 2021; Levine et al., 2020). This challenge is often exacerbated by the entanglement of epistemic and aleatoric types of uncertainty, as well as the complex propagation of uncertainties required in sequential decision making problems, as discussed in the following.

Uncertainty estimation in deep learning. Uncertainty estimation in deep reinforcement learning typically faces the presence of two distinct, epistemic and aleatoric types of uncertainty (Hüllermeier and Waegeman, 2020). The quantification of the aleatoric uncertainty type — uncertainty due to inherent randomness — is addressed by the dedicated framework of *distributional RL*, which models entire return distributions rather than expected returns (Bellemare et al., 2017). The quantification of epistemic uncertainty — uncertainty due to model ignorance — is often characterized by a trade-off between theoretical rigor and computational feasibility. *Bayesian inference* offers a normative framework for reasoning about model uncertainty by maintaining a posterior distribution over model parameters (Gelman and Shalizi, 2013; Jaynes, 2003; Neal, 1996). However, its application to high-dimensional deep NNs is generally intractable and must rely on alternative approximation techniques such as (amortized) variational inference (VI, Blei et al., 2017; Gal and Ghahramani, 2016; Kingma and Welling, 2014; Rezende and Mohamed, 2015) or approximate sampling methods like Markov chain Monte Carlo (Welling and Teh, 2011). As an appealing pragmatic alternative, *deep ensembles* have demonstrated remark-

1

able empirical performance by aggregating predictions from several independently trained networks (Lakshminarayanan et al., 2017). Their effectiveness has been shown for exploration (Osband et al., 2016) and offline RL (An et al., 2021), yet their computational and memory costs, which scale linearly with the number of models, remain a significant barrier, especially for large models. Complementing these are computationally efficient single-model methods like random network distillation (Burda et al., 2019b), and self-predictive errors (Guo et al., 2022) which are effective in practice but often lack the thorough theoretical understanding of their more costly counterparts.

Deep learning theory. A deeper theoretical understanding of how and why these uncertainty quantification methods operate in the context of deep learning has been advanced by recent progress in deep learning theory. The development of the neural tangent kernel (NTK) framework, in particular, has provided a new lens through which to analyze the behavior of these methods in the idealized limit of infinite network width (Jacot et al., 2018; Lee et al., 2020b). Using NTK theory, one can for example gain analytical insights that formally characterize and connect the predictions of deep ensembles to Bayesian inference (He et al., 2020). However, standard NTK formulations have not been widely translated to the learning pipelines in common RL settings with few exceptions (Lyle et al., 2022; Xiao et al., 2021).

1.4.2 Research Mission

From our preceding survey of the research landscape, we derive two central conclusions: first, that reliable uncertainty quantification — particularly for epistemic uncertainty — is of paramount importance for developing capable and trustworthy agents in sequential decision-making problems; and second, that the current state of research is characterized by a persistent tension. Methods with strong theoretical foundations, such as full Bayesian inference, tend to face significant hurdles in computational efficiency and scalability, while more scalable methods tend to be heuristic in nature, lacking a thorough analytical understanding or a motivation derived from theoretical principles.

The mission of this dissertation is to engage directly with this trade-off. We aim to develop novel algorithms and analyze existing ones to advance the state of epistemic uncertainty estimation towards methods that are both computationally efficient and theoretically well-motivated. It is our belief that such methods will prove to be more robust and reliable across a variety of distinct problems, while remaining computationally tractable for application with contemporary, large-scale NN architectures.

A recurrent theme in the research presented herein is a departure from the

"black-box" view that is sometimes applied to deep NNs. We find that such a perspective can conceal what lies at the root of epistemic uncertainty estimation in deep learning: the specific mechanisms by which neural networks perform inductive inference — that is, how they generalize from observed evidence to make predictions about novel inputs. Instead of obscuring these mechanisms, our approach seeks to use insights into this generalization behavior to design algorithms that are grounded in, and seek to leverage, this generalization behavior. In doing so, we draw upon recent results from deep learning theory as well as empirical findings from the broader deep learning and deep reinforcement learning literature (Bellemare et al., 2017; Dabney et al., 2018b; Jacot et al., 2018; Lee et al., 2018a; 2020b; Xiao et al., 2021).

We aim to develop and analyze these methods specifically within the domain of deep reinforcement learning. This field presents a particularly compelling setting, as it not only introduces a unique set of challenges for uncertainty quantification but also offers a richer, and importantly, a more measurable variety of benefits from uncertainty-aware algorithms. With the objectives of efficiency and theoretical motivation in mind, we focus on algorithms that estimate a particularly challenging form of uncertainty: the long-term, cumulative epistemic uncertainty that compounds over time in sequential decision-making. This type of uncertainty accounts not only for immediate, myopic uncertainties but also for the downstream knowledge gaps that may be encountered far in the future as part of a long sequence of interactions. This form of uncertainty is central to major open challenges in sequential decision-making like efficient exploration and the safe, reliable deployment of RL agents.

The translation of insights from deep learning theory and general deep learning to the specific context of deep RL is not always straightforward. The optimization pipelines and learning dynamics encountered in RL — often involving non-stationary data distributions and complex credit assignment problems — can influence NN behavior in distinct ways. This dissertation therefore also addresses the significant challenge of adapting results from different strands of machine learning for the specific conditions encountered in deep reinforcement learning. The central mission of this thesis can thus be summarized as: *to design and analyze computationally efficient, theoretically motivated uncertainty quantification methods for cumulative value uncertainty within deep reinforcement learning.*

1.5 Contents of This Thesis

This final section of the introduction details the specific scope and structure of the dissertation, outlining the guiding research questions, summarizing the main contributions, and providing a guide to the subsequent chapters.

1

1.5.1 Research Questions

To address the central mission outlined previously, our research is organized around a principal question that is subsequently investigated through several specific lines of inquiry. Within these lines of inquiry, we formulate four research questions, each corresponding to a dedicated chapter. In pursuing this approach, we tackle the broader challenge from several complementary angles, while progressing from modular multi-model methods towards more efficient, unified single-model solutions.

Principal research question. The central research question guiding this dissertation is:

How can we develop algorithms for the quantification of long-term cumulative uncertainties in deep reinforcement learning that reconcile computational efficiency with principled theoretical foundations?

To address this question, we pursue three main lines of inquiry, each associated with more specific technical questions.

1. Enhancing ensemble diversity. Our first line of research investigates uncertainty estimates provided by deep ensembles and whether such estimates can be improved by injecting diversity beyond independent parameter initializations through distinct architectural designs in the ensemble's members. This leads to our first specific question:

RQ1: *Can member-specific architectural choices in deep ensembles promote diverse generalization behaviors and thereby improve the quality of uncertainty estimates?*

2. Emulating ensembles with a single model. A central theme of this work is to develop efficient methods for uncertainty quantification by drawing from recent theoretical insights of deep learning theory. This line of research investigates supervised single-model methods as approximations to entire ensembles, providing theoretical grounding for popular existing methods and developing novel single-model approaches. This gives rise to two related research questions:

RQ2: *Can the predictive variance of supervised deep ensembles be approximated directly and accurately by a single neural network in the limit of infinite width?*

RQ3: *What is the theoretical nature of the uncertainty captured by random network distillation, as a prominent example of single-model heuristic methods, when analyzed in the infinite-width limit?*

3. Emulating ensembles of value functions with a single model. Finally, leveraging insights from the preceding investigations, this line of research investigates a single-model approach to directly estimate uncertainties of deep value functions for a broad range of policies. This inquiry is guided by our final research question:

RQ4: *Can the predictive variance of an ensemble of deep value functions be approximated directly and accurately by a single neural network in the limit of infinite width?*

1.5.2 Contributions

In addressing the previously stated research questions, this dissertation makes the following contributions to the field of uncertainty quantification in deep reinforcement learning:

- In response to RQ1, we introduce and analyze *diverse projection ensembles* for deep distributional reinforcement learning. We show that by explicitly incorporating distinct projection mechanisms in the ensemble's constituent models, we can achieve reliable uncertainty estimates with fewer models than ordinary deep ensembles. We show empirically, that leveraging such uncertainty estimates as an exploration bonus leads to significant performance improvements in challenging hard exploration tasks (Chapter 3).

- In response to RQ2, we propose *contextual similarity distillation (CSD)*, a novel single-model method for efficiently approximating the predictive variance of deep ensembles directly. Derived from insights using neural tangent kernel theory, CSD reframes direct variance prediction as a regression problem using kernel-based similarities as labels. Under certain idealized conditions, we show that this setting allows a single network to emulate the predictive uncertainty of an ensemble with infinite members. We show empirically that CSD achieves competitive performance on popular uncertainty-quantification benchmarks and serves as a reliable intrinsic reward in hard exploration tasks (Chapter 4).

- In response to RQ3, we provide a theoretical analysis of the widely-used *random network distillation (RND)* algorithm (Burda et al., 2019b). Using NTK theory, we formally establish an equivalence between the uncertainty signal produced by RND and the predictive variance deep ensembles in the limit of infinite width. Drawing from this insight, we further show that the RND algorithm can be modified to recover the finite-sample variance of exact posterior predictive distributions of Bayesian neural networks (BNNs), in the limit of infinite width. We thereby unify these distinct approaches within

1

a single theoretical framework and justify the use of RND as a principled method for uncertainty quantification in this idealized limit (Chapter 5).

- Finally, in response to RQ4, we develop *universal value-function uncertainty (UVU)*, an efficient single-model approach designed specifically to estimate long-term, cumulative epistemic uncertainty in universal value functions. The method measures uncertainty as self-predictive errors between an online learner trained with temporal difference learning on a synthetic reward signal derived from a fixed target network. We provide a theoretical analysis in the NTK regime that establishes an exact equivalence between the UVU error signal and the variance of a corresponding ensemble of universal value functions. We furthermore show that the approach is highly effective empirically in a challenging multi-task offline reinforcement learning setting (Chapter 6).

1.5.3 Thesis Outline

This dissertation is structured as follows:

- *Chapter 1* (this chapter) provides an introduction to the research topic, including its motivation, illustrative examples, and a guide to the structure of the dissertation.
- *Chapter 2* provides a background for the research in this thesis and reviews the current state of research.

Enhancing ensemble diversity.

- *Chapter 3 (RQ1)* presents work on diverse projection ensembles in distributional reinforcement learning, including theoretical analysis and empirical results on exploration tasks. Further background on distributional RL is provided within the chapter.

Emulating ensembles with a single model.

- *Chapter 4 (RQ2)* introduces contextual similarity distillation (CSD), detailing its theoretical derivation from NTK principles and evaluating its performance as an efficient single-model estimator of ensemble variance. Further background on NTK theory and supervised kernel regression is included.
- *Chapter 5 (RQ3)* focuses on the theoretical analysis of random network distillation and its modification towards the Bayesian framework. Further background on RND and Gaussian process regression is included.

Emulating ensembles of value functions with a single model.

- *Chapter 6 (RQ4)* introduces the universal value-function uncertainties (UVU) method. The chapter contains an algorithmic description, an NTK-based theoretical analysis, and empirical validation in offline reinforcement learning contexts. Further background on NTK theory and value-function uncertainties is included.

- *Chapter 7* provides a general discussion, synthesizing the findings from the preceding chapters, revisiting the research questions, suggesting limitations and directions for future research, and a final conclusion.

- The main body of the thesis is followed by a bibliography, appendices of supplementary material, and a list of the author's publications.

2

Background

2

The preceding chapter has motivated the central research goals of this dissertation, highlighting the critical role of uncertainty quantification in enhancing the performance and reliability of reinforcement learning agents. To lay a foundation for the novel contributions presented in the subsequent chapters, this chapter offers a broad and accessible review of the principles that underpin this work. While our aim here is to be comprehensive, we defer more specialized technical details to the dedicated background sections within each of the core content chapters, where they are most relevant.

Our review begins by delineating the mathematical formalism of Markov decision processes (MDPs), the standard framework for sequential decision-making under uncertainty. Building upon this, we will examine the core elements of reinforcement learning (RL), where agents aim to learn optimal strategies through interaction in environments with initially unknown dynamics, and its subsequent evolution into deep reinforcement learning through the integration of deep neural networks. The discussion will then shift to the specific challenges of uncertainty quantification within the deep RL paradigm. We first explore methods pertinent to *aleatoric uncertainty*, notably the framework of distributional reinforcement learning, before turning our focus to approaches for quantifying *epistemic uncertainty*, covering Bayesian inference, ensemble methods, and an assortment of other techniques. Finally, the chapter concludes with an introduction to perspectives from deep learning theory, which offer important insights into the behavior of deep neural networks, the central model class used throughout this thesis and field.

2.1 Markov Decision Processes

The predominant mathematical framework for modeling sequential decision-making problems under uncertainty, where outcomes are influenced by both the agent's actions and stochastic environmental factors, is the Markov decision process (MDP) (Bellman, 1957; Puterman, 2014). Initially described in the work of Bellman, MDPs formalize the decision-making process as an interaction between an agent and an environment evolving over discrete time steps. An MDP is formally defined by a tuple (i.e., an ordered list) $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$. Here, $\mathcal{S}$ denotes the set of possible states the environment can occupy, and $\mathcal{A}$ represents the set of actions available to the agent. The function $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$, often termed the transition probability function or transition kernel, specifies $P(s'|s, a)$, the probability of the environment transitioning to state $s' \in \mathcal{S}$ upon the agent taking action $a \in \mathcal{A}$ in state $s \in \mathcal{S}$. The reward function $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ defines the scalar reward[1] received by the agent after transition-

[1]Reward functions are sometimes defined as $R(s, a)$ or even $R(s)$. $R(s, a, s')$ is general and often interchangeable with $R(s, a)$ through $R(s, a) = \mathbb{E}_P[R(s, a, s')]$.

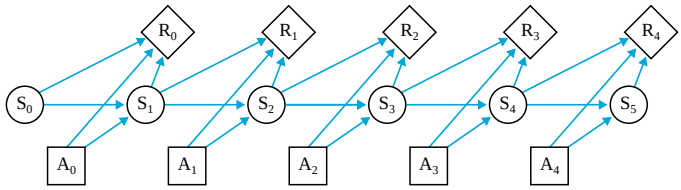


Figure 2.1: An illustration of a Markov decision process.

ing from state s to s' as a result of action a. Finally, $\gamma \in [0,1)$ is a discount factor that assigns a geometric series of weights to rewards, giving relative preference to immediate rewards over rewards encountered in the far future.

The Bellman equation. Within an MDP, the primary objective is typically to identify an optimal *policy*, denoted π^*. A policy $\pi(a|s)$ is a mapping from states to a probability distribution over actions, prescribing the agent's behavior. While various optimality criteria are described in the literature, the most widely adopted is the maximization of the expected discounted sum of future rewards $\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R_t]$, often referred to as the expected discounted return (Blackwell, 1962; Howard, 1960). The discount factor γ ensures that the sum of rewards remains bounded, even for infinite-horizon problems, and gives greater preference to immediate rewards. Central to determining optimal policies are *value functions*. The state-value function, $V^\pi(s)$, quantifies the expected discounted return starting from state s and subsequently following policy π. Similarly, the action-value function, $Q^\pi(s,a)$, assesses the expected discounted return upon taking action a in state s and thereafter adhering to policy π. Remarkably, these value functions satisfy a fundamental recursive relationship, described by the Bellman equations (Bellman, 1957). For a given policy π, the Bellman equation for $Q^\pi(s,a)$ is given by

$$Q^\pi(s,a) = \mathbb{E}_{s' \sim P(\cdot|s,a), a' \sim \pi(\cdot|s')}\left[R(s,a,s') + \gamma Q^\pi(s',a')\right]. \tag{2.1}$$

Equation 2.1 is a defining property of MDPs and underlies an overwhelming number of algorithms and methods that aim to solve them. It reveals that the value function, evaluated in the state-action pair (s,a), equals exactly the sum of the discounted expected value in the next state-action pair $\mathbb{E}_{P,\pi}[\gamma Q^\pi(s',a')]$ and the expected immediate reward $\mathbb{E}_P[R(s,a,s')]$. A perhaps even more crucial property of the Bellman equation is that the value function Q^π for a given policy π is its unique global solution. In other words, given an MDP and a policy π, if one is able to find a function that satisfies the corresponding Bellman equation for all states (or state-action pairs), then this function must be the unique value function of the policy π. Optimal policies, π^*, are those that

achieve the maximum possible value $V^*(s)$ or $Q^*(s,a)$ from any state or state-action pair, respectively (Puterman, 2014). In the most common settings, an optimal policy moreover acts greedily with respect to its optimal action-value function[2]: $\pi^*(a|s) = 1$ if $a = \arg\max_{\tilde{a}} Q^*(s,\tilde{a})$. If we furthermore assume that our MDP defines a probability distribution over initial states μ, value functions allow us to imbue policies with a definitive ranking: if a policy π on average achieves higher values $J(\pi) = \mathbb{E}_\mu[V^\pi(s)]$ in the initial states than another policy π', then π must be a better policy according to our optimality metric[3].

Value iteration. When the environment's dynamics P and reward function R are fully known, the problem of finding an optimal policy is often termed *planning*. For MDPs with finite state and action spaces, a prominent class of algorithms known as dynamic programming (DP) provides methods to compute optimal policies (Bellman, 1957). DP algorithms typically represent value functions and policies in tabular form, with distinct entries for each combination of state and action. Two foundational DP algorithms are value iteration and policy iteration. *Value iteration* starts with an arbitrary initial value table, $Q_0(s,a)$ (which may in fact not be the value function corresponding to any policy), and iteratively refines its estimates using an update rule derived from the Bellman equation: at each iteration k, the estimates are updated for all state-action pairs (s,a) according to

$$Q_{k+1}(s,a) \leftarrow \sum_{s' \in \mathcal{S}} P(s'|s,a)\big(R(s,a,s') + \gamma \max_{a' \in \mathcal{A}} Q_k(s',a')\big), \quad \forall (s,a) \in \mathcal{S} \times \mathcal{A}. \tag{2.2}$$

Intuitively, this update rule turns the *Bellman optimality equation* — the Bellman equation for a policy π^* that is greedy w.r.t. itself — into an update rule. Formally, we refer to this update rule as the *Bellman optimality operator*. Repeated application thereof is guaranteed to converge to the optimal value function, $Q_k \to Q^*$ as $k \to \infty$ (Puterman, 2014). Having established Q^*, an optimal deterministic policy π^* can be readily extracted from Q^*.

Policy iteration. *Policy Iteration*, alternatively, explicitly maintains and improves a policy. It begins with an initial policy π_0 and iterates between two steps (until convergence)

[2] Here, we have implicitly assumed that there exists a deterministic optimal policy, which is the case for most common unconstrained MDP formulations (Puterman, 2014).

[3] The definition of a start-state distribution is an optional addition to the definition of MDPs. Without such a criterion, a global ranking of policies becomes unattainable unless one defines alternative weightings over the state space.

1. *Policy evaluation:* Given the current policy π_k, compute the value function $Q^{\pi_k}(s,a)$. This is achieved by solving the Bellman equations (Eq. 2.1 with π_k) for all $(s,a) \in \mathcal{S} \times \mathcal{A}$ through repeated application of the Bellman operator for policy π_k (until convergence) with the update rule

$$Q_{j+1}(s,a) \leftarrow \sum_{s' \in \mathcal{S}} P(s'|s,a) \sum_{a' \in \mathcal{A}} \pi_k(a'|s') \big(R(s,a,s') + \gamma Q_j(s',a') \big), \quad (2.3)$$

   where $Q_j \to Q^{\pi_k}$ as $j \to \infty$. Again, we have turned the Bellman equation (here for the policy π_k) into an update rule, thereby obtaining what we refer to as the *Bellman operator* (or sometimes the *Bellman expectation operator*).

2. *Policy improvement:* Improve the policy by making it greedy with respect to the just-computed value function Q^{π_k} through the update rule

$$\pi_{k+1}(a|s) \leftarrow \begin{cases} 1 \quad \text{if} \quad a = \arg\max_{\tilde{a} \in \mathcal{A}} Q^{\pi_k}(s,\tilde{a}), \\ (\text{for ties, break arbitrarily}), \\ 0 \quad \text{otherwise}. \end{cases} \quad (2.4)$$

Policy iteration, too, is guaranteed to converge to an optimal policy π^* and its corresponding optimal value function Q^* (Howard, 1960; Puterman, 2014).

A mature body of literature details numerous extensions, hybrid algorithms (such as modified policy iteration), and generalizations of these DP methods. However, these methods may face computational and indeed theoretical limits for large or continuous state-action spaces. This is due to the so-called "curse of dimensionality" (Bellman, 1957), which will be addressed in more detail in Section 2.3. A further restrictive condition of DP methods is the assumption that the transition model and reward function are known *a priori*. The need to overcome these limitations, particularly the assumption of a known model, motivates the reinforcement learning paradigm, which focuses on *learning* from direct interactions when such prior knowledge is unavailable.

2.2 Reinforcement Learning

The DP methods discussed in the previous section solve Markov decision processes while assuming complete knowledge of the environment's transition probabilities P and reward function R. This assumption, however, severely limits their applicability in numerous real-world scenarios where such knowledge can not be assumed to be available a priori. To this end, RL presents a computational paradigm that emphasizes *learning algorithms* as a central means to

obtain optimal policies, obtained through direct interaction with an a priori unknown environment (Kaelbling et al., 1996; Sutton et al., 1998). In the RL setting, agents accumulate experience by continually executing actions and observing subsequent states and rewards. RL algorithms then leverage this interactional data to iteratively refine an internal model for decision-making, which may take various forms, such as value functions, policies, or explicit environment models. This paradigm of "learning to act" from experience opens avenues towards achieving mastery in problem domains that are otherwise too diverse or complex to solve analytically.

Q-Learning. Seminal work by Watkins and Dayan (1992) formally established this bridge between iterative learning algorithms and DP solutions for MDPs through the introduction of Q-learning. Q-learning is an algorithm that aims to directly *learn* the optimal action-value function, $Q^*(s,a)$, in a manner analogous to the value iteration algorithm introduced in Section 2.1. As in value iteration, Q-learning typically starts with an initially arbitrary table of Q-values. Through interaction with the environment, the agent then collects experiences — tuples of (state, action, reward, next state) — which are used to perform updates to the Q-table entries. These updates are driven by what we call the *empirical* Bellman optimality operator, where expectations over next states and rewards are replaced by their observed sample values. Specifically, for an experienced transition (s,a,r,s'), the Q-value $Q(s,a)$ is adjusted based on the observed reward r and the estimated maximum Q-value in the subsequent state s' with the update rule given by

$$Q_{k+1}(s,a) \leftarrow (1-\eta)Q_k(s,a) + \eta\big(r + \gamma \max_{a' \in \mathcal{A}} Q_k(s',a')\big), \tag{2.5}$$

where $0 < \eta < 1$ is a learning rate. Unlike value iteration, where all state-action pairs can be updated synchronously using due to the exhaustive prior knowledge of the environment, Q-learning updates are performed asynchronously and affect only the experienced state-action pairs. Due to the stochastic nature of empirical, sampled transitions and rewards, the Q-learning furthermore uses a *step size* or *learning rate* η. In RL, agents are actors in the environment. Unlike DP algorithms, RL algorithms are therefore also characterized by their data collection or *exploration* strategy and how it relates to the employed learning algorithm. Q-learning, on this account, is characterized by its *off-policy* nature: it can learn the optimal Q-function even from data collected by a behavioral policy that is different from the one implied by the current Q-table (e.g., data from a random or exploratory policy). Addressing the above-described traits inherent to RL — empirical updates and data collection — by assuming appropriate learning rate schedules and sufficient exploration of all state-action pairs, we

can guarantee that Q-learning converges to the optimal Q-function, Q^*, from which an optimal policy π^* can be derived (Tsitsiklis, 1994).

SARSA. In contrast to the off-policy Q-learning algorithm, the SARSA algorithm is an *on-policy* method that estimates the action-value function $Q^\pi(s,a)$ for the *current behavior policy* π (Rummery and Niranjan, 1994; Singh and Sutton, 1996). SARSA collects experiences by following the policy π and updates the Q-value of a state-action pair (s,a) based on the reward r received, the next state s', and crucially, the next action a' selected by the behavior policy π during collection. This results in the update rule

$$Q_{k+1}(s,a) \leftarrow (1-\eta)Q_k(s,a) + \eta\big(r + \gamma Q_k(s',a')\big). \tag{2.6}$$

The update thus uses an empirical sample of the Bellman operator specific to the behavior policy π. To ensure convergence to an optimal policy, SARSA must therefore balance exploration with exploitation within its behavior policy π and gradually shift towards greediness with respect to the learned Q-values. A common approach to this end is to employ an ϵ-greedy policy, which selects the action with the highest Q-value estimate but chooses a random action with a probability ϵ. By gradually decaying ϵ towards zero (and assuming appropriate learning rate schedules), SARSA also converges to the optimal policy and value function (Singh et al., 2000).

Actor-critic algorithms. The Q-learning and SARSA algorithms are examples of *value-based* RL methods, as their primary focus lies in learning a value function from which a policy is subsequently derived (typically by acting greedily). An alternative class of algorithms, termed *policy-based* methods, instead seeks to learn a parameterized policy directly without necessarily learning an explicit value function (Williams, 1992). These approaches typically optimize the policy directly to effect an increased probability of selecting actions correlated with high expected returns. The policy gradient theorem provides a theoretical basis for such updates, relating changes in policy parameters[4] to expected returns (Sutton et al., 1999). Policy-based methods possess certain advantages in continuous action spaces or when stochastic policies are desired. Further extending these ideas, *actor-critic methods* present a hybrid approach that aims to combine value-based and policy-based approaches by maintaining and learning distinct models for both a policy (the actor) and a value function (the critic) (Konda and Tsitsiklis, 1999). The critic estimates a value function (e.g., state-value or action-value) to evaluate the actions chosen by the actor, and the actor updates its policy parameters based on the critic's feedback, often in a direction suggested by the policy gradient.

[4] Tabular policies, too, can be considered parametric by regarding each table entry as a parameter.

Model-based RL. The majority of methods discussed thus far — value-based, policy-based, and actor-critic — are typically categorized as *model-free* reinforcement learning, as they do not require or learn an explicit model of the environment's transition dynamics $P(s'|s,a)$ or reward function $R(s,a,s')$. In contrast, *model-based* RL encompasses a significant body of techniques that explicitly learn an approximate model of the environment from interaction data. Once such a model, $\hat{P}$ and $\hat{R}$, is learned, the agent can, in principle, use planning algorithms (like value iteration or policy iteration) with this learned model to compute a policy, effectively performing planning using simulated experience (Sutton, 1991). Because the learned model is generally an approximation, model-based methods too require careful consideration of exploration strategies to gather data that allows the agent to refine a highly accurate model globally, that is, across the state-action space spanned by the environment.

Many foundational RL algorithms were initially developed and analyzed in *tabular* settings, where value functions, policies, and sometimes models are represented explicitly as lookup tables with entries for each discrete state or state-action pair. However, this approach suffers significantly from the so-called "curse of dimensionality" (Bellman, 1957), becoming computationally infeasible quickly as the size and dimensionality of state and action spaces grow. For most problems of practical interest, such spaces are vast or even continuous. This critical limitation thus stimulates the adoption of *function approximation* methods, where policies, value functions, or models are represented as parameterized functions that can generalize across numerous states and actions. The advent of deep neural networks as highly expressive and scalable function approximators has had a particularly transformative effect, leading to the field of *deep reinforcement learning*.

2.3 Deep Reinforcement Learning

The early tabular implementations of the foundational reinforcement learning algorithms as described previously encounter significant computational and indeed theoretical limitations when applied to problems with large or continuous state and action spaces. This challenge is a manifestation of the "curse of dimensionality" (Bellman, 1957), whereby the computational footprint for explicitly representing value functions or policies grows exponentially with the dimensionality of the state and action spaces. For instance, consider the classic pendulum swingup problem, as depicted in Fig. 2.2 (left): the agent must apply torque to a single joint to swing and balance a pendulum into an upright position. The joint angle and joint angle velocity of the pendulum constitute the state space of this environment and agents can choose between three

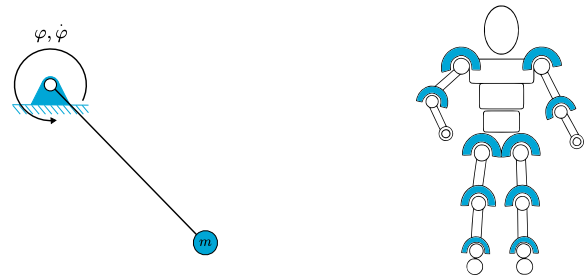


Figure 2.2: (*left*:) An illustration of a swingup pendulum. An agent must apply torque φ to swing the pendulum up into an upright position and maintain this position. (*right*:) An simplified illustration of a humanoid robot.

discrete actions (torque left, right, or none). Discretizing each dimension of this state space into 36 compartments, the resulting Q-value table would require $36 \cdot 36 \cdot 3 = 3,888$ entries. In contrast, consider a humanoid robot with 17 joints, leading to a state space of 40 dimensions, accounting for 17 joint angles, 17 joint velocities, and the robot's center of mass position and velocity in space. For simplicity, we will assume that the agent can, again, choose between three discrete levels of torque (left, right, or none) per joint[5]. A discretization of this state space with the same resolution of 36 distinct compartments as before would result in a Q-table with the astronomically large number of $36^{40} \cdot 3^{17} = 2.3076044 \cdot 10^{70}$ entries, rendering tabular approaches utterly intractable for any conceivable computing system.

Linear function approximation. As a consequence of this, the field of RL has gradually shifted focus towards the adoption of *function approximation* methods. Instead of maintaining explicit entries for every state or state-action pair, value functions, policies, or environment models are here represented by parameterized functions that can generalize from experienced situations to novel ones (Sutton et al., 1998). Linear function approximation, where a function such as the action-value $Q(s,a;\theta)$ is represented as a linear combination of a set of pre-defined basis functions or features $\phi_i(s,a)$ according to

$$Q(s,a;\theta) = \sum_{i=1}^{N_\theta} \theta_i \phi_i(s,a) = \theta^\top \phi(s,a). \tag{2.7}$$

[5]In practice, such a humanoid robot would require even higher-dimensional state and action descriptions. Typical simulated models of such robots account explicitly for positions and velocities of all body parts (376-dimensional) and use continuous action spaces (17-dimensional).

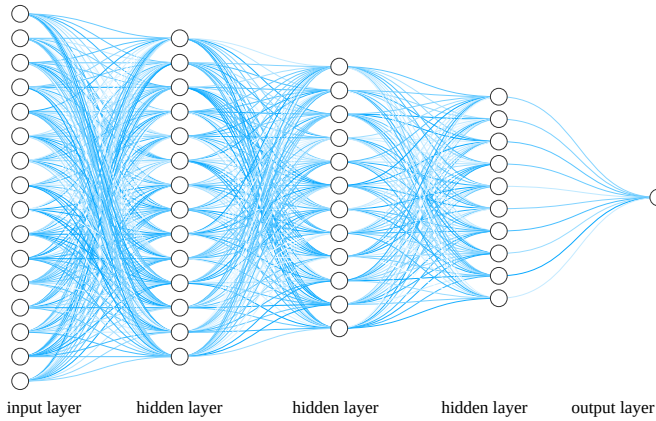


Figure 2.3: An illustration of a fully connected deep neural network.

is an approach widespread in earlier works and analytically well-understood. Here, $\phi(s,a)$ is a vector of N_ϕ feature values derived from the state-action pair, and $\theta = (\theta_1, \ldots, \theta_{N_\theta})^\top$ is the vector of learnable parameters. For the humanoid robot example, sensible feature mappings might include trigonometric functions of joint angles (e.g., $\sin(ik\varphi_j)$ for joints indexed by j a selection of k) and the joint velocities themselves. By adjusting a relatively small number of parameters[6] θ_i, the agent can shape the Q-value estimates across the entire continuous state-action space. The efficacy of linear function approximation, however, is tied heavily to the quality of the handcrafted feature space, whose design can prove to be highly complex and leads into deep waters of functional analysis.

Neural function approximation. Deep neural networks (DNNs) offer a powerful alternative to linear function approximation by extracting relevant features from data as part of an *end-to-end* learning process (Goodfellow et al., 2016; LeCun et al., 2015). Loosely inspired by biological neural computation, DNNs construct complex, nonlinear functions by composing multiple layers of simpler transformations. In their most simple form, neural networks process input signals (e.g., representing a state s) through a sequence of hidden layers, each comprising a fixed number of "neurons". Each neuron in a layer forms its individual output by computing a weighted sum of the outputs from the preceding layer, adds a bias term, and finally passes this sum through a nonlinear "activation" function $\sigma(\cdot)$. For a network with L layers in total, the computation for

[6]Small, naturally, is a rather subjective measure but in this case refers to the fact that the number of parameters is not tied exponentially to the number of states.

a neural Q-function $Q(s,a;\theta)$ could for instance be abstractly represented as

$$z^{(0)} = \begin{pmatrix} s \\ a \end{pmatrix}, \quad \text{(input)} \tag{2.8}$$

$$z^{(l)} = \sigma(W^{(l)} z^{(l-1)} + b^{(l)}), \quad \text{for } l = 1,\dots,L-1 \quad \text{(hidden layers)} \tag{2.9}$$

$$Q(s,a;\theta) = W^{(L)} z^{(L-1)} + b^{(L)}, \quad \text{(output layer)} \tag{2.10}$$

where $z^{(l)}$ is the vector of outputs for an intermediate layer l, $W^{(l)}$ are weight matrices, $b^{(l)}$ are bias vectors, and θ collectively denotes the learnable parameters, that is, the weights and biases $\{W^{(l)}, b^{(l)}\}_{l=1}^{L}$. We typically refer to neural networks as "deep" when we aggregate at least two hidden layers ($L > 2$). DNNs are so-called *universal function approximators*, capable in principle of representing arbitrarily complex functions (Cybenko, 1989; Hornik et al., 1989). A popular view among practitioners is that DNNs achieve this by learning a hierarchy of salient features, (e.g., from images, audio, graphics, text, etc), thereby autonomously performing the previously manual task of constructing feature spaces. Their integration into reinforcement learning, where the complex structure of objects like the value function often eludes simple intuition, has been transformative and defines the field of "deep reinforcement learning".

Deep RL algorithms. An early influential precursor in this domain was *temporal-difference-Gammon* (TD-Gammon), an algorithm that achieved master-level play in Backgammon, using a simple DNN class called multilayer perceptron, to approximate the value function (Tesauro et al., 1995). The contemporary era of deep RL has been shaped significantly by the deep Q-network (DQN) algorithm, which successfully learned to play a variety of Atari 2600 video games directly from raw pixel inputs using convolutional neural networks (Mnih et al., 2015). This was followed by landmark achievements such as AlphaGo and its successors (e.g., AlphaZero, MuZero) which attained superhuman performance in notoriously complex strategic board games like Go, chess, and shogi by combining DNNs for policy and value estimation with the planning algorithm Monte Carlo tree search (Schrittwieser et al., 2020; Silver et al., 2016; 2017). Beyond games, deep RL has demonstrated significant promise in robotics, enabling the learning of complex control and manipulation policies (Kalashnikov et al., 2018; Levine et al., 2016), in the control of fusion reactors (Degrave et al., 2022) and in the fine-tuning of large language models (Ouyang et al., 2022).

The remarkable successes of deep RL underscore its potential as a key technology for developing autonomous agents capable of operating effectively in complex, high-dimensional environments. This is largely attributable to the capacity of deep neural networks to learn highly expressive functions from raw data with minimal need for domain-specific feature engineering. However, the

integration of these highly expressive and often opaque function approximators into the pipeline of RL algorithms also introduces substantial challenges. The highly nonlinear nature of neural network functions often eludes thorough theoretical understanding and the rapid development of novel learning algorithms in practice often outpaces deep learning theory. Deep RL algorithms are moreover frequently noted for their training instability, sensitivity to hyperparameters, and substantial sample complexity, often requiring immense amounts of interaction data, which can be expensive or impractical to obtain in many real-world systems. The exploration-exploitation dilemma, a persistent challenge in most of RL, is arguably intensified in the deep RL setting due to the notoriously difficult and expensive quantification of predictive uncertainty of these highly non-linear function approximators. Developing a more profound understanding of this predictive uncertainty and devising reliable yet efficient mechanisms for its quantification are paramount, in our view, for enhancing the trustworthiness, safety, and ultimately, the widespread applicability of deep RL agents. These challenges thus form the primary motivation for the research presented in this dissertation.

2.4 Aleatoric Uncertainty in Deep Reinforcement Learning

Traditional reinforcement learning algorithms, as outlined in the previous sections, typically focus on estimating and optimizing the *expected* utility, most commonly the expected cumulative discounted reward, as reflected in the use of state or action-value functions. While the expectation serves as a crucial statistic for decision-making in many scenarios, the actual cumulative discounted reward, or *return* $Z = \sum_{t=0}^{\infty} \gamma^t R_t$, experienced by an agent is fundamentally a random variable. This randomness arises from inherent stochasticity in the environment's state transitions, the reward generation process, or the agent's own potentially stochastic policy. The resulting return distribution for a given state-action pair can in principle exhibit arbitrarily complex characteristics, such as multimodality, skewness, or various measures of dispersion (variance), all of which are obscured by merely considering the mean.

Distributional reinforcement learning departs from classical RL algorithms by explicitly aiming to learn the entire probability distribution of the random return $Z(s,a)$, rather than merely its expectation (Bellemare et al., 2017). Access to this complete distributional information captures the *aleatoric uncertainty* associated with returns and allows, for instance, the design of risk-sensitive agents that might prioritize strategies with lower return variance or optimize for specific quantiles of the return distribution (Chow et al., 2018; Morimura

et al., 2010).

The distributional Bellman equation. Analogous to the recursive Bellman equations for expected returns, we can formulate a distributional Bellman equation that characterizes the return distribution for a policy π. This equation states that the distribution of the random returns $Z^\pi(s,a)$ following action a in state s and policy π thereafter, is equal in law to the distribution of the sum of the immediate (random) reward $R(s,a,S')$ and the discounted random return $Z^\pi(S',A')$ in the subsequent state-action pair

$$Z^\pi(s,a) \overset{D}{=} R(s,a,S') + \gamma Z^\pi(S',A'), \tag{2.11}$$

where $S' \sim P(\cdot|s,a)$ is the random next state, $A' \sim \pi(\cdot|S')$ is the random next action, $R(s,a,S')$ is the random reward, and $\overset{D}{=}$ denotes equality in distribution. We used capital letters in this formulation to indicate random variables. As we did with the conventional Bellman equation in Section 2.1, we can turn the distributional Bellman equation into an update rule to obtain the *distributional Bellman operator* (Bellemare et al., 2017). As before, repeated application of the distributional Bellman operator to a (initially) random return distribution $Z_k(s,a)$ converges to the true return distribution, that is $Z_k \to Z^\pi$ as $k \to \infty$.

Deep distributional RL algorithms. However, not all analogies to the classical RL setting succeed trivially. For instance, ensuring convergence to a unique "optimal" return distribution Z^* (the return distribution corresponding to an optimal policy π^*) that satisfies a distributional Bellman optimality equation is more nuanced than in the scalar case. Indeed, the notion of a single "optimal distribution" is not straightforward (Rowland et al., 2019) and in fact many distinct optimal policies may exist with vastly different distributional return profiles. Furthermore, representing arbitrary probability distributions is inherently challenging. Return distributions can exhibit highly complex forms, necessitating specific and expressive function approximators even for a single state-action pair. This very complexity naturally places distributional RL in the vicinity of deep RL approaches, where the representational capacity of deep neural networks is leveraged to model these rich distributions. Various distributional RL algorithms thus parameterize the distribution of $Z(s,a)$ in different ways, for example, by learning explicit histograms of return distributions (Bellemare et al., 2017), by learning a set of quantile functions (Dabney et al., 2018a;b), or by modeling the return distributions moments (Nguyen-Tang et al., 2021). It may be worth noting that while deep distributional RL algorithms model the distribution of returns (aleatoric uncertainty), the parameters of the neural network used to represent this distribution are themselves learned from

finite data and are thus subject to *epistemic uncertainty*. This manifests in a so-to-speak second-order uncertainty about the learned return distribution (Cuzzolin, 2021). Interestingly, empirical evidence suggests that deep distributional RL algorithms often outperform classical "expected RL" algorithms, for which the underlying reason is not yet well-understood (Bellemare et al., 2023).

2.5 Epistemic Uncertainty in Deep Reinforcement Learning

Having addressed aleatoric uncertainty, which arises from inherent environmental and policy stochasticity, we now turn our attention to algorithms and methods designed to quantify and leverage *epistemic uncertainty*. Recall from Section 1.3 that epistemic uncertainty, unlike its aleatoric counterpart, is not intrinsic to the problem's stochastic dynamics but rather stems from limitations in the learned model itself, typically due to a lack of data.

2.5.1 Bayesian Inference

While the advent of deep reinforcement learning has paved the way for several remarkable achievements, it has at the same time amplified the necessity for reliable uncertainty quantification concerning the predictions of the highly complex neural function approximators employed. In most problems of interest, agents operate with a finite, often noisy, subset of all possible experiences obtainable through interaction with the environment. This limitation of data necessitates that learned models generalize across unobserved states and actions — a process of induction that is inherently imbued with epistemic uncertainty. A fundamental question thus arises: given a finite set of observations, how confident can we be that the model parameters inferred can generate correct or even adequate predictions in novel, unobserved scenarios (e.g., for value functions or policies)? The framework of Bayesian inference offers a principled and mathematically coherent approach to addressing this question of model uncertainty (Gelman and Shalizi, 2013; Jaynes, 2003).

Bayes' theorem. Bayesian inference derives its name from its mathematical centerpiece — Bayes' theorem — a fundamental probabilistic theorem attributed to Thomas Bayes that quantifies the probability of hypotheses in relation to observed evidence or information. In the context of parametric models, we now define a probabilistic model $p(f|x;\theta)$ that describes a probability distribution over outcomes f given x and the parametrization θ and thus encodes the aleatoric uncertainty of the underlying process. Given this probabilistic model and a set of observed data points $\mathcal{D}$, Bayes' rule quantifies

the *posterior distribution* over the parameters, $p(\theta|\mathcal{D})$ by

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})}. \tag{2.12}$$

Bayesian inference thus treats the parameters θ of a predictive model as random variables themselves. The posterior probability $p(\theta|\mathcal{D})$ is proportional to the product of two terms: the *prior distribution* $p(\theta)$, which encodes beliefs about the parameters before observing data, and the *likelihood function* $p(\mathcal{D}|\theta)$, which quantifies the probability of observing the entire data set $\mathcal{D}$ given a specific parameterization θ and is derived from out probabilistic model $p(f|x;\theta)$. The denominator, $p(\mathcal{D})$, known as the marginal likelihood or evidence, serves as a normalization constant ensuring the posterior is a valid probability distribution. We can use the prior distribution $p(\theta)$ to incorporate preferences for certain model properties (e.g., simplicity in the sense of Occam's razor or known problem symmetries), while the likelihood assesses how probable the observation is given a specific parameterization (e.g., a model can permit small deviations between predictions and observed outcomes by assuming the existence of Gaussian noise). Taken together, Bayes' theorem assigns a probability (or probability density) to every possible parameterization of a model, thereby defining the posterior distribution over models.

The posterior predictive distribution. A natural way to gauge the uncertainty in predictions $f(x;\theta)$ for any input x (including novel, unseen ones) is then to consider the entire spectrum of plausible models as represented by the posterior. This is achieved by forming the *posterior predictive distribution*, which averages the predictive distributions $p(f|x;\theta)$ of all possible parameter configurations θ, weighted by their posterior probabilities

$$p(f|x,\mathcal{D}) = \int_\theta p(f|x;\theta)p(\theta|\mathcal{D})d\theta. \tag{2.13}$$

The spread of this distribution (e.g., its variance, entropy, or credible intervals) then directly quantifies the uncertainty associated with the prediction $f(x;\theta)$ for x of our model. Notably, however, this distribution blurs the lines between aleatoric and epistemic uncertainty due to the averaging of the predictive distributions $p(f|x;\theta)$ of each parametrization. Indeed, the posterior predictive distribution $p(f|x,\mathcal{D})$ may exhibit a large spread due to high aleatoric uncertainties in each distribution $p(f|x;\theta)$ or due to the existence of highly disjoint but (near) deterministic predictions $p(f|x;\theta)$. Several approaches exist that aim to disentangle aleatoric and epistemic uncertainty in the posterior predictive distribution, for example by measuring the *total uncertainty* in Eq. (2.13) through its entropy and "substracting" from it the aleatoric uncertainty (the entropy in

$p(f|x;\theta)$ average over θ) (Depeweg et al., 2017). The remainder then represents the epistemic uncertainty in the posterior predictive distribution and implies an additive relationship between aleatoric and epistemic uncertainty. More approaches to this end exist and appropriate mechanisms for disentangling specific sources of uncertainty in this framework remain a subject of debate (Hüllermeier and Waegeman, 2020). Still, the Bayesian framework provides a principled and broadly applicable approach towards inferring a multitude of models and how to assess their compatibility with evidence.

Bayesian RL. Within reinforcement learning, the Bayesian paradigm has a rich history and offers a wide range of powerful mathematical tools, particularly in addressing the exploration-exploitation trade-off (Ghavamzadeh et al., 2015). We can apply Bayesian inference to maintain posterior distributions over value functions, policies, or even the parameters of a learned environment model in model-based RL (Depeweg et al., 2017; Strens, 2000). This uncertainty representation can be leveraged in various exploration strategies. For instance, model-free *posterior sampling* (or Thompson sampling) involves drawing a value function parameterization θ from the current posterior $p(\theta|\mathcal{D})$ at the beginning of each episode, and then acting greedily with respect to this sampled model for the duration of the episode (Osband et al., 2013; Russo et al., 2018). This mechanism drives exploration as it is unlikely to sample value functions known to be incompatible with the observed evidence, while different plausible hypotheses are tested and refuted over time. The greedy action selection of the agent makes is likely that those actions associated with high uncertainty are selected eventually, implementing a sort of *stochastic optimism* principle. In general, uncertainty-drive exploration methods often implement the "optimism in the face of uncertainty" principle, for instance by constructing UCBs on value estimates derived from the posterior distribution and selecting actions that maximize this optimistic upper bound (Srinivas et al., 2010). Such strategies directly incentivize probing actions with high epistemic uncertainty, under the assumption that these might lead to information gain and potential high rewards. Conversely, as highlighted in Section 1.3.2, quantified uncertainty can also inform conservative strategies that avoid actions with highly uncertain or potentially adverse outcomes.

Applying the full Bayesian inference framework to modern deep learning models and deep reinforcement learning agents, while conceptually appealing for its principled treatment of uncertainty, remains computationally formidable. The sheer scale of parameter spaces in deep neural networks (often millions to billions) and the typically highly complex form of their corresponding posterior distributions render exact computations of the posterior $p(\theta|\mathcal{D})$ generally infeasible. Consequently, practical implementations instead rely on

approximate inference techniques. Prominent among these are: VI methods, which approximate the true posterior with a simpler, tractable parametric distribution (e.g., a Gaussian) (Blundell et al., 2015; Graves, 2011; Houthooft et al., 2016); and Markov chain Monte Carlo (MCMC) methods, which aim to generate samples of the posterior directly rather explicitly representing its full distribution (Welling and Teh, 2011). While these approximate methods have enabled significant progress in applying Bayesian ideas to deep learning and deep RL, a persistent dilemma exists: computationally feasible approximations may be too coarse to retain the full theoretical benefits of the Bayesian approach, whereas more accurate approximations often face substantial hurdles in terms of scalability and computational feasibility. Furthermore, while the framework of Bayesian inference is by many considered to be the gold standard for uncertainty quantification, there also remain challenges of conceptual nature: the specification of meaningful prior distributions $p(\theta)$ over the immensely high-dimensional parameter spaces of neural networks is complex and often counterintuitive (Izmailov et al., 2021); the appropriate choice of likelihood functions $p(\mathcal{D}|\theta)$ that accurately reflect the complex error characteristics of more advanced training setups like model-free deep RL remain a subject of debate.

2.5.2 Ensemble Methods

Given the significant computational burden and approximation challenges associated with realizing accurate Bayesian inference methods for deep neural networks, ensemble methods have emerged as a widely adopted and highly effective pragmatic alternative for uncertainty quantification (Dietterich, 2000). The core principle of ensembling — the idea that an aggregation of multiple distinct hypotheses can lead to better judgments and serves as a measure of confidence — is a concept with deep historical roots, arguably predating modern statistics. The Greek philosopher Epicurus, for instance, advocated for a "principle of multiple explanations", maintaining that one should retain all explanations of a phenomenon that are consistent with the available evidence. The same foundational idea arguably underlies the Bayesian paradigm and the statistical bootstrap, one of the celebrated techniques of 20th-century statistics, which assesses the uncertainty of an estimator by training it on multiple datasets generated by resampling from the original data with replacement (Efron, 1982). Early works in machine learning apply similar ideas to neural networks, with Hansen and Salamon (1990) demonstrating that ensembles of neural networks can improve predictive accuracy, and Schapire (1990) showing how models can be trained sequentially to correct their predecessors' errors, an approach called boosting.

Deep ensembles. The pivotal role of ensembles as simple yet powerful uncertainty estimators in the context of more contemporary deep learning architectures was driven by seminal work of Lakshminarayanan et al. (2017). Here, they demonstrate that training a collection of identical networks independently, with the main source of diversity being their random parameter initializations, is sufficient to produce robust and well-calibrated uncertainty estimates. The prevailing hypothesis, supported by a growing body of empirical and theoretical work, is that this initial randomization is enough to cause the gradient-based optimization process to settle in diverse optima in the high-dimensional parameter space of large neural networks (Fort et al., 2019). While each of these resulting models explains the training data almost equally well, their divergent parameterizations cause them to extrapolate differently on novel inputs. The disagreement (e.g., the variance) in their predictions for a given input thus serves as a direct and effective measure of epistemic uncertainty.

Deep ensembles in RL. Deep ensembles have found numerous impactful applications in deep reinforcement learning, where their relative simplicity often allows them to function as a practical "drop-in" replacement for more complex Bayesian RL approaches. Frequently, they are used to drive efficient exploration; for instance, by randomly selecting one value function from the ensemble at the start of each episode and having the agent follow a corresponding greedy policy, the Bayesian method of posterior sampling (or Thompson sampling, cf. Sec. 2.5.1) can be effectively emulated (Osband et al., 2016; Russo et al., 2018). In other applications, the spread of ensemble predictions is used to directly construct confidence bounds around value estimates, enabling optimistic exploration strategies or informing conservative decision-making in safety-critical or offline learning contexts (Agarwal et al., 2020; Chua et al., 2018).

Despite their remarkable empirical success and relative ease of implementation, deep ensembles remain constrained by a straightforward limitation: the computational burden associated with training, storing, and performing inference with multiple full-scale models. This cost scales linearly with the number of ensemble members, creating a practical tension with the prevailing trend in deep learning, where conventional wisdom often suggests that computational resources are best invested in scaling up single-model capacities. Furthermore, while several connections to the Bayesian framework can be drawn under certain conditions (D'Angelo and Fortuin, 2021; He et al., 2020; Wilson and Izmailov, 2020), a complete and rigorous theoretical treatment of the uncertainty captured by practical deep ensembles remains an active and challenging area of research. Ultimately, the theoretical understanding of deep ensembles is

deeply intertwined with, and to some extent bottlenecked by, the fundamental challenge of understanding the behavior of deep neural networks themselves.

2.5.3 Other Approaches

Beyond the frameworks of Bayesian inference and ensemble techniques, a wide array of alternative techniques has been proposed to quantify or leverage uncertainty in deep learning and, by extension, in deep reinforcement learning.

Density estimation models. One notable model class includes *density estimation models*, which represent a natural and intuitive approach for identifying inputs that deviate from the training data distribution, a common proxy for epistemic uncertainty. These models aim to explicitly learn the probability density function $p(x)$ of the input data itself, with the underlying assumption that novel or OOD samples should be assigned a low probability density. Autoencoders, for instance, while primarily designed for dimensionality reduction and representation learning, have been repurposed for this task; they compress an input into a compact latent representation and then attempt to reconstruct the original input from this representation (Hinton and Salakhutdinov, 2006). The magnitude of the reconstruction error can be interpreted as an unnormalized, inverse measure of data likelihood, with high errors suggesting an unfamiliar input. More sophisticated approaches, such as *normalizing flows* (Dinh et al., 2017; Rezende and Mohamed, 2015), utilize carefully designed neural network architectures with invertible transformations. These allow normalizing flows to transform an initially simple probability distribution (e.g., a standard Gaussian) into a highly complex distribution capable of modeling the data $p(x)$ directly, providing properly normalized density estimates. Still, density models, particularly when implemented with deep neural networks, are complex systems in their own right. A thorough theoretical understanding of the complex behaviors of contemporary density models is hampered by the use of neural networks, and multiple empirical studies have cast significant doubt on their universal reliability for OOD detection. It has been shown that several deep density estimators (primarily deep generative models) assign higher likelihoods to certain out-of-distribution datasets than to the training data itself, a critical failure mode for uncertainty quantification (Choi et al., 2018; Nalisnick et al., 2019).

Self-predictive methods. Another class of methods utilizes *self-supervised prediction errors* as a proxy for novelty or uncertainty with substantial empirical success for guiding exploration in RL. Random network distillation (RND) exemplifies this approach (Burda et al., 2019b) by training a predictor network

to approximate the output of a fixed, randomly initialized target network on a set of observed states (or state-action pairs). The prediction error between the predictor and target network can then be used as an intrinsic reward, encouraging agents to visit states where the predictor performs poorly, that is, presumably novel states for which it has not yet learned an accurate mapping. Similar principles underlie other exploration strategies based on learning predictive models of environmental dynamics or other self-supervised objectives, where high prediction error signals a lack of understanding or familiarity (Guo et al., 2022; Pathak et al., 2017).

Other approaches. A number of techniques aim to more directly estimate epistemic uncertainty or generalization error. Some approaches involve training a secondary model to predict the primary model's error on unseen data (Lahlou et al., 2021). Others seek to augment the training objective of the primary neural network with regularization terms or gradient-based measures designed to explicitly flag inputs that deviate significantly from the training manifold (Hendrycks and Gimpel, 2017; Lee et al., 2018b; Van Amersfoort et al., 2020).

A common characteristic of the methods outlined in this section is their computational efficiency relative to full Bayesian inference or large ensembles, making them an attractive alternative for practical applications where computational resources are often a central constraint. However, a comprehensive theoretical understanding of how these methods operate, the precise nature of the "uncertainty" they capture, and their reliability across diverse tasks often remains less developed compared to more established theoretically principled frameworks. The development of methods that bridge this gap between computational feasibility and stronger theoretical motivation is a central theme of this dissertation.

2.6 Deep Learning Theory and Learning Dynamics

Owing to their intricate architectures and vast parameter spaces, deep neural networks are frequently treated as "black-box" function approximators — a perspective implicitly adopted in some of the preceding discussions for simplicity. A too simplistic view of these models, however, may undermine the potential for the principled design of neural network-based algorithms and a deeper understanding of their capabilities and failure modes. We thus turn our attention to theoretical frameworks that aim to capture the behavior of neural networks analytically. The objective of such a "deep learning theory" is to establish predictive theoretical models that can elucidate how typical design elements such as network architecture, parameter initialization, and loss function influence the properties of the function ultimately realized by a trained neural network.

Gradient descent. To examine the learning process of neural networks more closely, we consider a standard supervised training paradigm. A neural network $f(x;\theta)$ as introduced in Sec. 2.3, parameterized by a set of weights and biases collectively denoted by θ, is typically initialized with parameters drawn from some probability distribution. Given a training dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, the network is trained by iteratively adjusting its parameters to minimize a loss function $\mathcal{L}(\theta)$ that quantifies the discrepancy between its predictions and the ground-truth labels y_i. For a regression task, the mean squared error loss is a common choice

$$\mathcal{L}(\theta) = \frac{1}{2N} \sum_{i=1}^{N} \big(f(x_i;\theta) - y_i\big)^2 . \tag{2.14}$$

Optimization is most commonly performed via gradient descent (or stochastic variants thereof). Parameters are are here updated according to the rule

$$\theta_{k+1} = \theta_k - \eta \nabla_\theta \mathcal{L}(\theta_k), \tag{2.15}$$

where k indexes the iteration and η is the *learning rate*. Clearly, this iterative process outlines a well-defined dynamical system with specific trajectories of parameters θ_k, albeit in a complex high-dimensional space. The study of these so-called learning dynamics, which draws several interesting analogies to the study of particle systems in physics, forms an important branch of modern deep learning theory.

Analytically characterizing these training dynamics for typical neural networks, however, is extraordinarily challenging. This difficulty stems from several factors: (a) the immense dimensionality of the parameter space θ, ranging from millions to hundreds of billions; (b) the highly non-convex nature of the loss landscape $\mathcal{L}(\theta)$, which can feature numerous local minima, saddle points, and extensive flat regions (Choromanska et al., 2015; Dauphin et al., 2014); and (c) the complex, nonlinear dependencies between parameters across different layers. Consequently, deriving general and insightful theoretical models that precisely describe these dynamics for arbitrary network architectures remains largely an open problem(Bach, 2024; Roberts et al., 2022).

The neural tangent kernel. Significant theoretical progress has nevertheless been achieved by examining neural networks in certain idealized settings, most notably in the limit where the number of neurons in each hidden layer (the network width) tends to infinity. In this infinite-width limit, and under appropriate parameter initialization schemes, remarkable simplifications emerge (Jacot et al., 2018; Lee et al., 2018a; 2020b). One insightful formulation of neural

network functions can be obtained by a Taylor expansion, a widely applicable tool of calculus and general mathematical analysis, through

$$f(x;\theta) \approx f(x;\theta_0) + \nabla_\theta^\top f(x;\theta_0)(\theta - \theta_0), \tag{2.16}$$

where θ_0 is the set of parameters at initialization (the outcome of the random draw at initialization). A central observation is that, as the network width increases, individual parameters of θ need only deviate infinitesimally from their initial values θ_0 throughout training, while effecting substantial changes in the network function $f(x;\theta)$ to fit the training data. This leads to a key insight: wide neural networks, when trained with gradient descent, are well-described by the Taylor expansion (2.16) and thus behave like linear models, albeit in an extremely high-dimensional feature space $\phi(x) = \nabla_\theta f(x;\theta_0)$ that is implicitly defined by the network's architecture and random initialization. This means, conceptually, that the neural network evolves during gradient descent as if it were a linear model of the form $f(x;\theta) \approx \theta^\top \phi(x)$, inducing dynamics described by linear ordinary differential equations. The fixed features $\phi(x) = \nabla_\theta f(x;\theta_0)$ then ascribe the neural network function a similarity metric than can be thought of as the similarity of their gradient structures. We quantify such similarities between inputs x and x' as an inner product in this feature space by $\Theta(x,x') = \nabla_\theta^\top f(x;\theta_0)\nabla_\theta f(x';\theta_0)$, an object known as the *neural tangent kernel* (NTK, Jacot et al., 2018). Remarkably, under certain conditions including the infinite-width limit, this kernel becomes a deterministic object tied to the neural network architecture and initialization scheme, but not the individual outcome of the random initial parameter draws θ_0. The NTK moreover stays constant throughout training, allowing practitioners to derive insights into the role of architecture and parameter initializations on the dynamics and convergence solutions of these idealized neural networks.

Beyond the neural tangent kernel regime. The utility of neural tangent kernel (NTK) theory lies in its capacity to delineate how the functions learned by infinitely wide neural networks evolve, enabling the analysis of generalization properties and, pertinent to this dissertation, the behavior of deep ensembles from random initialization. However, it is important to acknowledge that this framework operates under idealized assumptions. Naturally, the requirement of infinite width is not met by practical networks, although sufficiently overparameterized (when parameters outnumber training samples) finite networks may exhibit behaviors consistent with NTK predictions. Perhaps more critically, the inherent linearization of the model in the NTK regime implies that the effective feature space in which wide neural networks operate do not adapt during training. This contrasts with the widely held view that a core component of deep learning performance lies in its ability to learn hierarchi-

cal representations that evolve in dependence of the training data (Bengio et al., 2013). Consequently, alternative theoretical frameworks are actively being explored, notably mean-field theory, which studies the evolution of the empirical distribution of neuron activations. While leading to more complex, nonlinear partial differential equations, these approaches offer the potential to capture the phenomenon of feature learning (Chizat and Bach, 2018; Mei et al., 2018), a concept eluding current NTK theory.

Within the scope of this dissertation, however, neural tangent kernel theory serves as a primary theoretical lens for several analyses. Its relative analytical tractability, compared to feature-learning regimes, provides valuable and tractable insights into the effects of random network initializations, the behavior of deep ensembles, and their connections to Bayesian inference, albeit in an idealized setting. We employ this framework to derive and understand theoretically motivated uncertainty quantification techniques to address the practical challenge of efficient uncertainty quantification in deep reinforcement learning.

3

Distributional Projection Ensembles

This chapter is based on work previously published as: M. A. Zanger, W. Böhmer, and M. T. J. Spaan. Diverse projection ensembles for distributional reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2024.
Author contributions are as follows: *M.A.Z.:* Conceptualization, Methodology, Formal Analysis, Experimental Implementation, Visualizations, Writing — Original Draft. *W.B.:* Supervision, Project Administration, Writing — Review & Editing. *M.T.J.S.:* Supervision, Project Administration, Funding Acquisition, Writing — Review & Editing.

Deep ensembles have established themselves as a reliable and practically feasible tool for quantifying epistemic uncertainty in deep learning. Underlying their efficacy is the principle, that different models occupy distinct solutions to the same learning problem. Ensembles of such models thus rely on the diversity of their constituent members. This chapter presents our first core contribution by investigating whether this diversity can be improved through explicit architectural design, rather than relying solely on random initialization. We focus specifically on the context of distributional reinforcement learning (RL), a paradigm where the full distribution of returns is learned, not just their expectation. In this setting, learned return distributions are typically approximated by projecting them onto a tractable, parametric family, a step that introduces a strong inductive bias into the learning process. We hypothesize that if all members of an ensemble share the same projection method, this shared bias may limit their functional diversity, thereby constraining the quality of the resulting uncertainty estimates.

To address this, this chapter explores the combination of several different projection and representation methods within a single distributional ensemble. We introduce *diverse projection ensembles*, a straightforward approach where diversity is actively promoted by constructing an ensemble from members with architecturally distinct projection operators. In doing so, this chapter provides a direct answer to our first research question (RQ1):

RQ1: *Can member-specific architectural choices in deep ensembles promote diverse generalization behaviors and thereby improve the quality of uncertainty estimates?*

Following an introduction to the topic, we provide a more detailed technical background on the mechanics of distributional RL and the use of projection methods therein. We then establish several theoretical properties of projection ensembles and derive a novel exploration algorithm that harnesses ensemble disagreement — measured by the average 1-Wasserstein distance between member distributions — as an intrinsic reward. Finally, we present a thorough empirical evaluation of our algorithm on the bsuite benchmark and the VizDoom environment (Kempka et al., 2016; Osband et al., 2020), demonstrating that diverse projection ensembles lead to significant performance improvements over existing methods, with the most pronounced gains in hard-exploration tasks.

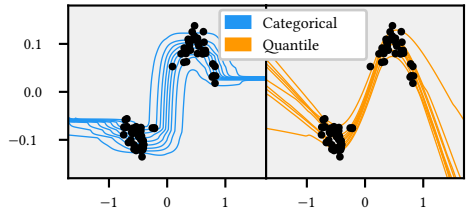
3.1 Introduction

In RL, agents interact with an unknown environment, aiming to acquire policies that yield high cumulative rewards. In pursuit of this objective, agents

must engage in a trade-off between information gain and reward maximization, a dilemma known as the exploration/exploitation trade-off. In the context of model-free RL, many algorithms designed to address this problem efficiently rely on a form of the *optimism in the face of uncertainty* principle (Auer, 2002) where agents act according to upper confidence bounds of value estimates. When using high-capacity function approximators (e.g., neural networks) the derivation of such confidence bounds is non-trivial. One popular approach fits an ensemble of approximations to a finite set of observations (Dietterich, 2000; Lakshminarayanan et al., 2017). Based on the intuition that a set of parametric solutions explains observed data equally well but provides diverse predictions for unobserved data, deep ensembles have shown particularly successful at quantifying uncertainty for novel inputs. An exploring agent may, for example, seek to reduce this kind of uncertainty by visiting unseen state-action regions sufficiently often, until ensemble members converge to almost equal predictions. This notion of reducible uncertainty is also known as *epistemic uncertainty* (Der Kiureghian and Ditlevsen, 2009; Hora, 1996).

A concept somewhat orthogonal to epistemic uncertainty is *aleatoric uncertainty*, that is the uncertainty associated with the inherent irreducible randomness of an event. The latter is the subject of the recently popular *distributional* branch of RL (Bellemare et al., 2017), which aims to approximate the distribution of returns, as opposed to its mean. While distributional RL naturally lends itself to risk-sensitive learning, several results show significant improvements over classical RL even when distributions are used only to recover the mean (Bellemare et al., 2017; Dabney et al., 2018b; Nguyen-Tang et al., 2021; Rowland et al., 2019; Yang et al., 2019). In general, the probability distribution of the random return may be arbitrarily complex and difficult to represent, prompting many recent advancements to rely on novel methods to *project* the unconstrained return distribution onto a set of representable distributions.

In this paper, we study the combination of different *projections* and *representations* in an ensemble of distributional value learners. In this setting, agents who seek to explore previously unseen states and actions can recognize such novel, out-of-distribution inputs by the diversity of member predictions: through learning, these predictions align with labels in frequently visited states and actions, while novel regions lead to disagreement. For this, the individual predictions for unseen inputs, hereafter also referred to as generalization behavior, are required to be sufficiently diverse. We argue that the projection step in distributional RL imposes an inductive bias that leads to such diverse generalization behaviors when joined with neural function approximation. We thus deem distributional projections instrumental to the construction of diverse ensembles, capable of effective separation of epistemic and aleatoric uncertainty. To illustrate the effect of the projection step in the function approximation set-



3

Figure 3.1: Toy 1D-regression: Black dots are training data with inputs x and labels y. Two models have been trained to predict the distribution $p(y|x)$ using a categorical projection (l.h.s.) and a quantile projection (r.h.s.). We plot contour lines for the $\tau = \{0.1, ..., 0.9\}$ quantiles of the predictive distributions over the interval $x \in \{-1.5, 1.5\}$.

ting, Fig. 3.1 shows a toy regression problem where the predictive distributions differ visibly for inputs x not densely covered by training data depending on the choice of projection.

Our main contributions are as follows:

(1) We introduce distributional *projection ensembles* and analyze their properties theoretically. In our setting, each model is iteratively updated toward the projected mixture over ensemble return distributions. We describe such use of distributional ensembles formally through a *projection mixture operator* and establish several of its properties, including contractivity and residual approximation errors.

(2) When using shared distributional temporal difference (TD) targets, ensemble disagreement is biased to represent distributional TD errors rather than errors w.r.t. the true return distribution. To this end, we derive a *propagation* scheme for epistemic uncertainty that relates absolute deviations from the true value function to distributional TD errors. This insight allows us to devise an optimism-based exploration algorithm that leverages a learned bonus for directed exploration.

(3) We implement these algorithmic elements in a deep RL setting and evaluate the resulting agent on the behavior suite (Osband et al., 2020), a benchmark collection of 468 environments, and a set of hard exploration problems in the visual domain *VizDoom* (Kempka et al., 2016). Our experiments show that *projection ensembles* aid reliable uncertainty estimation and exploration, outperforming baselines on most tasks, even when compared to significantly larger ensemble sizes.

3.2 Background

Throughout this work, we consider a finite *Markov Decision Process* (MDP, Bellman, 1957) of the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \gamma, P, \mu)$ as the default problem framework, where $\mathcal{S}$ is the finite state space, $\mathcal{A}$ is the finite action space, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathscr{P}(\mathbb{R})$ is the immediate reward distribution, $\gamma \in [0,1]$ is the discount factor, $P : \mathcal{S} \times \mathcal{A} \to \mathscr{P}(\mathcal{S})$ is the transition kernel, and $\mu : \mathscr{P}(\mathcal{S})$ is the start state distribution. Here, we write $\mathscr{P}(\mathcal{X})$ to indicate the space of probability distributions defined over some space $\mathcal{X}$. Given a state S_t at time t, agents draw an action A_t from a stochastic policy $\pi : \mathcal{S} \to \mathscr{P}(\mathcal{A})$ to be presented the random immediate reward $R_t \sim \mathcal{R}(\cdot|S_t, A_t)$ and the successor state $S_{t+1} \sim P(\cdot|S_t, A_t)$. Under policy π and transition kernel P, the discounted return is a random variable given by the discounted cumulative sum of random rewards according to $Z^\pi(s,a) = \sum_{t=0}^{\infty} \gamma^t R_t$, where $S_0 = s, A_0 = a$. Note that our notation will generally use uppercase letters to indicate random variables. Furthermore, we write $\mathcal{D}(Z^\pi(s,a)) \in \mathscr{P}(\mathbb{R})$ to denote the distribution of the random variable $Z^\pi(s,a)$, that is a state-action-dependent distribution residing in the space of probability distributions $\mathscr{P}(\mathbb{R})$. For explicit referrals, we label this distribution $\eta^\pi(s,a) = \mathcal{D}(Z^\pi(s,a))$. The expected value of $Z^\pi(s,a)$ is known as the state-action value $Q^\pi(s,a) = \mathbb{E}[Z^\pi(s,a)]$ and adheres to a temporal consistency condition described by the Bellman equation (Bellman, 1957)

$$Q^\pi(s,a) = \mathbb{E}_{P,\pi}[R_0 + \gamma Q^\pi(S_1, A_1) | S_0 = s, A_0 = a], \tag{3.1}$$

where $\mathbb{E}_{P,\pi}$ indicates that successor states and actions are drawn from P and π respectively. Moreover, the Bellman operator $T^\pi Q(s,a) := \mathbb{E}_{P,\pi}[R_0 + \gamma Q(S_1, A_1) | S_0 = s, A_0 = a]$ has the unique fixed point $Q^\pi(s,a)$.

3.2.1 Distributional Reinforcement Learning

The *distributional* Bellman operator $\mathcal{T}^\pi$ (Bellemare et al., 2017) is a probabilistic generalization of T^π and considers return distributions rather than their expectation. For notational convenience, we first define P^π to be the transition operator according to

$$P^\pi Z(s,a) :\overset{D}{=} Z(S_1, A_1), \qquad \text{where} \qquad S_1 \sim P(\cdot|S_0 = s, A_0 = a), \quad A_1 \sim \pi(\cdot|S_1), \tag{3.2}$$

and $\overset{D}{=}$ indicates an equality in distributional law (White, 1988). In this setting, the distributional Bellman operator is defined as

$$\mathcal{T}^\pi Z(s,a) :\overset{D}{=} R_0 + \gamma P^\pi Z(s,a). \tag{3.3}$$

The distributional counterpart $\mathcal{T}^\pi : \mathscr{P}(\mathbb{R})^{\mathcal{S}\times\mathcal{A}} \to \mathscr{P}(\mathbb{R})^{\mathcal{S}\times\mathcal{A}}$ to the classical Bellman operator has the unique fixed point $\mathcal{T}^\pi Z^\pi = Z^\pi$, that is the true return distribution Z^π. In the context of designing iterative algorithms, we will also refer to the identity $\mathcal{T}^\pi Z(s,a)$ as a bootstrap of the distribution $Z(s,a)$. For the analysis of many properties of $\mathcal{T}^\pi$, it is helpful to define a distance metric over the space of return distributions $\mathscr{P}(\mathbb{R})^{\mathcal{S}\times\mathcal{A}}$. Here, the supremum p-Wasserstein metric $\bar{w}_p : \mathscr{P}(\mathbb{R})^{\mathcal{S}\times\mathcal{A}} \times \mathscr{P}(\mathbb{R})^{\mathcal{S}\times\mathcal{A}} \to [0,\infty]$ has proven particularly useful. In the univariate case, $\bar{w}_p$ is given by

$$\bar{w}_p(\nu,\nu') = \sup_{s,a\in\mathcal{S}\times\mathcal{A}} \left(\int_0^1 |F^{-1}_{\nu(s,a)}(\tau) - F^{-1}_{\nu'(s,a)}(\tau)|^p d\tau\right)^{\frac{1}{p}}, \tag{3.4}$$

where $p \in [1,\infty)$, ν,ν' are any two state-action return distributions, and $F_{\nu(s,a)} : \mathbb{R} \to [0,1]$ is the CDF of $\nu(s,a)$. For notational brevity, we will use the notation $w_p(\nu(s,a),\nu'(s,a)) = w_p(\nu,\nu')(s,a)$ for the p-Wasserstein distance between distributions ν,ν', evaluated at (s,a). One of the central insights of previous works in distributional RL is that the operator $\mathcal{T}^\pi$ is a γ-contraction in $\bar{w}_p$ (Bellemare et al., 2017), meaning that we have $\bar{w}_p(\mathcal{T}^\pi\nu,\mathcal{T}^\pi\nu') \leq \gamma\bar{w}_p(\nu,\nu')$, a property that allows us (in principle) to construct convergent value iteration schemes in the distributional setting.

3.2.2 Categorical and Quantile Distributional RL

In general, we can not represent arbitrary probability distributions in $\mathscr{P}(\mathbb{R})$ and instead resort to parametric models capable of representing a subset $\mathscr{F}$ of $\mathscr{P}(\mathbb{R})$. Following Bellemare et al. (2023), we refer to $\mathscr{F}$ as a *representation* and define it to be the set of parametric distributions P_θ with $\mathscr{F} = \{P_\theta \in \mathscr{P}(\mathbb{R}) : \theta \in \Theta\}$. Furthermore, we define the *projection operator* $\Pi : \mathscr{P}(\mathbb{R}) \to \mathscr{F}$ to be a mapping from the space of probability distributions $\mathscr{P}(\mathbb{R})$ to the representation $\mathscr{F}$. Recently, two particular choices for representation and projection have proven highly performant in deep RL: the *categorical* and *quantile* model.

The **categorical representation** (Bellemare et al., 2017; Rowland et al., 2018) assumes a weighted mixture of K Dirac deltas δ_{z_k} with support at evenly spaced locations $z_k \in \{z_1,...,z_K\}$. The categorical representation is then given by the weighted sum $\mathscr{F}_C = \{\sum_{k=1}^K \theta_k\delta_{z_k} | \theta_k \geq 0, \sum_{k=1}^K \theta_k = 1\}$. The corresponding categorical projection operator Π_C maps a distribution ν from $\mathscr{P}(\mathbb{R})$ to a distribution in $\mathscr{F}_C$ by assigning probability mass inversely proportional to the distance to the closest z_k in the support $\{z_1,...,z_K\}$ for every point in the support of ν. For example, for a single Dirac distribution δ_x and assuming $z_k \leq x \leq z_{k+1}$ the projection is given by

$$\Pi_C\delta_x = \frac{z_{k+1}-x}{z_{k+1}-z_k}\delta_{z_k} + \frac{x-z_k}{z_{k+1}-z_k}\delta_{z_{k+1}}. \tag{3.5}$$

The corner cases are defined such that $\Pi_C \delta_x = \delta_{z_1}\ \forall x \leq z_1$ and $\Pi_C \delta_x = \delta_{z_K}\ \forall x \geq z_K$. It is straightforward to extend the above projection step to finite mixtures of Dirac distributions through $\Pi_C \sum_k p_k \delta_{z_k} = \sum_k p_k \Pi_C \delta_{z_k}$. The full definition of the projection Π_C is deferred to Appendix 3.8.4.

The **quantile representation** (Dabney et al., 2018b), like the categorical representation, comprises mixture distributions of Dirac deltas $\delta_{\theta_k}(z)$, but in contrast to the categorical representation, parametrizes their locations rather than probabilities. This yields a representation with the weighted sum $\mathscr{F}_Q = \{\sum_{k=1}^{K} \frac{1}{K} \delta_{\theta_k}(z) | \theta_k \in \mathbb{R}\}$. For some distribution $\nu \in \mathscr{P}(\mathbb{R})$, the quantile projection $\Pi_Q \nu$ is a mixture of K Dirac delta distributions with the particular choice of locations that minimizes the 1-Wasserstein distance between $\nu \in \mathscr{P}(\mathbb{R})$ and the projection $\Pi_Q \nu \in \mathscr{F}_Q$. The parametrization θ_k with minimal 1-Wasserstein distance is given by the evaluation of the inverse of the CDF, F_ν^{-1}, at midpoint quantiles $\tau_k = \frac{2k-1}{2K}$, $k \in \{1, ..., K\}$, s.t. $\theta_k = F_\nu^{-1}(\frac{2k-1}{2K})$. Equivalently, θ_k is the minimizer of the quantile regression (QR) loss (Koenker and Hallock, 2001), which is more amenable to gradient-based optimization. The loss is given by

$$\mathcal{L}_Q(\theta_k, \nu) = \mathbb{E}_{Z \sim \nu}[\rho_{\tau_k}(Z - \theta_k)], \tag{3.6}$$

where $\rho_\tau(u) = u(\tau - \mathbb{1}_{\{u \leq 0\}}(u))$ is an error function that assigns asymmetric weight to over- or underestimation errors and $\mathbb{1}$ denotes the indicator function.

3.3 Exploration with Distributional Projection Ensembles

This paper is foremost concerned with leveraging ensembles with diverse generalization behaviors induced by different representations and projection operators. To introduce the concept of distributional projection ensembles and their properties, we describe the main components in a formal setting that foregoes sample-based stochastic approximation and function approximation, before moving to a more practical deep RL setting in Section 3.4. We begin by outlining the *projection mixture operator* and its contraction properties. While this does not inform an exploration algorithm in its own right, it lays a solid algorithmic foundation for the subsequently derived exploration framework. Consider an ensemble $E = \{\eta_i(s,a) | i \in \{1, ..., M\}\}$ of M member distributions $\eta_i(s,a)$, each associated with a representation $\mathscr{F}_i$ and a projection operator Π_i. In this setting, we assume that each member distribution $\eta_i(s,a) \in \mathscr{F}_i$ is an element of the associated representation $\mathscr{F}_i$ and the projection operator $\Pi_i : \mathscr{P}(\mathbb{R}) \to \mathscr{F}_i$ maps any distribution $\nu \in \mathscr{P}(\mathbb{R})$ to $\mathscr{F}_i$ such that $\Pi_i \nu \in \mathscr{F}_i$. The set of representable uniform mixture distributions over E is then given by $\mathscr{F}_E = \{\eta_E(s,a) | \eta_E(s,a) = \frac{1}{M} \sum_i \eta_i(s,a), \eta_i(s,a) \in \mathscr{F}_i, i \in \{1, ..., M\}\}$. We

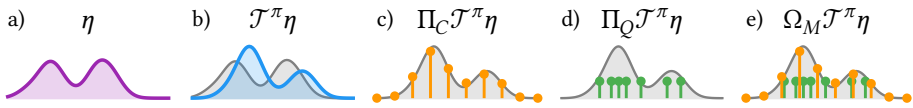


Figure 3.2: Illustration of the projection mixture operator with quantile and categorical projections.

can now define a central object in this paper, the projection mixture operator $\Omega_M : \mathscr{P}(\mathbb{R}) \to \mathscr{F}_E$, as follows:

$$\Omega_M\eta(s,a) = \frac{1}{M}\sum_{i=1}^{M} \Pi_i\eta(s,a). \tag{3.7}$$

Joining Ω_M with the distributional Bellman operator $\mathcal{T}^\pi$ yields the combined operator $\Omega_M\mathcal{T}^\pi$. Fig. 3.2 illustrates the intuition behind the operator $\Omega_M\mathcal{T}^\pi$: the distributional Bellman operator $\mathcal{T}^\pi$ is applied to a return distribution η (Fig. 3.2 a and b), then projects the resulting distribution with the individual projection operators Π_i onto M different representations $\eta_i = \Pi_i\mathcal{T}^\pi\eta \in \mathscr{F}_i$ (Fig. 3.2 c and d), and finally recombines the ensemble members into a mixture model in $\mathscr{F}_E$ (Fig. 3.2 e). In connection with iterative algorithms, we are often interested in the contractivity of the combined operator $\Omega_M\mathcal{T}^\pi$ to establish convergence. Proposition 3.1 delineates conditions under which we can combine individual projections Π_i such that the resulting combined operator $\Omega_M\mathcal{T}^\pi$ is a contraction mapping.

Proposition 3.1. *Let Π_i, $i \in \{1,\dots,M\}$ be projection operators $\Pi_i : \mathscr{P}(\mathbb{R}) \to \mathscr{F}_i$ mapping from the space of probability distributions $\mathscr{P}(\mathbb{R})$ to representations $\mathscr{F}_i$ and denote the projection mixture operator $\Omega_M : \mathscr{P}(\mathbb{R}) \to \mathscr{F}_E$ as defined in Eq. 3.7. Furthermore, assume that for some $p \in [1,\infty)$ each projection Π_i is bounded in the p-Wasserstein metric in the sense that for any two return distributions η,η' we have $w_p\big(\Pi_i\eta,\Pi_i\eta'\big)(s,a) \le c_i w_p\big(\eta,\eta'\big)(s,a)$ for a constant c_i. Then, the combined operator $\Omega_M\mathcal{T}^\pi$ is bounded in the supremum p-Wasserstein distance $\bar{w}_p$ by*

$$\bar{w}_p(\Omega_M\mathcal{T}^\pi\eta, \Omega_M\mathcal{T}^\pi\eta') \le \bar{c}_p\gamma\bar{w}_p(\eta,\eta') \tag{3.8}$$

and is accordingly a contraction so long as $\bar{c}_p\gamma < 1$, where $\bar{c}_p = \big(\sum_{i=1}^{M}\frac{1}{M}c_i^p\big)^{1/p}$.

The full proof is given Section 3.8. The contraction condition in Proposition 3.1 is naturally satisfied for example if all projections Π_i are non-expansions in a joint metric w_p. It is, however, more permissive in the sense that it only requires the joint modulus $\bar{c}_p$ to be limited, allowing for expanding operators in the ensemble for finite p. A contracting combined operator $\Omega_M\mathcal{T}^\pi$ allows us to formulate a simple convergent iteration scheme where in a sequence of

steps k, ensemble members are moved toward the projected mixture distribution according to $\hat{\eta}_{i,k+1} = \Pi_i \mathcal{T}^\pi \hat{\eta}_{E,k}$, yielding the $(k+1)$-th mixture distribution $\hat{\eta}_{E,k+1} = \frac{1}{M}\sum_{i=1}^{M} \hat{\eta}_{i,k+1}$. This procedure can be compactly expressed by

$$\hat{\eta}_{E,k+1} = \Omega_M \mathcal{T}^\pi \hat{\eta}_{E,k}, \quad \text{for} \quad k \in \{0,1,2,3,...\} \tag{3.9}$$

and has a unique fixed point which we denote $\eta_E^\pi = \hat{\eta}_{E,\infty}$.

3.3.1 Optimistic Bounds from Distributions

We proceed to describe how distributional projection ensembles can be leveraged for exploration. Our setting considers exploration strategies based on the UCB algorithm (Auer, 2002). In the context of model-free RL, provably efficient algorithms often rely on the construction of a bound, that overestimates the true state-action value with high probability (Jin et al., 2018; 2020). In other words, we are interested in finding an optimistic value $\hat{Q}^+(s,a)$ such that $\hat{Q}^+(s,a) \geq Q^\pi(s,a)$ with high probability. To this end, Proposition 3.2 relates an estimate $\hat{Q}(s,a)$ to the true value $Q^\pi(s,a)$ through a distributional error term.

Proposition 3.2. Let $\hat{Q}(s,a) = \mathbb{E}[\hat{Z}(s,a)]$ be a state-action value estimate where $\hat{Z}(s,a) \sim \hat{\eta}(s,a)$ is a random variable distributed according to an estimate $\hat{\eta}(s,a)$ of the true state-action return distribution $\eta^\pi(s,a)$. Further, denote $Q^\pi(s,a) = \mathbb{E}[Z^\pi(s,a)]$ the true state-action, where $Z^\pi(s,a) \sim \eta^\pi(s,a)$. We have that $Q^\pi(s,a)$ is bounded from above by

$$\hat{Q}(s,a) + w_1\big(\hat{\eta}, \eta^\pi\big)(s,a) \geq Q^\pi(s,a) \quad \forall (s,a) \in \mathcal{S} \times \mathcal{A},$$

where w_1 is the 1-Wasserstein distance metric.

The proof follows from the definition of the Wasserstein distance and is given in Section 3.8. Proposition 3.2 implies that, for a given distributional estimate $\hat{\eta}(s,a)$, we can construct an optimistic upper bound on $Q^\pi(s,a)$ by adding a bonus corresponding to the 1-Wasserstein distance between an estimate $\hat{\eta}(s,a)$ and the true return distribution $\eta^\pi(s,a)$, which we define as $b^\pi(s,a) = w_1(\hat{\eta}, \eta^\pi)(s,a)$ in the following. By adopting an optimistic action-selection with this guaranteed upper bound on $Q^\pi(s,a)$ according to

$$a = \underset{a \in \mathcal{A}}{\arg\max} [\hat{Q}(s,a) + b^\pi(s,a)], \tag{3.10}$$

we maintain that due to this constructed upper bound the resulting policy inherits efficient exploration properties of known optimism-based exploration methods. Note that in a convergent iteration scheme, we should expect the bonus $b^\pi(s,a)$ to almost vanish in the limit of infinite iterations. We thus refer to $b^\pi(s,a)$ as a measure of the epistemic uncertainty of the estimate $\hat{\eta}(s,a)$.

3.3.2 Propagation of Distributional Errors

By Proposition 3.2, an optimistic policy for efficient exploration can be derived from the distributional error $b^\pi(s,a)$. However, since we do not assume knowledge of the true return distribution $\eta^\pi(s,a)$, this error term requires estimation. The primary purpose of this section is to establish such an estimator by propagating distributional TD errors. This is necessary because the use of TD backups prohibits a consistent uncertainty quantification in values (described extensively in the Bayesian setting for example by Fellows et al. 2021). The issue is particularly easy to see by considering the backup in a single (s,a) tuple: even if every estimate $\hat{\eta}_i(s,a)$ in an ensemble fits the backup $\mathcal{T}^\pi\hat{\eta}_E(s,a)$ accurately, this does not imply $\hat{\eta}_i(s,a) = \eta^\pi(s,a)$ as the TD backup may have been incorrect. Even a well-behaved ensemble (in the sense that its disagreement reliably measures prediction errors) in this case quantifies errors w.r.t. the bootstrapped target $\Omega_M\mathcal{T}^\pi\hat{\eta}_E(s,a)$, rather than the true return distribution $\eta^\pi(s,a)$.

To establish a bonus estimate that allows for optimistic action selection in the spirit of Proposition 3.2, we now derive a propagation scheme for epistemic uncertainty in the distributional setting. More specifically, we find that an upper bound on the bonus $b^\pi(s,a)$ satisfies a temporal consistency condition, similar to the Bellman equations, that relates the total distributional error $w_1(\hat{\eta},\eta_E^\pi)(s,a)$ to a *one-step* error $w_1(\hat{\eta},\Omega_M\mathcal{T}^\pi\hat{\eta})(s,a)$ that is more amenable to estimation.

Theorem 3.3. Let $\hat{\eta}(s,a) \in \mathscr{P}(\mathbb{R})$ be an estimate of the true return distribution $\eta^\pi(s,a) \in \mathscr{P}(\mathbb{R})$, and denote the projection mixture operator $\Omega_M : \mathscr{P}(\mathbb{R}) \to \mathscr{F}_E$ with members Π_i and bounding moduli c_i and $\bar{c}_p$ as defined in Proposition 3.1. Furthermore, assume $\Omega_M\mathcal{T}^\pi$ is a contraction mapping with fixed point η_E^π. We then have for all $(s,a) \in \mathcal{S}\times\mathcal{A}$

$$w_1\big(\hat{\eta},\eta_E^\pi\big)(s,a) \;\le\; w_1\big(\hat{\eta},\Omega_M\mathcal{T}^\pi\hat{\eta}\big)(s,a) + \bar{c}_1\,\gamma\,\mathbb{E}\big[w_1\big(\hat{\eta},\eta_E^\pi\big)(S_1,A_1)\big|S_0=s,A_0=a\big],$$

where $S_1 \sim P(\cdot|S_0=s,A_0=a)$ and $A_1 \sim \pi(\cdot|S_1)$.

The proof is given in Section 3.8 and exploits the triangle inequality property of the Wasserstein distance. It may be worth noting that Theorem 3.3 is a general result that is not restricted to the use of projection ensembles. It is, however, a natural complement to the iteration described in Eq. (3.9) in that it allows us to reconcile the benefits of bootstrapping diverse ensemble mixtures with optimistic action selection for directed exploration. To this end, we devise a separate iteration procedure aimed at finding an approximate upper bound on $w_1(\hat{\eta},\eta_E^\pi)(s,a)$. Denoting the k-th iterate of the bonus estimate $\hat{b}_k(s,a)$, we have by Theorem 3.3 that the iteration

$$\hat{b}_{k+1}(s,a) = w_1\big(\hat{\eta},\Omega_M\mathcal{T}^\pi\hat{\eta}\big)(s,a) + \bar{c}_1\gamma\,\mathbb{E}_{P,\pi}\big[\hat{b}_k(S_1,A_1)\big|S_0=s,A_0=a\big]\,,$$

$\forall (s,a) \in \mathcal{S} \times \mathcal{A}$ and converges to an upper bound[1] on $w_1(\hat{\eta}, \eta_E^\pi)(s,a)$. Notably, this iteration requires only a local error estimate $w_1\big(\hat{\eta}, \Omega_M \mathcal{T}^\pi \hat{\eta}\big)(s,a)$ and is more amenable to estimation through our ensemble.

We conclude this section with the remark that the use of projection ensembles may clash with the intuition that epistemic uncertainty should vanish in convergence. This is because each member inherits irreducible approximation errors from the projections Π_i. In Section 3.8, we provide general bounds for these errors and show that residual errors can be controlled through the number of atoms K in the specific example of an ensemble based on the quantile and categorical projections.

3.4 Deep Distributional Projection Ensembles

Section 3.3 has introduced the concept of projection ensembles in a formal setting. In this section, we aim to transcribe the previously derived algorithmic components into a deep RL algorithm that departs from several of the previous assumptions. Specifically, this includes 1) control with a greedy policy, 2) sample-based stochastic approximation, 3) nonlinear function approximation, and 4) gradient-based optimization. While this sets the following section apart from the theoretical setting considered in Section 3.3, we hypothesize that diverse projection ensembles bring to bear several advantages in this scenario. The underlying idea is that distributional projections and the functional constraints they entail offer an effective tool to impose diverse generalization behaviors on an ensemble, yielding a more reliable tool for out-of-distribution sample detection. In particular, we implement the above-described algorithm with a neural ensemble comprising the models of the two popular deep RL algorithms quantile regression deep Q-network (QR-DQN) (Dabney et al., 2018b) and *categorical Q-networks* (C51, Bellemare et al., 2017).

In particular, we propose projection ensemble deep Q-network (PE-DQN), a deep RL algorithm that combines the quantile and categorical projections (Bellemare et al., 2017; Dabney et al., 2018b) into a diverse ensemble to drive exploration and learning stability. Our parametric model consists of the mixture distribution $\eta_{E,\theta}$ parametrized by θ. We construct $\eta_{E,\theta}$ as an equal mixture between a quantile and a categorical representation, each parametrized through a neural network (NN) with K output logits where we use the notation θ_{ik} to mean the k-th logit of the network parametrized by the parameters θ_i of the i-th model in the ensemble. We consider a sample transition (s,a,r,s',a') where a' is chosen greedily according to $\mathbb{E}_{Z \sim \eta_{E,\theta}(s',a')}[Z]$. Dependencies on (s,a) are hereafter dropped for conciseness by writing $\theta_{ik} = \theta_{ik}(s,a)$ and $\theta'_{ik} = \theta_{ik}(s',a')$.

[1] To see the convergence, note that the sequence is equivalent to an iteration with T^π in an Markov decision process (MDP) with the deterministic immediate reward $w_1\big(\hat{\eta}, \Omega_M \mathcal{T}^\pi \hat{\eta}\big)(s,a)$.

Projection losses. Next, we assume that bootstrapped return distributions are generated by a set of delayed parameters $\tilde{\theta}$, as is common (Mnih et al., 2015). The stochastic (sampled) version of the distributional Bellman operator $\hat{\mathcal{T}}^\pi$, applied to the target ensemble's mixture distribution $\eta_{E,\tilde{\theta}}$ yields

$$\hat{\mathcal{T}}^\pi \eta_{E,\tilde{\theta}} = \frac{1}{2} \sum_{i=1}^{M=2} \sum_{k=1}^{K} p(\tilde{\theta}'_{ik}) \delta_{r+\gamma z(\tilde{\theta}'_{ik})}. \tag{3.11}$$

3

Instead of applying the projection mixture Ω_M analytically, as done in Section 3.3, the parametric estimates $\eta_{E,\theta}$ are moved incrementally towards a projected target distribution through gradient descent on a loss function.

In the *quantile* representation, we augment the classical quantile regression loss (Koenker and Hallock, 2001) with an importance-sampling ratio $Kp(\tilde{\theta}'_{ij})$ to correct for the non-uniformity of atoms from the bootstrapped distribution $\hat{\mathcal{T}}^\pi \eta_{E,\tilde{\theta}}$. For a set of fixed quantiles τ_k, the loss $\mathcal{L}_1$ is given by

$$\mathcal{L}_1\big(\eta_{\theta_1}, \Pi_Q \hat{\mathcal{T}}^\pi \eta_{E,\tilde{\theta}}\big) = \sum_{i=1}^{M=2} \sum_{k,j=1}^{K} Kp(\tilde{\theta}'_{ij}) \Big(\rho_{\tau_k}\big(r + \gamma z(\tilde{\theta}'_{ij}) - \theta_{1k}\big)\Big). \tag{3.12}$$

The *categorical* model minimizes the KL divergence between the projected bootstrap distribution $\Pi_C \hat{\mathcal{T}}^\pi \eta_{E,\tilde{\theta}}$ and an estimate η_{θ_2}. The corresponding loss is given by

$$\mathcal{L}_2\big(\eta_{\theta_2}, \Pi_C \hat{\mathcal{T}}^\pi \eta_{E,\tilde{\theta}}\big) = D_{KL}\big(\Pi_C \hat{\mathcal{T}}^\pi \eta_{E,\tilde{\theta}} \| \eta_{\theta_2}\big). \tag{3.13}$$

As $\hat{\mathcal{T}}^\pi \eta_{E,\tilde{\theta}}$ is a mixture of Dirac distributions, the definition of the projection Π_C according to Eq. 3.5 can be applied straightforwardly to obtain the projected bootstrap distribution $\Pi_C \hat{\mathcal{T}}^\pi \eta_{E,\tilde{\theta}}$.

Uncertainty propagation. We aim to estimate a state-action dependent bonus $b_\vartheta(s,a)$ in the spirit of Theorem 3.3 and the subsequently derived iteration with a set of parameters ϑ. For this, we estimate the local error estimate $w_1(\eta_{E,\theta}, \Omega_M \hat{\mathcal{T}}^\pi \eta_{E,\theta})(s,a)$ as the average ensemble disagreement $w_{\text{avg}}(s,a) = 1/(M(M-1)) \sum_{i,j=1}^{M} w_1(\eta_{\theta_i}, \eta_{\theta_j})(s,a)$. The bonus $b_\vartheta(s,a)$ can then be learned in the same fashion as a regular value function with the local uncertainty estimate $w_{\text{avg}}(s,a)$ as an intrinsic reward. This yields the exploratory action-selection rule

$$a_\epsilon = \arg\max_{a \in \mathcal{A}} \big(\mathbb{E}_{Z \sim \eta_{E,\theta}(s,a)}[Z] + \beta b_\vartheta(s,a) \big), \tag{3.14}$$

where β is a hyperparameter to control the policy's drive towards exploratory actions. Further details on our implementation and an illustration of the difference between local error estimates and bonus estimates in practice are given in Appendix A.1.2 and Appendix A.1.3.

3.5 Empirical Analysis

Our experiments are designed to provide us with a better understanding of how PE-DQN operates, in comparison to related algorithms as well as in relation to its algorithmic elements. To this end, we aimed to keep codebases and hyperparameters between all implementations equal up to algorithm-specific parameters, which we optimized with a grid search on a selected subsets of problems. Further details regarding the experimental design and implementations are provided in Appendix A.1.

We outline our choice of baselines briefly: Bootstrapped deep Q-network (BDQN) with prior functions (BDQNP, Osband et al., 2019) approximates posterior sampling of a parametric value function by combining statistical bootstrapping with additive prior functions in an ensemble of deep Q-network (DQN) agents. *Information-directed sampling* (IDS-categorical Q-network (C51), Nikolov et al., 2019) builds on the BDQN architecture but acts according to an information-gain ratio for which Nikolov et al. (2019) estimate aleatoric uncertainty (noise) with the categorical C51 model. In contrast, decaying left-truncated variance (DLTV) QR-DQN (Mavrin et al., 2019) uses a distributional value approximation based on the quantile representation and follows a decaying exploration bonus of the left-truncated variance.

3.5.1 Distributional Projections and Generalization Behavior

First, we examine empirically the influence of the projection step in deep distributional RL on generalization behaviors. For this, we probe the influence of the quantile and categorical projections on generalization through an experiment that evaluates exploration in a reward-free setting. Specifically, we equip agents with an action-selection rule that maximizes a particular statistic $\$[Z]$ of the predictive distribution $\hat{\eta}(s,a)$ according to

$$a = \arg\max_{a \in \mathcal{A}} \left(\$[Z]\right), Z \sim \hat{\eta}(s,a). \tag{3.15}$$

The underlying idea is that this selection rule leads to exploration of novel state-action regions only if high values of the statistic are correlated with high epistemic uncertainty. For example, if we choose a quantile representation with $\$[Z]$ to be the variance of the distribution, we recover a basic form of the exploration algorithm DLTV-QR-DQN (Mavrin et al., 2019). Fig. 3.3 shows

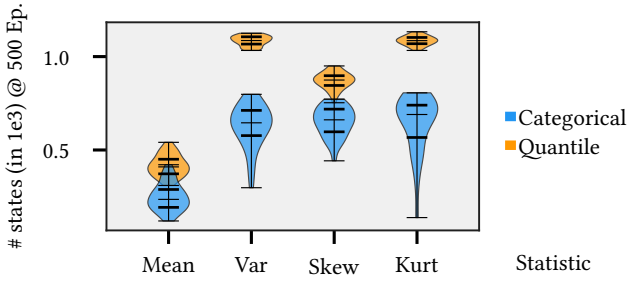
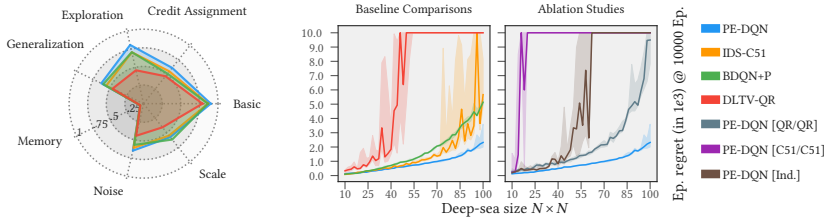


Figure 3.3: Deep-sea exploration with different statistics. Higher means more exploration. Bars represent medians and interquartile ranges of 30 seeds.



(a) (b)

Figure 3.4: (a) Summary of bsuite experiments. Wide is better. (b) Median episodic regret for deep sea sizes up to 100. Low is better. Shaded regions are the interquartile range of 10 seeds.

the results of this study for the first four statistical moments on the deep exploration benchmark *deep sea* with size 50. Except for the mean (the greedy policy), the choice of projection influences significantly whether the statistic-maximizing policy leads to more exploration, implying that the generalization behaviour of the 2nd to 4th moment of the predictive distributions is shaped distinctly by the employed projection.

3.5.2 The Behaviour Suite

In order to assess the learning process of agents in various aspects on a wide range of tasks, we evaluate PE-DQN on the behavior suite (bsuite) (Osband et al., 2020), a battery of benchmark problems constructed to assess key properties of RL algorithms. The suite consists of 23 tasks with up to 22 variations in size or seed, totaling 468 environments.

Comparative evaluation. Fig. 3.4 (a) shows the results of the entire bsuite experiment, summarized in seven *core capabilities*. These capability scores are

computed as proposed by Osband et al. (2020) and follow a handcrafted scoring function per environment. For example, exploration capability is scored by the average regret in the sparse-reward environments *deep sea*, *stochastic deep sea*, and *cartpole swingup*. The full set of results is provided in Appendix A.1. Perhaps unsurprisingly, PE-DQN has its strongest performance in the exploration category but we find that it improves upon baselines in several more categories. Note here that PE-DQN uses substantially fewer models than the baselines, with a total of 4 distributional models compared to the 20 DQN models used in the ensembles of both bootstrapped deep Q-network + priors (BDQNP) and information-directed sampling (IDS), where the latter requires an additional C51 model.

3.5.3 The Deep Sea and Ablations

Deep sea is a hard exploration problem in the behavior suite and has recently gained popularity as an exploration benchmark (Flennerhag et al., 2020; Janz et al., 2019; Osband et al., 2019). It is a sparse reward environment where agents can reach the only rewarding state at the bottom right of an $N \times N$ grid through a unique sequence of actions in an exponentially growing trajectory space. We ran an additional experiment on deep sea with grid sizes up to 100; double the maximal size in the behavior suite. Fig. 3.4 (b) shows a summary of this experiment where we evaluated episodic regret, that is the number of non-rewarding episodes with a maximum budget of 10000 episodes. PE-DQN scales more gracefully to larger sizes of the problem than the baselines, reducing the median regret by roughly half. The r.h.s. plot in Fig. 3.4 (b) shows the results of ablation studies designed to provide a more nuanced view of PE-DQN's performance; the baselines labeled PE-DQNQR/QR and PE-DQNC51/C51 use the same bonus estimation step as PE-DQN except that ensemble members consist of equivalent models with *the same projections and representations*. Conversely, PE-DQN [Ind.] uses PE-DQN's diverse projection ensemble and employs an optimistic action-selection directly with the ensemble disagreement $w_{\text{avg}}(s, a)$ but trains models independently and accordingly does not make use of an uncertainty propagation scheme in the spirit of Theorem 3.3. Both components lead to a pronounced difference in exploration capability and rendered indispensable to PE-DQN's overall performance.

3.5.4 The VizDoom Environment

We investigate PE-DQN's behavior in a high-dimensional visual domain. The *VizDoom* environment *MyWayHome* (Kempka et al., 2016) tasks agents with finding a (rewarding) object by navigating in a maze-like map with ego-perspective pixel observations as seen in Fig. 3.5 (a). Following work by

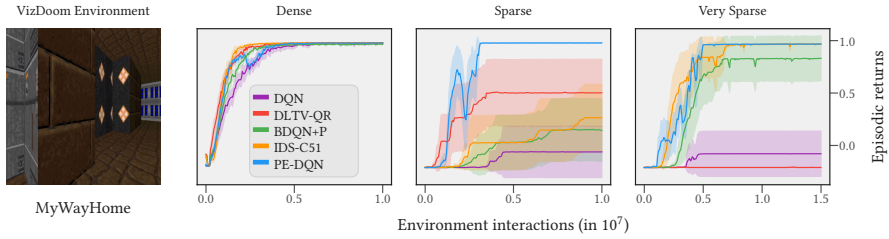


Figure 3.5: (a) Visual observation in the VizDoom environment (Kempka et al., 2016). (b) Mean learning curves in different variations of the *MyWayHome* VizDoom environment. Shaded regions are 90% Student's t confidence intervals from 10 seeds.

Pathak et al. (2017), we run three variations of this experiment where the reward sparsity is increased by spawning the player further away from the goal object. Learning curves for all algorithms are shown in Fig. 3.5 (b). Among the tested algorithms, only PE-DQN finds the object across all 10 seeds in all environments, indicating particularly reliable novelty detection. Interestingly, the sparse domain proved harder to baseline algorithms which we attribute to the "forkedness" of the associated map (see Appendix A.1). This result moreover shows that diverse projection ensembles scale gracefully to high-dimensional domains while using significantly fewer models than the ensemble-based baselines.

3.6 Related Work

Our work builds on a swiftly growing body of literature in distributional RL (Bellemare et al., 2017; Morimura et al., 2010). In particular, several of our theoretical results rely on works by Rowland et al. (2018) and Dabney et al. (2018b), who first provided contraction properties with categorical and quantile projections in distributional RL respectively. Numerous recently proposed algorithms (Dabney et al., 2018a; Nguyen-Tang et al., 2021; Rowland et al., 2019; Yang et al., 2019) are based on novel representations and projections, typically with an increased capacity to represent complex distributions. In contrast to our approach, however, these methods have no built-in functionality to estimate epistemic uncertainty. To the best of our knowledge, our work is the first to study the combination of different projection operators and representations in the context of distributional RL.

Several works, however, have applied ensemble techniques to distributional approaches. For example, Clements et al. (2019), Eriksson et al. (2022),

and Hoel et al. (2023) use ensembles of distributional models to derive aleatoric and epistemic risk measures. Lindenberg et al. (2020) use an ensemble of agents in independent environment instances based on categorical models to drive performance and stability. Jiang et al. (2024) leverage quantile-based ensembles to drive exploration in contextual MDPs, while Nikolov et al. (2019) combine a deterministic Q-ensemble with a distributional categorical model for information-directed sampling. In a broader sense, the use of deep ensembles for value estimation and exploration is widespread (Chen et al., 2017; Fellows et al., 2021; Flennerhag et al., 2020; Osband et al., 2016; 2019). A notable distinction between such algorithms is whether ensemble members are trained independently or whether joint TD backups are used. Our work falls into the latter category which typically requires a propagation mechanism to estimate value uncertainty rather than uncertainty in TD targets (Fellows et al., 2021; Janz et al., 2019; Moerland et al., 2017). Our proposed propagation scheme establishes a temporal consistency between *distributional* TD errors and errors w.r.t. the true return distribution. In contrast to the related uncertainty Bellman equation (O'Donoghue et al., 2018), our approach applies to the distributional setting and devises uncertainty propagation from the perspective of error decomposition, rather than posterior variance.

3.7 Conclusion

In this work, we have introduced projection ensembles for distributional RL, a method combining models based on different parametric representations and projections of return distributions. We provided a theoretical analysis that establishes convergence conditions and bounds on residual approximation errors that apply to general compositions of such projection ensembles. Furthermore, we introduced a general propagation method that reconciles one-step distributional TD errors with optimism-based exploration. PE-DQN, a deep RL algorithm, empirically demonstrates the efficacy of diverse projection ensembles on exploration tasks and showed performance improvements on a wide range of tasks. We believe our work opens up a number of promising avenues for future research. For example, we have only considered the use of uniform mixtures over distributional ensembles in this work. A continuation of this approach may aim to use a diverse collection of models less conservatively, aiming to exploit the strengths of particular models in specific regions of the state-action space.

3.8 Proofs

This section provides proofs for the theoretical claims and establishes further results on the residual approximation error incurred by our method.

3.8.1 Contractivity of Projection Mixtures

Before stating supporting lemmas and proofs of the results in Section 3.3, we recall several basic properties of the p-Wasserstein distances which we will find useful in the subsequent proofs. Derivations of these properties can for example be found in an overview by Mariucci and Reiß (2018).

P.1 The p-Wasserstein distances satisfy the *triangle inequality*, that is

$$w_p(X,Y) \leq w_p(X,Z) + w_p(Z,Y).$$

P.2 For random variables X and Y and an auxiliary variable Z independent of X and Y, the p-Wasserstein metric satisfies the inequality

$$w_p(X+Z,Y+Z) \leq w_p(X,Y).$$

P.3 For a real-valued scalar $a \in \mathbb{R}$, we have

$$w_p(aX,aY) = |a| w_p(X,Y).$$

Lemma 3.4. *Let* $\nu = \sum_{i=1}^{M} \frac{1}{M}\nu_i$, $\nu' = \sum_{i=1}^{M} \frac{1}{M}\nu_i'$ *be two mixture distributions* $\nu, \nu' \in \mathscr{P}(\mathbb{R})$. *Furthermore denote* $w_p(\nu,\nu')$ *the* p*-Wasserstein metric between* ν *and* ν'. *Then* w_p^p *satisfies*

$$w_p^p(\nu,\nu') \leq \frac{1}{M} \sum_{i=1}^{M} w_p^p(\nu_i,\nu_i').$$

Proof. The Wasserstein distance in its general form is expressed in terms of couplings between the probability measures ν and ν' according to

$$w_p(\nu,\nu') = \inf_{\mu \in \Gamma(\nu,\nu')} \mathbb{E}_{(x,y)\sim\mu}[|x-y|^p]^{1/p}, \tag{3.16}$$

where $\Gamma(\nu,\nu')$ is the set of all couplings between ν and ν', i.e. joint distributions on $\mathscr{P}(\mathbb{R}^2)$ with marginals ν and ν'. Now suppose for each i we have a coupling $\mu_i(x,y) \in \Gamma(\nu_i,\nu_i')$ such that

$$\mathbb{E}_{(x,y)\sim\mu_i}[|x-y|^p] = \inf_{\mu \in \Gamma(\nu_i,\nu_i')} \mathbb{E}_{(x,y)\sim\mu}[|x-y|^p] = w_p^p(\nu_i,\nu_i'). \tag{3.17}$$

Since by definition $\mu_i(x,y)$ is a coupling of ν_i and ν_i', the mixture of couplings $\bar{\mu}(x,y) = \sum_{i=1}^{M} \frac{1}{M}\mu_i(x,y)$ is then a valid coupling of ν and ν', since $\int_y \bar{\mu}(x,y)dy = \nu(x)$ and as $\int_{x'} \bar{\mu}(x',y)dx' = \nu'(x)$. We can thus write

$$w_p^p(\nu,\nu') = \inf_{\mu\in\Gamma(\nu,\nu')} \mathbb{E}_{(x,y)\sim\mu}[|x-y|^p] \tag{3.18}$$

$$\leq \mathbb{E}_{(x,y)\sim\bar{\mu}}[|x-y|^p] \tag{3.19}$$

$$= \sum_{i=1}^{M} \frac{1}{M}\mathbb{E}_{(x,y)\sim\mu_i}[|x-y|^p] \tag{3.20}$$

$$= \sum_{i=1}^{M} \frac{1}{M} w_p^p(\nu_i,\nu_i'). \tag{3.21}$$

□

We now restate Proposition 3.1 for convenience.

Proposition 3.1. *Let $\Pi_i, i \in \{1,...,M\}$ be projection operators $\Pi_i : \mathscr{P}(\mathbb{R}) \to \mathscr{F}_i$ mapping from the space of probability distributions $\mathscr{P}(\mathbb{R})$ to representations $\mathscr{F}_i$ and denote the projection mixture operator $\Omega_M : \mathscr{P}(\mathbb{R}) \to \mathscr{F}_E$ as defined in Eq. 3.7. Furthermore, assume that for some $p \in [1,\infty)$ each projection Π_i is bounded in the p-Wasserstein metric in the sense that for any two return distributions η,η' we have $w_p(\Pi_i\eta,\Pi_i\eta')(s,a) \leq c_i w_p(\eta,\eta')(s,a)$ for a constant c_i. Then, the combined operator $\Omega_M\mathcal{T}^\pi$ is bounded in the supremum p-Wasserstein distance $\bar{w}_p$ by*

$$\bar{w}_p(\Omega_M\mathcal{T}^\pi\eta, \Omega_M\mathcal{T}^\pi\eta') \leq \bar{c}_p\gamma\bar{w}_p(\eta,\eta') \tag{3.8}$$

and is accordingly a contraction so long as $\bar{c}_p\gamma < 1$, where $\bar{c}_p = (\sum_{i=1}^{M} \frac{1}{M} c_i^p)^{1/p}$.

Proof. Due to the assumption of the proposition, we have $w_p(\Pi_i\nu,\Pi_i\nu') \leq$

$c_i w_p(\nu, \nu')$. With Lemma 3.4 and the γ-contractivity of $\mathcal{T}^\pi$, it follows that

$$\bar{w}_p^p(\Omega_M \mathcal{T}^\pi \eta, \Omega_M \mathcal{T}^\pi \eta') = \bar{w}_p^p\Big(\sum_{i=1}^M \tfrac{1}{M} \Pi_i \mathcal{T}^\pi \eta, \sum_{i=1}^M \tfrac{1}{M} \Pi_i \mathcal{T}^\pi \eta'\Big) \tag{3.22}$$

$$\leq \frac{1}{M} \sum_{i=1}^M \bar{w}_p^p(\Pi_i \mathcal{T}^\pi \eta, \Pi_i \mathcal{T}^\pi \eta') \tag{3.23}$$

$$\leq \frac{1}{M} \sum_{i=1}^M c_i^p \bar{w}_p^p(\mathcal{T}^\pi \eta, \mathcal{T}^\pi \eta') \tag{3.24}$$

$$\leq \frac{1}{M} \sum_{i=1}^M c_i^p \gamma^p \bar{w}_p^p(\eta, \eta') \tag{3.25}$$

$$= \gamma^p \bar{w}_p^p(\eta, \eta') \frac{1}{M} \sum_{i=1}^M c_i^p . \tag{3.26}$$

The state then finally follows by taking the p-th root, yielding the joint modulus $\bar{c}_p = (\sum_{i=1}^M \frac{1}{M} c_i^p)^{1/p}$. □

3.8.2 Optimistic Bounds from Distributions

We restate Proposition 3.2 for convenience.

Proposition 3.2. Let $\hat{Q}(s,a) = \mathbb{E}[\hat{Z}(s,a)]$ be a state-action value estimate where $\hat{Z}(s,a) \sim \hat{\eta}(s,a)$ is a random variable distributed according to an estimate $\hat{\eta}(s,a)$ of the true state-action return distribution $\eta^\pi(s,a)$. Further, denote $Q^\pi(s,a) = \mathbb{E}[Z^\pi(s,a)]$ the true state-action, where $Z^\pi(s,a) \sim \eta^\pi(s,a)$. We have that $Q^\pi(s,a)$ is bounded from above by

$$\hat{Q}(s,a) + w_1\big(\hat{\eta}, \eta^\pi\big)(s,a) \geq Q^\pi(s,a) \quad \forall (s,a) \in \mathcal{S} \times \mathcal{A},$$

where w_1 is the 1-Wasserstein distance metric.

Proof. We begin by stating a property that relates the expected value $\mathbb{E}[X]$ to the CDF of X under the condition that the expectation $\mathbb{E}[X]$ is well-defined and finite. The property is an extension to the property of the expectation of nonnegative variables which itself is a consequence of Fubini's Theorem (see for example Ibe 2014 for this). Let $X \sim \nu$ and write F_ν for the CDF of ν, then:

$$\mathbb{E}[X] = \int_0^\infty \big(1 - F_\nu(x)\big) dx - \int_{-\infty}^0 F_\nu(x) dx . \tag{3.27}$$

Now, suppose an auxiliary variable X' is distributed according to the law ν'. It then follows that

$$\begin{aligned}
\left|\mathbb{E}[X]-\mathbb{E}[X']\right| &= \left|\int_0^\infty \big(F_{\nu'}(x)-F_\nu(x)\big)dx - \int_{-\infty}^0 \big(F_\nu - F_{\nu'}(x)\big)dx\right| && (3.28)\\
&= \left|\int_{-\infty}^\infty F_{\nu'}(x)-F_\nu(x)dx\right| && (3.29)\\
&\le \int_{-\infty}^\infty \left|F_{\nu'}(x)-F_\nu(x)\right|dx && (3.30)\\
&= w_1(\nu,\nu'), && (3.31)
\end{aligned}$$

where the last step was obtained by a change of variables in the definition of the 1-Wasserstein distance:

$$\begin{aligned}
w_1(\nu,\nu') &= \int_0^1 |F_\nu^{-1}(\tau)-F_{\nu'}^{-1}(\tau)|d\tau && (3.32)\\
&= \int_{\mathbb{R}} |F_\nu(x)-F_{\nu'}(x)|dx. && (3.33)
\end{aligned}$$

The result of Proposition 3.2 is obtained by rearranging. □

3.8.3 Propagation of Distributional Errors

Before stating the proof of Theorem 3.3, we formalize the notion of a push-forward distribution which will be useful in a more explicit description of the distributional Bellman operator $\mathcal{T}^\pi$. Our notation here follows the detailed exposition by Bellemare et al. (2023).

Definition 3.5. For a function $f : \mathbb{R}\to\mathbb{R}$ and a random variable Z with distribution $\nu = \mathcal{D}(Z)$, $\nu\in\mathscr{P}(\mathbb{R})$, the *pushforward distribution* $f_\#\nu\in\mathscr{P}(\mathbb{R})$ of ν through f is defined as

$$f_\#\nu(B) = \nu(f^{-1}(B)), \quad \forall B\in\mathcal{B}(\mathbb{R}),$$

where $\mathcal{B}$ are the Borel subsets of $\mathbb{R}$.

Equivalently to Definition 3.5, we may write $f_\#\nu = \mathcal{D}(f(Z))$. By defining a bootstrap transformation $b_{r,\gamma} : \mathbb{R}\to\mathbb{R}$ with $b_{r,\gamma} = r+\gamma x$, we can state a more explicit definition of the distributional Bellman operator $\mathcal{T}^\pi$ according to Definition 3.6.

Definition 3.6. [Distributional Bellman Operator (Bellemare et al., 2017)] The distributional Bellman operator $\mathcal{T}^\pi : \mathscr{P}(\mathbb{R})^{\mathcal{S}\times\mathcal{A}}\to\mathscr{P}(\mathbb{R})^{\mathcal{S}\times\mathcal{A}}$ is given by

$$\big(\mathcal{T}^\pi\eta\big)(s,a) = \mathbb{E}\big[(b_{R_0,\gamma})_\#\eta(S_1,A_1)\big|S_0=s,A_0=a\big],$$

where $S_1\sim P(\cdot|S_0=s,A_0=a)$, $\quad A_1\sim\pi(\cdot|S_1)$.

Lemma 3.7. *Let $(b_{r,\gamma})_{\#}\nu \in \mathscr{P}(\mathbb{R})$ be the pushforward distribution of $\nu \in \mathscr{P}(\mathbb{R})$ through $b_{r,\gamma} : \mathbb{R} \to \mathbb{R}$. Then we have for two distributions ν, ν' and the 1-Wasserstein distance w_1 that*

$$w_1\big((b_{r,\gamma})_{\#}\nu, (b_{r,\gamma})_{\#}\nu'\big) = \gamma w_1(\nu, \nu').$$

Proof. The proof follows from the definition of the 1-Wasserstein distance. Let $Z \sim \nu$ and $Z' \sim \nu'$ be two independent random variables, then

$$\begin{aligned} w_1\big((b_{r,\gamma})_{\#}\nu, (b_{r,\gamma})_{\#}\nu'\big) &= w_1\big(\mathcal{D}(r+\gamma Z), \mathcal{D}(r+\gamma Z')\big) && (3.34)\\ &= \int_0^1 \big|F^{-1}_{(b_{0,\gamma})_{\#}\nu}(\tau) - F^{-1}_{(b_{0,\gamma})_{\#}\nu'}(\tau)\big| d\tau && (3.35)\\ &= |\gamma| w_1(\nu, \nu'). && (3.36) \end{aligned}$$

□

We now restate Theorem 3.3 for convenience.

Theorem 3.3. *Let $\hat{\eta}(s,a) \in \mathscr{P}(\mathbb{R})$ be an estimate of the true return distribution $\eta^\pi(s,a) \in \mathscr{P}(\mathbb{R})$, and denote the projection mixture operator $\Omega_M : \mathscr{P}(\mathbb{R}) \to \mathscr{F}_E$ with members Π_i and bounding moduli c_i and $\bar{c}_p$ as defined in Proposition 3.1. Furthermore, assume $\Omega_M \mathcal{T}^\pi$ is a contraction mapping with fixed point η^π_E. We then have for all $(s,a) \in \mathcal{S} \times \mathcal{A}$*

$$w_1\big(\hat{\eta}, \eta^\pi_E\big)(s,a) \;\leq\; w_1\big(\hat{\eta}, \Omega_M \mathcal{T}^\pi \hat{\eta}\big)(s,a) + \bar{c}_1 \gamma\, \mathbb{E}\big[w_1\big(\hat{\eta}, \eta^\pi_E\big)(S_1, A_1) \big| S_0 = s, A_0 = a\big],$$

where $S_1 \sim P(\cdot|S_0 = s, A_0 = a)$ and $A_1 \sim \pi(\cdot|S_1)$.

Proof. Since $\eta^\pi_E(s,a)$ is the fixed point of the combined operator $\Omega_M \mathcal{T}^\pi$, we have that $\Omega_M \mathcal{T}^\pi \eta^\pi_E(s,a) = \eta^\pi_E(s,a)$. From the triangle inequality it follows that

$$w_1\big(\hat{\eta}, \eta^\pi_E\big)(s,a) \leq w_1\big(\hat{\eta}, \Omega_M \mathcal{T}^\pi \hat{\eta}\big)(s,a) + w_1\big(\Omega_M \mathcal{T}^\pi \hat{\eta}, \Omega_M \mathcal{T}^\pi \eta^\pi_E\big)(s,a). \tag{3.37}$$

Furthermore, for the second term on the r.h.s. in Eq. 3.37 the following holds:

$$\begin{aligned} w_1\big(\Omega_M \mathcal{T}^\pi \hat{\eta}, \Omega_M \mathcal{T}^\pi \eta^\pi_E\big)(s,a) &= w_1\big(\tfrac{1}{M}\sum_{i=1}^M \Pi_i \mathcal{T}^\pi \hat{\eta}, \tfrac{1}{M}\sum_{i=1}^M \Pi_i \mathcal{T}^\pi \eta^\pi_E\big)(s,a) && (3.38)\\ &\leq \tfrac{1}{M}\sum_{i=1}^M c_i w_1\big(\mathcal{T}^\pi \hat{\eta}, \mathcal{T}^\pi \eta^\pi_E\big)(s,a) && (3.39)\\ &= \bar{c}_1 w_1\big(\mathcal{T}^\pi \hat{\eta}, \mathcal{T}^\pi \eta^\pi_E\big)(s,a). && (3.40) \end{aligned}$$

Under slight abuse of the assumptions in Section 3.2, we here consider an immediate reward distribution with finite support on $\mathcal{R}$ to simplify the following derivation. In this case, we can write out the expectation in Definition 3.6 as

$$\big(\mathcal{T}^\pi \hat{\eta}\big)(s,a) = \sum_{r\in\mathcal{R}}\sum_{s'\in\mathcal{S}}\sum_{a'\in\mathcal{A}} Pr(R_0 = r, A_1 = a', S_1 = s' | S_0 = s, A_0 = a)\big((b_{r,\gamma})_{\#}\hat{\eta}(s',a')\big), \tag{3.41}$$

where $Pr(\cdot)$ is the joint probability distribution given by the transition kernel $P(\cdot|s,a)$, the immediate reward distribution $\mathcal{R}(\cdot|s,a)$, and the policy $\pi(\cdot|S')$. Thus, by Lemma 3.4 and Lemma 3.7 it follows that

$$\begin{aligned}
&\bar{c}_1 w_1\big(\mathcal{T}^\pi\hat{\eta}, \mathcal{T}^\pi\eta_E^\pi\big)(s,a) \\
&\quad\le \bar{c}_1 \mathbb{E}\big[w_1\big((b_{R_0,\gamma})_{\#}\hat{\eta}(S_1,A_1), (b_{R_0,\gamma})_{\#}\eta_E^\pi(S_1,A_1)\big)\big|S_0 = s, A_0 = a\big] &(3.42)\\
&\quad= \bar{c}_1\gamma \mathbb{E}\big[w_1\big(\hat{\eta},\eta_E^\pi\big)(S_1,A_1)\big|S_0 = s, A_0 = a\big], &(3.43)
\end{aligned}$$

where $S_1 \sim P(\cdot|S_0 = s, A_0 = a)$ and $A_1 \sim \pi(\cdot|S')$. The proof is completed by rearranging. □

3.8.4 Additional Proofs

We provide additional theoretical results below.

Residual Epistemic Uncertainty

Due to a limitation to finite-dimensional representations and the use of varying projections, our algorithm incurs residual approximation errors which may not vanish even in convergence. In the context of epistemic uncertainty quantification, this is unfortunate as it can frustrate exploration or lead to overconfident predictions. Specifically, the undesired properties are twofold: 1) Even in convergence, the fixed point η_E^π does not equal the true return distribution (bias). 2) Even in the fixed point η_E^π, the ensemble disagreement w_{avg} does not vanish. Often, however, we may be able to upper bound and control the error incurred due to the projections Π_i. In this case, Propositions 3.8 and 3.9 provide upper bounds on both types of errors as a function of bounded projection errors.

Proposition 3.8. Let Ω_M be a projection mixture operator with individual projections Π_i defined as in Eq. (3.7). Further, assume each projection Π_i is upper bounded by $w_p(\Pi_i\nu,\nu) \le d_i$ for some $p \in [1,\infty)$. Then, the p-Wasserstein distance between the fixed point $\eta_E^\pi(s,a) = \Omega_M\mathcal{T}^\pi\eta_E^\pi(s,a)$ and the true return distribution $\eta^\pi(s,a) = \mathcal{T}^\pi\eta^\pi(s,a)$ satisfies

$$w_p(\eta_E^\pi,\eta^\pi)(s,a) \;\le\; \frac{\bar{d}_p}{1-\bar{c}_p\gamma} \quad \forall (s,a)\in\mathcal{S}\times\mathcal{A}, \qquad \text{where} \qquad \bar{d}_p \;=\; \Big(\sum_{i=1}^{M}\frac{1}{M}d_i^p\Big)^{1/p}.$$

Proof. To show the desired property, we will use Proposition 3.1 and Lemma 3.4. We omitted the dependency on (s,a) in this section for brevity. It follows then from the triangle inequality that

$$w_p(\eta_E^\pi, \eta^\pi) \leq w_p(\Omega_M \mathcal{T}^\pi \eta_E^\pi, \Omega_M \eta^\pi) + w_p(\Omega_M \eta^\pi, \eta^\pi) \tag{3.44}$$

$$= w_p(\Omega_M \mathcal{T}^\pi \eta_E^\pi, \Omega_M \mathcal{T}^\pi \eta^\pi) + w_p(\Omega_M \eta^\pi, \eta^\pi) \tag{3.45}$$

$$\leq \bar{c}_p \gamma w_p(\eta_E^\pi, \eta^\pi) + w_p(\frac{1}{M} \sum_{i=1}^{M} \Pi_i \eta^\pi, \eta^\pi) \tag{3.46}$$

$$\leq \bar{c}_p \gamma w_p(\eta_E^\pi, \eta^\pi) + (\frac{1}{M} \sum_{i=1}^{M} w_p^p(\Pi_i \eta^\pi, \eta^\pi))^{1/p}. \tag{3.47}$$

Per the assumption of Proposition 3.8 and by rearranging we obtain the desired result. □

Proposition 3.9. *Let* $w_{avg} = \frac{1}{M(M-1)} \sum_{i,j=1}^{M} w_p(\hat{\eta}_i, \hat{\eta}_j)$ *be the average ensemble disagreement and assume individual projections* Π_i *are bounded by* $w_p(\Pi_i \nu, \nu) \leq d_i$. *For an ensemble* E *whose mixture distribution equals exactly the fixed point* $\eta_E^\pi(s,a) = \Omega_M \mathcal{T}^\pi \eta_E^\pi(s,a)$, *the average ensemble disagreement* w_{avg} *satisfies the inequality*

$$w_{avg}(s,a) \leq \frac{2M}{M-1} \bar{d} \quad \forall (s,a) \in \mathcal{S} \times \mathcal{A}, \qquad \text{where} \qquad \bar{d} = \frac{1}{M} \sum_{i=1}^{M} d_i.$$

Proof. In the fixed point $\eta_E^\pi(s,a) = \Omega_M \mathcal{T}^\pi \eta_E^\pi(s,a)$, the distributional error estimated by $w_{\text{avg}}(s,a)$ does not vanish, unlike the ground truth error given by $w_1(\eta_E^\pi, \Omega_M \mathcal{T}^\pi \eta_E^\pi)(s,a) = 0$. The shown property upper bounds this mismatch and is a direct consequence of the assumption $w_p(\Pi_i \nu, \nu) \leq d_i$ which postulates an upper bound on the error introduced by the projection Π_i in terms of the p-Wasserstein distance. The average disagreement is given by

$$w_{\text{avg}}(s,a) = \frac{1}{M(M-1)} \sum_{i,j=1}^{M} w_p(\hat{\eta}_i, \hat{\eta}_j)(s,a). \tag{3.48}$$

The proof is given by applying the triangle inequality and the assumption of the proposition with

$$w_p(\hat{\eta}_i, \hat{\eta}_j) = w_p(\Pi_i \eta_E^\pi, \Pi_j \eta_E^\pi) \tag{3.49}$$

$$\leq w_p(\Pi_i \eta_E^\pi, \eta_E^\pi) + w_p(\eta_E^\pi, \Pi_j \eta_E^\pi) \tag{3.50}$$

$$\leq d_i + d_j. \tag{3.51}$$

Plugging in and rearranging yields the desired result. □

Lemma 3.10. *[Projection error of the categorical projection (Rowland et al., 2018)] For any distribution* $\nu \in \mathscr{P}([z_{\min}, z_{\max}])$ *with support on the interval* $[z_{\min}, z_{\max}]$ *and a categorical projection as defined in Eq.* (3.5) *with* K *atoms* $z_k \in \{z_1, ..., z_K\}$ *s.t.* $z_1 \geq Z_{\min}$ *and* $z_K \leq z_{\max}$*, the error incurred by the projection* Π_C *is upper bounded in the* 1*-Wasserstein distance by the identity*

$$w_1(\Pi_C \nu, \nu) \leq \left[\sup_{1 \leq k \leq K} (z_{k+1} - z_k) \right].$$

Proof (restated). The proof uses the duality between the 1-Wasserstein distance and the 1-Cramér distance stating

$$l_1(\nu, \nu') = \int_{\mathbb{R}} |F_\nu(x) - F_{\nu'}(x)| dx = \int_0^1 |F_\nu^{-1}(\tau) - F_{\nu'}^{-1}(\tau)| d\tau = w_1(\nu, \nu'), \tag{3.52}$$

and can be obtained by a change of variables. The l_1 formulation simplifies the analysis of the categorical projection, yielding

$$\begin{aligned}
w_1(\Pi_C \nu, \nu) &= \int_{\mathbb{R}} |F_{\Pi_C \nu}(x) - F_\nu(x)| dx && (3.53)\\
&\leq \sum_{k=1}^{K-1} (z_{k+1} - z_k) |F_{\Pi_C \nu}(z_k) - F_\nu(z_k)| && (3.54)\\
&\leq \sum_{k=1}^{K-1} (z_{k+1} - z_k) |F_\nu(z_{k+1}) - F_\nu(z_k)| && (3.55)\\
&\leq \left[\sup_{1 \leq k \leq K} (z_{k+1} - z_k) \right] \sum_{k=1}^{K-1} |F_\nu(z_{k+1}) - F_\nu(z_k)| && (3.56)\\
&\leq \left[\sup_{1 \leq k \leq K} (z_{k+1} - z_k) \right]. && (3.57)
\end{aligned}$$

□

Lemma 3.11. *[Projection error of the quantile projection (Dabney et al., 2018b)] For any distribution* $\nu \in \mathscr{P}([z_{\min}, z_{\max}])$ *with support on the interval* $[z_{\min}, z_{\max}]$ *and a quantile projection defined according to Eq.* (3.6) *with* K *equally weighted locations* $\theta_k \in \{\theta_1, ..., \theta_K\}$*, the error incurred by the projection* Π_Q *is bounded in the* 1*-Wasserstein distance by the identity*

$$w_1(\Pi_Q \nu, \nu) \leq \frac{z_{\max} - z_{\min}}{K}.$$

Proof(restated). The projection Π_Q is given by

$$\Pi_Q \nu = \tfrac{1}{K} \textstyle\sum_{k=1}^K \delta_{F_\nu^{-1}(\tau_k)}, \qquad \text{where } \tau_k = \frac{2k-1}{2K}. \tag{3.58}$$

The desired identity $w_1(\Pi_Q \nu, \nu)$ is accordingly given by the continuous integral

$$w_1(\Pi_Q \nu, \nu) = \int_0^1 |F^{-1}_{\Pi_Q \nu}(\tau) - F^{-1}_{\nu}(\tau)| d\tau \,, \tag{3.59}$$

and can be rewritten in terms of a sum of piecewise expectations

$$w_1(\Pi_Q \nu, \nu) = \sum_{k=1}^{K} \frac{1}{K} \mathbb{E}_{X \sim \nu}\left[|X - F^{-1}_{\nu}(\tfrac{2k-1}{2K})| \,\middle|\, F^{-1}_{\nu}(\tfrac{k-1}{K}) < X \leq F^{-1}_{\nu}(\tfrac{k}{K}) \right]. \tag{3.60}$$

From this, it follows that

$$\begin{aligned} w_1(\Pi_Q \nu, \nu) &\leq \frac{1}{K}(F^{-1}_{\nu}(1) - F^{-1}_{\nu}(0)) && (3.61) \\ &\leq \frac{z_{\max} - z_{\min}}{K} \,. && (3.62) \end{aligned}$$

□

Corollary 3.12. *Let $\eta^{\pi}_E(s,a)$ be the fixed point return distribution for an ensemble of the categorical and quantile projections with the mixture operator $\Omega_M \eta(s,a) = 1/2\,\Pi_Q \eta(s,a) + 1/2\,\Pi_C \eta(s,a)$. Furthermore, suppose the return distribution $\eta^{\pi}_E(s,a)$ has bounded support on the interval $(R_{\max} - R_{\min})/(1-\gamma)$ where $R_{\max}$ and $R_{\min}$ denote the maximum and minimum immediate reward of the MDP. The average ensemble disagreement $w_{avg}(s,a)$ is then bounded by*

$$w_{avg}(s,a) \leq \frac{4(R_{\max} - R_{\min})}{(1-\gamma)K} \,.$$

Proof. The result follows straightforwardly from Proposition 3.9 and Lemmas 3.10, 3.11. □

The Categorical Projection

The full definition of the categorical (or also Cramér) projection as stated by Rowland et al. (2018) is given below.

Definition 3.13. [Categorical projection (Rowland et al., 2018)] For a set of fixed locations $z_1, ..., z_K$ where $z_1 < z_2 < ... < z_K$, let $h_{z_k} : \mathbb{R} \to [0,1]$ be the hat function centered around z_k for $k = 1, ..., K$ given by

$$h_{z_k}(x) = \begin{cases} \frac{z_{k+1} - x}{z_{k+1} - z_k} & \text{for } x \in [z_k, z_{k+1}] \quad \text{and } 1 \leq k < K, \\ \frac{x - z_{k-1}}{z_k - z_{k-1}} & \text{for } x \in [z_{k-1}, z_k] \quad \text{and } 1 < k \leq K, \\ 1 & \text{for } x \leq z_1 \quad \text{and } k = 1, \\ 1 & \text{for } x \geq z_K \quad \text{and } k = K, \\ 0 & \text{otherwise.} \end{cases}$$

Furthermore, let the categorical representation $\mathscr{F}_C$ be defined as a finite mixture of Dirac deltas $\mathscr{F}_C = \{\sum_{k=1}^{K} \theta_k \delta_{z_k} | \theta_k \geq 0, \sum_{k=1}^{K} \theta_k = 1\}$. The categorical projection operator $\Pi_C : \mathscr{P}(\mathbb{R}) \to \mathscr{F}_C$ of a distribution $\nu \in \mathscr{P}(\mathbb{R})$ is then defined as

$$\Pi_C \nu = \sum_{k=1}^{K} \mathbb{E}_{\omega \sim \nu}[h_{z_k}(\omega)] \delta_{z_k}.$$

4

Contextual Similarity Distillation

The work presented in this chapter is to appear as: M. A. Zanger, P. R. Van der Vaart, W. Böhmer, and M. T. J. Spaan. Contextual similarity distillation: Ensemble uncertainties with a single model. To appear in *International Conference on Learning Representations (ICLR)*, 2026.
Author contributions are as follows: *M.A.Z.:* Conceptualization, Methodology, Formal Analysis, Experimental Implementation, Visualizations, Writing — Original Draft. *P.R.V.:* Formal Analysis, Experimental Implementation, Writing — Review & Editing. *W.B.:* Supervision, Project Administration, Writing — Review & Editing. *M.T.J.S.:* Supervision, Project Administration, Funding Acquisition, Writing — Review & Editing.

In the previous chapter, we demonstrated that the uncertainty estimates of deep ensembles can be enhanced by instilling distinct inductive biases into the constituent members, thereby promoting a diversity in architecturally enforced generalization behaviors. This chapter takes a more ambitious step: we investigate whether it is possible to capture the predictive uncertainty of an entire ensemble within a single, computationally efficient model. This approach directly addresses our second *research question* (RQ2):

RQ2: *Can the predictive variance of supervised deep ensembles be approximated directly and accurately by a single neural network in the limit of infinite width?*

Our main idea is based on the analytically tractable predictive variance of deep ensembles in the infinite width limit governed by the neural tangent kernel (NTK). While analytically tractable, direct computation of this expression is typically prohiitive as it requires very large kernel matrix inversions. However, we identify that the crucial matrix inverse component of this expression can be re-framed as the solution to a unique supervised learning task: a *contextualized kernel regression* problem, where kernel similarities between "contextual" query points and training samples themselves serve as the regression targets.

This insight forms the basis of our proposed method, contextual similarity distillation (CSD). CSD is a novel framework that turns this theoretical regression problem into a practical, gradient-based training pipeline for single neural networks, allowing us to predict ensemble variances directly, without ever instantiating the full ensemble. In this chapter, we first detail this theoretical derivation. We then describe the practical CSD algorithm and its properties, one of which is its ability to leverage unlabeled data augmentations. Finally, we present a comprehensive empirical evaluation on a range of out-of-distribution detection benchmarks and hard-exploration reinforcement learning tasks, showing that CSD performs competitively with, and sometimes superior to, strong ensemble-based baselines at a fraction of the computational cost.

4.1 Introduction

With the deployment of increasingly large deep learning systems to real-world applications, efficient uncertainty quantification has become an essential challenge of modern deep learning. Assessing the reliability in predictions is crucial in applications ranging from OOD detection to deep reinforcement learning (RL), where uncertainty estimation is used to drive exploration, stabilize offline learning, increase data efficiency, or to design cautious, safety-aware

agents. A necessary condition for designing and deploying such agents is their ability to quantify uncertainty reliably and efficiently.

Bayesian methods for deep neural networks address this challenge with a solid theoretical footing (Goan and Fookes, 2020; Izmailov et al., 2021; Pearce et al., 2020) but often require coarse approximations or costly sampling from a complex posterior. To this end, deep ensembles from random initializations (Lakshminarayanan et al., 2017; Osband et al., 2016; Qin et al., 2022) have emerged as a simple but reliable method for estimating predictive uncertainty in neural networks. While usually more efficient than full Bayesian inference, the computational cost of training several models remains a burden, particularly with increasing parameter spaces.

In this paper, we introduce contextual similarity distillation (CSD), a novel single-model approach that directly estimates the variance of a random initialization ensemble of deep neural networks (DNNs) without ever training or evaluating such an ensemble in the first place. The theoretical motivation for our approach is derived from recent work characterizing the learning dynamics of wide neural networks through the NTK (Jacot et al., 2018; Lee et al., 2020b). Under some conditions, this setting allows us to describe deep ensembles and in particular their predictive variance by the NTK Gaussian process (NTK GP, He et al., 2020), providing an analytical expression for ensemble uncertainties. Although one can in principle solve these analytical expressions explicitly without requiring training of an ensemble of models, these computations quickly become infeasible when considering large models or datasets, as frequently encountered in the field of RL.

In contrast, CSD is amenable to regular training pipelines based on gradient descent and approximates predictive ensemble variance with a single forward pass. We derive our method from the insight that ensemble variance can be obtained as the result of a structured supervised regression problem, where labels correspond to kernel similarities between training points and a test point x_t. As a result, one can obtain the predictive variance of a deep ensemble for a known query point x_t by training a single neural network (NN) on a regression task using gradient descent and a carefully designed label function dependent on x_t. We then extend this "single-query" approach to work efficiently for arbitrary queries x_t by formulating a contextualized regression model that involves regression tasks with a family of context-dependent label functions. This formulation moreover enables CSD to refine its uncertainty estimates by leveraging unlabeled data, for example from a target domain of interest or from data augmentation techniques, an approach that has proven extraordinarily successful in the field of self-supervised and representation learning (Caron et al., 2021; Chen et al., 2020; Guo et al., 2022).

We analyze the practical effectiveness of CSD through an empirical eval-

uation on a variety of distribution shift detection tasks(Van Amersfoort et al., 2020) using the FashionMNIST, MNIST, KMNIST, and NOTMNIST datasets(Clanuwat et al., 2018; Deng, 2012; Xiao et al., 2017). We moreover use CSD to generate an exploration signal on sparse-reward reinforcement learning problems from the visual RL benchmark VizDOOM (Kempka et al., 2016). Empirically, CSD consistently achieves competitive and sometimes superior uncertainty estimation to finite deep ensembles and other baseline methods while maintaining lower computational cost. We believe these results establish CSD as a both principled and scalable alternative to ensemble-based uncertainty quantification and exploration methods.

4.2 Background

For our default framework, we consider a finite *Markov Decision Process* (MDP, Bellman, 1957) of the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \gamma, P, \mu)$, with state space $\mathcal{S}$, action space $\mathcal{A}$, immediate reward distribution $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathscr{P}(\mathbb{R})$, discount $\gamma \in [0,1]$, transition kernel $P : \mathcal{S} \times \mathcal{A} \to \mathscr{P}(\mathcal{S})$, and the start state distribution $\mu : \mathscr{P}(\mathcal{S})$. Here, $\mathscr{P}(\mathcal{Z})$ indicates the space of probability distributions over some space $\mathcal{Z}$ and random variables are denoted with uppercase letters. Given a state S_t at time t, agents choose an action A_t from a stochastic policy $\pi : \mathcal{S} \to \mathscr{P}(\mathcal{A})$ and subsequently receives the immediate reward $R_t \sim \mathcal{R}(\cdot|S_t, A_t)$ and observes next state $S_{t+1} \sim P(\cdot|S_t, A_t)$. The expected discounted sum of future rewards, conditioned on a particular state s and action a is known as the state-action value and is given by $Q^\pi(s,a) = \mathbb{E}_{P,\pi}[\sum_{t=0}^{\infty} \gamma^t R_t | S_0 = s, A_0 = a]$. This value function adheres to a temporal consistency condition described by the Bellman equation (Bellman, 1957)

$$Q^\pi(s,a) = \mathbb{E}_{P,\pi}[R_0 + \gamma Q^\pi(S_1, A_1) | S_0 = s, A_0 = a], \tag{4.1}$$

where $\mathbb{E}_{P,\pi}[\cdot]$ indicates that S_1 and A_1 are drawn from P and π respectively. The expected return of a policy π can compactly be expressed through the state-action value and the starting state distribution through

$$J(\pi) = \mathbb{E}_{S_0 \sim \mu, A_0 \sim \pi}[Q^\pi(S_0, A_0)]. \tag{4.2}$$

The objective of reinforcement learning is to find an optimal policy π^* that maximizes the above equation $\pi^* = \arg\max J(\pi)$.

4.2.1 Exploration in Reinforcement Learning

A fundamental challenge in attaining an optimal policy π^* lies in the exploration-exploitation trade-off: an agent must decide whether to exploit its current knowledge to maximize returns or whether to explore novel actions in

order to discover better strategies. Efficient exploration is particularly crucial in high-dimensional or sparse-reward settings, where naive strategies such as random exploration require prohibitive amounts of interactions.

A widely used approach to exploration is *optimism in the face of uncertainty* (Auer, 2002; Auer et al., 2008), where agents prioritize actions with high epistemic uncertainty in value estimates. In the context of model-free RL, provably efficient algorithms often rely on the construction of an UCB that overestimates the true optimal value $Q^{\pi^*}(s,a)$ with high probability (Jin et al., 2018; 2020; Neustroev and de Weerdt, 2020). This may be implemented by adding a well-chosen exploration bonus $b(s,a)$ to value estimates according to

$$Q^{\text{opt}}(s,a) = Q^{\pi}(s,a) + b(s,a). \tag{4.3}$$

In small state-action spaces, such bonuses can be derived from count-based concentration inequalities (Bellemare et al., 2016; Jin et al., 2020), whereas high-dimensional, continuous domains usually require function approximation, significantly complicating efficient uncertainty estimation (Burda et al., 2019b; Ghavamzadeh et al., 2015; Lakshminarayanan et al., 2017; Osband et al., 2016).

With the widespread use of deep neural networks, deep ensembles (Lakshminarayanan et al., 2017) based on random initialization have become a dominant tool for quantifying epistemic uncertainty in high-dimensional continuous spaces (Chen et al., 2017; He et al., 2020; Osband et al., 2019). An informal intuition behind the effectiveness of ensembles is the tendency of randomly initialized NNs to converge to diverse minima in the training loss landscape (Fort et al., 2019), leading to higher prediction diversity for unseen inputs. The variance among ensemble members can then be used to measure the model's uncertainty for a specific input.

4.2.2 Neural Tangent Kernel Gaussian Processes

In order to better understand the properties of deep ensembles and to design better exploration algorithms, an analytical description of deep neural networks and their learning dynamics is desirable. While a general framework remains elusive, significant progress has been made in the field of deep learning theory. In particular, seminal works by Jacot et al. (2018) and Lee et al. (2020b) have shown that wide neural networks trained by gradient descent are well-described by their linearized training dynamics and thus predictable.

For this, let neural networks be parametrized functions $f(x,\theta_t) : \mathbb{R}^n \to \mathbb{R}$ and denote training data $\mathcal{X} = \{x_i \in \mathbb{R}^n | i \in \{1,...,N_D\}\}$ and training labels $\mathcal{Y} = \{y_i \in \mathbb{R} | i \in \{1,...,N_D\}\}$. We assume training is performed using gradient descent with infinitesimal step sizes, also referred to as gradient flow. The initialization weights θ_0 are drawn i.i.d. from a normal distribution $\theta_0 \sim \mathcal{N}$, and deep ensembles are formed by training multiple independently initialized neural

network functions. We furthermore assume so-called NTK-parametrization, which scales forward and backward passes in proportion to layer widths (see Jacot et al., 2018; Lee et al., 2020b, for details).

A key result by Lee et al. (2020b) is that in the limit of infinite layer widths, the training dynamics of deep networks are described exactly by a Taylor expansion around the parameter initialization θ_0. In this setting, the NTK $\Theta(x, x') : \mathbb{R}^{n \times n} \to \mathbb{R}$, first described by Jacot et al. (2018), emerges as the defining function governing learning dynamics:

$$\Theta_0(x, x') = \nabla_\theta^\top f(x, \theta_0) \nabla_\theta f(x', \theta_0). \tag{4.4}$$

The NTK can be interpreted as a similarity measure between inputs based on gradient representations of the inputs x and x'. Crucially, Jacot et al. (2018) find that in the limit of infinite layer width, $\Theta(x, x')$ becomes deterministic despite random weight initialization $\Theta_0(x, x') = \Theta(x, x')$ and remains constant throughout training, inducing analytically solvable training dynamics. As a result, the post-training NN function $f(x, \theta_\infty)$ can be characterized as a deterministic function of the random initialization $f(x, \theta_0)$ through

$$f(x, \theta_\infty) = f(x, \theta_0) + \Theta(x, \mathcal{X}) \Theta(\mathcal{X}, \mathcal{X})^{-1} (y - f(\mathcal{X}, \theta_0)). \tag{4.5}$$

Here, we have overloaded notation to indicate the vectorization $\Theta(x, \mathcal{X}) \in \mathbb{R}^{1 \times N_D}$, $\Theta(\mathcal{X}, \mathcal{X}) \in \mathbb{R}^{N_D \times N_D}$, and so forth. The matrix $\Theta(\mathcal{X}, \mathcal{X})$ is also known as the training Gram matrix, as we will refer to it. Further extending this framework, He et al. (2020) demonstrate that by introducing suitable function priors on $f(x, \theta_0)$, akin to the well-known randomized prior functions by Osband et al. (2019), the post-training function is described by a *Gaussian process* (GP, Rasmussen and Williams, 2006):

$$f(\mathcal{X}_t, \theta_\infty) \sim \mathcal{N}(\mu_\infty(\mathcal{X}_t), \Sigma_\infty(\mathcal{X}_t)), \tag{4.6}$$

with mean and covariance given by

$$\begin{aligned} \mu_\infty(\mathcal{X}_t) &= \Theta(\mathcal{X}_t, \mathcal{X}) \Theta(\mathcal{X}, \mathcal{X})^{-1} y, \\ \Sigma_\infty(\mathcal{X}_t) &= \Theta(\mathcal{X}_t, \mathcal{X}_t) - \Theta(\mathcal{X}_t, \mathcal{X}) \Theta(\mathcal{X}, \mathcal{X})^{-1} \Theta(\mathcal{X}, \mathcal{X}_t), \end{aligned}$$

where $\mathcal{X}_t$ is an arbitrary test data set. An outline of the derivation of Equations 4.5 and 4.6 is provided in Appendix 4.8.1. Consequently, the variance of an ensemble over infinite random initializations is given by

$$\mathbb{V}[f(x, \theta_\infty)] = \Theta(x, x) - \Theta(x, \mathcal{X}) \Theta(\mathcal{X}, \mathcal{X})^{-1} \Theta(\mathcal{X}, x). \tag{4.7}$$

The above expression provides us with a theoretical footing for understanding the behavior and uncertainty estimates of deep ensembles. In the following sections we will describe our approach for estimating Eq. 4.7 not as the result of training several random models but deterministically with a single model.

4.3 Contextual Similarity Distillation

We now proceed to describe our approach, CSD. The main objective of our method is to approximate the variance of an infinite deep ensemble, as described by Eq. 4.7, directly with a single model.

4.3.1 Ensemble Variance Predictions for A Priori Queries

We introduce the underlying idea of CSD in the simplified setting of *a priori known* test points. Given a test query point x_t, it is our goal is to estimate the variance $\mathbb{V}[f(x_t, \theta_\infty)]$ of an ensemble of independently initialized NNs, trained on a dataset $\mathcal{X}$. It is important to note that one could in principle obtain this variance via the NTK Gaussian process (GP) by solving Eq. 4.7. This, however, requires inversion of the potentially very large Gram matrix $\Theta(\mathcal{X}, \mathcal{X})$, which becomes computationally prohibitive for most datasets and models of interest, including RL applications where sample sizes can go into the billions.

Instead of solving Eq. 4.7 directly, we leverage an alternative perspective that arises naturally from the learning dynamics of wide neural networks. Specifically, we begin with the simple observation that the variance of a wide ensemble at a test point x_t can be computed efficiently as the solution to a regular supervised regression problem of a single model with a particular label function. For this, let $g(x, \tilde{\theta}_t)$ be a NN of the same architecture as $f(x, \theta_t)$, thus inducing an equal NTK $\Theta_g(x, x') = \Theta(x, x')$. Recall that the post-training NN function $g(x, \tilde{\theta}_\infty)$ with squared loss on $\mathcal{Y}$ is given by

$$g(x, \tilde{\theta}_\infty) = g(x, \tilde{\theta}_0) + \Theta_g(x, \mathcal{X})\Theta_g(\mathcal{X}, \mathcal{X})^{-1}(\mathcal{Y} - g(\mathcal{X}, \tilde{\theta}_0)). \tag{4.8}$$

It is straightforward to see that for small function initialization[1] $g(x, \tilde{\theta}_0) \approx 0$, $\forall x$ the r.h.s. of this expression, when choosing the label function $\mathcal{Y}_{x_t}(\mathcal{X}) = \Theta(\mathcal{X}, x_t)$, simplifies to

$$g_{x_t}(x, \tilde{\theta}_\infty) = \Theta(x, \mathcal{X})\Theta(\mathcal{X}, \mathcal{X})^{-1}\Theta(\mathcal{X}, x_t), \tag{4.9}$$

where we used the subscript x_t to indicate the function's dependence on the label function $\mathcal{Y}_{x_t}$. This identity now recovers exactly the problematic right term of Eq. 4.7 containing the Gram inversion $\Theta(\mathcal{X}, \mathcal{X})^{-1}$. Note that $g_{x_t}(x, \tilde{\theta}_\infty)$ is obtained "naturally" as the result of gradient-based regression, without requiring explicit inversion of $\Theta(\mathcal{X}, \mathcal{X})$ or training of a large ensemble at any point. The ensemble variance in a query point x_t can be obtained as

$$\mathbb{V}[f(x_t, \theta_\infty)] = \Theta(x_t, x_t) - g_{x_t}(x_t, \tilde{\theta}_\infty), \tag{4.10}$$

[1]For example, small function initialization can simply be obtained by redefining $\hat{f}(x, \theta_t) := f(x, \theta_t) - f(x, \theta_0)$.

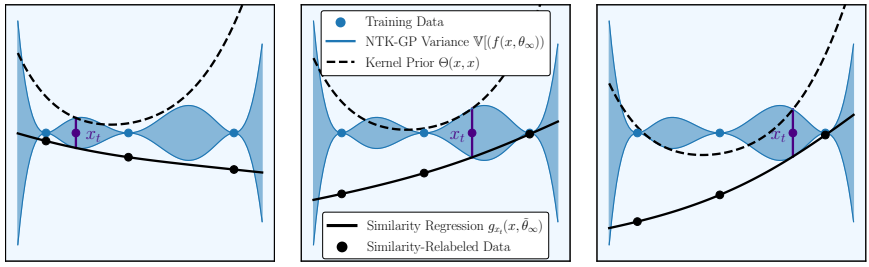


Figure 4.1: Illustration of regression tasks with query-dependent NTK similarities as labels. The difference between the kernel prior function $\Theta(x,x)$ (dotted line) and the post-training regression function $g_{x_t}(x,\tilde{\theta}_\infty)$ matches exactly ensemble variance in x_t. Plots from left to right depict the same principle, but for different query points x_t.

4

which can be computed efficiently. Fig. 4.1 illustrates the above-described process of obtaining expression 4.10 geometrically. While simple, we believe this formulation provides a crucial insight: uncertainty estimation for a NN can be phrased as a singular prediction problem of kernel similarities.

4.3.2 Ensemble Variance Estimation for Arbitrary Query Points

In the above derivation, we outlined an efficient method for obtaining ensemble variances at a specific test query point x_t *known a priori*. An obvious limitation of this approach, however, is that the used labeling function $y_{x_t}(\mathcal{X}) = \Theta(\mathcal{X}, x_t)$ and by extension the model $g_{x_t}(x,\tilde{\theta}_\infty)$ is inherently dependent on the test point x_t and not usable for arbitrary queries.

To overcome this limitation, we now formulate a *contextualized regression model* $g(x,c,\tilde{\theta}_t)$, where c serves as a context variable that determines the label function used during training of the function $g(x,c,\tilde{\theta}_t)$. Specifically, instead of defining a label function that depends on a single fixed test query x_t, we construct a family of label functions parameterized by the context c, $y_c(\mathcal{X}) = \Theta(\mathcal{X}, c)$. This means that for a set of context data $\mathcal{C} = \{c_i \in \mathbb{R}^n | i \in \{1,...,N_C\}\}$, the model $g(x,c,\tilde{\theta}_t)$ is optimized to solve a supervised regression problem associated with labels $y_c(\mathcal{X})$.

Intuitively, this approach can be interpreted as an attempt to interpolate between multiple regression solutions that were trained on the same dataset $\mathcal{X}$ but with different label functions $y_c(\mathcal{X})$. Geometrically, this corresponds to conjoining the functions g_{x_t} in Fig. 4.1 along a new dimension c. So long as $g(x,c,\tilde{\theta}_\infty)$ maintains the approximate dynamics of $g_c(x,\tilde{\theta}_\infty)$, this model can be evaluated quickly for arbitrary test points by setting $c = x_t$ in

$$g(x,c,\tilde{\theta}_\infty) \approx \Theta(x,\mathcal{X})\Theta(\mathcal{X},\mathcal{X})^{-1}\Theta(\mathcal{X},c). \tag{4.11}$$

This generalization accordingly enables ensemble variance estimation across arbitrary points x without requiring a separate regression solution for each individual query by computing

$$\mathbb{V}[f(x,\theta_\infty)] \approx \Theta(x,x) - g(x,x,\tilde{\theta}_\infty). \tag{4.12}$$

An intuitive interpretation of the function $g(x,x,\tilde{\theta}_\infty)$ is that it captures an ensemble's confidence gained through observing the training data $\mathcal{X}$, weighted by its similarity to x. The resulting variance of Eq. 4.12 can then be understood as the difference between a prior uncertainty term $\Theta(x,x)$ and the confidence term $g(x,x,\tilde{\theta}_\infty)$. One should note at this point, that the evaluation of $g(x,c,\tilde{\theta}_\infty)$ for contexts $c \notin \mathcal{C}$ not used during training requires g to generalize to novel c. Furthermore, the introduction of the context variable c may influence the training dynamics of g, putting this approach into the realm of approximate algorithms. We have added a section to Appendix 4.6 that discusses and summarizes used approximations and their implications for practical settings.

Finetuning variance estimates with context data. Before proceeding to describe our practical setup, we outline a property of contextualized similarity distillation that emerges through the above-described modeling choices. Our theoretical motivation highlights that *exact* ensemble variances (in the NTK regime) can be obtained when the test point x_t is known a priori. The implication of the subsequent formulation as a contextualized regression problem is that, when available, one can include unlabeled context data $\mathcal{C}$ during training to obtain better uncertainty estimates in the domain of interest, as we will show later in the experimental section. This property also opens up the possibility of using unlabeled data augmentations to improve uncertainty estimation, an approach that has proven extraordinarily successful in the field of self-supervised and representation learning (Caron et al., 2021; Chen et al., 2020; Guo et al., 2022) and not easily incorporated with standard approaches for uncertainty estimation (Burda et al., 2019b; Gal and Ghahramani, 2016; Lakshminarayanan et al., 2017).

4.3.3 Deep Contextualized Similarity Distillation

Building on this theoretical basis, we proceed to describe a setting for contextualized similarity distillation with deep neural networks. This section outlines algorithmic design choices we found to be computationally efficient while maintaining the approach's theoretical motivation.

First, we parameterize the contextualized regression model $g(x,c,\tilde{\theta}_\infty)$ as an inner product between a feature vector $\phi(x,\tilde{\theta}_f)$ and a context vector $\psi(c,\tilde{\theta}_c)$ as

$$g(x,c,\tilde{\theta}_\infty) = \phi(x,\tilde{\theta}_f)^\top \psi(c,\tilde{\theta}_c). \tag{4.13}$$

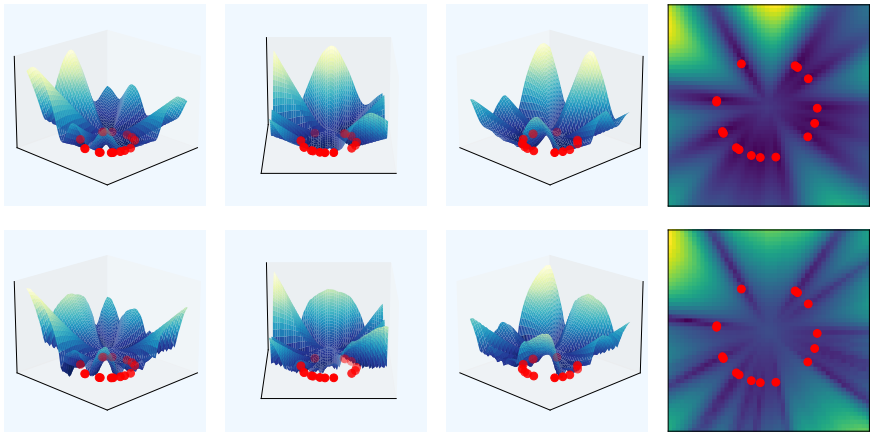


Figure 4.2: *Top Row*: Variance of an ensemble of 100 randomly initialized neural networks on a 2D toy regression task. Red dots are training points. *Bottom Row*: Variance prediction by CSD with a single model on the same regression task.

Conceptually, this parametrization can be thought of as introducing a context dependent final layer of weights, represented by $\psi(c, \tilde{\theta}_c)$, to the regression model g. Computationally, this inner product parametrization bears the advantage that $g(\mathcal{X}, \mathcal{C}, \tilde{\theta}_\infty) \in \mathbb{R}^{N_D \times N_C}$ can be evaluated quickly without requiring explicit forward passes for each pairing $(x_i \in \mathcal{X}, c_j \in \mathcal{C})$.

Second, we approximate the NTK prior $\Theta(x, x')$ with partial gradients. Given that $\Theta(x, x')$ is not involved in backward gradient computations, computing the full analytical or empirical prior kernel functions $\Theta(x, x')$ is often not computationally prohibitive, but can pose a burden for models with large parameter spaces. We find that gradients with respect to only the last layer weights θ_0^L are sufficient in practice and further accelerate computation. Assuming, the last layer of f is a dense layer such that $f(x, \theta_0) = \varphi(x, \theta_0^{1:L-1})^\top \theta_0^L$, we have

$$\Theta^L(x, x') = \nabla_{\theta_0^L}^\top f(x, \theta_0) \nabla_{\theta_0^L} f(x', \theta_0) = \varphi(x, \theta_0^{1:L-1})^\top \varphi_f(x', \theta_0^{1:L-1}). \tag{4.14}$$

The resulting training pipeline for $g(x, c, \tilde{\theta}_t)$ involves a simple supervised regression task with minimization of the squared loss, where (x_i, c_i) are sampled randomly from $\mathcal{X}$ and $\mathcal{C}$

$$\mathcal{L}(\tilde{\theta}_t) = \frac{1}{N} \sum_i^N \frac{1}{2} \big(g(x_i, c_i, \tilde{\theta}_t) - \Theta^L(x_i, c_i) \big)^2. \tag{4.15}$$

Lastly, we propose several choices for the context data $\mathcal{C}$. We find that the arguably simplest choice, that is to reuse the training set $c_i \sim \mathcal{X}$, works well in

practice and is easily implemented. In addition, it is possible to apply data augmentations to the training samples $\mathcal{X}$ when using as context data. For this, we employ the well-established set of augmentations from the contrastive learning literature (Chen et al., 2020). We note here, that designing novel data augmentation techniques for the purpose of uncertainty quantification is a promising avenue (see for example works by Wen et al. (2020) and Wu and Williamson (2024)). Unlike contrastive learning and many other self-supervised methods, our approach does not require data augmentations to preserve the nature of the original label and can in principle use any unlabeled data. Finally, when available, unlabeled data from the test distribution of interest can be used and often provides an additional improvement in uncertainty estimation, as we will show empirically.

4.4 Empirical Evaluation

Our empirical evaluation aims to provide us with a better understanding of CSD in practice. Given that our approach introduces approximations beyond the theoretical framework, we investigate whether CSD maintains its theoretically motivated properties in practice with high-dimensional problem and parameter spaces. Specifically, we aim to assess whether CSD provides a scalable alternative to deep ensembles and other established methods in uncertainty quantification, including Monte Carlo dropout (Gal and Ghahramani, 2016), a Bayesian neural network (BNN) based on Markov chain Monte Carlo sampling (BNN - MCMC, Garriga-Alonso and Fortuin, 2021), a Laplace approximated Bayesian NN (BNN - Laplace, Immer et al., 2021), deep ensembles of sizes 3 and 15 (ENS, Lakshminarayanan et al., 2017) and random network distillation (RND, Burda et al., 2019b). Furthermore, we analyze how algorithmic design choices, such as the choice of context data, influence uncertainty estimates. Lastly, we seek to evaluate our approach's efficacy as an exploration signal for deep reinforcement learning agents on sparse-reward visual exploration tasks from the VizDoom (Kempka et al., 2016) suite.

4.4.1 Distribution Shift Detection

Following prior work (Immer et al., 2021; Rudner et al., 2022; Van Amersfoort et al., 2020), we evaluate uncertainty estimates in image classification under distribution shift, where a model trained on an in-distribution dataset is evaluated on inputs from a shifted distribution.

In particular, we train models on one of the FashionMNIST, MNIST, KMNIST, NotMNIST datasets and evaluate uncertainty estimates on the other, shifted datasets and a perturbed version of the in-distribution dataset. Well-

Table 4.1: Distribution Shift Detection. Test accuracy and average OOD detection metrics across MNIST, FashionMNIST, KMNIST, NotMNIST. OOD metrics are evaluated for each ID dataset against the remaining OOD datasets and a perturbed version of the ID dataset.

Method	Acc.	AUROC	AUPR-IN	AUPR-OUT
MCD	94.39 ± 0.10	85.67 ± 0.21	81.73 ± 0.34	86.44 ± 0.20
BNN-MCMC	87.70 ± 0.38	83.17 ± 0.60	82.65 ± 0.66	82.28 ± 0.71
BNN-Laplace	90.86 ± 0.62	81.38 ± 0.73	79.43 ± 0.84	81.84 ± 0.66
RND	96.18 ± 0.05	**94.40** ± 0.41	94.17 ± 0.63	**94.01** ± 0.31
ENS(3)	96.91 ± 0.04	92.30 ± 0.09	92.83 ± 0.10	91.37 ± 0.11
ENS(15)	**97.18** ± 0.03	94.00 ± 0.07	**94.70** ± 0.07	92.99 ± 0.06
CSD	96.29 ± 0.07	**96.63** ± 0.35	**96.94** ± 0.39	**96.19** ± 0.32
CSD-Aug.	96.28 ± 0.06	**98.22** ± 0.14	**98.51** ± 0.13	**97.80** ± 0.17
CSD-OOD.	96.30 ± 0.06	**98.57** ± 0.14	**98.86** ± 0.12	**98.19** ± 0.15

calibrated epistemic uncertainty estimates will correlate with dataset shift, such that out-of-distribution samples are likely to be rated more uncertain than in-distribution samples. To compare methods quantitatively, we use the threshold-independent AUROC metric, as well as the AUPR curve for in-distribution (ID) and OOD samples. The AUROC metric can be interpreted as the likelihood of an OOD sample receiving higher uncertainty than an ID sample, while AUPR-IN and AUPR-OUT provide additional sensitivity to dataset size and the choice of the positive class. For these metrics, Table 4.1 reports the average and standard deviation over 10 seeds, averaged over all permutations of ID and OOD datasets, along with average test accuracy. Full detailed results are provided in the supplementary material.

To analyze the role of the used context data, we evaluate three versions of CSD: a baseline that only uses training data (CSD), a variant incorporating data augmentations to training samples (CSD-Aug.), and a model using context data from the evaluation distribution (CSD-OOD). Even in the basic version, CSD demonstrates highly effective distribution shift detection, surpassing baseline methods on a variety of datasets while requiring only a single model. Our results furthermore suggest that incorporating data augmentations and target-distribution context data indeed significantly improves performance.

4.4.2 Exploration in VizDoom

We now evaluate CSD in a reinforcement learning task with high-dimensional observation spaces and sparse rewards. For this, we consider visual navigation tasks in the VizDOOM environment, where agents explore a 3D maze-like environment with ego-perspective image observations. The agent is tasked with

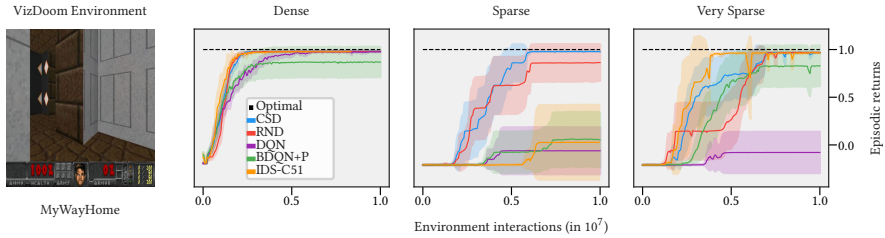


Figure 4.3: (Left): Visual observation in the VizDoom environment (Kempka et al., 2016). (From Second Left to Right): Mean learning curves in variations of VizDoom *MyWayHome*. Shaded regions are 90% Student's t confidence intervals from 10 seeds.

reaching a goal while receiving a minimal constant negative reward except upon successful completion, where a reward of 1 is given. We consider three variations of the task, where agents are initialized at increasing distances from the goal, defining progressively harder exploration tasks (details provided in Appendix B.1.2).

We use a deep Q-network (DQN) agent (Mnih et al., 2015) as a base algorithm and include uncertainty estimates by CSD as an intrinsic reward (full details provided in Appendix B.1). For a comparative evaluation, we compare the performance of CSD-based exploration with several baseline algorithms, including DQN (Mnih et al., 2015), random network distillation (random network distillation (RND), Burda et al., 2019b), bootstrapped Q-networks (BDQNP, Osband et al., 2019), and *information-directed sampling* (IDS, Nikolov et al., 2019). Fig. 4.3 shows mean learning curves across 10 random seeds. Interestingly, the sparse version of the environment appears to be the hardest, a circumstance we believe is due to the spawning point lying in a sidearm of the maze map. Of the tested methods, only CSD was able to find the goal across all seeds and environments, with RND performing most competitively.

4.5 Related Work

Our work builds on the extensive body of literature in the field of uncertainty quantification in deep learning and reinforcement learning. Ensemble learning (Dietterich, 2000) has emerged as on the most effective and reliable approaches to uncertainty estimation (Lakshminarayanan et al., 2017) and has been widely adopted in the deep reinforcement learning literature. In particular, ensembles can be used for efficient exploration by sampling random models (Osband and Van Roy, 2017; Osband et al., 2016; Qin et al., 2022), by constructing upper confidence bounds for exploration bonuses (Chen et al., 2017; O' Donoghue et al., 2018) or by estimating information gain (Nikolov et al., 2019).

Several works moreover rely on deep ensembles to reduce overestimation and improve learning stability (Chen et al., 2021; Fujimoto et al., 2018; Haarnoja et al., 2018), extending to the challenging offline setting (Agarwal et al., 2020; An et al., 2021; Smit et al., 2021).

A number of previous works have focused on reducing ensemble size, notably by disaligning the Jacobian of networks (An et al., 2021), adding repulsive loss terms (Sheikh et al., 2022), or through architectural diversification (Osband et al., 2019; Zanger et al., 2024). Notably, various works aim to quantify epistemic uncertainty with a single model (Burda et al., 2019b; Filos et al., 2021; Guo et al., 2022; Lahlou et al., 2021; Pathak et al., 2017), often by measuring prediction errors. To the best of our knowledge, few single-model methods in the field offer an interpretation as ensemble or posterior uncertainty.

In a broader sense, ensembles have been studied extensively from a Bayesian perspective (D'Angelo and Fortuin, 2021; Hoffmann and Elster, 2021). In particular, some of our work relies on the NTK GP characterization of deep ensembles by He et al. (2020), who, in turn, rely on seminal work by seminal work on the NTK by Jacot et al. (2018) and Lee et al. (2020b). Subsequent analysis has used the NTK to disentangle ensemble variance (Kobayashi et al., 2022). Recent works (Wilson et al., 2025) rely on NTK theory to derive a sampling-based uncertainty estimator, while Calvo-Ordoñez et al. (2024) construct uncertainty estimates using several regression models. In contrast to the latter, our method uses a contextualized regression model that allows for single-model uncertainty estimates in a deep learning setting.

4.6 Limitations and Assumptions

As our method relies on several approximations, we include a discussion that aims to provide an overview of the approximate nature of our method and in which settings it is exact or where deviations may be more likely.

The first central approximation we make is to model neural networks with dynamics governed by a deterministic and constant NTK. Jacot et al. (2018) show that this is the case for fully connected NNs with NTK parametrization trained on a squared loss. The implied dynamics are solved assuming gradient flow, that is with infinitesimal step sizes and full-batch gradients. Jacot et al. (2018) and Lee et al. (2020b) moreover show that convergence and final generalization behavior is empirically well-described by wide but finite architectures including fully connected NNs, CNNs and residual architectures, trained with stochastic gradient descent. The function initialization scheme proposed by He et al. (2020) allows for a GP interpretation of NNs from random initialization and largely relies on the same assumptions as the above-described works.

Our theoretical motivation, outlined in Sections 4.3.1 and 4.3.2, relies on

the GP description of deep ensembles and the implied assumptions. Given this setting, that is assuming NTK parametrization with infinite widths, function initialization according to He et al. (2020), and gradient flow with squared loss, the derivation for single-query ensemble variances in Section 4.3.1 is exact. In our contextualized model described in Section 4.3.2, we introduce an additional approximation through the introduction of an explicit context variable c, which may interfere with the training dynamics of $g(x, c, \tilde{\theta})$. Let training tuples be $x^c = (x, c)$ and $\mathcal{X}^c = \{x_1^c, x_2^c, ..., x_{N_T}^c\}$ and let the NTK of g be $\Theta_g((x,c),(x',c')) = \nabla_{\tilde{\theta}}^\top g(x,c,\tilde{\theta}_0)\nabla_{\tilde{\theta}} g(x',c',\tilde{\theta}_0)$. The analogous regression solution to the function $g(x, c, \tilde{\theta})$ by minimizing the loss in Eq. 4.15 becomes

$$g(x,c,\tilde{\theta}_\infty) = \Theta_g(x^c, \mathcal{X}^c)\Theta_g(\mathcal{X}^c, \mathcal{X}^c)^{-1}\Theta(\mathcal{X}^c). \tag{4.16}$$

A natural setting in which these training dynamics recover Eq. 4.11 is when gradients are independent between context, that is $\Theta_g((x,c),(x,c')) = 0$ if $c \neq c'$ and maintain the gradient structure of $\Theta(x, x')$ with $\Theta_g((x,c),(x',c)) = \Theta(x,x')$, $\forall c \in \mathcal{C}$. As this setting would hardly permit meaningful interpolations and extrapolations between different contexts c, one engages in a trade off between generalization capability towards general contexts c and interference in the training dynamics.

In our practical setting, we furthermore approximate the NTK prior function with partial gradients as outlined in Eq. 4.14 of Section 4.3.3. The influence of this approximation choice generally depends on architecture, but we found it to perform well in our experiments using deep convolutional and residual architectures. Lastly, the RL exploration setting involves data streams rather than fixed datasets $\mathcal{X}$, further deviating from the earlier delineated dynamics. Understanding the influence of this non stationarity on training dynamics is an open problem, and we believe countermeasures like periodic resets (D'Oro et al., 2023) are a promising avenue for future research.

4.7 Conclusion

This work introduced *contextual similarity distillation* (CSD), a novel single-model approach for uncertainty quantification that estimates the predictive variance of an ensemble with a single model and forward pass. By reframing ensemble variance estimation as a structured regression problem, CSD enables efficient uncertainty estimation without requiring the training of multiple models, stochastic forward passes, or explicit kernel matrix inversion. Instead, phrasing predictive variance estimation as a contextualized regression problem is amenable to standard training pipelines with DNNs and gradient descent.

We implemented CSD in a deep learning setting and performed a comparative evaluation on a variety of distribution shift detection and reinforcement learning tasks. Empirically, we found that CSD provides uncertainty estimates competitive and sometimes superior to deep ensembles and other alternatives on all tasks. This makes CSD an attractive option for guiding exploration in RL, as our experiments on high-dimensional exploration tasks confirmed. Our results furthermore confirmed that our approach can leverage unlabeled target domain data and data augmentations to further refine uncertainty estimates. We believe our work opens up several avenues for future research, including applications in model-based and offline RL, or the use of more refined data augmentation techniques.

Our findings, we believe, position CSD as a scalable alternative to deep ensembles, offering a principled and computationally efficient method for uncertainty quantification in deep learning.

4.8 Proofs

This section provides informal derivations of the learning dynamics used in this chapter.

4.8.1 Linearized Neural Network Learning Dynamics

For completeness, we briefly outline a sketch for how the GP interpretation of wide neural networks governed by NTK dynamics described in Expression 4.6 can be obtained. This section largely follows the seminal works by Jacot et al. (2018), Lee et al. (2020b) and He et al. (2020), to whom we refer readers interested in further details.

We begin by constructing a first-order Taylor expansion of the neural network function $f(x, \theta_0)$ around its initialization parameters θ_0:

$$f_{\text{lin}}(x, \theta_t) = f(x, \theta_0) + \nabla_\theta^\top f(x, \theta_0)(\theta_t - \theta_0). \tag{4.17}$$

When trained on $\mathcal{X}$ and $\mathcal{Y}$ with the squared error loss $\mathcal{L} = \frac{1}{2}\|f_{\text{lin}}(\mathcal{X}; \theta_t) - \mathcal{Y}\|^2$, gradient flow with a learning rate α induces an evolution of θ_t according to

$$\frac{\mathrm{d}}{\mathrm{d}t}\theta_t = -\alpha \nabla_\theta \mathcal{L} = -\alpha \nabla_\theta f_{\text{lin}}(\mathcal{X}, \theta_t) \nabla_{f_{\text{lin}}(\mathcal{X}, \theta_t)} \mathcal{L}. \tag{4.18}$$

In function space, this evolution translates to the expression

$$\frac{\mathrm{d}}{\mathrm{d}t} f_{\text{lin}}(x; \theta_t) = \nabla_\theta^\top f_{\text{lin}}(x, \theta_t) \frac{\mathrm{d}}{\mathrm{d}t}\theta_t = -\alpha \Theta_0(x, \mathcal{X})(f_{\text{lin}}(\mathcal{X}; \theta_t) - \mathcal{Y}), \tag{4.19}$$

where $\Theta_0(x,x') = \nabla_\theta^\top f(x,\theta_0)\nabla_\theta f(x',\theta_0)$ is the (empirical) tangent kernel of $f_{\text{lin}}(x,\theta_t)$. Since this linearization has constant gradients $\nabla_\theta f(x,\theta_0)$, the resulting differential equation is linear and solvable. For the substitution $v_t = (f_{\text{lin}}(\mathcal{X};\theta_t) - \mathcal{Y})$, we obtain the training error dynamics $\frac{\mathrm{d}}{\mathrm{d}t}v_t = -\alpha\Theta_0(\mathcal{X},\mathcal{X})v_t$ to which an exponential ansatz yields the solution

$$f_{\text{lin}}(\mathcal{X};\theta_t) - \mathcal{Y} = e^{-\alpha t\Theta_0(\mathcal{X},\mathcal{X})}(f(\mathcal{X};\theta_0) - \mathcal{Y}), \tag{4.20}$$

where the matrix exponential $e^{-\alpha t\Theta_0(\mathcal{X},\mathcal{X})}$ was used. Plugging Eq. 4.20 back into Eq. 4.19, one arrives at the identity

$$\frac{\mathrm{d}}{\mathrm{d}t}f_{\text{lin}}(x;\theta_t) = -\alpha\Theta_0(x,\mathcal{X})e^{-\alpha t\Theta_0(\mathcal{X},\mathcal{X})}(f(\mathcal{X};\theta_0) - \mathcal{Y}). \tag{4.21}$$

This differential expression is explicit in its terms such that we can obtain a solution by integration through

$$f_{\text{lin}}(x;\theta_t) = f(x,\theta_0) + \int_0^t \frac{\mathrm{d}}{\mathrm{d}t'} f_{\text{lin}}(x,\theta_{t'})\,\mathrm{d}t' \tag{4.22}$$

$$= f(x,\theta_0) + \Theta_0(x,\mathcal{X})\Theta_0(\mathcal{X},\mathcal{X})^{-1}(e^{-\alpha t\Theta(\mathcal{X},\mathcal{X})} - I)(f(\mathcal{X},\theta_0) - \mathcal{Y}), \tag{4.23}$$

which recovers Eq. 4.5 for $t \to \infty$. A central result by Jacot et al. (2018) and extended in the linearized setting by Lee et al. (2020b) is that, as layer widths of the neural network go to infinity, the NTK $\Theta_0(x,x')$ becomes deterministic and constant and the linear approximation $f_{\text{lin}}(x;\theta_t)$ becomes exact w.r.t. the original function $\lim_{\text{width}\to\infty} f_{\text{lin}}(x;\theta_t) = f(x,\theta_t)$.

4.8.2 Distribution of Neural Network Functions

Having established the training dynamics of a linearized neural network and its idealized limit in infinite width, we now aim to express how functions output functions of neural networks distribute as a consequence of random weight initializations.

Rewriting the (infinite width) post-training test and training functions as an affine transformation of the initialization yields

$$\begin{pmatrix} f(\mathcal{X}_t,\theta_\infty) \\ f(\mathcal{X},\theta_\infty) \end{pmatrix} = \begin{pmatrix} I & -\Theta(\mathcal{X}_t,\mathcal{X})\Theta(\mathcal{X},\mathcal{X})^{-1} \\ 0 & 0 \end{pmatrix}\begin{pmatrix} f(\mathcal{X}_t,\theta_0) \\ f(\mathcal{X},\theta_0) \end{pmatrix} + \begin{pmatrix} \Theta(\mathcal{X}_t,\mathcal{X})\Theta(\mathcal{X},\mathcal{X})^{-1}\mathcal{Y} \\ \mathcal{Y} \end{pmatrix}. \tag{4.24}$$

For the earlier described parametrization of f, the set of initial predictions is known to follow a multivariate Gaussian distribution (Lee et al., 2018a) described by the neural network Gaussian process (NNGP) $f(\mathcal{X},\theta_0) \sim \mathcal{N}(0,\kappa(\mathcal{X},\mathcal{X}))$

(and analogously for $\mathcal{X}_t$), where

$$\kappa(\mathcal{X}_t, \mathcal{X}_t) = \mathbb{E}_{\theta_0}\left[f(\mathcal{X}_t, \theta_0) f(\mathcal{X}_t, \theta_0)^\top \right]. \tag{4.25}$$

Affine transformations of multivariate Gaussian random variables $X \sim \mathcal{N}(\mu_X, \Sigma_X)$ with $Y = a + BX$ are, in turn, multivariate Gaussian random variables with distribution $Y \sim \mathcal{N}(a + B\mu_X,\ B\Sigma_X B^\top)$. We here omit explicit derivations and rearrangements for brevity. As a consequence, Eq. 4.24 with initialization covariance from Eq. 4.25 is also described by a multivariate Gaussian with mean and covariance given by

$$\begin{aligned}
\mathbb{E}_{\theta_0}[f(\mathcal{X}_t, \theta_\infty)] &= \Theta(\mathcal{X}_t, \mathcal{X})\Theta(\mathcal{X}, \mathcal{X})^{-1} y\,, \\
\mathrm{Cov}(f(\mathcal{X}_t, \theta_\infty)) &= \kappa(\mathcal{X}_t, \mathcal{X}_t) - \Theta(\mathcal{X}_t, \mathcal{X})\Theta(\mathcal{X}, \mathcal{X})^{-1}\kappa(\mathcal{X}, \mathcal{X})\Theta(\mathcal{X}, \mathcal{X})^{-1}\Theta(\mathcal{X}, \mathcal{X}_t) \\
&\quad - (\Theta(\mathcal{X}_t, \mathcal{X})\Theta(\mathcal{X}, \mathcal{X})^{-1}\kappa(\mathcal{X}, \mathcal{X}_t) + \text{h.c.}),
\end{aligned} \tag{4.26}$$

where h.c. refers to the Hermitian conjugate of the preceding term. He et al. (2020) then introduce constant "correction" terms to the function initialization described in Eq. 4.25, in particular such that $\kappa(x, x') = \Theta(x, x')$. This simplifies Expression 4.26 significantly and now permits a GP interpretation with the final expression given by Eq. 4.6.

5

An Analysis of Random Network Distillation

This chapter is based on unpublished work: M. A. Zanger, Y. Wu, W. Böhmer, and M. T. J. Spaan. On the Equivalence of Random Network Distillation, Deep Ensembles, and Bayesian Inference, 2025.

Author contributions are as follows: *M.A.Z.:* Conceptualization, Methodology, Formal Analysis, Experimental Implementation, Visualizations, Writing — Original Draft. *Y.W.:* Discussions, Experimental Implementation. *W.B.:* Supervision, Project Administration, Writing — Review & Editing. *M.T.J.S.:* Supervision, Project Administration, Funding Acquisition, Writing — Review & Editing.

In addition to designing novel methods for principled uncertainty quantification, as done in the previous chapter, another promising path to this end is to theoretically analyze existing, empirically successful but less understood approaches. Among such methods, random network distillation (RND) is a prominent example due to its simplicity and effectiveness in driving exploration (Burda et al., 2019b). RND operates on a simple principle: it measures novelty via the prediction error of an online network trained to match the outputs of a fixed, randomly initialized target network. Despite its widespread use, the theoretical foundations of RND have remained largely unexplored, and it has been unclear what form of uncertainty its self-predictive error signal truly captures. This chapter aims to bridge this theoretical gap by providing a formal analysis of RND, thereby addressing our third research question (RQ3):

RQ3: *What is the theoretical nature of the uncertainty captured by random network distillation, as a prominent example of single-model heuristic methods, when analyzed in the infinite-width limit?*

To answer this, we analyze RND within the neural tangent kernel (NTK) framework. Our analysis reveals two central findings. First, we establish that the RND error signal is not merely a heuristic but is, in the infinite-width limit, formally equivalent to the predictive variance of a corresponding deep ensemble. This result provides a strong theoretical justification for RND's empirical success as a measure of epistemic uncertainty.

Second, building on this equivalence, we show that by strategically constructing the RND target function — a technique inspired by prior-shaping in Bayesian deep ensembles (He et al., 2020) — we can devise a *Bayesian RND* algorithm. We prove that the error distribution of this modified algorithm directly mirrors the centered posterior predictive distribution of an infinitely wide Bayesian neural network (BNN). Based on this, we further derive a practical posterior sampling algorithm using Bayesian RND. Collectively, these findings provide a unified theoretical perspective that situates RND within the principled frameworks of deep ensembles and Bayesian inference, and offer new avenues for developing efficient yet theoretically-grounded uncertainty quantification methods.

5.1 Introduction

Quantifying predictive uncertainty remains a cornerstone of reliable machine learning and underpins applications from safe robotics to efficiently exploring agents and autonomous scientific discovery. Bayesian inference is widely regarded as a theoretical gold-standard to this end (Goan and Fookes, 2020; Neal, 1996) but its application to neural networks is typically intractable in prac-

tice, requiring approximations of simplified posteriors through variational inference (VI, Blei et al., 2017; Gal and Ghahramani, 2016; Kingma and Welling, 2014) or complex sampling mechanisms through Markov chain Monte Carlo approaches (MCMC, Chen et al., 2014; Garriga-Alonso and Fortuin, 2021; Liu and Wang, 2016). Deep ensembles (Dietterich, 2000; Lakshminarayanan et al., 2017) on the other hand maintain several independently initialized models to quantify predictive variance as uncertainty. Due to their simplicity and relative practical reliability, deep ensembles have become a widely established alternative to Bayesian approaches for uncertainty quantification in deep learning (Abdar et al., 2021). However, both ensemble methods and approximate Bayesian methods typically incur substantial computational and memory costs, in particular for larger-scale models, motivating lighter-weight alternatives.

RND (Burda et al., 2019b) offers one such approach: by training a *predictor network* to mimic the outputs of a fixed, randomly initialized *target network*, RND produces a simple novelty or uncertainty signal via the squared prediction error. RND has seen empirical success in exploration, out-of-distribution detection, and continual learning (Burda et al., 2019b; Matthews et al., 2024; Nikulin et al., 2023), yet the theoretical understanding of the nature of its uncertainty estimates remains blurry. In particular, it is unclear how — or whether — the RND error relates to the principled uncertainties produced for example by Bayesian inference or deep ensembles.

In this paper, we establish these missing theoretical connections by analyzing random network distillation in the idealized setting of infinite network width. In particular, we establish a Gaussian process (GP) interpretation of the self-predictive RND errors in the limit of infinitely wide neural networks, drawing on NTK theory (Jacot et al., 2018; Lee et al., 2020b). Our three main contributions are:

1. *Ensemble equivalence with Standard RND:* We prove that, in the idealized infinite width limit, the squared prediction errors of standard RND coincide exactly with the variance of a deep ensemble.

2. *Posterior equivalence with Bayesian RND:* By carefully engineering the RND target function, we design a *Bayesian RND* variant whose error distribution matches that of the exact Bayesian posterior predictive distribution of a neural network in the limit of infinite width.

3. *Posterior sampling with Bayesian RND:* Based on a multi-headed Bayesian RND model, we devise a posterior sampling algorithm that produces i.i.d. samples of the exact Bayesian posterior predictive distribution of neural networks in the limit of infinite width.

This unifying perspective on the uncertainty estimates produced by RND, deep ensembles, and Bayesian inference provides a novel understanding and theoretical support for the empirical effectiveness of RND and suggests avenues for future research directions towards principled Bayesian inference with minimal computational overhead.

5.2 Background

We begin by establishing notation, defining RND formally, and briefly introducing the theoretical framework used in our analysis. We denote $f(x;\theta) : \mathbb{R}^{d_{\text{in}}} \to \mathbb{R}^K$ a neural network function characterized by its parameters $\theta \in \mathbb{R}^P$. We primarily consider standard architectures such as fully connected feedforward neural networks. We will furthermore overload notation to concatenate function outputs, for example indicating a set $\mathcal{X} = \{x_i \in \mathbb{R}^{d_{\text{in}}}\}_{i=1}^{N_D}$ and the corresponding function output as a column vector $f(\mathcal{X};\theta_t) = (f(x_i;\theta_t))_{i=1}^{N_D}$, where $f(\mathcal{X};\theta_t) \in \mathbb{R}^{N_D \times K}$ or matrix-valued identities $\Sigma(\mathcal{X},\mathcal{X}) = (\Sigma(x_i,x_j))_{i,j=1}^{N_D}$, where $\Sigma(\mathcal{X},\mathcal{X}) \in \mathbb{R}^{N_D \times N_D}$. For conciseness our notation will furthermore use a shorthand for covariance and kernel matrices denoting $\Sigma_{\mathcal{X}\mathcal{X}} \equiv \Sigma(\mathcal{X},\mathcal{X})$. In the following we briefly review two methods pertinent to this work.

Random network distillation. Random network distillation (Burda et al., 2019b) is an uncertainty quantification technique that employs two neural networks of identical architecture: A fixed, randomly initialized *target network* $g(x;\psi_0) : \mathbb{R}^{d_{\text{in}}} \to \mathbb{R}^K$, and a *predictor network* $u(x;\vartheta_t)$, where parameters ϑ_t are subject to optimization via gradient descent. In particular, the predictor is trained to minimize the expected squared difference to the target network's output on a set of data points $\mathcal{X} = \{x_i \in \mathbb{R}^{d_{\text{in}}}\}_{i=1}^{N_D}$

$$\mathcal{L}_{\text{rnd}}(\vartheta_t) = \frac{1}{2}\|u(\mathcal{X};\vartheta_t) - g(\mathcal{X};\psi_0)\|_2^2. \tag{5.1}$$

It is common to design RND with a multi headed architecture with output dimension K and individual output heads $\{u_i(x;\vartheta_t)\}_{i=1}^K$, and $\{g_i(x;\psi_0)\}_{i=1}^K$, where the sum of squared prediction errors $\epsilon_i(x;\vartheta_t,\psi_0) = u_i(x;\vartheta_t) - g_i(x;\psi_0)$ at a test point x serves as an uncertainty or novelty signal

$$\epsilon^2(x;\vartheta_t,\psi_0) = \frac{1}{K}\sum_{i=1}^{K}\big(u_i(x;\vartheta_t) - g_i(x;\psi_0)\big)^2. \tag{5.2}$$

Gaussian processes and infinite width. In our analysis, we will frequently use the framework of GPs (Rasmussen and Williams, 2006) to model distributions over random functions. A univariate GP defines a distribution over functions

$f^0 \sim \mathcal{GP}(\mu^0, \Sigma^0)$ characterized by a mean function $\mu^0 : \mathbb{R}^{d_{\text{in}}} \to \mathbb{R}$ and a covariance (kernel) function $\Sigma^0 : \mathbb{R}^{d_{\text{in}}} \times \mathbb{R}^{d_{\text{in}}} \to \mathbb{R}$ such that $f_0(\mathcal{X}_T)$ follows a multivariate Gaussian distribution $f_0(\mathcal{X}_T) \sim \mathcal{N}(\mu^0(\mathcal{X}_T), \Sigma^0(\mathcal{X}_T, \mathcal{X}_T))$ for any finite set of evaluation points $\mathcal{X}_T = \{x_i^{\text{Test}}\}_{i=1}^{N_T}$. We can condition a prior GP $\mathcal{N}(\mu^0(\mathcal{X}_T), \Sigma^0(\mathcal{X}_T, \mathcal{X}_T))$ on training data $\mathcal{X} = \{x_i\}_{i=1}^{N_D}$ and labels $\mathcal{Y} = \{y_i\}_{i=1}^{N_D}$ to obtain a posterior GP whose *posterior predictive distribution* is Gaussian with mean and covariance given by

$$\mu(\mathcal{X}_T) = \mu^0(\mathcal{X}_T) + \Sigma^0_{\mathcal{X}_T\mathcal{X}}(\Sigma^0_{\mathcal{X}\mathcal{X}})^{-1}\big(\mathcal{Y} - \mu^0(\mathcal{X})\big), \tag{5.3}$$

$$\Sigma_{\mathcal{X}_T\mathcal{X}_T} = \Sigma^0_{\mathcal{X}_T\mathcal{X}_T} - \Sigma^0_{\mathcal{X}_T\mathcal{X}}(\Sigma^0_{\mathcal{X}\mathcal{X}})^{-1}\Sigma^0_{\mathcal{X}\mathcal{X}_T}. \tag{5.4}$$

Our theoretical analysis is situated in the infinite-width limit of neural networks. In this regime, previous work has shown that neural networks at initialization are described by a GP, known as the neural network Gaussian process (NNGP) (Lee et al., 2018a), $f(\mathcal{X}_T; \theta_0) \sim \mathcal{GP}(0, \kappa_{\mathcal{X}_T\mathcal{X}_T})$ with $\kappa_{\mathcal{X}_T\mathcal{X}_T} = \mathbb{E}_{\theta_0}[f(\mathcal{X}_T;\theta_0) f(\mathcal{X}_T;\theta_0)^\top]$ being the NNGP kernel function.

5.3 Equivalence of Random Network Distillation & Deep Ensembles

In this work, we aim to characterize formally the relationship between the error signals as measured by random network distillation and the predictive variance of deep neural network ensembles. Before treating multivariate output dimensions in section 5.3.1, we first consider scalar function outputs for simplicity, i.e. $f, u, g : \mathbb{R}^{d_{\text{in}}} \to \mathbb{R}$. In our analysis, we consider fully connected neural networks $f(x;\theta_t)$ of L layers of width $n_1, \ldots, n_L = n$, parametrized by θ_t at time t. The forward computation of such networks is defined recursively with $z_i^l(x;\theta_t^{\leq l})$ denoting the i-th output of layer l and

$$z_i^l(x, \theta_t^{\leq l}) = \sigma_b b_i^l + \frac{\sigma_w}{\sqrt{n_{l-1}}} \sum_{j=1}^{n_{l-1}} w_{ij}^l x_j^l(x), \quad x_j^l(x) = \phi(z_j^{l-1}(x;\theta_t^{\leq l-1})), \tag{5.5}$$

where $\theta_t^{\leq l}$ denotes the parameters $\{w^1, b^1, \ldots, w^l, b^l\}$ up to layer l, σ_b and σ_w denote scaling parameters of the forward computation, and $\phi : \mathbb{R} \to \mathbb{R}$ is a Lipschitz-continuous nonlinearity. In Eq. (5.5), $n_0 = d_{in}$ and $x^1(x) = x$. The output of a scalar-output neural network is then given by $f(x;\theta_t) = z^L(x;\theta_t^{\leq L})$. In our analysis, we assume that parameters are initialized i.i.d. from a normal distribution[1] $\theta_0 \sim \mathcal{N}(0, I)$).

[1] This forward computation is also known as the neural tangent kernel parametrization and differs from the common settings in that the variance scalings σ_b and σ_w affect both forward and gradient computations. This condition gives well-behaved gradients in the infinite-width limit.

Within this setting, it is our goal to leverage the predictable generalization behavior of neural networks in the limit of infinite-width $n \to \infty$ to characterize the self-predictive errors of random network distillation exactly for any input, allowing us to draw a formal connection to the variance of deep ensembles. Our analysis considers the training dynamics under *gradient flow*, the continuous-time limit of gradient descent $\frac{d}{dt}\theta_t = -\nabla_\theta \mathcal{L}(\theta_t)$, applied to a squared error loss. Under gradient flow with such a squared loss $\mathcal{L}(\theta_t) = \frac{1}{2}\|f(\mathcal{X};\theta_t) - \mathcal{Y}\|_2^2$, the evolution of a neural network (NN) function f is described by a differential equation in function space

$$\frac{\mathrm{d}}{\mathrm{d}t} f(x;\theta_t) = \nabla_\theta^\top f(x;\theta_t) \frac{\mathrm{d}}{\mathrm{d}t}\theta_t = -\nabla_\theta^\top f(x;\theta_t) \nabla_\theta \mathcal{L}(\theta_t) \tag{5.6}$$

$$= -\nabla_\theta^\top f(x;\theta_t) \nabla_\theta f(\mathcal{X};\theta_t)(f(\mathcal{X};\theta_t) - \mathcal{Y}) \tag{5.7}$$

$$\equiv -\Theta_t(x,\mathcal{X})(f(\mathcal{X};\theta_t) - \mathcal{Y}). \tag{5.8}$$

5

The above learning dynamics are governed by a gradient similarity function, called the *neural tangent kernel* (NTK, Jacot et al., 2018), $\Theta_t(x,x') = \nabla_\theta^\top f(x;\theta_t)\nabla_\theta f(x';\theta_t)$, which is itself a dynamic object, resulting in highly nonlinear, generally intractable differential equations.

A remarkable result can however be derived from examining neural networks in the idealized limit of infinite width: 1.) due to regularity effects akin to the law of large numbers, the inner product kernel $\Theta_0(x,x')$ at initialization is deterministic despite the random initialization of neural network parameters θ_0; 2.) the inner product kernel $\Theta_t(x,x')$ remains constant throughout t under gradient flow (Jacot et al., 2018; Lee et al., 2020b). In particular, this means the infinite-width limit yields $\lim_{n\to\infty}\Theta_0(x,x') = \lim_{n\to\infty}\Theta_t(x,x') \equiv \Theta(x,x')$ and leads to significantly simplified dynamics and converting Eq. 5.6 into a linear ordinary differential equation. The now linear ODE 5.6 can be solved analytically, leading to the following characterization of the converged function $f(x;\theta_\infty)$ for $t \to \infty$.

Proposition 5.1. (Jacot et al., 2018)(Post-convergence neural network function) In the limit of infinite layer widths $n \to \infty$ and infinite time $t \to \infty$, the output function of a neural network $f(x;\theta_\infty)$ with NTK parametrization according to Eq. 5.5 is given by

$$f(x;\theta_\infty) = f(x;\theta_0) - \Theta_{x\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}\big(\mathcal{Y} - f(\mathcal{X};\theta_0)\big),$$

where we used the shorthand $\Theta_{xx'} \equiv \Theta(x,x')$.

Proof sketch. By taking the infinite width limit $n \to \infty$, we obtain a linear ODE from Eq. (5.6). Through an exponential ansatz, its explicit solution with initial

condition $f(x;\theta_0)$ is given by $f(x;\theta_t) = f(x;\theta_0) + \Theta_{x\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}(I - e^{-t\Theta_{\mathcal{X}\mathcal{X}}})(\mathcal{Y} - f(\mathcal{X};\theta_0))$. Assuming the training Gram matrix $\Theta_{\mathcal{X}\mathcal{X}}$ is positive definite (and thus invertible), the exponential term decays to zero as $t \to \infty$, yielding the kernel regression formula in Proposition (5.1). See Jacot et al. (2018) and Appendix 5.8.1.

Note that Proposition 5.1 reveals the final network function $f(x;\theta_\infty)$ as a deterministic transformation of the initial random function $f(x;\theta_0)$. This transformation solely depends on the training data $\mathcal{X}$ training labels $\mathcal{Y}$ and the fixed NTK Θ, which depends deterministically on the network architecture and the parameter initialization scheme. Crucially, this characterization of the final NN function $f(x;\theta_\infty)$ permits an analytical description of the generalization behavior of f solely with objects known prior to training. We can furthermore leverage the deterministic dependency of $f(x;\theta_\infty)$ on the random initialization function $f(x;\theta_0)$ to obtain an analytical expression of the distribution of post-convergence functions. As described in Section 5.2, Lee et al. (2018a) show that $f(x;\theta_0)$, in the infinite width limit, follows a specific GP described by the NNGP $f(x;\theta_0) \sim \mathcal{GP}(0,\kappa_{xx'})$ characterized by a covariance function $\kappa_{xx'} = \kappa(x,x')$, leading to the post-convergence distribution described in Proposition 5.2.

Proposition 5.2. (Lee et al., 2020b)(Distribution of post-convergence neural network functions) Let $f(\mathcal{X}_T;\theta_\infty)$ be the converged output function of a NN on a set of testpoints $\mathcal{X}_T$ under the conditions of Proposition 5.1. The distribution of post-convergence NN functions over random initializations $\theta_0 \sim \mathcal{N}(0,I)$ is Gaussian with mean and covariance given by

$$\begin{aligned}\mathbb{E}[f(\mathcal{X}_T,\theta_\infty)] &= \Theta_{\mathcal{X}_T\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}\mathcal{Y},\\ \Sigma^f_{\mathcal{X}_T\mathcal{X}_T}(\theta_\infty) &= \kappa_{\mathcal{X}_T\mathcal{X}_T} + \Theta_{\mathcal{X}_T\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}\kappa_{\mathcal{X}\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}\Theta_{\mathcal{X}\mathcal{X}_T}\\ &\quad - \left(\Theta_{\mathcal{X}_T\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}\kappa_{\mathcal{X}\mathcal{X}_T} + h.c.\right),\end{aligned}$$

where h.c. refers to the Hermitian conjugate of the preceding term.

Proof sketch. We use the fact that $f(x;\theta_\infty)$ can be written as a linear combination of the test initialization $f(x;\theta_0)$ and the training initialization $f(\mathcal{X};\theta_0)$. Both these identities are described probabilistically by the NNGP $f(x;\theta_0) \sim \mathcal{GP}(0,\kappa_{xx'})$, and $f(\mathcal{X};\theta_0) \sim \mathcal{GP}(0,\kappa_{\mathcal{X}\mathcal{X}})$. Applying a linear transformation to a GP yields another GP (Rasmussen and Williams, 2006), meaning $f(x;\theta_\infty)$ also follows a GP. Propagating the prior covariance κ through the linear transformation described by Proposition 5.1 reveals the expression for the post-convergence covariance function $\Sigma^f_{\mathcal{X}_T\mathcal{X}_T}(\theta_\infty)$ given in Proposition 5.2. See also Appendix 5.8.1 or Lee et al. (2020b).

Note that the variance of the distribution of post-convergence functions $f(x;\theta_\infty)$ as described in Proposition 5.2 at any test point x, $\mathbb{V}[f(x;\theta_\infty)] =$

$\Sigma^{f}_{xx}(\theta_\infty)$, represents the predictive variance of an infinite ensemble of infinitely wide neural networks from independently drawn random initializations and trained to convergence on data $(\mathcal{X}, \mathcal{Y})$.

With this probabilistic understanding of idealized deep ensembles and their predictive variance for arbitrary inputs x, we now aim to draw analogous conclusions regarding the self-predictive errors $\epsilon(x;\vartheta_\infty,\psi_0)$ of a converged RND model in the limit of infinite network width. This setup involves training a predictor $u(x;\vartheta_t)$ to match a fixed random target function $g(x;\psi_0)$. Intuitively, the expected errors ought to vanish for training points in $\mathcal{X}$ and remain non-zero elsewhere[2], inheriting the randomness and generalization behaviors of the functions u and g. Using an analogous derivation as in Proposition 5.2, we can now formalize this intuition in the proposition below.

Proposition 5.3. (Distribution of post-convergence RND errors) Under the conditions of Proposition 5.1, let $u(x;\vartheta_\infty)$ be a converged predictor network trained on data $\mathcal{X}$ with labels from a fixed target network $g(\mathcal{X};\psi_0)$. Initialization parameters ϑ_0, ψ_0 are drawn i.i.d. $\vartheta_0, \psi_0 \sim \mathcal{N}(0, I)$ resulting in NNGPs $u(x;\vartheta_0) \sim \mathcal{GP}(0, \kappa^u(x,x'))$ and $g(x;\psi_0) \sim \mathcal{GP}(0, \kappa^g(x,x'))$. The RND error at convergence $\epsilon(\mathcal{X}_T;\vartheta_\infty,\psi_0)$ is Gaussian with zero mean and covariance

$$\begin{aligned}\mathbb{E}[\epsilon(\mathcal{X}_T,\vartheta_\infty,\psi_0)] &= 0\,,\\ \Sigma^{\epsilon}_{\mathcal{X}_T\mathcal{X}_T}(\vartheta_\infty,\psi_0) &= \kappa^{\epsilon}_{\mathcal{X}_T\mathcal{X}_T} + \Theta_{\mathcal{X}_T\mathcal{X}}\Theta^{-1}_{\mathcal{X}\mathcal{X}}\kappa^{\epsilon}_{\mathcal{X}\mathcal{X}}\Theta^{-1}_{\mathcal{X}\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}_T}\\ &\quad - \left(\Theta_{\mathcal{X}_T\mathcal{X}}\Theta^{-1}_{\mathcal{X}\mathcal{X}}\kappa^{\epsilon}_{\mathcal{X}\mathcal{X}_T} + h.c.\right),\end{aligned}$$

where $\kappa^{\epsilon}_{xx'} = \kappa^{u}_{xx'} + \kappa^{g}_{xx'}$ is the covariance kernel of the initialization errors $\epsilon(x;\vartheta_0,\psi_0) = u(x;\vartheta_0) - g(x;\psi_0)$.

Proof sketch. Consider the error function $u(x;\vartheta_\infty) - g(x;\psi_0)$, a sum of the random post-convergence function $u(x;\vartheta_\infty)$ and the fixed random target function $g(x;\psi_0)$. By the same argument as for Proposition 5.1, this error function is a linear transformation of its initialization $u(x;\vartheta_0) - g(x;\psi_0)$, which is a sum of two independent NNGPs. We can continue following analogous arguments for Proposition 5.2 to arrive at the conclusion that the converged error function $u(x;\vartheta_\infty) - g(x;\psi_0)$ itself is a GP with zero-mean and covariance with an altered prior NNGP kernel $\kappa^{\epsilon}(x,x')$ composed of the online prior kernel $\kappa^{u}_{xx'}$ and the target prior kernel $\kappa^{g}_{xx'}$. See also Appendix 5.8.1.

Corollary 5.4. (Equivalence in expectation between RND errors and ensemble variance) Under the conditions of Proposition 5.3, let $\epsilon(x;\vartheta_\infty,\psi_0)$ be the converged RND

[2] This is assuming no true invariances are encoded in the network architecture, as is the case in the fully connected feedforward networks considered here.

error as defined in Eq. (5.2). *Moreover, let* $\mathbb{V}[f(x;\theta_\infty)]$ *denote the variance of converged NN functions over initializations* θ_0. *For an architectural equivalence between* f, u, *and* g *and i.i.d. parameter initialization* $\theta_0, \vartheta_0, \psi_0 \sim \mathcal{N}(0, I)$, *the expected norm of the RND error* $\epsilon^2(x;\vartheta_\infty,\psi_0)$ *coincides with the ensemble variance*

$$\mathbb{E}_{\vartheta_0,\psi_0}\left[\epsilon^2(x;\vartheta_\infty,\psi_0)\right] = \mathbb{V}_{\theta_0}[f(x;\theta_\infty)] \tag{5.9}$$

Proof sketch. Corollary 5.4 follows straighforwardly from Proposition 5.3 by using $\kappa^u(x,x') = \kappa^g(x,x')$. Taking the trace of the covariance matrix and dividing by 2, we recover the predictive ensemble variance $\mathbb{V}_{\theta_0}[f(x;\theta_\infty)]$.

Corollary 5.4 formally shows that, for an architectural equivalence between ensemble, predictor and target network, the expected RND errors directly quantify the predictive variance of the corresponding infinite ensemble model described by Proposition 5.2. To the best of our knowledge, it is the first formal analysis of random network distillation in the NTK regime and reveals a first theoretical motivation for the popular algorithm: in the idealized infinite-width setting, *expected RND errors exactly quantify the variance of deep ensembles for any input* x.

5.3.1 Multi-Headed Random Network Distillation

So far, our analysis has considered the *average* behavior of single-output networks for simplicity. While insightful in its own right, this setting does not reflect most common practical implementations of random network distillation and instead, if taken literally, would imply an ensemble of random network distillation models. To connect with common practical implementations that typically use multi-headed architectures for enhanced reliability and efficiency, we aim to understand the basic probabilistic relation between different function outputs $f_i(x;\theta_t)$ and $f_j(x';\theta_t)$ of a NN with shared hidden layers in the infinite-width limit. The result below identifies this relationship as a statistical independence between the different *random* network outputs $f_i(x;\theta_t)$ and $f_j(x';\theta_t)$ for any time t during gradient flow optimization.

Proposition 5.5. *(Independence of NN functions) Under the conditions of Proposition 5.1, the random output functions* $f_i(x;\theta_t)$ *of a NN with* K *output dimensions and shared hidden layers are mutually independent with covariance*

$$\Sigma^{ij}_{xx'}(\theta_t) = \mathbb{E}[f_i(x;\theta_t) f_j(x';\theta_t)] = \begin{cases} \Sigma^f_{xx'}(\theta_t) & \text{if } i = j, \\ 0 & \text{if } i \neq j, \end{cases}$$

on the interval $t \in [0,\infty)$.

Proof sketch. The property follows from known results that state the independence between output dimensions of the NNGP kernel κ and the NTK Θ (Arora et al., 2019; Jacot et al., 2018; Lee et al., 2018a). For both kernel functions, the proof proceeds by induction, where the independence property between output dimensions is propagated layer-wise. The induction start is equal for both kernels, where first layer outputs, as well as gradients are linear transformations of the Gaussian first-layer weights. Both the NNGP and NTK permit a recursive formulation, through which the independence property can be propagated layer-wise, constituting the induction step. Combined with the learning dynamics of wide NNs, we can conclude that the individual function outputs of a multi-headed NN, too, are statistically independent for any time t on the interval $[0, \infty)$. See Appendix 5.8.1 or Lee et al. (2018a) and Jacot et al. (2018).

Notably, this decoupling holds despite the shared hidden layers and is an artifact of the learning dynamics exhibited in the infinite width limit and the NTK regime. In the absence of feature learning in this regime, output functions become statistically independent despite sharing a network body. By virtue of this independence property, we can translate the earlier single-function results regarding the distribution of RND errors (Proposition 5.3 and Corollary 5.4) to the multi-headed setting. Our first main result then establishes an equivalence between the errors of the multi-headed RND algorithm, a widely used architecture in practice, and the variance of a finite-sized deep ensemble.

Theorem 5.6. (Distributional equivalence between multi-headed RND and finite deep ensembles) Under the conditions of Proposition 5.1, let $u_i(x;\vartheta_\infty), g_i(x;\psi_0)$ be the i-th output of predictor and target networks respectively with K output dimensions. Denote their sample mean RND error $\bar{\epsilon}^2(x;\vartheta_\infty,\psi_0) = \frac{1}{K}\sum_{i=1}^{K}\epsilon_i^2(x;\vartheta_\infty,\psi_0)$. Moreover, let $\{f(x;\theta_\infty^i)\}_{i=1}^{K+1}$ be an ensemble of $K+1$ independently initialized NNs. Denote its sample variance $\bar{\sigma}_f^2(x;\theta_\infty^{i\ldots K+1}) = \frac{1}{K}\sum_{i=1}^{K+1}(f(x;\theta_\infty^i) - \frac{1}{K+1}\sum_{j=1}^{K+1} f(x;\theta_\infty^j))^2$. We have that

$$\frac{1}{2}\bar{\epsilon}^2(x;\vartheta_\infty,\psi_0) \overset{D}{=} \bar{\sigma}_f^2(x;\theta_\infty^{i\ldots K+1}), \tag{5.10}$$

where $\overset{D}{=}$ indicates an equality in distribution, namely by a scaled Chi-squared distribution $\bar{\sigma}_f^2(x;\theta_\infty^{i\ldots K+1}) \sim \frac{\Sigma_{xx}^f(\theta_\infty)}{K}\chi^2(K)$ with scale $\Sigma_{xx}^f(\theta_\infty)$ given by the analytical variance as given in Proposition 5.2.

Proof sketch. By Proposition 5.5, the function heads $\{u_i(x;\vartheta_\infty)\}_{i=1}^{K}$ are K independent predictors, each trained to match their independent targets $g_i(x;\psi_0)$.

Thus, the errors $\{\epsilon_i(x;\vartheta_\infty,\psi_0)\}_{i=1}^K$ are i.i.d. samples from the error distribution outlined in Proposition 5.4. In particular, $\bar{\epsilon}^2$ is the empirical mean of i.i.d. samples from a Gaussian which is known to be Chi-squared distributed. Similarly, we have that the ensemble $\{f(x;\theta_\infty^i)\}_{i=1}^{K+1}$ are $K+1$ i.i.d. samples from the GP defined in Proposition 5.2, again yielding the known Chi-squared distribution for its sample variance $\bar{\sigma}_f^2(x;\theta_\infty^{i\ldots K+1})$. See Appendix 5.8.1.

Theorem 5.6 establishes an equality in distribution between the empirical error of a multi-headed RND architecture and the empirical variance of a finite ensemble of neural networks in the limit of infinite width. Our result, to the best of our knowledge, is the first to formalize this equivalence and provides a theoretical motivation for the use of RND and its common multi-headed architecture as an uncertainty quantification technique.

In a broader sense, we believe this analysis is insightful to many practitioners using random network distillation by establishing a strong link between theory and practice. It is, to the best of our knowledge, the first result that connects the self-predictive errors of RND in its common multi-headed architecture to the predictive variance of finite deep ensembles. Still, the NTK-based perspective applies to an inherently idealized regime and naturally opens up new avenues for investigation. Understanding the relationship between RND networks and deep ensembles at finite width, where feature learning impacts behavior, remains a critical open question beyond the scope of our current framework. Yet, intriguing possibilities also arise within the infinite-width setting itself: Could the properties of the RND target network be deliberately chosen or modified? Exploring different target initializations offers a computationally inexpensive lever to shape the uncertainty signal captured by RND. Indeed, pursuing this very direction, the next section investigates how a specific adaptation of the RND target network allows us to establish a direct correspondence not just with ensemble variance, but with the principled uncertainty quantification provided by Bayesian posterior inference.

5.4 Equivalence of Random Network Distillation and Bayesian Posteriors

Having formulated an equivalence between standard random network distillation and deep ensemble variance, we now proceed to investigate how theoretical connections to the Bayesian inference framework can be established by invoking deliberate changes to the standard random network distillation algorithm, namely by modifying the fixed target function g. Our goal is to show that the RND error signal itself can, under specific conditions, be interpreted as a draw from a Bayesian posterior predictive distribution.

To this end, we briefly recall Bayesian inference with the classical Gaussian

linear model. We define a regression model as $f(x;\theta) = \phi(x)^\top\theta$ with a feature mapping $\phi : \mathbb{R}^{d_{\text{in}}} \to \mathbb{R}^{d_P}$, and a prior distribution over the parameters $p(\theta) \sim \mathcal{N}(0,\Sigma^0)$. The prior distribution $p(\theta)$ implicitly defines a GP prior $f^0(x;\theta) \sim \mathcal{GP}(0,\phi(x)^\top\Sigma^0\phi(x'))$, with the prior kernel $K_{xx'} = \phi(x)^\top\Sigma^0\phi(x')$. Within this linear model[3], we look to infer a posterior distribution over functions given observations $\mathcal{X} = \{x_i \in \mathbb{R}^{d_{\text{in}}}\}_{i=1}^{N_D}$ and labels $\mathcal{Y} = \{y_i \in \mathbb{R}\}_{i=1}^{N_D}$. Owing to our prior choice, this can be done by simply conditioning the joint Gaussian predictions of the regression model $f(x;\theta)$ on the data $\mathcal{X},\mathcal{Y}$, a simple probabilistic operation in the case of Gaussian random variables. We obtain a conditional *posterior predictive* distribution over functions as

$$p(f|x,\mathcal{X},\mathcal{Y}) \sim \mathcal{N}\big(K_{x\mathcal{X}}K_{\mathcal{X}\mathcal{X}}^{-1}\mathcal{Y},\, K_{xx} - K_{x\mathcal{X}}K_{\mathcal{X}\mathcal{X}}^{-1}K_{\mathcal{X}x}\big). \tag{5.11}$$

We can contrast this result with the GP governing the distribution of converged NN functions in Proposition 5.2 to see a disparity in the functional structure of the covariance functions. While Proposition 5.2 and Proposition 5.3, too, specify GPs, they do not permit an interpretation as a Bayesian posterior predictive distribution (Lee et al., 2020b) due to the presence of two (in general) distinct kernel functions, namely the NNGP kernel κ and the NTK Θ. However, inspection of Proposition (5.3) and Eq. (5.11) suggests a path: if the prior kernel components within $\Sigma^\epsilon_{xx'}(\vartheta_\infty,\psi_0)$, namely $\kappa^\epsilon_{xx'}$, could be aligned with the dynamics kernel $\Theta_{xx'}$ (i.e., if $\kappa^\epsilon \propto \Theta$), then the resulting covariance structure simplifies to the desired Bayesian posterior form of

$$f(x;\theta_\infty) \sim \mathcal{N}\big(\Theta_{x\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}\mathcal{Y},\quad \Theta_{xx} - \Theta_{x\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}\Theta_{\mathcal{X}x}\big). \tag{5.12}$$

An important insight here is that Eq. 5.12 now is the *exact Bayesian posterior predictive distribution of a neural network in the infinite width limit*, which corresponds to a kernel regression model with the NTK as a GP prior $\mathcal{GP}(0,\Theta_{xx'})$ and conditioned on the data $(\mathcal{X},\mathcal{Y})$.

The idea of aligning the prior and dynamic kernels has been previously explored by He et al. (2020) to construct *Bayesian ensembles* where the predictive distribution of the ensemble matches the posterior predictive distribution of the *NTK-GP*. We propose that a similar alignment can be achieved in the RND framework by constructing the target function $g(x;\psi_0)$ to assume a specific form. The idea is to design a target $\tilde{g}(x;\vartheta_0,\psi_0)$ such that when a predictor $u(x;\vartheta_0)$ is trained to match it, the resulting "Bayesian" error distribution $\epsilon^b(x;\vartheta_\infty,\vartheta_0,\psi_0) = u(x;\vartheta_\infty) - \tilde{g}(x;\vartheta_0,\psi_0)$ behaves like a draw from the posterior of a Bayesian model whose prior kernel is the NTK $\Theta_{xx'}$ itself[4].

[3] We use a noise-free regression model for ease of notation here, but extensions to the noisy case by including an observation noise term $\sigma_n^2 I$ in the kernel matrix inversions (cf. Eq. (5.11)-(5.12)) are straightforward.

[4] The newly constructed target function $\tilde{g}(x;\vartheta_0,\psi_0)$ uses both ϑ_0 and ψ_0 for reasons that will become clear in the remainder of section.

In the random network distillation algorithm, the prior kernel $\kappa^{\epsilon^b}_{xx'}$ of initialization errors $\epsilon^b(x;\vartheta_0,\vartheta_0,\psi_0) = u(x;\vartheta_0) - \tilde{g}(x;\vartheta_0,\psi_0)$ is given by the sum of the online prior kernel and the target prior kernel $\kappa^{\epsilon^b}_{xx'} = \kappa^u_{xx'} + \kappa^{\tilde{g}}_{xx'}$ (cf. Proposition 5.3), provided that u and $\tilde{g}$ follow independent GPs. To obtain an error prior kernel that aligns with the NTK such that $\kappa^{\epsilon^b}_{xx'} = \Theta_{xx'}$, one may thus construct the target prior such that it satisfies $\kappa^{\tilde{g}}_{xx'} = \Theta_{xx'} - \kappa^u_{xx'}$. To this end, a closer inspection of the relation between the NNGP kernel $\kappa^u_{xx'}$ and the NTK $\Theta_{xx'}$ is instructive. For this purpose, we will view the online network $u(x;\vartheta_0)$ as a random feature model with its forward computation path as described in Eq. 5.5. Let in this scenario $x^L(x)$ denote the output vector, or the post-activations, before the final linear layer and denote the last-layer parameters at initialization $t = 0$ as (w^L, b^L). We can write the NN output at initialization $u(x;\vartheta_0)$ as

$$u(x;\vartheta_0) = \sigma_b b^L + \frac{\sigma_w}{\sqrt{n_{L-1}}} \sum_{i=1}^{n_{L-1}} w_i^L x_i^L(x), \tag{5.13}$$

that is, as a simple linear model of the random final post-activations $x^L(x)$. Viewing the function in Eq. (5.13) as a random feature model leads to a crucial insight: since the last-layer weights and biases (w^L, b^L) are assumed to be initialized i.i.d. from a standard normal $(w^L, b^L) \sim \mathcal{N}(0, I)$, Eq. (5.13) describes a (random) affine transformation of a Gaussian vector[5] whose covariance in the limit $n \to \infty$ is quantified by the NNGP kernel $\kappa^u_{xx'}$ given by

$$\kappa^u_{xx'} = \mathbb{E}[u(x;\vartheta_0)u(x';\vartheta_0)] = \sigma_b^2 + \sigma_w^2 \mathbb{E}[x_i^L(x)x_i^L(x')]. \tag{5.14}$$

Let us now compare this expression for the prior kernel $\kappa^u_{xx'}$ of the online network with its dynamics kernel $\Theta_{xx'}$. In particular, we will split the dynamics kernel $\Theta_{xx'}$ into a last-layer component

$$\Theta^L_{xx'} = \nabla^\top_{\{w^L,b^L\}} u(x;\vartheta_0) \nabla_{\{w^L,b^L\}} u(x';\vartheta_0)$$

and a component summarizing all preceding parameters

$$\Theta^{\leq L-1}_{xx'} = \nabla^\top_{\vartheta^{\leq L-1}} u(x;\vartheta_0) \nabla_{\vartheta^{\leq L-1}} u(x';\vartheta_0)$$

[5] To see the correspondence in Eq. 5.14, first notice that due to the i.i.d. initialization of (w^L, b^L), any cross-products (e.g., involving elements indexed with $i \neq j$) vanish in the expectation $\mathbb{E}[u(x;\vartheta_0)u(x';\vartheta_0)]$. The expectation thus becomes $\mathbb{E}[u(x;\vartheta_0)u(x';\vartheta_0)] = \mathbb{E}_{w^{\leq L},b^{\leq L}}[\sigma_b^2 + \frac{\sigma_w^2}{n_{L-1}} \sum_{i=1}^{n_{L-1}} x_i^L(x)x_i^L(x')]$. By linearity, the expectation on the r.h.s. can be pulled inside the sum and by symmetry we have that $\mathbb{E}_{w^{\leq L},b^{\leq L}}[x_i^L(x)x_i^L(x')]$ is independent of i, s.t. $\mathbb{E}_{w^{\leq L},b^{\leq L}}[\frac{\sigma_w^2}{n_{L-1}} \sum_{i=1}^{n_{L-1}} x_i^L(x)x_i^L(x')] = \sigma_w^2 \mathbb{E}[x_i^L(x)x_i^L(x')]$.

such that $\Theta_{xx'} = \Theta^L_{xx'} + \Theta^{\leq L-1}_{xx'}$. Since $u(x;\vartheta_0)$ is linear in the last-layer parameters $\{w^L, b^L\}$ (cf. Eq. 5.13), we make the crucial observation that the last-layer NTK component $\Theta^L_{xx'}$ equals the NNGP prior kernel[6] $\Theta^L_{xx'} = \kappa^u_{xx'}$. This property gives a clear instruction for engineering the prior kernel of the target network: by constructing $\kappa^{\tilde{g}}_{xx'}$ such that $\kappa^{\tilde{g}}_{xx'} = \Theta^{\leq L-1}_{xx'}$ and independently from $\kappa^u_{xx'}$, we obtain an error prior as

$$\kappa^{\epsilon^b}_{xx'} = \kappa^{\tilde{g}}_{xx'} + \kappa^u_{xx'} = \Theta^L_{xx'} + \Theta^{\leq L-1}_{xx'} = \Theta_{xx'} \,. \tag{5.15}$$

In the following, we will thus aim to construct a target function $\tilde{g}(x;\vartheta_0,\psi_0)$ with the desired property $\kappa^{\tilde{g}}_{xx'} = \Theta^{\leq L-1}_{xx'}$, in particular by modeling $\tilde{g}$ as a linear function in the feature space corresponding to gradients in earlier layers. This approach has also previously been demonstrated by He et al. (2020) to develop Bayesian ensembles.

5

Proposition 5.7. (Bayesian RND target function) Under the conditions of Proposition 5.1, let $u(x;\vartheta_0)$ and $g(x;\psi_0)$ be neural networks of L layers with parameters $\vartheta_0, \psi_0 \sim \mathcal{N}(0, I)$ i.i.d. Moreover, let $\psi^L_0 = \{w^L, b^L\}$ denote the last-layer parameters of ψ_0 and $\psi^{\leq L-1}_0$ the parameters of all preceding layers. Suppose the target function $\tilde{g}(x;\vartheta_0,\psi_0)$ is given by

$$\tilde{g}(x;\vartheta_0,\psi_0) = \nabla^\top_{\vartheta_0} u(x;\vartheta_0)\psi^*_0 \,,$$

where $\psi^*_0 = \{\psi^{\leq L-1}_0, 0_{dim(\psi^L_0)}\}$ is a copy of ψ_0 with its last-layer weights set to 0. In the infinite width limit $n \to \infty$, $\tilde{g}(x;\vartheta_0,\psi_0)$ distributes by construction as $\tilde{g}(x;\vartheta_0,\psi_0) \sim \mathcal{GP}(0, \kappa^{\tilde{g}}_{xx'})$ where $\kappa^{\tilde{g}}_{xx'} = \Theta^{\leq L-1}_{xx'}$.

Proof sketch. The function $\tilde{g}(x;\vartheta_0,\psi_0)$ is by construction equivalent to a linear function with the (random) feature map $\nabla_{\vartheta^{\leq L-1}_0} u(x;\vartheta_0)$ given by the gradient of parameters in the pre-final layers and with a parameter vector $\psi^{\leq L-1}_0$. Conditioned on ϑ_0, the random function $\tilde{g}(x;\vartheta_0,\psi_0)$ is thus an affine transformation of the Gaussian vector $\psi^{\leq L-1}_0$ and thus a GP itself, at any width n. Using the central results by Jacot et al. (2018) that $\Theta_{0,xx'} \to \Theta_{xx'}$ as $n \to \infty$ and appealing to the bounded convergence theorem, the limiting distribution of the *unconditioned* random function $\tilde{g}(x;\vartheta_0,\psi_0)$, too, becomes Gaussian with the deterministic covariance $\Theta^{\leq L-1}_{xx'}$.

[6] To see this correspondence, notice that the last-layer gradient inner product $\nabla^\top_{\{w^L,b^L\}} u(x;\vartheta_0) \nabla_{\{w^L,b^L\}} u(x';\vartheta_0)$ reduces to the sum $\sigma_b^2 + \frac{\sigma_w^2}{n_{L-1}} \sum_{i=1}^{n_{L-1}} x^L_i(x) x^L_i(x')$, where the r.h.s. sum tends to its expectation in the limit $n_{L-1} \to \infty$ given that summands are identically distributed (as before by symmetry) and independent (which is shown more rigorously for example in Sec. 5.8.1).

While the specific form of the kernel $\Theta^{\leq L-1}_{xx'} = \Theta_{xx'} - \Theta^{L}_{xx'}$ seems unusual as a standalone prior, it is crucially important in shaping the final error distribution. This is because with the altered "Bayesian" target function $\tilde{g}(x;\vartheta_0,\psi_0)$ we can shape the covariance structure of errors at initialization by satisfying Eq. 5.15, appealing to Proposition (5.3). With the engineered target function $\tilde{g}(x;\vartheta_0,\psi_0)$, the learning dynamics of an RND model where the predictor network $u(x;\vartheta_t)$ learns to mimic $\tilde{g}(\mathcal{X};\vartheta_0,\psi_0)$ can be shaped in the desired way. Our central statement is that the distribution of the error between the converged predictor $u(x;\vartheta_\infty)$ and the target function $\tilde{g}(x;\vartheta_0,\psi_0)$ will then no longer reflect the variance of deep ensembles trained with gradient descent, but will instead directly embody the characteristics of a Bayesian posterior predictive distribution derived from the NTK-GP prior. Theorem 5.8 formalizes this result.

Theorem 5.8. *(Distribution of Bayesian RND errors) Under the conditions of Proposition 5.1, let $u(x;\vartheta_\infty)$ be a converged predictor network trained on data $\mathcal{X}$ with labels from the fixed target function $\tilde{g}(\mathcal{X};\vartheta_0,\psi_0)$ as defined in Proposition 5.7. Let parameters ϑ_0,ψ_0 be drawn i.i.d. $\vartheta_0,\psi_0 \sim \mathcal{N}(0,I)$. The post-convergence Bayesian RND error $\epsilon^b(\mathcal{X}_T;\vartheta_\infty,\vartheta_0,\psi_0) = u(\mathcal{X}_T;\vartheta_\infty) - \tilde{g}(\mathcal{X}_T;\vartheta_0,\psi_0)$ on a test set $\mathcal{X}_T$ is Gaussian with zero mean and covariance*

$$\Sigma^{\epsilon^b}_{\mathcal{X}_T\mathcal{X}_T}(\vartheta_\infty,\vartheta_0,\psi_0) = \Theta_{\mathcal{X}_T\mathcal{X}_T} - \Theta_{\mathcal{X}_T\mathcal{X}}\Theta^{-1}_{\mathcal{X}\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}_T},$$

and thus recovers the covariance of the exact Bayesian posterior predictive distribution of an infinitely wide neural network with the corresponding NTK $\Theta_{xx'}$.

Proof sketch. The result follows by combining Proposition 5.3 and Proposition 5.7, provided that the GP governing the predictor initialization $\kappa^{u}_{xx'}$ and the target function $\kappa^{\tilde{g}}_{xx'}$ are independent. Owing to the fact that the parameters ϑ_0 and ψ_0 are drawn independently, the independence between $u(x;\vartheta_0)$ and $\tilde{g}(x;\vartheta_0,\psi_0)$ is apparent by rewriting the covariance $\mathbb{E}[u(x;\vartheta_0)\tilde{g}(x;\vartheta_0,\psi_0)]$ in terms of conditional expectations on ϑ_0 by the law of total expectation. Furthermore, since $\Theta_{xx'} = \Theta^{L}_{xx'} + \Theta^{\leq L-1}_{xx'}$ and $\kappa^{\tilde{g}}_{xx'} = \Theta^{\leq L-1}_{xx'}$, $\kappa^{u}_{xx'} = \Theta^{L}_{xx'}$, we have that $\kappa^{\epsilon^b}_{xx'} = \Theta_{xx'}$. In other words, the GP kernel of initial errors aligns with the NTK of the online predictor, such that the distribution of post-convergence errors in Proposition 5.3 simplifies significantly. This same covariance function indeed also defines the posterior predictive distribution of infinitely wide neural networks as described by the GP with prior $\mathcal{GP}(0,\Theta_{xx'})$ and conditioned on $(\mathcal{X},\mathcal{Y})$.

Theorem 5.8 shows that with a carefully engineered target function, the RND error signal $\epsilon^b(x;\vartheta_\infty,\vartheta_0,\psi_0) = u(x;\vartheta_\infty) - \tilde{g}(x;\vartheta_0,\psi_0)$ is no longer just related to ensemble variance, but rather becomes a direct sample from the centered posterior predictive distribution of a Bayesian model whose prior ker-

nel is the NTK itself. This novel result provides a direct bridge between RND and Bayesian inference in the limit of infinite network width, providing a useful insight: the error signal generated by this modified RND procedure is not merely a heuristic measure of distance, but is itself a random draw from the (centered) Bayesian posterior predictive distribution of an NTK-based GP. This direct distributional equivalence has immediate practical implications, for example prescribing rather straightforwardly how this Bayesian form of RND can be used for exact posterior sampling. By applying Proposition 5.5 to the multi-headed Bayesian RND architecture[7], in contrast to obtaining samples from deep ensembles as done in Theorem 5.6, we now obtain several independent samples from the centered posterior predictive distribution through $\epsilon_i^b(x;\vartheta_\infty,\vartheta_0,\psi_0) = u_i(x;\vartheta_\infty) - \tilde{g}_i(x;\vartheta_0,\psi_0)$. The below corollary details how this can be leveraged to conduct a posterior sampling procedure, requiring access only to a mean estimate and a single Bayesian RND model.

5

Corollary 5.9 (Posterior Sampling via Bayesian RND). *Let* $\mathcal{N}\left(\mu^b(x), \Sigma^b_{xx'}\right)$ *be the posterior predictive distribution of an infinitely wide neural network conditioned on* x *with mean* $\mu^b(x) = \Theta_{x\mathcal{X}}\Theta^{-1}_{\mathcal{X}\mathcal{X}}\mathcal{Y}$ *and covariance* $\Sigma^b_{xx'} = \Theta_{xx'} - \Theta_{x\mathcal{X}}\Theta^{-1}_{\mathcal{X}\mathcal{X}}\Theta_{\mathcal{X}x'}$. *Suppose* $\tilde{\mu}(x;\theta_\infty) \approx \mu^b(x)$ *is an estimate of the mean function and let* $\{\epsilon_i^b(x;\vartheta_\infty,\vartheta_0,\psi_0)\}_{i=1}^K$ *be error functions of a* K*-head Bayesian RND model as defined in Theorem 5.8.*
The following procedure generates (at most K*) independent samples from the conditional posterior predictive distribution* $\mathcal{N}\left(\mu^b(x), \Sigma^b_{xx'}\right)$*:*

1. *sample* $i \sim \mathcal{U}[1,K]$
2. *compute* $\tilde{\mu}_i(x) = \tilde{\mu}(x;\theta_\infty) + \epsilon_i^b(x;\vartheta_\infty,\vartheta_0,\psi_0)$
3. $\tilde{\mu}_i(x)$ *is an i.i.d. sample from the conditional posterior predictive distribution* $\mathcal{N}\left(\mu^b(x), \Sigma^b_{xx'}\right)$

Proof sketch. The result follows directly from Theorem (5.8) and application of the independence argument of Proposition (5.5) to the multi-headed setting.

Corollary 5.9 shows that given an estimator of the posterior predictive mean, a modified Bayesian RND setup can be used perform direct posterior sampling that faithfully represents Bayesian posterior uncertainty in the NTK limit. This offers a pathway to performing exact Bayesian inference through

[7] In a multi-headed architecture, the Bayesian target function described in Proposition 5.8 becomes a JVP. Several common machine learning libraries (e.g., JAX (Bradbury et al., 2018) offer dedicated algorithms to compute such JVPs efficiently.

the lens of network distillation, provided that the target and predictor networks initializations are carefully managed.

This completes our theoretical development, first showing an equivalence of RND in the NTK regime to ensemble variance and now, through specific modifications to its target function, to the generation of independent samples from exact Bayesian posterior predictive distributions.

5.5 Related Work

A substantial body of research studies the analytical learning dynamics of deep learning, particularly in the infinite-width limit. Central to our analysis are seminal works characterizing the NNGP (Lee et al., 2018a) at initialization, the dynamics-governing NTK (Jacot et al., 2018), and the evolution of wide networks as linear models (Lee et al., 2020b). This provides a theoretical framework for analytical descriptions of deep ensembles (Dietterich, 2000; Lakshminarayanan et al., 2017), with subsequent studies using NTK theory to precisely characterize ensemble variances under various conditions, including observation noise (Calvo-Ordoñez et al., 2024; Kobayashi et al., 2022; Yang, 2019). A central line of work for our paper is the connection between deep ensembles and Bayesian inference in infinite-width NTK regime. Notably, He et al. (2020) demonstrate how to construct "Bayesian ensembles", an approach we adapt to construct "Bayesian RND" algorithms. The broader link between deep ensembles and approximations of Bayesian posteriors has been studied extensively (D'Angelo and Fortuin, 2021; Izmailov et al., 2021; Osband et al., 2019). More recently, NTK-based approaches have been used for single-model uncertainty estimation (Zanger et al., 2025a) or ad-hoc uncertainty quantification (Wilson et al., 2025).

While uncertainty quantification has a rich body of literature within reinforcement learning, the application of NTK theory to RL settings is still developing. Several works have leveraged linearized learning dynamics in RL, including in overparameterized settings (Xiao et al., 2021), for neural networks with single or multiple layers (Cai et al., 2019; Wai et al., 2020), to analyze generalization (Lyle et al., 2022), and to derive provably optimistic value functions (Yang et al., 2020). Concurrent work to ours studies the infinite-width limit of an RND-like estimator for value function uncertainty (Zanger et al., 2025b). More broadly, deep ensembles and Bayesian methods are widely used in RL, driving exploration (Chen et al., 2017; Ishfaq et al., 2021; Nikolov et al., 2019; Osband et al., 2016; 2019; Zanger et al., 2024), enabling robust offline and off-policy learning (An et al., 2021; Chen et al., 2021; Lee et al., 2021), and ensuring safety (Hoel et al., 2023; Lee et al., 2022; Lütjens et al., 2019). Our work provides a theoretical basis for RND (Burda et al., 2019b), which belongs to a

class of computationally cheaper, single-model methods whose theoretical underpinnings are typically less understood (Guo et al., 2022; Lahlou et al., 2021; Pathak et al., 2017; Sensoy et al., 2018; Van Amersfoort et al., 2020).

5.6 Limitations and Assumptions

We provide an overview of the primary assumptions underpinning our analysis and discuss their relation to practical settings. The foremost assumption is that our analysis operates within the NTK regime. This framework presupposes the asymptotic limit of infinitely wide neural networks and a so-called NTK-parametrization of forward computations that ensures network dynamics linearize around their initialization, leading to "lazy" learning with kernel regression behavior. This idealized setting naturally deviates from practical implementations involving finite-width networks. Nonetheless, a significant body of work has demonstrated that predictions from NTK theory can remain remarkably accurate for sufficiently wide, modern architectures, providing a reasonable approximation of their behavior (e.g., Lee et al., 2020a; Samarin et al., 2020; Seleznova and Kutyniok, 2022).

Furthermore, our derivations assume training via full-batch gradient flow, which corresponds to gradient descent with an infinitesimal step size. This abstains from the use of stochastic minibatch optimizers, which are standard in practice. While beyond our current scope, extensions of NTK analysis to incorporate the effects of stochastic gradient noise do exist (e.g., Cao and Gu, 2019; Nitanda and Suzuki, 2021; Yang, 2019). Finally, our analysis considers a fixed training dataset $\mathcal{X}$. This contrasts with prominent applications of RND, particularly in online reinforcement learning, where the agent interacts with an environment and learns from an inherently non-stationary data stream. Characterizing how these equivalences with ensembles and Bayesian posteriors evolve under such distribution shifts remains an important open question.

5.7 Discussion

In this work, we have established a novel theoretical understanding of random network distillation (RND) by connecting it to the principled uncertainty frameworks of deep ensembles and Bayesian inference. By analyzing these techniques within the unifying setting of infinitely wide neural networks, we provide a clear analytical interpretation for the empirically successful RND algorithm. Our analysis yields a twofold equivalence: first, we prove that the squared error of standard RND exactly recovers the predictive variance of deep ensembles in the NTK regime. Second, we demonstrate that the RND framework is more versatile; by strategically engineering the RND target function,

the resulting error signal can be made to directly mirror the centered posterior predictive distribution of an NTK-governed GP, that is, the exact posterior predictive distribution of neural networks in the infinite width limit. This "Bayesian RND" variant furthermore allows for posterior sampling procedures that produce i.i.d. samples from this posterior. Our work thereby unifies RND, ensembles, and Bayesian inference under a single theoretical lens from infinite width perspective.

Crucially, our findings hold under the assumptions infinite-width and the NTK regime, a setting where networks effectively linearize and operate as kernel machines with a fixed kernel. This "lazy" training regime, while analytically tractable and predictive for very wide networks, does not capture the phenomenon of feature learning. The degree to which our established equivalences translate to practical, finite-width networks that learn features remains a significant open question.

Conversely, the clear conditions for this theoretical equivalence also suggest a lens from which to approach future research: deviations between RND, ensembles, and Bayesian posteriors in practice must arise from departures from the NTK regime. Characterizing these deviations could lead to novel techniques and a deeper understanding of computationally efficient approaches in Bayesian deep learning, operating well outside the kernelized infinite-width setting. Furthermore, substantial empirical investigation is needed to quantify the gap between our theory and practice, and to understand precisely when and why the uncertainty signals from these methods diverge. Perhaps the most exciting direction, however, is the concept of *target engineering*. *Bayesian RND* is but one example of incorporating diverse prior knowledge into the RND error signal through deliberate target functions. This suggests a path towards creating computationally efficient, single-model uncertainty estimators with the flexibility to encode structured priors, for example attending to task-specific features, already extant neural network models, or symmetric function priors, simply by modifying the target network's structure or initialization. Further research to this end can open paths towards creating computationally efficient, single-model uncertainty estimators with tailored, structured priors, promising more robust and principled applications in fields from active learning to safe reinforcement learning.

5.8 Proofs

This section provides extended proofs for our analysis of RND.

5.8.1 Ensemble Equivalence

Our first result states the equivalence of self-predictive errors of RND and predictive variance of deep ensembles in the infinite-width NTK regime.

Proof of Proposition 5.1

We restate Proposition 5.1 for convenience.

Proposition 5.1. *(Jacot et al., 2018)(Post-convergence neural network function) In the limit of infinite layer widths $n \to \infty$ and infinite time $t \to \infty$, the output function of a neural network $f(x;\theta_\infty)$ with NTK parametrization according to Eq. 5.5 is given by*

$$f(x;\theta_\infty) = f(x;\theta_0) - \Theta_{x\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}\big(\mathcal{Y} - f(\mathcal{X};\theta_0)\big),$$

where we used the shorthand $\Theta_{xx'} \equiv \Theta(x,x')$.

Proof. The proof is centered around the learning dynamics of a neural network under gradient descent, whereby we assume the limit of infinitesimal step size for simplicity. This setting is also referred to as "gradient flow". The driving force behind the learning dynamics of parameters θ_t is gradient flow optimization on the loss

$$\mathcal{L}(\theta_t) = \frac{1}{2}\| f(\mathcal{X},\theta_t) - \mathcal{Y} \|_2^2, \tag{5.16}$$

with the subsequent evolution of parameters by

$$\frac{\mathrm{d}}{\mathrm{d}t}\theta_t = -\alpha \nabla_\theta \mathcal{L}(\theta_t), \tag{5.17}$$

where α is a learning rate. From this, we can obtain the parameter space differential equation

$$\frac{\mathrm{d}}{\mathrm{d}t}\theta_t = -\alpha \nabla_\theta f(\mathcal{X},\theta_t)\big(f(\mathcal{X},\theta_t) - \mathcal{Y}\big). \tag{5.18}$$

In order to translate this expression to a function-space view through a first-order Taylor expansion of f around its initialization parameters θ_0:

$$f_{\mathrm{lin}}(x,\theta_t) = f(x,\theta_0) + \nabla_\theta^\top f(x,\theta_0)(\theta_t - \theta_0). \tag{5.19}$$

The use of a linearized neural network function simplifies the analysis in two aspects: 1.) the linearization offers a simple translation of the parameter space

evolution $\frac{\mathrm{d}}{\mathrm{d}t}\theta_t$ to a function-space evolution and 2.) the linearized neural network function $f_{\text{lin}}(x,\theta_t)$ results in linear dynamics, simplifying the earlier derived differential equation to a linear ODE. The evolution of f_{lin} is then obtained by taking the time-derivative of Eq. (5.19) and plugging in the parameter evolution for a linearized function from Eq. (5.18) such that

$$\frac{\mathrm{d}}{\mathrm{d}t} f_{\text{lin}}(x,\theta_t) = -\alpha \nabla_\theta^\top f(x,\theta_0) \nabla_\theta f(\mathcal{X},\theta_0) \big(f_{\text{lin}}(\mathcal{X},\theta_t) - \mathcal{Y}\big). \tag{5.20}$$

Let us denote the training error of f_{lin} at time t with $\delta_t = f_{\text{lin}}(\mathcal{X},\theta_t) - \mathcal{Y}$ and accordingly write

$$\frac{\mathrm{d}}{\mathrm{d}t}\delta_t = -\alpha \Theta^0_{\mathcal{X}\mathcal{X}} \delta_t, \tag{5.21}$$

where $\Theta^0_{\mathcal{X}\mathcal{X}}$ denotes the empirical tangent kernel at initialization $\Theta^0_{\mathcal{X}\mathcal{X}} = \nabla_\theta^\top f(\mathcal{X},\theta_0) \nabla_\theta f(\mathcal{X},\theta_0)$. The differential equation (5.21) is a linear ODE system to which an exponential ansatz provides the explicit solution

$$\delta_t = e^{-\alpha t \Theta^0_{\mathcal{X}\mathcal{X}}} \delta_0, \tag{5.22}$$

where $e^{\Theta_{\mathcal{X}\mathcal{X}}} = \sum_{k=0}^{\infty} \frac{1}{k!} (\Theta_{\mathcal{X}\mathcal{X}})^K$ is the matrix exponential. We plug this result back in the linearized function space differential equation 5.20 to obtain

$$\frac{\mathrm{d}}{\mathrm{d}t} f_{\text{lin}}(x,\theta_t) = -\alpha \Theta^0_{x\mathcal{X}} e^{-\alpha t \Theta^0_{\mathcal{X}\mathcal{X}}} \big(f(\mathcal{X},\theta_0) - \mathcal{Y}\big). \tag{5.23}$$

In this form, we can solve for $f_{\text{lin}}(x,\theta_t)$ directly by integration

$$f_{\text{lin}}(x,\theta_t) = f(x,\theta_0) + \int_0^t \frac{\mathrm{d}}{\mathrm{d}t'} f_{\text{lin}}(x,\theta_{t'})\,\mathrm{d}t' \tag{5.24}$$

$$= f(x,\theta_0) + \Theta^0_{x\mathcal{X}} (\Theta^0_{\mathcal{X}\mathcal{X}})^{-1} \Big(e^{-\alpha t \Theta^0_{\mathcal{X}\mathcal{X}}} - I\Big) \big(f(\mathcal{X},\theta_0) - \mathcal{Y}\big). \tag{5.25}$$

Remarkably, the linearized and true learning dynamics become increasingly aligned with increasing neural network width. Jacot et al. (2018) and Lee et al. (2020b) show that as network width increases, the required individual movement of parameters $\theta_t - \theta_0$ to effect sufficient movement in the output function $f(x,\theta_t)$ decreases. In the limit of infinite width $n \to \infty$, the linearization of f then becomes exact $\lim_{n\to\infty} f_{\text{lin}}(x,\theta_t) = f(x,\theta_t)$. Under the outlined training dynamics, the same limit furthermore causes the NTK to become deterministic (despite random weight initializations) and stationary $\lim_{n\to\infty} \Theta^0_{xx'} = \Theta^t_{xx'} = \Theta_{xx'}$. Thus, the convergenced function at time $t \to \infty$ is described by

$$f(x,\theta_\infty) = f(x,\theta_0) - \Theta_{x\mathcal{X}} \Theta^{-1}_{\mathcal{X}\mathcal{X}} \big(f(\mathcal{X},\theta_0) - \mathcal{Y}\big). \tag{5.26}$$

□

Proof of Proposition 5.2

We restate Proposition 5.2 for convenience.

Proposition 5.2. (Lee et al., 2020b)(Distribution of post-convergence neural network functions) Let $f(\mathcal{X}_T;\theta_\infty)$ be the converged output function of a NN on a set of testpoints $\mathcal{X}_T$ under the conditions of Proposition 5.1. The distribution of post-convergence NN functions over random initializations $\theta_0 \sim \mathcal{N}(0,I)$ is Gaussian with mean and covariance given by

$$\begin{aligned}\mathbb{E}[f(\mathcal{X}_T,\theta_\infty)] &= \Theta_{\mathcal{X}_T\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}y\,,\\ \Sigma^f_{\mathcal{X}_T\mathcal{X}_T}(\theta_\infty) &= \kappa_{\mathcal{X}_T\mathcal{X}_T} + \Theta_{\mathcal{X}_T\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}\kappa_{\mathcal{X}\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}\Theta_{\mathcal{X}\mathcal{X}_T}\\ &\quad - \left(\Theta_{\mathcal{X}_T\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}\kappa_{\mathcal{X}\mathcal{X}_T} + h.c.\right),\end{aligned}$$

where h.c. refers to the Hermitian conjugate of the preceding term.

Proof. The proof builds on the previous result of Proposition 5.1 providing a closed-form expression for the post-convergence function as a deterministic function of its initialization, here evaluated for a set of test points $\mathcal{X}_T$

$$f(\mathcal{X}_T,\theta_\infty) = f(\mathcal{X}_T,\theta_0) - \Theta_{\mathcal{X}_T\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}\big(f(\mathcal{X},\theta_0) - y\big)\,. \tag{5.27}$$

To be precise, the post-convergence predictions $f(\mathcal{X}_T,\theta_\infty)$ can be written as an affine transformation of the vector $(f(\mathcal{X}_T,\theta_0), f(\mathcal{X},\theta_0)^\top)^\top$. This yields the block matrix equation

$$\begin{pmatrix} f(\mathcal{X}_T,\theta_\infty)\\ f(\mathcal{X},\theta_\infty)\end{pmatrix} = \\ \begin{pmatrix} I & -\Theta(\mathcal{X}_T,\mathcal{X})\Theta(\mathcal{X},\mathcal{X})^{-1}\\ 0 & 0\end{pmatrix}\begin{pmatrix} f(\mathcal{X}_T,\theta_0)\\ f(\mathcal{X},\theta_0)\end{pmatrix} + \begin{pmatrix}\Theta(\mathcal{X}_T,\mathcal{X})\Theta(\mathcal{X},\mathcal{X})^{-1}y\\ y\end{pmatrix}. \tag{5.28}$$

We recall that, at initialization, neural networks in the infinite width limit distribute to a GP called NNGP (Lee et al., 2018a) as

$$f(\mathcal{X}_T,\theta_0) \sim \mathcal{GP}(0,\kappa_{\mathcal{X}_T\mathcal{X}_T}) \text{ with } \kappa_{\mathcal{X}_T\mathcal{X}_T} = \mathbb{E}_{\theta_0}[f(\mathcal{X}_T,\theta_0)f(\mathcal{X}_T,\theta_0)^\top]. \tag{5.29}$$

The block eq. (5.28) thus describes an affine transformation of a GP itself. We have that affine transformations of multivariate Gaussian random variables $X \sim \mathcal{N}(\mu_X,\Sigma_X)$ with $Y = a + BX$ distribute Gaussian themselves with $Y \sim \mathcal{N}(a + B\mu_X,\ B\Sigma_X B^\top)$. Application to Eq. 5.28 and rearrangement then yields the post-convergence GP with mean and covariance

$$\mathbb{E}[f(\mathcal{X}_T,\theta_\infty)] = \Theta_{\mathcal{X}_T\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}y\,, \tag{5.30}$$

$$\Sigma^f_{\mathcal{X}_T\mathcal{X}_T}(\theta_\infty) = \\ \kappa_{\mathcal{X}_T\mathcal{X}_T} + \Theta_{\mathcal{X}_T\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}\kappa_{\mathcal{X}\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}\Theta_{\mathcal{X}\mathcal{X}_T} - \left(\Theta_{\mathcal{X}_T\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}\kappa_{\mathcal{X}\mathcal{X}_T} + \text{h.c.}\right), \tag{5.31}$$

where h.c. refers to the Hermitian conjugate of the preceding term. This completes the proof. □

Proof of Proposition 5.3

We restate Proposition 5.2 for convenience.

Proposition 5.3. (Distribution of post-convergence RND errors) Under the conditions of Proposition 5.1, let $u(x;\vartheta_\infty)$ be a converged predictor network trained on data $\mathcal{X}$ with labels from a fixed target network $g(\mathcal{X};\psi_0)$. Initialization parameters ϑ_0,ψ_0 are drawn i.i.d. $\vartheta_0,\psi_0 \sim \mathcal{N}(0,I)$ resulting in NNGPs $u(x;\vartheta_0) \sim \mathcal{GP}(0,\kappa^u(x,x'))$ and $g(x;\psi_0) \sim \mathcal{GP}(0,\kappa^g(x,x'))$. The RND error at convergence $\epsilon(\mathcal{X}_T;\vartheta_\infty,\psi_0)$ is Gaussian with zero mean and covariance

$$\begin{aligned}\mathbb{E}[\epsilon(\mathcal{X}_T,\vartheta_\infty,\psi_0)] &= 0\,,\\ \Sigma^\epsilon_{\mathcal{X}_T\mathcal{X}_T}(\vartheta_\infty,\psi_0) &= \kappa^\epsilon_{\mathcal{X}_T\mathcal{X}_T} + \Theta_{\mathcal{X}_T\mathcal{X}}\Theta^{-1}_{\mathcal{X}\mathcal{X}}\kappa^\epsilon_{\mathcal{X}\mathcal{X}}\Theta^{-1}_{\mathcal{X}\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}_T}\\ &\quad - \big(\Theta_{\mathcal{X}_T\mathcal{X}}\Theta^{-1}_{\mathcal{X}\mathcal{X}}\kappa^\epsilon_{\mathcal{X}\mathcal{X}_T} + h.c.\big)\,,\end{aligned}$$

where $\kappa^\epsilon_{xx'} = \kappa^u_{xx'} + \kappa^g_{xx'}$ is the covariance kernel of the initialization errors $\epsilon(x;\vartheta_0,\psi_0) = u(x;\vartheta_0) - g(x;\psi_0)$.

Proof. This proposition considers the post-convergence distribution of self-predictive errors as produced by RND. The online predictor $u(x;\vartheta_t)$ undergoes learning dynamics under the same conditions as outlined in the derivation of Proposition 5.1, albeit with the self-predictive loss

$$\mathcal{L}(\vartheta_t) = \frac{1}{2}\| u(\mathcal{X},\vartheta_t) - g(\mathcal{X},\psi_0)\|_2^2\,. \tag{5.32}$$

This, by analogy to Proposition 5.1, implies that the online predictor $u(x;\vartheta_t)$ converges as $t\to\infty$ to the function

$$u(x,\vartheta_\infty) = u(x,\vartheta_0) - \Theta_{x\mathcal{X}}\Theta^{-1}_{\mathcal{X}\mathcal{X}}\big(u(\mathcal{X},\vartheta_0) - g(\mathcal{X},\psi_0)\big)\,. \tag{5.33}$$

For a set of test points $\mathcal{X}_T$, the error $\epsilon(\mathcal{X}_T;\vartheta_\infty,\psi_0) = u(\mathcal{X}_T;\vartheta_\infty) - g(\mathcal{X}_T;\psi_0)$ at convergence can thus be written as the affine transformation

$$\epsilon(\mathcal{X}_T;\vartheta_\infty,\psi_0) = \epsilon(\mathcal{X}_T;\vartheta_0,\psi_0) - \Theta_{\mathcal{X}_T\mathcal{X}}\Theta^{-1}_{\mathcal{X}\mathcal{X}}\epsilon(\mathcal{X};\vartheta_0,\psi_0)\,. \tag{5.34}$$

and the corresponding block matrix equation

$$\begin{pmatrix}\epsilon(\mathcal{X}_T;\vartheta_\infty,\psi_0)\\ \epsilon(\mathcal{X};\vartheta_\infty,\psi_0)\end{pmatrix} = \begin{pmatrix} I & -\Theta_{\mathcal{X}_T\mathcal{X}}\Theta^{-1}_{\mathcal{X}\mathcal{X}}\\ 0 & 0\end{pmatrix}\begin{pmatrix}\epsilon(\mathcal{X}_T;\vartheta_0,\psi_0)\\ \epsilon(\mathcal{X};\vartheta_0,\psi_0)\end{pmatrix}. \tag{5.35}$$

The errors are thus themselves Gaussian with $\epsilon(\mathcal{X}_T;\vartheta_\infty,\psi_0) \sim \mathcal{GP}(0,\kappa^\epsilon_{\mathcal{X}_T\mathcal{X}_T})$ where $\kappa^\epsilon_{\mathcal{X}_T\mathcal{X}_T} = \mathbb{E}_{\vartheta_0,\psi_0}[\epsilon(\mathcal{X}_T;\vartheta_0,\psi_0)\epsilon(\mathcal{X}_T;\vartheta_0,\psi_0)^\top]$. The latter term describes the distribution of self-predictive errors at initialization, which is a simple sum of two independent NNGPs $\epsilon(\mathcal{X}_T;\vartheta_0,\psi_0) = u(\mathcal{X}_T;\vartheta_0) - g(\mathcal{X}_T;\psi_0)$ such that $\kappa^\epsilon_{\mathcal{X}_T\mathcal{X}_T} = \kappa^u_{\mathcal{X}_T\mathcal{X}_T} + \kappa^g_{\mathcal{X}_T\mathcal{X}_T}$, completing the proof. □

5

Proof of Proposition 5.5

Before treating Proposition 5.5 we first derive two known results concerning the independence and recursive character of the NNGP kernel and the NTK. We assume forward computations of $f(x;\theta_t)$ are defined according to Eq. 5.5. To avoid confusion with indices i,j we will in this section use the notation $\kappa(x,x')$ rather than $\kappa_{xx'}$ to denote the function inputs x,x' (and similarly for $\Theta(x,x')$).

Proposition 5.10. *(Lee et al., 2018a) (Recursive NNGP formulation) At initialization $t=0$ and in the limit $n\to\infty$, the i-th output at layer l, $z_i^l(x;\theta_0^{\leq l})$, converges to a GP with zero mean and covariance function $\kappa_{ii}^l(x,x')$ given by*

$$\kappa_{ii}^1(x,x') = \frac{\sigma_w^2}{n_0}x^\top x' + \sigma_b^2, \quad \text{and} \quad k_{ij}^1(x,x') = 0, \quad \text{if } i\neq j, \tag{5.36}$$

$$\kappa_{ii}^l(x,x') = \sigma_b^2 + \sigma_w^2 \mathbb{E}_{z_i^{l-1}\sim\mathcal{GP}(0,\kappa_{ii}^{l-1})}[\phi(z_i^{l-1}(x;\theta_0^{\leq l-1}))\phi(z_i^{l-1}(x';\theta_0^{\leq l-1}))], \tag{5.37}$$

$$\text{and} \quad \kappa_{ij}^l(x,x') = 0, \quad \text{if } i\neq j, \tag{5.38}$$

and we have $\kappa_{ii}^l(x,x') = \kappa^l(x,x')$, $\forall i$.

Proof. We prove the proposition by induction. The induction assumption is that if outputs at layer $l-1$ satisfy a GP structure

$$z_i^{l-1} \sim \mathcal{GP}(0,\kappa^{l-1}), \tag{5.39}$$

with the covariance function defined as

$$\kappa_{ij}^{l-1}(x,x') = \mathbb{E}[z_i^{l-1}(x;\theta_0^{\leq l-1})z_j^{l-1}(x';\theta_0^{\leq l-1})] = \begin{cases} k^{l-1}(x,x') & \text{if } i=j, \\ 0 & \text{if } i\neq j, \end{cases} \tag{5.40}$$

then, outputs at layer l follow

$$z_i^l(x) \sim \mathcal{GP}(0,\kappa^l), \tag{5.41}$$

where the NNGP kernel at layer l is given by:

$$\kappa_{ii}^l(x,x') = \mathbb{E}[z_i^l(x;\theta_0^{\leq l})z_i^l(x';\theta_0^{\leq l})] = \kappa^l(x,x'), \quad \forall i, \tag{5.42}$$

$$\kappa_{ij}^l(x,x') = \mathbb{E}[z_i^l(x;\theta_0^{\leq l})z_j^l(x';\theta_0^{\leq l})] = 0, \quad \text{if } i\neq j. \tag{5.43}$$

with the recursive definition

$$\kappa^l(x,x') = \sigma_b^2 + \sigma_w^2 \mathbb{E}_{z_i^{l-1}\sim\mathcal{GP}(0,k^{l-1})}[\phi(z_i^{l-1}(x;\theta_0^{\leq l-1}))\phi(z_i^{l-1}(x';\theta_0^{\leq l-1}))]. \tag{5.44}$$

Base case ($l = 1$). At layer $l = 1$ we have:

$$z_i^1(x;\theta_0^{\leq 1}) = \frac{\sigma_w}{\sqrt{n_0}} \sum_{j=1}^{n_0} w_{ij}^1 x_j + \sigma_b b_i^1 . \tag{5.45}$$

This is an affine transform of Gaussian random variables; thus, $z_i^1(x;\theta_0^{\leq 1})$ distributes Gaussian with

$$z_i^1(x) \sim \mathcal{GP}(0, \kappa^1), \tag{5.46}$$

with kernel

$$\kappa^1(x,x') = \frac{\sigma_w^2}{n_0} x^\top x' + \sigma_b^2 = \kappa_{ii}^1(x,x'), \quad \text{and} \quad \kappa_{ij}^1 = 0, \quad \text{if } i \neq j, \tag{5.47}$$

where the independence follows from the fact that $z_i^1(x;\theta_0^{\leq 1})$ is computed from separate, independent rows of weights and biases.

Induction step $l > 1$. For layers $l > 1$ we have

$$z_i^l(x;\theta_0^{\leq l}) = \sigma_b b_i^l + \frac{\sigma_w}{\sqrt{n_{l-1}}} \sum_{j=1}^{n_{l-1}} w_{ij}^l x_j^l(x), \quad x_j^l(x) = \phi(z_j^{l-1}(x;\theta_0^{\leq l-1})). \tag{5.48}$$

By the induction assumption, $z_j^{l-1}(x;\theta_0^{\leq l-1})$ are generated by independent GPs. Hence, $x_i^l(x)$ and $x_j^l(x)$ are independent for $i \neq j$. Consequently, $z_i^l(x;\theta_0^{\leq l})$ is a sum of independent random variables. By the CLT (as $n_1, \ldots, n_L \to \infty$) the tuple $\{z_i^l(x;\theta_0^{\leq l}), z_i^l(x';\theta_0^{\leq l})\}$ tends to be jointly Gaussian, with covariance given by:

$$\begin{aligned} &\mathbb{E}[z_i^l(x;\theta_0^{\leq l}) z_i^l(x';\theta_0^{\leq l})] = \\ &\quad \sigma_b^2 + \sigma_w^2 \mathbb{E}_{z_i^{l-1} \sim \mathcal{GP}(0,\kappa^{l-1})}[\phi(z_i^{l-1}(x;\theta_0^{\leq l-1}))\phi(z_i^{l-1}(x';\theta_0^{\leq l-1}))] . \end{aligned} \tag{5.49}$$

Moreover, as z_i^l and z_j^l for $i \neq j$ are defined through independent rows of the parameters w^l, b^l and independent pre-activations $x^l(x)$, we have

$$\kappa_{ij}^l = \mathbb{E}[z_i^l(x) z_j^l(x')] = 0, \quad \text{if } i \neq j, \tag{5.50}$$

and thus completing the proof. □

Proposition 5.11. *(Jacot et al., 2018) (Recursive NTK formulation) In the limit* $n \to \infty$*, the neural tangent kernel* $\Theta_{ii}^l(x,x')$ *of the i-th output* $z_i^l(x;\theta_0^{\leq l})$ *at layer l, defined as the gradient inner product*

$$\Theta_{ii}^l(x,x') = \nabla_{\theta^l}^\top z_i^l(x;\theta_0^{\leq l}) \nabla_{\theta^l} z_i^l(x';\theta_0^{\leq l}), \tag{5.51}$$

is given recursively by

$$\Theta^1_{ii}(x,x') = \kappa^1_{ii}(x,x') = \frac{\sigma_w^2}{n_0} x^\top x' + \sigma_b^2, \quad \textit{and} \quad \Theta^1_{ij}(x,x') = 0, \quad \textit{if } i \neq j, \tag{5.52}$$

$$\Theta^l_{ii}(x,x') = \Theta^{l-1}_{ii}(x,x')\dot{\kappa}^{l-1}_{ii}(x,x') + \kappa^l_{ii}(x,x'), \tag{5.53}$$

$$\tag{5.54}$$

where

$$\dot{\kappa}^l_{ii}(x,x') = \sigma_w^2 \mathbb{E}_{z_i^{l-1} \sim \mathcal{GP}(0,\kappa^{l-1}_{ii})}[\dot{\phi}(z_i^{l-1}(x;\theta_0^{\leq l-1}))\dot{\phi}(z_i^{l-1}(x';\theta_0^{\leq l-1}))], \tag{5.55}$$

and

$$\Theta^l_{ij}(x,x') = \nabla^\top_{\theta^l} z^l_i(x;\theta_0^{\leq l}) \nabla_{\theta^l} z^l_j(x';\theta_0^{\leq l}) = 0 \quad \textit{if } i \neq j. \tag{5.56}$$

5

Proof. The proof is by induction. The induction assumption is that if gradients satisfy at layer $l-1$

$$\Theta^{l-1}_{ij}(x,x') = \nabla^\top_{\theta^{l-1}} z^{l-1}_i(x;\theta_0^{\leq l-1}) \nabla_{\theta^{l-1}} z^{l-1}_j(x';\theta_0^{\leq l-1}) = \begin{cases} \Theta^{l-1}(x,x') & \text{if } i = j, \\ 0 & \text{if } i \neq j, \end{cases} \tag{5.57}$$

then at layer l we have

$$\Theta^l_{ij}(x,x') = \begin{cases} \Theta^{l-1}_{ii}(x,x')\dot{\kappa}^l_{ii}(x,x') + \kappa^l_{ii}(x,x') & \text{if } i = j, \\ 0 & \text{if } i \neq j. \end{cases} \tag{5.58}$$

Base case ($l = 1$). At layer $l = 1$, we have

$$z^1_i(x;\theta_0^{\leq 1}) = \sigma_b b^1_i + \frac{\sigma_w}{\sqrt{n_0}} \sum_j^{n_0} w^1_{ij} x_j, \tag{5.59}$$

and the gradient inner product is given by:

$$\nabla^\top_{\theta^1} z^1_i(x;\theta_0^{\leq 1}) \nabla_{\theta^1} z^1_i(x';\theta_0^{\leq 1}) = \frac{\sigma_w^2}{n_0} x^\top x' + \sigma_b^2 = \kappa^1_{ii}(x,x'). \tag{5.60}$$

Inductive step ($l > 1$). For layers $l > 1$, we split parameters $\theta^l = \theta^{l-1} \cup \{w^l, b^l\}$ and split the inner product by

$$\Theta^l_{ii}(x,x') = \underbrace{\nabla^\top_{\theta^{l-1}} z^l_i(x;\theta_0^{\leq l}) \nabla_{\theta^{l-1}} z^l_i(x';\theta_0^{\leq l})}_{l.h.s} + \underbrace{\nabla^\top_{\{w^l,b^l\}} z^l_i(x;\theta_0^{\leq l}) \nabla_{\{w^l,b^l\}} z^l_i(x;\theta_0^{\leq l})}_{r.h.s}. \tag{5.61}$$

Note that the above *r.h.s* involves gradients w.r.t. last-layer parameters, i.e. the post-activation outputs of the previous layer, and by the same arguments as in the NNGP derivation of Proposition 5.10, this is a sum of independent post activations s.t. in the limit $n_{l-1} \to \infty$

$$\nabla^{\top}_{\{w^l,b^l\}} z^l_i(x;\theta_0^{\leq l}) \nabla_{\{w^l,b^l\}} z^l_j(x';\theta_0^{\leq l}) = \begin{cases} k^l_{ii}(x,x'), & i=j, \\ 0, & i \neq j. \end{cases} \tag{5.62}$$

For the *l.h.s.*, we first apply chain rule to obtain

$$\nabla_{\theta^{l-1}} z^l_i(x;\theta_0^{\leq l}) = \frac{\sigma_w}{\sqrt{n_{l-1}}} \sum_j^{n_{l-1}} w^l_{ij} \dot{\phi}(z^{l-1}_j(x;\theta_0^{\leq l-1})) \nabla_{\theta^{l-1}} z^{l-1}_j(x;\theta_0^{\leq l-1}). \tag{5.63}$$

The gradient inner product of outputs i and j thus reduces to

$$\begin{aligned} &\nabla^{\top}_{\theta^{l-1}} z^l_i(x;\theta_0^{\leq l}) \nabla_{\theta^{l-1}} z^l_j(x';\theta_0^{\leq l}) = \\ &\quad \frac{\sigma_w^2}{n_{l-1}} \sum_k^{n_{l-1}} w^l_{ik} w^l_{jk} \dot{\phi}(z^{l-1}_k(x;\theta_0^{\leq l-1})) \dot{\phi}(z^{l-1}_k(x';\theta_0^{\leq l-1})) \Theta^{l-1}_{kk}(x,x'). \end{aligned} \tag{5.64}$$

By the induction assumption $\Theta^{l-1}_{kk}(x,x') = \Theta^{l-1}(x,x')$ and again by the independence of the rows w^l_i and w^l_j for $i \neq j$, the above expression converges in the limit $n_{l-1} \to \infty$ to an expectation with

$$\Theta^l_{ij}(x,x') = \begin{cases} \Theta^{l-1}(x,x') \dot{\kappa}^l_{ii}(x,x') + \kappa^l_{ii}(x,x') & i=j, \\ 0 & i \neq j, \end{cases} \tag{5.65}$$

thereby completing the proof. □

We now restate Proposition 5.5 for convenience.

Proposition 5.5. (Independence of NN functions) Under the conditions of Proposition 5.1, the random output functions $f_i(x;\theta_t)$ of a NN with K output dimensions and shared hidden layers are mutually independent with covariance

$$\Sigma^{ij}_{xx'}(\theta_t) = \mathbb{E}[f_i(x;\theta_t) f_j(x';\theta_t)] = \begin{cases} \Sigma^f_{xx'}(\theta_t) & \text{if } i=j, \\ 0 & \text{if } i \neq j, \end{cases}$$

on the interval $t \in [0,\infty)$.

Proof. We begin by deriving the training dynamics for the output $f_i(x;\theta_t)$ analogously to the proof of Proposition 5.1. We denote by $\mathcal{Y}_i$ the labels used to train the function $f_i(x;\theta_t)$. By Proposition 5.11, the training dynamics of $f_i(x;\theta_t)$ and

$f_j(x;\theta_t)$ are decoupled for $i \neq j$ and we can thus derive Eq. 5.24 analogously for individual output heads i. Taking the infinite width limit, we obtain at time t

$$f_i(x;\theta_t) = f_i(x;\theta_0) + \Theta_{ii}(x,\mathcal{X})\Theta_{ii}(\mathcal{X},\mathcal{X})^{-1}\Big(e^{-\alpha t\Theta_{ii}(\mathcal{X},\mathcal{X})} - I\Big)(f_i(\mathcal{X};\theta_0) - y_i). \tag{5.66}$$

Thus, the output head $f_i(x;\theta_t)$ at time t is a deterministic function of its own initialization only, which itself is characterized by a GP $f_i(x;\theta_0) \sim \mathcal{GP}(0,\kappa_{ii}(x,x'))$ that is independent of output heads $j \neq i$ by Proposition 5.10. And thus, since $f_i(x;\theta_t)$ is an affine transform of its own independent initialization terms $f_i(x;\theta_0)$ and $f_i(\mathcal{X};\theta_0)$, it too must follow an independent GP with $\mathbb{E}_{\theta_0}[f_i(x;\theta_t)f_i(x';\theta_t)] = \Sigma(x,x';\theta_t)$ and in particular $\mathbb{E}_{\theta_0}[f_i(x;\theta_t)f_j(x';\theta_t)] = 0$ if $i \neq j$. □

5

Proof of Theorem 5.6

We restate Theorem 5.6 for convenience.

Theorem 5.6. (Distributional equivalence between multi-headed RND and finite deep ensembles) Under the conditions of Proposition 5.1, let $u_i(x;\vartheta_\infty), g_i(x;\psi_0)$ be the i-th output of predictor and target networks respectively with K output dimensions. Denote their sample mean RND error $\bar{\epsilon}^2(x;\vartheta_\infty,\psi_0) = \frac{1}{K}\sum_{i=1}^K \epsilon_i^2(x;\vartheta_\infty,\psi_0)$. Moreover, let $\{f(x;\theta_\infty^i)\}_{i=1}^{K+1}$ be an ensemble of $K+1$ independently initialized NNs. Denote its sample variance $\bar{\sigma}_f^2(x;\theta_\infty^{i\ldots K+1}) = \frac{1}{K}\sum_{i=1}^{K+1}(f(x;\theta_\infty^i) - \frac{1}{K+1}\sum_{j=1}^{K+1} f(x;\theta_\infty^j))^2$. We have that

$$\frac{1}{2}\bar{\epsilon}^2(x;\vartheta_\infty,\psi_0) \overset{D}{=} \bar{\sigma}_f^2(x;\theta_\infty^{i\ldots K+1}), \tag{5.10}$$

where $\overset{D}{=}$ indicates an equality in distribution, namely by a scaled Chi-squared distribution $\bar{\sigma}_f^2(x;\theta_\infty^{i\ldots K+1}) \sim \frac{\Sigma_{xx}^f(\theta_\infty)}{K}\chi^2(K)$ with scale $\Sigma_{xx}^f(\theta_\infty)$ given by the analytical variance as given in Proposition 5.2.

Proof. The proof follows by combining the results of Propositions (5.3) and (5.5). We define a multiheaded RND predictor with K output heads $\{u_i(x,\vartheta_t)\}_{i=1}^K$ and a fixed multiheaded target network $\{g_i(x_t;\psi_0)\}_{i=1}^K$ of equivalent architecture as u_i (i.e., both corresponding to the same NTK Θ) with the corresponding prediction errors $\{\epsilon_i(x;\vartheta_t,\psi_0)\}_{i=1}^K$ accordingly. Let $u_i(x,\vartheta_t)$ be trained such that each head i is trained to match the i-th target output $g_i(x;\psi_0)$.

By Proposition 5.5, the predictions of online predictor heads $\{u_i(x,\vartheta_t)\}_{i=1}^K$ at time t and fixed target networks $\{g_i(x_t;\psi_0)\}_{i=1}^K$ are each mutually independent

with

$$\mathbb{E}_{\vartheta_0}[u_i(x;\vartheta_t)u_j(x;\vartheta_t)] = 0\,, \quad \text{if } i \neq j\,, \tag{5.67}$$

and

$$\mathbb{E}_{\psi_0}[g_i(x;\psi_0)g_j(x;\psi_0)] = 0\,, \quad \text{if } i \neq j\,. \tag{5.68}$$

As a consequence, we also have that

$$\mathbb{E}_{\vartheta_0,\psi_0}[\epsilon_i(x;\vartheta_t,\psi_0)\epsilon_j(x;\vartheta_t,\psi_0)] = 0\,, \quad \text{if } i \neq j\,. \tag{5.69}$$

As previously established in the proof of Proposition 5.5, the multi-headed functions $\{\epsilon_i(x;\vartheta_t,\psi_0)\}_{i=1}^K$ follow equivalent learning dynamics as their scalar-output counterparts. The post-convergence distribution of individual heads $\epsilon_i(x;\vartheta_\infty,\psi_0)$ must therefore equal the scalar-output post-convergence distribution established in Proposition 5.3. Consequently, the errors $\{\epsilon_i(x;\vartheta_t,\psi_0)\}_{i=1}^K$ are independent and identically distributed draws from a Gaussian with mean and covariance

$$\begin{aligned}
&\mathbb{E}[\epsilon(x,\vartheta_\infty,\psi_0)] = 0\,,\\
&\quad \Sigma^\epsilon_{xx'}(\vartheta_\infty,\psi_0) = \kappa^\epsilon_{xx'} + \Theta_{x\mathcal{X}}\Theta^{-1}_{\mathcal{X}\mathcal{X}}\kappa^\epsilon_{\mathcal{X}\mathcal{X}}\Theta^{-1}_{\mathcal{X}\mathcal{X}}\Theta_{\mathcal{X}x'} - \left(\Theta_{x\mathcal{X}}\Theta^{-1}_{\mathcal{X}\mathcal{X}}\kappa^\epsilon_{\mathcal{X}x'} + \text{h.c.}\right),
\end{aligned}$$

where $\kappa^\epsilon_{xx'} = \kappa^u_{xx'} + \kappa^g_{xx'}$. The sample mean square given by $\frac{1}{2}\bar{\epsilon}^2(x;\vartheta_\infty,\psi_0) = \frac{1}{2K}\sum_{i=1}^K \epsilon_i^2(x;\vartheta_\infty,\psi_0)$ is then known to follow a scaled Chi-squared distribution with K degrees of freedom

$$\tfrac{1}{2}\bar{\epsilon}^2(x;\vartheta_\infty,\psi_0) \sim \frac{\frac{1}{2}\Sigma^\epsilon_{xx}(\vartheta_\infty,\psi_0)}{K}\chi^2(K) \tag{5.70}$$

where $\Sigma^\epsilon_{xx}(\vartheta_\infty,\psi_0)$ is the variance of the GP described in Proposition 5.3.

Conversely, a set of $K+1$ independent neural networks arranged to a deep ensemble $\{f(x;\theta^i_\infty)\}_{i=1}^{K+1}$ in the infinite width limit $n \to \infty$ and at convergence $t \to \infty$ are by definition i.i.d. samples from the GP described in Proposition 5.2. As before, the empirical variance defined as $\bar{\sigma}^2_f(x;\theta^{i\ldots K+1}_\infty) = \frac{1}{K}\sum_{i=1}^{K+1}\big(f(x;\theta^i_\infty) - \frac{1}{K+1}\sum_{j=1}^{K+1} f(x;\theta^j_\infty)\big)^2$ distributes as a scaled Chi-squared distribution with K degrees of freedom

$$\bar{\sigma}^2_f(x;\theta^{i\ldots K+1}_\infty) \sim \frac{\Sigma^f_{xx}(\theta_\infty)}{K}\chi^2(K)\,, \tag{5.71}$$

where $\Sigma^f_{xx}(\theta_\infty)$ is the variance of the GP described in Proposition 5.2.

Finally, as we assume equal architecture and i.i.d. initialization of u, g, and f, we have that $\kappa^\epsilon_{xx'} = \kappa^u_{xx'} + \kappa^g_{xx'} = 2\kappa^u_{xx'} = 2\kappa_{xx'}$ and accordingly $\frac{1}{2}\Sigma^\epsilon_{xx}(\vartheta_\infty,\psi_0) = \Sigma^f_{xx}(\theta_\infty)$, completing the proof. □

5.8.2 Posterior Equivalence

This section contains proofs for results pertaining to the equivalence of self-predictive errors of "Bayesian RND" and the variance of Bayesian posterior predictive distributions of neural networks in the infinite width limit.

Proof of Proposition 5.7

We restate Proposition 5.7 for convenience.

Proposition 5.7. (Bayesian RND target function) Under the conditions of Proposition 5.1, let $u(x;\vartheta_0)$ and $g(x;\psi_0)$ be neural networks of L layers with parameters $\vartheta_0, \psi_0 \sim \mathcal{N}(0, I)$ i.i.d. Moreover, let $\psi_0^L = \{w^L, b^L\}$ denote the last-layer parameters of ψ_0 and $\psi_0^{\leq L-1}$ the parameters of all preceding layers. Suppose the target function $\tilde{g}(x;\vartheta_0,\psi_0)$ is given by

$$\tilde{g}(x;\vartheta_0,\psi_0) = \nabla_{\vartheta_0}^\top u(x;\vartheta_0)\psi_0^*,$$

where $\psi_0^* = \{\psi_0^{\leq L-1}, 0_{dim(\psi_0^L)}\}$ is a copy of ψ_0 with its last-layer weights set to 0. In the infinite width limit $n \to \infty$, $\tilde{g}(x;\vartheta_0,\psi_0)$ distributes by construction as $\tilde{g}(x;\vartheta_0,\psi_0) \sim \mathcal{GP}(0, \kappa_{xx'}^{\tilde{g}})$ where $\kappa_{xx'}^{\tilde{g}} = \Theta_{xx'}^{\leq L-1}$.

Proof. The proof will show that in the limit $n \to \infty$ the function $\tilde{g}(x;\vartheta_0,\psi_0)$ converges to a GP $\tilde{g}(x;\vartheta_0,\psi_0) \sim \mathcal{GP}(0, \Theta_{xx'}^{\leq L-1})$ by Lévy's continuity theorem, which we recall informally below.

Theorem 5.12. (Lévy's continuity theorem) Let $\{Z_n\}_{n=1}^\infty$ be a sequence of $\mathbb{R}^n$-valued random variables. Their characteristic functions $\varphi_{Z_n}(t)$ for some $t \in \mathbb{R}^n$ are given by

$$\varphi_{Z_n}(t) = \mathbb{E}[e^{it^\top Z_n}], \tag{5.72}$$

where i is the imaginary unit. If in the limit $n \to \infty$ the sequence of characteristic functions converges pointwise to a function

$$\varphi_{Z_n}(t) \to \varphi(t) \quad \forall t \in \mathbb{R}^n, \tag{5.73}$$

then Z_n converges in distribution to a random variable Z

$$Z_n \xrightarrow{D} Z, \tag{5.74}$$

whose characteristic function is $\varphi_Z(t) = \varphi(t)$

Rigorous proof can be found for example in Durrett (2019).

We begin by rewriting the function $\tilde{g}(x;\vartheta_0,\psi_0)$ as a linear model with

$$\tilde{g}(x;\vartheta_0,\psi_0) = \nabla_\vartheta^\top u(x;\vartheta_0)\psi_0^* \tag{5.75}$$

$$= \nabla_{\vartheta^{\leq L-1}}^\top u(x;\vartheta_0)\psi_0^{\leq L-1}. \tag{5.76}$$

Since $\psi_0^{\leq L-1}$ is an independent draw from ϑ_0 by assumption, $\tilde{g}(x;\vartheta_0,\psi_0)$ is a random affine transform of the Gaussian vector $\psi_0^{\leq L-1}$. For more precise treatment of the distribution of $\tilde{g}(x;\vartheta_0,\psi_0)$, we write $\tilde{G}(\mathcal{X}_T)$ to denote the random variable corresponding to the function evaluations of $\tilde{g}$ on a test set $\mathcal{X}_T$. Conditioned on ϑ_0 (i.e., fixing the affine transform), we thus have that $\tilde{G}(\mathcal{X}_T)|\vartheta_0 \sim \mathcal{GP}(0, \Theta_{0,\mathcal{X}_T\mathcal{X}_T}^{\leq L-1})$, where $\Theta_{0,\mathcal{X}_T\mathcal{X}_T}^{\leq L-1} = \nabla_{\vartheta^{\leq L-1}}^\top u(\mathcal{X}_T;\vartheta_0)\nabla_{\vartheta^{\leq L-1}} u(\mathcal{X}_T;\vartheta_0)$ is the empirical NTK matrix of u. Note that this statement holds irrespective of the network width n.

Next, we show that the unconditional law of $\tilde{G}(\mathcal{X}_T)$, too, tends to a GP in the limit $n \to \infty$. To this end, we examine the distribution of the unconditioned random vector $\tilde{G}(\mathcal{X}_T)$ through its characteristic function

$$\varphi_{\tilde{G}(\mathcal{X}_T)}(t) = \mathbb{E}[e^{it^\top \tilde{G}(\mathcal{X}_T)}]. \tag{5.77}$$

This characteristic function $\varphi_{\tilde{G}(\mathcal{X}_T)}(t)$ uniquely defines the distribution of $\tilde{G}(\mathcal{X}_T)$ (Durrett, 2019). By the law of total expectation, the characteristic function of the unconditional variable $\tilde{G}(\mathcal{X}_T)$ can then be written as

$$\varphi_{\tilde{G}(\mathcal{X}_T)}(t) = \mathbb{E}_{\vartheta_0}\left[\mathbb{E}[e^{it^\top \tilde{G}(\mathcal{X}_T)}|\vartheta_0]\right]. \tag{5.78}$$

As stated above, the conditional distribution of $\tilde{G}(\mathcal{X}_T)|\vartheta_0$ is a zero-mean Gaussian with the empirical covariance $\Theta_{0,\mathcal{X}_T\mathcal{X}_T}^{\leq L-1}$, to which we can show the conditional characteristic function is given by (Durrett, 2019)

$$\mathbb{E}[e^{it^\top \tilde{G}(\mathcal{X}_T)}|\vartheta_0] = e^{-\frac{1}{2}t^\top \Theta_{0,\mathcal{X}_T\mathcal{X}_T}^{\leq L-1} t}. \tag{5.79}$$

Plugging this back into Eq. 5.78 gives

$$\varphi_{\tilde{G}(\mathcal{X}_T)}(t) = \mathbb{E}_{\vartheta_0}[e^{-\frac{1}{2}t^\top \Theta_{0,\mathcal{X}_T\mathcal{X}_T}^{\leq L-1} t}]. \tag{5.80}$$

We now use the known result by Jacot et al. (2018) that, as $n \to \infty$ we have that $\Theta_{0,\mathcal{X}_T\mathcal{X}_T} \to \Theta_{\mathcal{X}_T\mathcal{X}_T}$ in probability and accordingly $\Theta_{0,\mathcal{X}_T\mathcal{X}_T}^{\leq L-1} \to \Theta_{\mathcal{X}_T\mathcal{X}_T}^{\leq L-1}$ converges to a deterministic kernel matrix. Moreover, since the Gram matrix $\Theta_{0,\mathcal{X}_T\mathcal{X}_T}^{\leq L-1}$ is positive semidefinite in general, the term $e^{-\frac{1}{2}t^\top \Theta_{0,\mathcal{X}_T\mathcal{X}_T}^{\leq L-1} t}$ is bounded

and continuous. By bounded convergence (Durrett, 2019), we can then conclude that we also have convergence of the characteristic function through

$$\lim_{n\to\infty} \varphi_{\tilde{G}(\mathcal{X}_T)}(t) = \lim_{n\to\infty} \mathbb{E}_{\vartheta_0}[e^{-\frac{1}{2}t^\top \Theta^{\leq L-1}_{0,\mathcal{X}_T\mathcal{X}_T} t}] \tag{5.81}$$

$$= e^{-\frac{1}{2}t^\top \Theta^{\leq L-1}_{\mathcal{X}_T\mathcal{X}_T} t}. \tag{5.82}$$

As stated earlier, for a Gaussian random vector Z with $Z \sim \mathcal{GP}(0, \Theta^{\leq L-1}_{\mathcal{X}_T\mathcal{X}_T})$ its characteristic function is given by $e^{-\frac{1}{2}t^\top \Theta^{\leq L-1}_{\mathcal{X}_T\mathcal{X}_T} t}$. Invoking Lévy's continuity theorem, the pointwise convergence of $\varphi_{\tilde{G}(\mathcal{X}_T)}(t)$ to this exact limit $\varphi_{\tilde{G}(\mathcal{X}_T)}(t) \to e^{-\frac{1}{2}t^\top \Theta^{\leq L-1}_{\mathcal{X}_T\mathcal{X}_T} t}$ then implies convergence in distribution of $\tilde{G}(\mathcal{X}_T) \xrightarrow{D} Z$ and we can thus conclude $\tilde{g}(x;\vartheta_0,\psi_0) \sim \mathcal{GP}(0, \Theta^{\leq L-1}_{xx'})$. □

Proof of Theorem 5.8

We restate Proposition 5.7 for convenience.

Theorem 5.8. (Distribution of Bayesian RND errors) Under the conditions of Proposition 5.1, let $u(x;\vartheta_\infty)$ be a converged predictor network trained on data $\mathcal{X}$ with labels from the fixed target function $\tilde{g}(\mathcal{X};\vartheta_0,\psi_0)$ as defined in Proposition 5.7. Let parameters ϑ_0,ψ_0 be drawn i.i.d. $\vartheta_0,\psi_0 \sim \mathcal{N}(0,I)$. The post-convergence Bayesian RND error $\epsilon^b(\mathcal{X}_T;\vartheta_\infty,\vartheta_0,\psi_0) = u(\mathcal{X}_T;\vartheta_\infty) - \tilde{g}(\mathcal{X}_T;\vartheta_0,\psi_0)$ on a test set $\mathcal{X}_T$ is Gaussian with zero mean and covariance

$$\Sigma^{\epsilon^b}_{\mathcal{X}_T\mathcal{X}_T}(\vartheta_\infty,\vartheta_0,\psi_0) = \Theta_{\mathcal{X}_T\mathcal{X}_T} - \Theta_{\mathcal{X}_T\mathcal{X}}\Theta^{-1}_{\mathcal{X}\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}_T},$$

and thus recovers the covariance of the exact Bayesian posterior predictive distribution of an infinitely wide neural network with the corresponding NTK $\Theta_{xx'}$.

Proof. The result follows from the independence of the two GPs of interest in the limit $n \to \infty$. First, this is $\tilde{g}(x;\vartheta_0,\psi_0) \sim \mathcal{GP}(0, \Theta^{\leq L-1}_{xx'})$ and second, $u(x;\vartheta_0) \sim \mathcal{GP}(0, \Theta^{L}_{xx'})$. In the following, we will show that the two GPs are in the limit $n \to \infty$ independent processes such that Eq. 5.15 applies.

We first write for any two points x, x' the covariance

$$\mathrm{Cov}[\tilde{g}(x;\vartheta_0,\psi_0), u(x';\vartheta_0)] = \mathbb{E}[\tilde{g}(x;\vartheta_0,\psi_0)u(x';\vartheta_0)]. \tag{5.83}$$

As ψ_0 is drawn independently of ϑ_0, the conditional expectation can be written as

$$\mathbb{E}[\tilde{g}(x;\vartheta_0,\psi_0)u(x';\vartheta_0)|\vartheta_0] = u(x';\vartheta_0)\mathbb{E}[\tilde{g}(x;\vartheta_0,\psi_0)|\vartheta_0] \tag{5.84}$$

$$= u(x';\vartheta_0)\mathbb{E}[\nabla^\top_{\vartheta^{\leq L-1}} u(x;\vartheta_0)\psi_0^{\leq L-1}|\vartheta_0] \tag{5.85}$$

$$= u(x';\vartheta_0)\cdot 0, \tag{5.86}$$

and by the law of total expectation

$$\mathbb{E}[\tilde{g}(x;\vartheta_0,\psi_0)u(x';\vartheta_0)] = \mathbb{E}_{\vartheta_0}\big[\mathbb{E}[\tilde{g}(x;\vartheta_0,\psi_0)u(x';\vartheta_0)|\vartheta_0]\big] \tag{5.87}$$

$$= 0\,. \tag{5.88}$$

We conclude that the two GPs $\tilde{g}(x;\vartheta_0,\psi_0) \sim \mathcal{GP}(0,\Theta^{\leq L-1}_{xx'})$ and $u(x;\vartheta_0) \sim \mathcal{GP}(0,\Theta^{L}_{xx'})$ are mutually independent such that the initialization kernel $\kappa^{\epsilon^b}_{xx'}$ is given as

$$\kappa^{\epsilon^b}_{xx'} = \Theta_{xx'}\,. \tag{5.89}$$

This is because $\Theta_{xx'} = \Theta^{L}_{xx'} + \Theta^{\leq L-1}_{xx'}$ and $\kappa^{\tilde{g}}_{xx'} = \Theta^{\leq L-1}_{xx'}$, $\kappa^{u}_{xx'} = \Theta^{L}_{xx'}$ are mutually independent. □

6

Universal Value-Function Uncertainties

The work presented in this chapter is to appear as: M. A. Zanger, M. Weltevrede, Y. Oren, P. R. Van der Vaart, C. Horsch, W. Böhmer, and M. T. J. Spaan. Universal value-function uncertainties. To appear in *International Conference on Learning Representations (ICLR)*, 2026.
Author contributions are as follows: *M.A.Z.:* Conceptualization, Methodology, Formal Analysis, Experimental Implementation, Visualizations, Writing — Original Draft. *M.W.:* Experimental Implementation, Writing — Review & Editing. *Y.O.:* Experimental Implementation, Writing — Review & Editing. *P.R.V.:* Formal Analysis, Writing — Review & Editing. *C.H.:* Experimental Implementation, Writing — Review & Editing. *W.B.:* Supervision, Project Administration, Writing — Review & Editing. *M.T.J.S.:* Supervision, Project Administration, Funding Acquisition, Writing — Review & Editing.

Estimating epistemic uncertainty in value functions is a crucial challenge for many aspects of reinforcement learning (RL), including efficient exploration, safe or conservative decision-making, and offline RL. The preceding chapters have established that single-model methods can efficiently approximate the uncertainty of deep ensembles and have laid a theoretical foundation for the popular random network distillation (RND) algorithm, a previously little-understood approach for single-model uncertainty quantification. However, these approaches typically quantify *myopic* uncertainty — the uncertainty associated with an immediate, one-step prediction. To guide decision-making in sequential problems, this myopic signal must then be propagated through a value function, often via an intrinsic reward mechanisms, to inform long-term, forward-looking behavior. This final research chapter takes the next logical step, investigating whether this additional propagation mechanism can be avoided altogether, circumvented by a single model designed to estimate long-term, cumulative uncertainty directly. This line of inquiry synthesizes insights from our previous investigations and addresses our final *research question* (RQ4):

RQ4: *Can the predictive variance of an ensemble of deep value functions be approximated directly and accurately by a single neural network in the limit of infinite width?*

To answer this question, this chapter introduces universal value-function uncertainty (UVU), a novel method that adapts the self-predictive error paradigm for the specific challenges of value-based reinforcement learning. Similar in spirit to RND, UVU quantifies uncertainty as the squared prediction error between an online learner and a fixed, randomly initialized target network. The crucial distinction, however, lies in its training procedure: the online network is trained using *temporal difference (TD) learning* with a synthetic reward signal derived from the fixed target. This design ensures that the resulting error signal inherently incorporates future uncertainties, directly reflecting policy-conditional, cumulative value uncertainty for any given policy.

In this chapter, we first present the full algorithmic details of the UVU method. We then provide an extensive theoretical analysis within the neural tangent kernel (NTK) framework, proving that in the infinite-width limit, the UVU error signal is exacolortly equivalent to the variance of a corresponding ensemble of universal value functions. Finally, we demonstrate the practical efficacy of our approach in a challenging multi-task offline reinforcement learning setting. Our empirical results show that UVU serves as a reliable estimator of agent capability and achieves performance comparable or superior to very large ensembles of universal value functions, while offering the simplicity and substantial computational savings of a single-model method.

6.1 Introduction

Deep reinforcement learning (RL) has emerged as an essential paradigm for addressing difficult sequential decision-making problems (Mnih et al., 2015; Silver et al., 2016; Vinyals et al., 2019) but a more widespread deployment of agents to real-world applications remains challenging. Open problems such as efficient exploration, scalable offline learning and safety pose persistent obstacles to this transition. Central to these capabilities is the quantification of *epistemic uncertainty*, an agent's uncertainty due to limited data. In the context of RL, uncertainty estimation relating to the *value function* is of particular importance as it reflects uncertainty about long-term consequences of actions.

However, computationally tractable estimation of value-function uncertainty remains a challenge. Bayesian RL approaches, both in its model-based (Ghavamzadeh et al., 2015) and model-free (Dearden et al., 1998) flavors, typically come with sound theoretical underpinnings but face significant computational hurdles due to the general intractability of posterior inference. Theoretical guarantees of the latter are moreover often complicated by the use of training procedures like temporal difference (TD) learning with bootstrapping. Conversely, deep ensembles (Lakshminarayanan et al., 2017) have emerged as a reliable standard for practical value uncertainty estimation in deep RL (Chen et al., 2017; Osband et al., 2016). Empirically, independently trained value functions from random initialization provide effective uncertainty estimates that correlate well with true estimation errors. Although in general more tractable than full posterior inference, this approach remains computationally challenging for larger models where a manyfold increase in computation and memory severely limits scalability. Various single-model approaches like random network distillation (RND) (Burda et al., 2019b), pseudo counts (Bellemare et al., 2016) or intrinsic curiosity (Pathak et al., 2017) efficiently capture *myopic* epistemic uncertainty but require additional propagation mechanisms to obtain value uncertainties (Janz et al., 2019; O'Donoghue et al., 2018; Zhou et al., 2020) and often elude a thorough theoretical understanding. We conclude that there persists a lack of computationally efficient single-model approaches with the ability to directly estimate policy-dependent value uncertainties with a strong theoretical foundation.

To this end, we introduce *universal value-function uncertainties (UVU)*, a novel method designed to estimate epistemic uncertainty of value functions for any given policy using a single-model architecture. Similar in spirit to the well-known RND algorithm, UVU quantifies uncertainty through a prediction error between an online learner u and a fixed, randomly initialized target network g. Crucially, and in contrast to the regression objective of RND, UVU optimizes its online network u using TD learning with a synthetic reward r_g

generated entirely from the target network g. By construction, the reward r_g implies a value learning problem to which the target function g itself is a solution, forcing the online learner u to recover g through minimization of TD losses. UVU then quantifies uncertainty as the squared prediction error between online learner and fixed target function. Unlike previous methods, our design requires no training of multiple models (e.g., ensembles) nor separate value and uncertainty models (e.g., RND, ICM). Furthermore, we design UVU as a universal policy-conditioned model (comparable to universal value function approximators (Schaul et al., 2015)), that is, it takes as input a state, action, and policy encoding and predicts the epistemic uncertainty associated with the value function for the encoded policy.

A key contribution of our work is a thorough theoretical analysis of UVU using the framework of *neural tangent kernels* (NTK, Jacot et al., 2018). Specifically, we characterize the learning dynamics of wide neural networks with TD losses and gradient descent to obtain closed-form solutions for the convergence and generalization behavior of neural network value functions. In the limit of infinite network width, we then show that prediction errors generated by UVU are equivalent to the variance of an ensemble of universal value functions, both in expectation and with finite sample estimators.

We validate UVU empirically on an offline multi-task benchmark from the minigrid suite where agents are required to reject tasks they cannot perform to achieve maximal scores. We show that UVU's uncertainty estimates perform comparably to large deep ensembles, while drastically reducing the computational footprint.

6.2 Background

We frame our work within the standard Markov decision process (MDP) (Bellman, 1957) formalism, defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \gamma, P, \mu)$. Here, $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathscr{P}(\mathbb{R})$ is the distribution of immediate rewards, $\gamma \in [0, 1)$ is the discount factor, $P : \mathcal{S} \times \mathcal{A} \to \mathscr{P}(\mathcal{S})$ is the transition probability kernel, and $\mu : \mathscr{P}(\mathcal{S})$ is the initial state distribution. An RL agent interacts with this environment by selecting actions according to a policy $\pi : \mathcal{S} \to \mathscr{P}(\mathcal{A})$. At each timestep t, the agent is in state S_t, takes action $A_t \sim \pi(\cdot|S_t)$, receives a reward $R_t \sim \mathcal{R}(\cdot|S_t, A_t)$, and transitions to a new state $S_{t+1} \sim P(\cdot|S_t, A_t)$. We quantify the merit of taking actions $A_t = a$ in state $S_t = s$ and subsequently following policy π by the action-value function, or Q-function $Q^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, which accounts for the cumulative discounted future rewards and adheres to a recursive consistency condition described by the

Bellman equation

$$Q^\pi(s,a) = \mathbb{E}_{\mathcal{R},\pi,P}[R_0 + \gamma Q^\pi(S_1, A_1) | S_0 = s, A_0 = a]. \tag{6.1}$$

The agent's objective then is to find policies that maximize the expected returns $J(\pi) = \mathbb{E}_{S_0 \sim \mu, A_0 \sim \pi(\cdot|S_0)}[Q^\pi(S_0, A_0)]$.

Often, we may be interested in agents capable of operating a variety of policies to achieve different goals. UVFAs (Schaul et al., 2015) address this by conditioning value functions additionally on an encoding $z \in \mathcal{Z}$. This encoding specifies a current policy context, indicating for example a task or goal. We denote such *universal* Q-functions as $Q(s,a,z)$. In the context of this work, we consider z to be a parameterization or indexing of a specific policy $\pi(\cdot|s,z)$, or in other words $Q : \mathcal{S} \times \mathcal{A} \times \mathcal{Z} \to \mathbb{R}, \quad Q(s,a,z) \equiv Q^{\pi(\cdot|s,z)}(s,a)$.

Both in the single and multi task settings, obtaining effective policies may require efficient exploration and an agent's ability to reason about *epistemic* uncertainty. This source of uncertainty, in contrast to *aleatoric* uncertainty, stems from a lack of knowledge and may in general be reduced by the acquisition of data. In the context of RL, we make an additional distinction between *myopic uncertainty* and *value uncertainty*.

6.2.1 Myopic Uncertainty and Neural Tangent Kernels

Myopic uncertainty estimation methods, such as RND or ensembles predicting immediate rewards or next states, quantify epistemic uncertainty without explicitly accounting for future uncertainties along trajectories. We first briefly recall the RND algorithm (Burda et al., 2019b), before introducing the NTK (Jacot et al., 2018) framework.

Random network distillation comprises two neural networks: A fixed, randomly initialized *target network* $g(x;\psi_0)$, and a *predictor network* $u(x;\vartheta_t)$. The online predictor $u(x;\vartheta_t)$ is trained via gradient descent to minimize a square loss between its own predictions and the target network's output on a set of data points $\mathcal{X} = \{x_i \in \mathbb{R}^{d_{\text{in}}}\}_{i=1}^{N_D}$. The RND prediction error at a test point x then serves as an uncertainty or novelty signal. The loss and error function of RND are then given as

$$\mathcal{L}_{\text{rnd}}(\theta_t) = \tfrac{1}{2}(u(\mathcal{X};\theta_t) - g(\mathcal{X};\psi_0))^2 \text{ and } \epsilon^2_{\text{rnd}}(x;\vartheta_t,\psi_0) = \tfrac{1}{2}(u(x;\vartheta_t) - g(x;\psi_0))^2 . \tag{6.2}$$

This mechanism relies on the idea that the predictor network recovers the outputs of the target network only for datapoints contained in the dataset $x_i \in \mathcal{X}$, while a measurable error ϵ^2_{rnd} persists for out-of-distribution test samples $x_T \notin \mathcal{X}$, yielding a measure of epistemic uncertainty.

Next, we introduce the framework of neural tangent kernels, an analytical framework we intend to employ for the study of neural network and deep ensemble behavior. Consider a neural network $f(x,\theta_t) : \mathbb{R}^{n_{\text{in}}} \to \mathbb{R}$ with hidden layer widths $n_1,\dots,n_L = n$ and inputs $x \in \mathbb{R}^{n_{\text{in}}}$, a dataset $\mathcal{X}$, and labels $\mathcal{Y} = \{y_i \in \mathbb{R}\}_{i=1}^{N_D}$. Inputs x_i may, for example, be state-action tuples and labels y_i may be rewards. The network parameters $\theta_0 \in \mathbb{R}^{n_{\text{p}}}$ are initialized randomly $\theta_0 \sim \mathcal{N}(0,1)$ and updated with gradient descent with infinitesimal step sizes, also called gradient flow. In the limit of infinite width n, the function initialization $f(\cdot,\theta_0)$, as shown by Lee et al. (2018a), is equivalent to a Gaussian process (GP) prior with a specific kernel $\kappa : \mathbb{R}^{n_{\text{in}}} \times \mathbb{R}^{n_{\text{in}}} \to \mathbb{R}$ called the neural network Gaussian process (NNGP). The functional evolution of f through gradient flow is then governed by a *gradient* inner product kernel $\Theta : \mathbb{R}^{n_{\text{in}}} \times \mathbb{R}^{n_{\text{in}}} \to \mathbb{R}$ yielding

$$\Theta(x,x') = \nabla_\theta^\top f(x,\theta_0)\nabla_\theta f(x',\theta_0) \text{ and } \kappa(x,x') = \mathbb{E}[f(x,\theta_0)f(x',\theta_0)]\,. \tag{6.3}$$

Remarkably, seminal work by Jacot et al. (2018) showed that in the limit of infinite width and appropriate parametrization[1], the kernel Θ becomes deterministic and remains constant throughout training. This limiting kernel, referred to as the neural tangent kernel (NTK), leads to analytically tractable training dynamics for various loss functions, including the squared loss $\mathcal{L}(\theta_t) = \frac{1}{2}\|f(\mathcal{X};\theta_t) - \mathcal{Y}\|_2^2$. Owing to this, one can show (Jacot et al., 2018; Lee et al., 2020b) that for $t \to \infty$ post convergence function evaluations $f(\mathcal{X}_T,\theta_\infty)$ on a set of test points $\mathcal{X}_T$, too, are Gaussian with mean $\mathbb{E}[f(\mathcal{X}_T,\theta_\infty)] = \Theta_{\mathcal{X}_T\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}\mathcal{Y}$ and covariance

$$\begin{aligned}&\text{Cov}[f(\mathcal{X}_T,\theta_\infty)] = \\ &\quad \kappa_{\mathcal{X}_T\mathcal{X}_T} - \left(\Theta_{\mathcal{X}_T,\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}\kappa_{\mathcal{X}\mathcal{X}_T} + h.c.\right) + \Theta_{\mathcal{X}_T\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}\kappa_{\mathcal{X}\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}}^{-1}\Theta_{\mathcal{X}\mathcal{X}_T},\end{aligned} \tag{6.4}$$

where *h.c.* denotes the Hermitian conjugate of the preceding term and we used the shorthands $\Theta_{\mathcal{X}_1\mathcal{X}_2} = \Theta(\mathcal{X}_1,\mathcal{X}_2)$ and $\kappa_{\mathcal{X}_1\mathcal{X}_2} = \kappa(\mathcal{X}_1,\mathcal{X}_2)$. This expression provides a closed-form solution for the epistemic uncertainty captured by an infinite ensemble of neural networks (NNs) in the NTK regime trained with square losses. For example, the predictive variances of such ensembles are easily obtained as the diagonal entries of Eq. 6.4. While requiring an idealized setting, NTK theory offers a solid theoretical grounding for quantifying the behavior of deep ensembles and, by extension, myopic uncertainty estimates from related approaches. However, this analysis does not extend to value functions trained with TD losses and bootstrapping as is common in practical reinforcement learning settings.

[1] so-called NTK parametrization scales forward/backward passes appropriately, see Jacot et al. (2018)

6.2.2 Value Uncertainty

In contrast to myopic uncertainties, value uncertainty quantifies a model's lack of knowledge in the value $Q^\pi(s,a)$. As such it inherently depends on future trajectories induced by policies π. Due to this need to account for accumulated uncertainties over potentially long horizons, value uncertainty estimation typically renders more difficult than its myopic counterpart.

A widely used technique(An et al., 2021; Chen et al., 2017; Osband et al., 2016) to this end is the use of deep ensembles of value functions $Q(s,a,\theta_t) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ from random initializations θ_0. Q-functions are trained on transitional data $\mathcal{X}_{TD} = \{s_i, a_i\}_{i=1}^{N_D}$, $\mathcal{X}'_{TD} = \{s'_i, a'_i\}_{i=1}^{N_D}$, and $r = \{r_i\}_{i=1}^{N_D}$, where s'_i are samples from the transition kernel P and a'_i are samples from a policy π. Q-functions are then optimized through gradient descent on a temporal difference (TD) loss given by

$$\mathcal{L}(\theta_t) = \frac{1}{2} \| [\gamma Q^\pi(\mathcal{X}'_{TD}, \theta_t)]_{\text{sg}} + r - Q^\pi(\mathcal{X}_{TD}, \theta_t) \|_2^2, \tag{6.5}$$

where $[\cdot]_{\text{sg}}$ indicates a stop-gradient operation. Due to the stopping of gradient flow through $Q(\mathcal{X}', \theta_t)$, we refer to this operation as semi-gradient updates. Uncertainty estimates can then be obtained as the variance $\sigma_q^2(s,a) = \mathbb{V}_{\theta_0}[Q(s,a,\theta_t)]$ between ensembles of Q-functions from random initializations. While empirically successful, TD-trained deep ensembles are not as well understood as the supervised learning setting outlined in the previous section 6.2.1. Due to the use of bootstrapped TD losses, the closed-form NTK regime solutions in Eq. 6.4 do not apply to deep value function ensembles.

An alternative to the above approach is the propagation of myopic uncertainty estimates. Several prior methods(Luis et al., 2023; O'Donoghue et al., 2018; Zhou et al., 2020) formalize this setting under a model-based perspective, where transition models $\tilde{P}(\cdot|s,a)$ are sampled from a Bayesian posterior conditioned on transition data up to t. For acyclic MDPs, this setting permits a consistency condition similar to the Bellman equation that upper bounds value uncertainties recursively. While this approach devises a method for obtaining value uncertainties from propagated myopic uncertainties, several open problems remain, such as the tightness of model-free bounds of this kind (Janz et al., 2019; Van der Vaart et al., 2025) as well as how to prevent *underestimation* of these upper bounds due to the use of function approximation (Rashid et al., 2020; Zanger et al., 2024).

6.3 Universal Value-Function Uncertainties

Our method, *universal value-function uncertainties* (UVU), measures epistemic value uncertainty as the prediction errors between an online learner and a fixed

target network, similar in spirit to random network distillation (Burda et al., 2019b). However, while RND quantifies myopic uncertainty through immediate prediction errors, UVU modifies the training process of the online learner such that the resulting prediction errors reflect value-function uncertainties, that is, uncertainty about long-term returns under a given policy.

Our method centers around the interplay of two distinct neural networks: an online learner $u(s,a,z,\vartheta_t) : \mathcal{S}\times\mathcal{A}\times\mathcal{Z}\to\mathbb{R}$, parameterized by weights ϑ_t, and a fixed, randomly initialized target network $g(s,a,z,\psi_0) : \mathcal{S}\times\mathcal{A}\times\mathcal{Z}\to\mathbb{R}$, parameterized by weights ψ_0. Given a transition (s,a,s') and policy encoding z, we draw subsequent actions a' from a policy $\pi(\cdot|s',z)$. Then, we use the fixed target network g to generate synthetic rewards as

$$r_g^z(s,a,s',a') = g(s,a,z,\psi_0) - \gamma\, g(s',a',z,\psi_0). \tag{6.6}$$

While the weights ψ_0 of the target network remain fixed at initialization, the online network u is trained to minimize a TD loss using the synthetic reward r_g^π. Given a dataset $\mathcal{X} = \{s_i,a_i,z_i\}_{i=1}^{N_D}$, we have

$$\mathcal{L}(\vartheta_t) = \frac{1}{2N_D}\sum_i^{N_D}\Big(\gamma\big[u(s_i',a_i',z_i,\vartheta_t)\big]_{\text{sg}} + r_g^z(s_i,a_i,s_i',a_i') - u(s_i,a_i,z_i,\vartheta_t)\Big)^2, \tag{6.7}$$

where $[\cdot]_{\text{sg}}$ indicates a stop-gradient operation. For any tuple (s,a,z) ($\in\mathcal{X}$ or not), we measure predictive uncertainties as squared prediction errors between the learner and the target function

$$\epsilon(s,a,z,\vartheta_t,\psi_0)^2 = \big(u(s,a,z,\vartheta_t) - g(s,a,z,\psi_0)\big)^2. \tag{6.8}$$

The intuition behind this design is that, by construction, the value-function associated with policy $\pi(\cdot|s,z)$ and the synthetic rewards $r_g^z(s,a,s',a')$ exactly equals the fixed target network $g(s,a,z,\psi_0)$. As a sanity check, note that the target function $g(s,a,z,\psi_0)$ itself satisfies the Bellman equation for the policy $\pi(\cdot|s,z)$ and the synthetic reward definition in Eq. (6.6), constituting a *random value function* to r_g^z and hence achieves zero-loss according to Eq. (6.7). Therefore, if the dataset $\mathcal{X}$ sufficiently covers the dynamics induced by $\pi(\cdot|s,z)$, the online network $u(s,a,z,\vartheta_0)$ is able to recover $g(s,a,z,\psi_0)$ exactly, nullifying prediction errors. However, when data coverage is incomplete for the evaluated policy, minimization of the TD loss 6.7 is not sufficient for the online network $u(s,a,z,\vartheta_0)$ to recover target network predictions $g(s,a,z,\psi_0)$. This discrepancy is captured by the prediction errors, which quantify epistemic uncertainty regarding future gaps of the available data.

6.3.1 Building Intuition by an Example

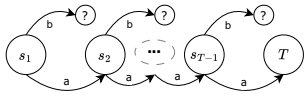


Figure 6.1: Chain MDP of length N with unexplored actions b.

To build intuition for how UVU operates and captures value uncertainty, we first consider a tabular setting with a simple chain MDP as illustrated in Figure 6.1. Suppose we collect data from a deterministic policy π_d using action a exclusively. Given this dataset, suppose we would like to estimate the uncertainty associated with the value $Q^{\pi(\cdot|s,z)}(s,a)$ of a policy $\pi(\cdot|s,z)$ that differs from the data-collection policy in that it chooses action "b" in s_3. In our tabular setting, we then initialize random tables u_{sa} and g_{sa}. For every transition (s_t, a_t, s_{t+1}) contained in our single-trajectory dataset, we draw $a_{t+1} \sim \pi(\cdot|s,z)$, compute the reward $r_{g,t}$ as $r_{g,t} = g_{s_t a_t} - \gamma g_{s_{t+1} a_{t+1}}$ and update table entries with the rule $u_{s_t a_t} \leftarrow r_{g,t} + \gamma u_{s_{t+1} a_{t+1}}$. Fig. 6.2 visualizes this process for several independently initialized tables (rows in Fig. 6.2) for the data-collecting policy π_d (left), and for the altered policy $\pi(\cdot|s,z)$ (right), which chooses action "b" in s_3. We outline how this procedure yields uncertainty estimates: We first note, that one may regard g as a randomly generated value-function, for which we derive the corresponding reward function as r_g. As g_{sa}, by construction, is the value-function corresponding to r_g, one may expect that the update rule applied to u_{sa} causes u_{sa} to recover g_{sa}. Crucially, however, this is only possible if sufficient data is available for the evaluated policy. When a policy diverges from available data, as occurs under $\pi(\cdot|s,z)$ in s_3, this causes an effective truncation of the collected trajectory. Consequently, $u_{s_1 a}$ and $u_{s_2 a}$ receive updates from $u_{s_3 b}$, which remains at its initialization, rather than inferring the reward-generating function g_{sa}. In the absence of long-term data, the empirical Bellman equations reflected in our updates do not uniquely determine the underlying value function g_{sa}. Indeed, both u_{sa} and g_{sa} incur zero TD-error in the r.h.s. of Fig. 6.2, yet differ significantly from each other. It is this ambiguity that UVU errors $(g_{sa} - u_{sa})^2$ quantify. To ensure u recovers g, longer rollouts under the policy $\pi(\cdot|s,z)$ are required to sufficiently constrain the solution space dictated by the Bellman equations (as seen in Fig. 6.2 left).

Figure 6.3 illustrates uncertainty estimates for the shown chain MDP using neural networks and for a whole family of policies $\pi(\cdot|s,z)$ which select the unexplored action b with probability $1-z$. We analyze the predictive variance of an ensemble of 128 universal Q-functions, each conditioned on the policy $\pi(\cdot|s,z)$. In the bottom row, we plot the squared prediction error of a single UVU model, averaged over 128 independent heads. Both approaches show peaked uncertainty in early sections, as policies are more likely to choose the unknown action "b" eventually, and low uncertainty closer to the terminal state and for z close to 1. A comparison with RND is provided in the Appendix C.1.3.

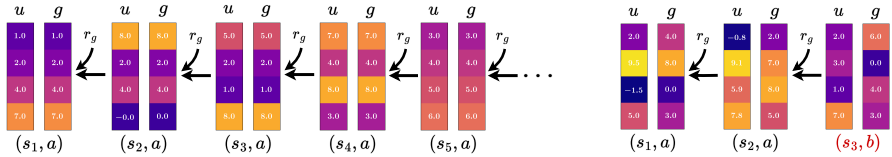


Figure 6.2: (*left*:) Illustration of uncertainty estimation in tabular UVU with 4 independently initialized tables for u and g. Access to full trajectory data allows u to recover g. (*right*:) By executing action "b", trajectories are effectively truncated, preventing u from recovering g. All plots use $\gamma = 0.7$.

6.4 What do Universal Value-Function Uncertainties Represent?

While the previous section provided intuition for UVU, we now derive an analytical characterization of the uncertainties captured by the prediction errors ϵ between a converged online learner u and the fixed target g. We turn to NTK theory to characterize the generalization properties of the involved neural networks in the limit of infinite width, allowing us to draw an exact equality between the squared predictions errors of UVU and the variance of universal value function ensembles.

In the following analysis, we use the notational shorthand $x = (s, a, z)$ and $x' = (s', a', z)$ and denote a neural network $f(x, \theta_t)$ with hidden layer widths $n_1, \ldots, n_L = n$, transitions from $\mathcal{X} = \{(s_i, a_i, z_i)\}_{i=1}^{N_D}$ to $\mathcal{X}' = \{(s_i', a_i', z_i)\}_{i=1}^{N_D}$, where $a_i' \sim \pi(\cdot|s_i', z_i)$, and rewards $r = \{r_i\}_{i=1}^{N_D}$. The evolution of the parameters θ_t under gradient descent with infinitesimal step sizes, also called gradient flow, is driven by the minimization of TD losses with

$$\frac{\mathrm{d}}{\mathrm{d}t}\theta_t = -\alpha \nabla_\theta \mathcal{L}(\theta_t), \quad \text{and} \quad \mathcal{L}(\theta_t) = \frac{1}{2} \| [\gamma f(\mathcal{X}', \theta_t)]_{\mathrm{sg}} + r - f(\mathcal{X}, \theta_t) \|_2^2 . \tag{6.9}$$

We study the dynamics induced by this parameter evolution in the infinite-width limit $n \to \infty$. In this regime, the learning dynamics of f become linear as the NTK becomes deterministic and stationary, permitting explicit closed-form expressions for the evolution of the function $f(x, \theta_t)$. In particular, we show that the post convergence function $\lim_{t \to \infty} f(x, \theta_t)$ is given by

$$f(x, \theta_\infty) = f(x, \theta_0) - \Theta_{x\mathcal{X}} \left(\Theta_{\mathcal{X}\mathcal{X}} - \gamma \Theta_{\mathcal{X}'\mathcal{X}} \right)^{-1} \left(f(\mathcal{X}, \theta_0) - (\gamma f(\mathcal{X}', \theta_0) + r) \right), \tag{6.10}$$

where $\Theta_{xx'}$ is the NTK of f. Proof is given in Appendix 6.9.1. This identity is useful to our analysis as it delineates any converged function $f(x, \theta_\infty)$ trained with TD losses 6.9 through its initialization $f(x, \theta_0)$. Theorem 6.1 leverages

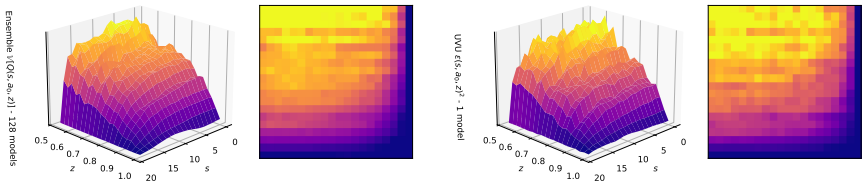


Figure 6.3: *From left to right, (1. and 2.):* Variance of an ensemble of 128 universal Q-functions trained on a chain MDP dataset. *(3. and 4.):* Value uncertainty as measured by UVU prediction errors with a single 128-headed model. All plots evaluate the "*a*" action of the chain MDP.

this deterministic dependency to express the distribution of post convergence functions over random initializations θ_0.

Theorem 6.1. Let $f(x,\theta_t)$ be a NN with L hidden layers of width $n_1,\ldots,n_L = n$ trained with gradient flow to reduce the TD loss $\mathcal{L}(\theta_t) = \frac{1}{2}\|\gamma[f(\mathcal{X}',\theta_t)]_{sg} + r - f(\mathcal{X},\theta_t)\|_2^2$. In the limit of infinite width $n\to\infty$ and time $t\to\infty$, the distribution of predictions $f(\mathcal{X}_T,\theta_\infty)$ on a set of test points $\mathcal{X}_T$ converges to a Gaussian with mean and covariance given by

$$\mathbb{E}_{\theta_0}\big[f(\mathcal{X}_T,\theta_\infty)\big] = \Theta_{\mathcal{X}_T\mathcal{X}}\Delta_{\mathcal{X}}^{-1}r,$$
$$\mathrm{Cov}_{\theta_0}\big[f(\mathcal{X}_T,\theta_\infty)\big] =$$
$$\kappa_{\mathcal{X}_T\mathcal{X}_T} - (\Theta_{\mathcal{X}_T\mathcal{X}}\Delta_{\mathcal{X}}^{-1}\Lambda_{\mathcal{X}_T} + h.c.) + (\Theta_{\mathcal{X}_T\mathcal{X}}\Delta_{\mathcal{X}}^{-1}(\Lambda_{\mathcal{X}} - \gamma\Lambda_{\mathcal{X}'})\Delta_{\mathcal{X}}^{-1^\top}\Theta_{\mathcal{X}\mathcal{X}_T}),$$

where $\Theta_{xx'}$ is the NTK, $\kappa_{xx'}$ is the neural network Gaussian process (NNGP) kernel, h.c. denotes the Hermitian conjugate, and

$$\Delta_{\tilde{\mathcal{X}}} = \Theta_{\mathcal{X}\tilde{\mathcal{X}}} - \gamma\Theta_{\mathcal{X}',\tilde{\mathcal{X}}}, \quad \text{and} \quad \Lambda_{\tilde{\mathcal{X}}} = \kappa_{\mathcal{X}\tilde{\mathcal{X}}} - \gamma\kappa_{\mathcal{X}',\tilde{\mathcal{X}}}.$$

Proof is provided in Appendix 6.9.1. Theorem 6.1 is significant as it allows us to formalize explicitly the expected behavior and uncertainties of neural networks trained with semi-gradient TD losses, including universal value function ensembles and the prediction errors of UVU. In particular, the variance of an ensemble of universal Q-functions $Q(\mathcal{X}_T,\theta_\infty)$ over random initializations θ_0 is readily given by the diagonal entries of the covariance matrix $\mathrm{Cov}[Q(\mathcal{X}_T,\theta_\infty)]$. Applied to the UVU setting, Theorem 6.1 gives an expression for the converged online network $u(x,\vartheta_\infty) = \Theta_{x\mathcal{X}}\Delta_{\mathcal{X}}^{-1}r_g^z$ trained with the synthetic rewards $r_g^z = g(\mathcal{X},\psi_0) - \gamma g(\mathcal{X}',\psi_0)$. From this, It is straightforward to obtain the distribution of post convergence prediction errors $\frac{1}{2}\epsilon(x,\vartheta_\infty,\psi_0)^2$. In Corollary 6.2, we use this insight to conclude that the expected squared prediction errors of UVU precisely match the variance of value functions $Q(x,\theta_\infty)$ from random initializations θ_0.

Corollary 6.2. Under the conditions of Theorem 6.1, let $u(x,\vartheta_\infty)$ be a converged online predictor trained with synthetic rewards generated by the fixed target network $g(x,\psi_0)$ with $r_g^z = g(\mathcal{X},\psi_0) - \gamma g(\mathcal{X}',\psi_0)$. Furthermore denote the variance of converged universal Q-functions $\mathbb{V}_{\theta_0}[Q(x,\theta_\infty)]$. Assume u, g, and Q are architecturally equal and parameters are drawn i.i.d. $\theta_0,\vartheta_0,\psi_0 \sim \mathcal{N}(0,1)$. The expected squared prediction error coincides with Q-function variance

$$\mathbb{E}_{\vartheta_0,\psi_0}\left[\tfrac{1}{2}\epsilon(x,\vartheta_\infty,\psi_0)^2\right] = \mathbb{V}_{\theta_0}\left[Q(x,\theta_\infty)\right], \tag{6.11}$$

where the l.h.s. expectation and r.h.s. variance are taken over random initializations $\vartheta_0,\psi_0,\theta_0$.

Proof is given in Appendix 6.9.1. This result provides the central theoretical justification for UVU: in the limit of infinite width, our measure of uncertainty, the expected squared prediction error between the online and target network, is mathematically equivalent to the variance one would obtain by training an ensemble of universal Q-functions.

In practice, we are moreover interested in the behavior of finite estimators, that is, ensemble variances are estimated with a finite number of models. We furthermore implement UVU with a number of multiple independent heads u_i and g_i with shared hidden layers. Corollary 6.3 shows that the distribution of the sample mean squared prediction error from M heads is identical to the distribution of the sample variance of an ensemble of $M+1$ independently trained universal Q-functions.

Corollary 6.3. Under the conditions of Theorem 6.1, consider online and target networks with M independent heads u_i, g_i, $i = 1,\dots,M$, each trained to convergence with errors $\epsilon_i(x,\vartheta_\infty,\psi_0)$. Let $\frac{1}{2}\bar{\epsilon}(x,\vartheta_\infty,\psi_0)^2 = \frac{1}{2M}\sum_{i=1}^{M}\epsilon_i(x,\vartheta_\infty,\psi_0)^2$ be the sample mean squared prediction error over M heads. Moreover, consider $M+1$ independent converged Q-functions $Q_i(x;\theta_\infty)$ and denote their sample variance $\bar{\sigma}^2_Q(x,\theta_\infty) = \frac{1}{M}\sum_{i=1}^{M+1}(Q_i(x;\theta_\infty) - \bar{Q}(x;\theta_\infty))^2$, where $\bar{Q}$ is the sample mean. The two estimators are identically distributed according to a scaled Chi-squared distribution

$$\tfrac{1}{2}\bar{\epsilon}(x,\vartheta_\infty,\psi_0)^2 \overset{D}{=} \bar{\sigma}^2_Q(x,\theta_\infty), \quad \bar{\sigma}^2_Q(x,\theta_\infty) \sim \frac{\sigma^2_Q}{M}\chi^2(M), \tag{6.12}$$

with M degrees of freedom and $\sigma^2_Q(x,\theta_\infty) = \mathbb{V}_{\theta_0}[Q(x,\theta_\infty)]$ is the analytical variance of converged Q-functions given by Theorem 6.1.

Proof is provided in Appendix 6.9.2. The distributional equivalence of these finite sample estimators provides theoretical motivation for using a multi headed architecture with shared hidden layers within a single UVU model and

its use as an estimator for ensemble variances of universal Q-functions. While the assumptions of infinite width and gradient flow are theoretical idealizations, several empirical results suggest that insights from the NTK regime can translate well to practical finite width deep learning models (Lee et al., 2020b; Liu et al., 2020; Tsilivis and Kempe, 2022), motivating further empirical investigation in Section 6.5.

6.5 Empirical Analysis

Our empirical analysis is designed to assess whether UVU can effectively quantify value function uncertainty in practical settings, comparing its performance against established baselines, particularly deep ensembles. Specifically, we aim to address the following questions:

1. Does the theoretical motivation for UVU hold in practice and do its uncertainty estimates enable effective decision-making comparable to deep ensembles?

2. How are uncertainty estimates generated by UVU affected by deviations from our theoretical analysis, namely finite network width?

To address these questions, we focus on an offline multitask RL setting with incomplete data where reliable uncertainty estimation is crucial to attain high performance.

6.5.1 Experimental Setup

In our experimental analysis, we use an offline variant of the `GoToDoor` environment from the Minigrid benchmark suite (Chevalier-Boisvert et al., 2023). An example view is shown in Figure 6.4 (c). In this task, the agent navigates a grid world containing four doors of different colors, placed at random locations and receives a task specification z indicating a target door color. Upon opening the correct door, the agent receives a reward and is placed in a random different location. Episodes are of fixed length and feature a randomly generated grid layout and random door positions / colors. In our experiments, we use variations of different difficulties by increasing maximum grid sizes.

Dataset collection. A dataset $\mathcal{D} = \{(s_i, a_i, r_i, z_i, s_i',)\}_{i=1}^{N_D}$ is collected using a policy that performs expertly but systematically fails for certain task/grid combinations (e.g., it can not successfully open doors on the "north" wall, irrespective of color or grid layout). Policies seeking to improve upon the behavior policy thus ought to deviate from the dataset, inducing value uncertainty.

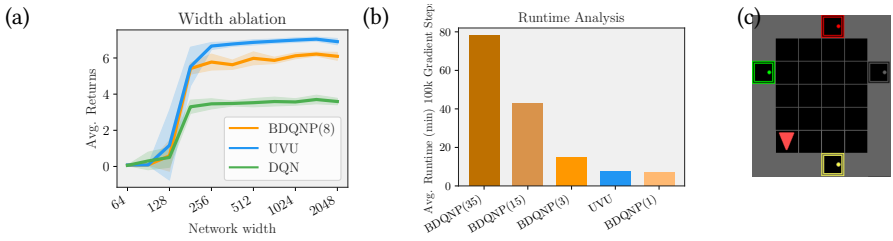


Figure 6.4: (a) Ablation on GoToDoor-10 with different network widths. Shaded region indicates standard deviations over 5 seeds. (b) Runtime of various ensemble sizes vs. UVU. Ensembles are implemented with vmap in JAX(Bradbury et al., 2018). (c) Illustration of the GoToDoor environment. The agent (red triangle) must navigate to the door indicated by the task specification z.

Task rejection protocol. All baselines implement an agent based on *deep Q-networks* (DQN, Mnih et al., 2015) trained in an offline fashion on $\mathcal{D}$. As the agents aim to learn an optimal policy for all grids and tasks contained in $\mathcal{D}$, the resulting greedy policy tends to deviate from the available data when the collecting policy is suboptimal. We employ a task-rejection protocol to quantify an agent's ability to recognize this divergence and the associated value uncertainty. As most task/grid combinations are contained in $\mathcal{D}$, though with varying levels of policy expertise, myopic uncertainty is not sufficient for fulfilling this task. Specifically, upon encountering the initial state s_0, the agent is given opportunity to reject a fixed selection of tasks (here door colors). It is subsequently given one of the remaining, non-rejected tasks and performance is measured by the average return achieved on the attempted task. Successful agents must thus either possess uncertainty estimates reliable enough to consistently reject tasks associated with a data/policy mismatch or rely on out-of-distribution generalization. Similar protocols, known as accuracy rejection curves, have been used widely in the supervised learning literature (Nadeem et al., 2009).

6.5.2 Results

We conduct experiments according to the above protocol and perform a quantitative evaluation of UVU and several baseline algorithms. All agents are trained offline and use the basic deep Q-network (DQN) architecture (Mnih et al., 2015) adapted for universal value functions, taking the task encoding z as an additional input to the state (details are provided in Appendix C.1). Specifically, we compare UVU against several baselines: A DQN baseline with random task rejection (DQN); Bootstrapped DQN with randomized priors (BDQNP) (Osband et al., 2019); A DQN adaptation of random network distillation (DQN-RND) (Burda et al., 2019b) and a version adapted with the

Table 6.1: Results of offline multitask RL with task rejection on different variations of the GoToDoor environment. Results are average evaluation returns of the best-performing policy over 10^5 gradient steps and intervals are 90% student's t confidence intervals.

N	DQN	BDQNP(3)	BDQNP(15)	BDQNP(35)	DQN-RND	DQN-RND-P	UVU (Ours)
5	5.50 ± .15	8.69 ± .24	**10.50 ± .04**	**10.58 ± .03**	3.94 ± .50	**10.41 ± .12**	**10.54 ± .03**
6	4.93 ± .12	7.66 ± .09	9.39 ± .04	**9.57 ± .04**	1.99 ± .40	9.28 ± .12	**9.54 ± .03**
7	4.58 ± .09	6.61 ± .16	8.49 ± .05	**8.75 ± .06**	2.66 ± .43	8.12 ± .23	**8.73 ± .04**
8	4.06 ± .12	5.91 ± .10	7.68 ± .05	7.92 ± .05	2.53 ± .54	7.40 ± .14	**8.03 ± .04**
9	3.66 ± .09	5.04 ± .08	6.69 ± .07	7.03 ± .13	2.39 ± .38	6.39 ± .19	**7.29 ± .10**
10	3.39 ± .11	4.64 ± .14	6.09 ± .13	**6.53 ± .16**	2.25 ± .48	5.64 ± .17	**6.72 ± .12**

uncertainty prior mechanism proposed by Zanger et al. (2024) (DQN-RND-P). Except for the DQN baseline, all algorithms reject tasks based on the highest uncertainty estimate, given the initial state s_0 and action a_0, which is chosen greedily by the agent.

Table 6.1 shows the average return achieved by each method on the GoToDoor experiment across different maximum grid sizes, with average runtimes displayed in Fig. 6.4 (b). This result addresses our first research question regarding the practical effectiveness of UVU compared to ensembles and other baseline methods. As shown, the standard DQN baseline performs significantly worse than uncertainty-based algorithms, indicating that learned Q-functions do not generalize sufficiently to counterbalance inadequate uncertainty estimation. Both small and large ensembles significantly improve performance by leveraging uncertainty to reject tasks and policies associated with missing data. RND-based agents perform well when intrinsic reward priors are used. Our approach scores highly and outperforms many of the tested baselines with statistical significance, indicating that it is indeed able to effectively quantify value uncertainty using a single-model multi-headed architecture.

We furthermore ablate UVU's dependency on network width, given that our theoretical analysis is situated in the infinite width limit. Fig. 6.4 (a) shows that UVU's performance scales similarly with network width to DQN and BDQNP baselines, indicating that finite-sized networks, provided appropriate representational capacity, are sufficient for effective uncertainty estimates.

6.6 Related Work

A body of literature considers the quantification of value function uncertainty in the context of exploration. Early works (Dearden et al., 1998; Engel et al., 2005) consider Bayesian adoptions of model-free RL algorithms. More recent works provide theoretical analyses of the Bayesian model-free setting and correct applications thereof (Fellows et al., 2021; Schmitt et al., 2023; Van der Vaart et al., 2025), which is a subject of debate due to the use TD losses. Several works

furthermore derive provably efficient model-free algorithms using frequentist upper bounds on values in tabular (Jin et al., 2018; Strehl et al., 2006) and linear settings (Jin et al., 2020). Similarly, Yang et al. (2020) derive provably optimisic bounds of value functions in the NTK regime, but in contrast to our work uses local bonuses to obtain these. The exact relationship between bounds derived from local bonuses and the functional variance in ensemble or Bayesian settings remains open.

The widespread use and empirical success of ensembles for uncertainty quantification in deep learning (Dietterich, 2000; Lakshminarayanan et al., 2017) has motivated several directions of research towards a better theoretical understanding of their behavior. Following seminal works by Jacot et al. (2018) and Lee et al. (2020b) who characterize NN learning dynamics in the NTK regime, a number of works have connected deep ensembles to Bayesian interpretations (D'Angelo and Fortuin, 2021; He et al., 2020). Moreover, a number of papers have studied the learning dynamics of model-free RL: in the overparametrized linear settings (Xiao et al., 2021); in neural settings for single (Cai et al., 2019) and multiple layers (Wai et al., 2020); to analyze generalization behavior (Lyle et al., 2022) with linear and second-order approximations. It should be noted that the aforementioned do not focus on probabilistic descriptions of posterior distributions in the NTK regime. In contrast, our work provides probabilistic closed-form solutions for this setting with semi-gradient TD learning.

In practice, the use of deep ensembles is common in RL, with applications ranging from efficient exploration (Chen et al., 2017; Nikolov et al., 2019; Osband et al., 2016; 2019; Zanger et al., 2024) to off-policy or offline RL (An et al., 2021; Chen et al., 2021; Lee et al., 2021) and conservative or safe RL (Hoel et al., 2023; Lee et al., 2022; Lütjens et al., 2019). Single model methods that aim to reduce the computational burden of ensemble methods typically operate as myopic uncertainty estimators (Burda et al., 2019b; Lahlou et al., 2021; Pathak et al., 2017; Zanger et al., 2025a) and require additional mechanisms (Janz et al., 2019; Luis et al., 2023; O'Donoghue et al., 2018; Zhou et al., 2020).

6.7 Limitations and Assumptions

In this section, we detail central theoretical underpinnings and idealizations upon which our theoretical analysis is built.

A central element of our theoretical analysis is the representation of neural network learning dynamics via the NTK, an object in the theoretical limit of infinite network width. The established NTK framework, where the kernel is deterministic despite random initialziation and and constant throughout training, typically applies to fully connected networks with NTK parameterization,

optimized using a squared error loss (Jacot et al., 2018). Our framework instead accommodates a semi-gradient TD loss, and thereby introduces an additional prerequisite for ensuring the convergence of these dynamics: the positive definiteness of the matrix expression $\Theta_{\mathcal{X}\mathcal{X}} - \gamma\Theta_{\mathcal{X}'\mathcal{X}}$. This particular constraint is more a characteristic inherent to the TD learning paradigm itself than a direct consequence of the infinite-width abstraction. Indeed, the design of neural network architectures that inherently satisfy such stability conditions for TD learning continues to be an active area of contemporary research (Gallici et al., 2024; Yue et al., 2023). The modeling choice of semi-gradient TD losses moreover does not incorporate the use of target networks, where bootstrapped values do not only stop gradients but are generated by a separate network altogether that slowly moves towards the online learner. Our analysis moreover considers the setting of *offline policy evaluation*, that is, we do not assume that additional data is acquired during learning and that policies evaluated for value learning remain constant. The assumption of a fixed, static dataset diverges from the conditions of online reinforcement learning with control, where the distribution of training data $(\mathcal{X}, \mathcal{X}')$ typically evolves as the agent interacts with its environment, both due to its collection of novel transitions and due to adjustments to the policy, for example by use of a Bellman optimality operator. Lastly, our theoretical model assumes, primarily for simplicity, that learning occurs under gradient flow with infinitesimally small step sizes and with updates derived from full-batch gradients. Both finite-sized gradient step sizes and stochastic minibatching has been treated in the literature, albeit not in the TD learning setting (Jacot et al., 2018; Lee et al., 2020b; Liu et al., 2020; Yang, 2019). We believe our analysis could be extended to these settings without major modifications.

6.8 Discussion

In this work, we introduced *universal value-function uncertainties* (UVU), an efficient single-model method for uncertainty quantification in value functions. Our method measures uncertainties as prediction error between a fixed, random target network and an online learner trained with a TD loss. This induces prediction errors that reflect long-term, policy-dependent uncertainty rather than myopic novelty. One of our core contributions is a thorough theoretical analysis of this approach via neural tangent kernel theory, which, in the limit of infinite network width, establishes an equivalence between UVU errors and the variance of ensembles of universal value functions. Empirically, UVU achieves performance comparable and sometimes superior to sizeable deep ensembles and other baselines in challenging offline task-rejection settings, while offering substantial computational savings.

We believe our work opens up several avenues for future research: Although our NTK analysis provides a strong theoretical backing, it relies on idealized assumptions, notably the limit of infinite network width, as outlined above. Our experiments suggest UVU's performance is robust in practical finite-width regimes (Figure 6.4), yet bridging this gap between theory and practice remains an area for future work. On a related note, analysis in the NTK regime typically eludes feature learning. Combinations of UVU with representation learning approaches such as self-predictive auxiliary losses (Fujimoto et al., 2023; Guo et al., 2022; Schwarzer et al., 2020) are, in our view, a very promising avenue for highly challenging exploration problems. Furthermore, while our approach estimates uncertainty for given policies, it does not devise a method for obtaining diverse policies and encodings thereof. We thus believe algorithms from the unsupervised RL literature(Touati and Ollivier, 2021; Zheng et al., 2023) naturally integrate with our approach. In conclusion, we believe UVU provides a strong foundation for future developments in uncertainty-aware agents that are both capable and computationally feasible.

6.9 Proofs

This section provides proofs and further theoretical results for UVU.

6.9.1 Infinite-Width Learning Dynamics

We begin by deriving learning dynamics for general functions with TD losses and gradient descent, before analyzing the post training distribution of deep ensembles and prediction errors of UVU.

Linearized Learning Dynamics with Temporal Difference Losses

We analyze the learning dynamics of a function trained using semi-gradient TD losses on a fixed dataset of transitions $\mathcal{X}, \mathcal{X}'$. Let $f(x, \theta_t)$ denote a NN of interest with depth L and widths $n_1, \ldots, n_{L-1} = n$.

Proposition 6.4. In the limit of infinite width $n \to \infty$ and infinite time $t \to \infty$, the function $f(x, \theta_t)$ converges to

$$f(x, \theta_\infty) = f(x, \theta_0) - \Theta_{x\mathcal{X}} \left(\Theta_{\mathcal{X}\mathcal{X}} - \gamma \Theta_{\mathcal{X}'\mathcal{X}}\right)^{-1} \left(f(\mathcal{X}, \theta_0) - (\gamma f(\mathcal{X}', \theta_0) + r)\right), \tag{6.13}$$

where $\Theta_{xx'}$ is the neural tangent kernel of f.

Proof. We begin by linearizing the function f around its initialization parameters θ_0:

$$f_{\text{lin}}(x, \theta_t) = f(x, \theta_0) + \nabla_\theta^\top f(x, \theta_0)(\theta_t - \theta_0). \tag{6.14}$$

We assume gradient descent updates with infinitesimal step size and a learning rate α on the loss

$$\mathcal{L}(\theta_t) = \tfrac{1}{2}\|\gamma f_{\text{lin}}(\mathcal{X}', \theta_t)_{\text{sg}} + r - f_{\text{lin}}(\mathcal{X}, \theta_t)\|_2^2, \tag{6.15}$$

yielding the parameter evolution

$$\frac{\mathrm{d}}{\mathrm{d}t}\theta_t = -\alpha \nabla_\theta \mathcal{L}(\theta_t). \tag{6.16}$$

Setting $w_t = \theta_t - \theta_0$ and find the learning dynamics:

$$\frac{\mathrm{d}}{\mathrm{d}t} w_t = -\alpha \nabla_\theta f(\mathcal{X}, \theta_0)\big(f_{\text{lin}}(\mathcal{X}, \theta_t) - (\gamma f_{\text{lin}}(\mathcal{X}', \theta_t) + r)\big). \tag{6.17}$$

Thus, the evolution of the linearized function is given by

$$\frac{\mathrm{d}}{\mathrm{d}t} f_{\text{lin}}(x, \theta_t) = -\alpha \nabla_\theta^\top f(x, \theta_0) \nabla_\theta f(\mathcal{X}, \theta_0)\big(f_{\text{lin}}(\mathcal{X}, \theta_t) - (\gamma f_{\text{lin}}(\mathcal{X}', \theta_t) + r)\big). \tag{6.18}$$

Letting $\delta_{\text{TD}}(\theta_t) = f_{\text{lin}}(\mathcal{X}, \theta_t) - (\gamma f_{\text{lin}}(\mathcal{X}', \theta_t) + r)$, we obtain the differential equation

$$\frac{\mathrm{d}}{\mathrm{d}t}\delta_{\text{TD}}(\theta_t) = -\alpha\big(\Theta^{t_0}_{\mathcal{X}\mathcal{X}} - \gamma \Theta^{t_0}_{\mathcal{X}'\mathcal{X}}\big)\delta_{\text{TD}}(\theta_t), \tag{6.19}$$

where $\Theta^{t_0}_{xx'} = \nabla_\theta^\top f(x, \theta_0) \nabla_\theta f(x', \theta_0)$ is the (empirical) tangent kernel corresponding to $f_{\text{lin}}(x, \theta_t)$. Since the linearization $f_{\text{lin}}(x, \theta_t)$ has constant gradients $\nabla_\theta f(x, \theta_0)$, the above differential equation is linear and solvable so long as the matrix $\Theta^{t_0}_{\mathcal{X}\mathcal{X}} - \gamma \Theta^{t_0}_{\mathcal{X}'\mathcal{X}}$ is positive definite. With an exponential ansatz, we obtain the solution

$$\delta_{\text{TD}}(\theta_t) = e^{-\alpha t(\Theta^{t_0}_{\mathcal{X}\mathcal{X}} - \gamma \Theta^{t_0}_{\mathcal{X}'\mathcal{X}})}\delta_{\text{TD}}(\theta_0), \tag{6.20}$$

where e^X is a matrix exponential. Reintegrating yields the explicit evolution of predictions

$$f_{\text{lin}}(x, \theta_t) = f(x, \theta_0) + \int_0^t \frac{\mathrm{d}}{\mathrm{d}t'} f_{\text{lin}}(x, \theta_{t'})\, \mathrm{d}t' \tag{6.21}$$

$$= f(x, \theta_0) - \Theta^{t_0}_{x\mathcal{X}}\big(\Theta^{t_0}_{\mathcal{X}\mathcal{X}} - \gamma \Theta^{t_0}_{\mathcal{X}'\mathcal{X}}\big)^{-1}\Big(e^{-\alpha t(\Theta^{t_0}_{\mathcal{X}\mathcal{X}} - \gamma \Theta^{t_0}_{\mathcal{X}'\mathcal{X}})} - I\Big)\delta_{\text{TD}}(\theta_0). \tag{6.22}$$

Jacot et al. (2018) show that in the limit of infinite layer widths of the neural network, the NTK $\Theta^{t_0}_{xx'}$ becomes deterministic and constant $\Theta^{t_0}_{xx'} \to \Theta_{xx'}$. As a consequence, the linear approximation $f_{\text{lin}}(x; \theta_t)$ becomes exact w.r.t. the original function $\lim_{\text{width}\to\infty} f_{\text{lin}}(x; \theta_t) = f(x, \theta_t)$ (Lee et al., 2020b).

□

Remark on the constancy of the NTK in TD learning. We note here, that our proof assumed the results by Jacot et al. (2018) to hold for the case of semi-gradient TD updates, namely that the NTK becomes deterministic and constant $\Theta^{t_0}_{xx'} \to \Theta_{xx'}$ in the limit of infinite width under the here shown dynamics. First, the determinacy of the NTK at initialization follows from the law of large numbers and applies in our case equally as in the least squares case. The constancy of the NTK throughout training is established by Theorem 2 in Jacot et al. (2018), which we restate informally below.

Theorem 6.5. (Jacot et al., 2018) In the limit of infinite layer widths $n \to \infty$ and $n = n_1, \ldots, n_L$, the kernel $\Theta^{t_0}_{xx'}$ converges uniformly on the interval $t \in [0, T]$ to the constant neural tangent kernel

$$\Theta^{t_0}_{xx'} \to \Theta_{xx'},$$

provided that the integral $\int_0^T \|d_t\|_2 \, dt$ stays bounded. Here, $d_t \in \mathbb{R}^{N_D}$ is the training direction of the parameter evolution such that $\frac{\mathrm{d}}{\mathrm{d}t}\theta_t = -\alpha \nabla_\theta f(\mathcal{X}, \theta) d_t$

In the here studied case of semi-gradient TD learning, the parameter evolution (as outlined above in Eq. (6.17)) is described by the gradient $\nabla_\theta f(\mathcal{X}, \theta_0)$ and the *training direction* d_t according to

$$\frac{\mathrm{d}}{\mathrm{d}t}\theta_t = -\alpha \nabla_\theta f(\mathcal{X}, \theta_0) \underbrace{\left(f_{\text{lin}}(\mathcal{X}, \theta_t) - (\gamma f_{\text{lin}}(\mathcal{X}', \theta_t) + r) \right)}_{d_t}, \tag{6.23}$$

where the training direction is given by $d_t = f_{\text{lin}}(\mathcal{X}, \theta_t) - (\gamma f_{\text{lin}}(\mathcal{X}', \theta_t) + r) = \delta_{TD}(\theta_t)$. Provided that the matrix $\Theta^{t_0}_{\mathcal{X}\mathcal{X}} - \gamma \Theta^{t_0}_{\mathcal{X}'\mathcal{X}}$ is positive definite, the norm of the training direction $\|d_t\|_2$ decays exponentially by Eq. 6.20. This implies

$$\|d_t\|_2 < \|d_0\|_2 e^{-t\lambda_{\min}}, \tag{6.24}$$

where $\lambda_{\min}$ is the smallest eigenvalue of $\Theta^{t_0}_{\mathcal{X}\mathcal{X}} - \gamma \Theta^{t_0}_{\mathcal{X}'\mathcal{X}}$. Assuming $\Theta^{t_0}_{\mathcal{X}\mathcal{X}} - \gamma \Theta^{t_0}_{\mathcal{X}'\mathcal{X}}$ is positive definite, $\lambda_{\min}$ is positive and as a consequence, we have

$$\int_0^\infty \|d_t\|_2 \, dt < \int_0^\infty \|d_0\|_2 e^{-t\lambda_{\min}} \, dt < \infty, \tag{6.25}$$

bounding the required integral of Theorem 6.5 for any T and establishing $\Theta^{t_0}_{xx'} \to \Theta_{xx'}$ uniformly on the interval $[0, \infty)$ (see Theorem 2 in Jacot et al. (2018) for detailed proof for the last statement).

We note, however, that the condition for $\Theta^{t_0}_{\mathcal{X}\mathcal{X}} - \gamma \Theta^{t_0}_{\mathcal{X}'\mathcal{X}}$ to be positive definite is, for any $\gamma > 0$, stronger than in the classical results for supervised learning with least squares regression. While $\Theta_{\mathcal{X}\mathcal{X}}$ can be guaranteed to be

positive definite for example by restricting $\mathcal{X}$ to lie on a unit-sphere, $x_i \in \mathcal{X}$ to be unique, and by assuming non-polynomial nonlinearities in the neural network (so as to prevent rank decay in the network expressivity), the condition is harder to satisfy in the TD learning setting. Here, the eigenspectrum of $\Theta^{t_0}_{\mathcal{X}\mathcal{X}} - \gamma\Theta^{t_0}_{\mathcal{X}'\mathcal{X}}$ tends to depend on the transitions $\mathcal{X} \to \mathcal{X}'$ themselves and thus is both dependent on the discount γ as well as the interplay between gradient structures of the NTK and the MDP dynamics.

We also note here, that this is not primarily a limitation of applying NTK theory to TD learning, but is reflected in practical experience: TD learning can, especially in offline settings, indeed be instable and diverge. Instability of this form is thus inherent to the learning algorithm rather than an artifact of our theoretical treatment. Informally, one approach towards guaranteeing positive definiteness of $\Theta^{t_0}_{\mathcal{X}\mathcal{X}} - \gamma\Theta^{t_0}_{\mathcal{X}'\mathcal{X}}$ is by enforcing diagonal dominance, appealing to the Gershgorin circle theorem (Gerschgorin, 1931). For a matrix $A = [a_{ij}]$, every real eigenvalue λ must lie in

$$a_{ii} - R_i \leq \lambda \leq a_{ii} + R_i\,, \tag{6.26}$$

where $R_i = \sum_{i \neq j} |a_{ij}|$ is the sum of off-diagonal elements of a row i. In other words, a lower bound on the smallest real eigenvalue can be increased by increasing diagonal entries a_{ii} while decreasing off-diagonal elements a_{ij}. In the TD learning setting, this translates to gradient conditioning, e.g., by ensuring $\|\nabla_\theta f(x,\theta)\|_2 = \|\nabla_\theta f(x',\theta)\|_2 = C$ for any pair x, x', guaranteeing cross-similarities to be smaller than self-similarities. Indeed several recent works pursue similar strategies to stabilize offline TD learning (Gallici et al., 2024; Yue et al., 2023) and rely on architectural elements like layer normalization (Ba et al., 2016) to shape gradient norms.

Post Training Function Distribution with Temporal Difference Dynamics

We now aim to establish the distribution of post-training functions $f(x, t_\infty)$ when initial parameters θ_0 are drawn randomly i.i.d. For the remainder of this section, we will assume the infinite width limit, s.t. $f_{\text{lin}}(x,\theta_\infty) = f(x,\theta_\infty)$ and $\Theta^{t_0}_{xx'} = \Theta_{xx'}$. The post-training function $f(x,\theta_\infty)$ is given by

$$f(x,\theta_\infty) = f(x,\theta_0) - \Theta_{x\mathcal{X}}\big(\Theta^{t_0}_{\mathcal{X}\mathcal{X}} - \gamma\Theta^{t_0}_{\mathcal{X}'\mathcal{X}}\big)^{-1}\big(f(\mathcal{X},\theta_0) - (\gamma f(\mathcal{X}',\theta_0) + r)\big), \tag{6.27}$$

and is thus a deterministic function of the initialization θ_0.

Theorem 6.1. Let $f(x,\theta_t)$ be a NN with L hidden layers of width $n_1,\ldots,n_L = n$ trained with gradient flow to reduce the TD loss $\mathcal{L}(\theta_t) = \frac{1}{2}\|\gamma[f(\mathcal{X}',\theta_t)]_{sg} + r - f(\mathcal{X},\theta_t)\|_2^2$. In the limit of infinite width $n \to \infty$ and time $t \to \infty$, the distribution

of predictions $f(\mathcal{X}_T,\theta_\infty)$ on a set of test points $\mathcal{X}_T$ converges to a Gaussian with mean and covariance given by

$$\mathbb{E}_{\theta_0}\big[f(\mathcal{X}_T,\theta_\infty)\big] = \Theta_{\mathcal{X}_T\mathcal{X}}\Delta_{\mathcal{X}}^{-1}r,$$
$$\mathrm{Cov}_{\theta_0}\big[f(\mathcal{X}_T,\theta_\infty)\big] =$$
$$\kappa_{\mathcal{X}_T\mathcal{X}_T} - (\Theta_{\mathcal{X}_T\mathcal{X}}\Delta_{\mathcal{X}}^{-1}\Lambda_{\mathcal{X}_T} + h.c.) + (\Theta_{\mathcal{X}_T\mathcal{X}}\Delta_{\mathcal{X}}^{-1}(\Lambda_{\mathcal{X}} - \gamma\Lambda_{\mathcal{X}'})\Delta_{\mathcal{X}}^{-1^\top}\Theta_{\mathcal{X}\mathcal{X}_T}),$$

where $\Theta_{xx'}$ is the NTK, $\kappa_{xx'}$ is the NNGP kernel, h.c. denotes the Hermitian conjugate, and

$$\Delta_{\tilde{\mathcal{X}}} = \Theta_{\mathcal{X}\tilde{\mathcal{X}}} - \gamma\Theta_{\mathcal{X}'\tilde{\mathcal{X}}}, \quad \text{and} \quad \Lambda_{\tilde{\mathcal{X}}} = \kappa_{\mathcal{X}\tilde{\mathcal{X}}} - \gamma\kappa_{\mathcal{X}'\tilde{\mathcal{X}}}.$$

Proof. We begin by introducing a column vector of post-training function evaluations on a set of test points $\mathcal{X}_T$, and the training data $\mathcal{X}$ and $\mathcal{X}'$. Moreover, we introduce the shorthand

$$\Delta_{\mathcal{X}} = \Theta_{\mathcal{X}\mathcal{X}} - \gamma\Theta_{\mathcal{X}'\mathcal{X}}, \tag{6.28}$$

and similarly $\Delta_{\mathcal{X}'} = \Theta_{\mathcal{X}\mathcal{X}'} - \gamma\Theta_{\mathcal{X}'\mathcal{X}'}$. The vector can then be compactly described in block matrix notation by

$$\underbrace{\begin{pmatrix} f(\mathcal{X}_T,\theta_\infty) \\ f(\mathcal{X},\theta_\infty) \\ f(\mathcal{X}',\theta_\infty) \end{pmatrix}}_{f^\infty} =$$
$$\underbrace{\begin{pmatrix} I & -\Theta_{\mathcal{X}_T\mathcal{X}}\Delta_{\mathcal{X}}^{-1} & \gamma\Theta_{\mathcal{X}_T\mathcal{X}}\Delta_{\mathcal{X}}^{-1} \\ I & -\Theta_{\mathcal{X}\mathcal{X}}\Delta_{\mathcal{X}}^{-1} & \gamma\Theta_{\mathcal{X}\mathcal{X}}\Delta_{\mathcal{X}}^{-1} \\ I & -\Theta_{\mathcal{X}'\mathcal{X}}\Delta_{\mathcal{X}}^{-1} & \gamma\Theta_{\mathcal{X}'\mathcal{X}}\Delta_{\mathcal{X}}^{-1} \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} f(\mathcal{X}_T,\theta_0) \\ f(\mathcal{X},\theta_0) \\ f(\mathcal{X}',\theta_0) \end{pmatrix}}_{f^0} + \underbrace{\begin{pmatrix} \Theta_{\mathcal{X}_T\mathcal{X}}\Delta_{\mathcal{X}}^{-1}r \\ \Theta_{\mathcal{X}\mathcal{X}}\Delta_{\mathcal{X}}^{-1}r \\ \Theta_{\mathcal{X}'\mathcal{X}}\Delta_{\mathcal{X}}^{-1}r \end{pmatrix}}_{b}. \tag{6.29}$$

Lee et al. (2018a) show that neural networks with random Gaussian initialization θ_0 (including NTK parametrization) are described by the NNGP $f(\mathcal{X}_T,\theta_0) \sim \mathcal{N}(0,\kappa_{\mathcal{X}_T\mathcal{X}_T})$ with $\kappa_{\mathcal{X}_T\mathcal{X}_T} = \mathbb{E}[f(\mathcal{X}_T,\theta_0)f(\mathcal{X}_T,\theta_0)^\top]$. By extension, the initializations f^0 are jointly Gaussian with zero mean and covariance matrix

$$\mathrm{Cov}[f^0] = \underbrace{\begin{pmatrix} \kappa_{\mathcal{X}_T\mathcal{X}_T} & \kappa_{\mathcal{X}_T\mathcal{X}} & \kappa_{\mathcal{X}_T\mathcal{X}'} \\ \kappa_{\mathcal{X}\mathcal{X}_T} & \kappa_{\mathcal{X}\mathcal{X}} & \kappa_{\mathcal{X}\mathcal{X}'} \\ \kappa_{\mathcal{X}'\mathcal{X}_T} & \kappa_{\mathcal{X}'\mathcal{X}} & \kappa_{\mathcal{X}'\mathcal{X}'} \end{pmatrix}}_{K}. \tag{6.30}$$

As the post-training function evaluations f^∞ given in Eq. (6.29) are affine transformations of the multivariate Gaussian random variables $f^0 \sim \mathcal{N}(0,K)$, they themselves are multivariate Gaussian with distribution $f^\infty \sim \mathcal{N}(b, AKA^\top)$.

We are content with obtaining an expression for the distribution of $f(\mathcal{X}_T, \theta_\infty)$ and thus in the following focus on the top-left entry of the block matrix $(AKA^\top)_{11}$. For notational brevity, we introduce the following shorthand notations

$$\Lambda_{\tilde{\mathcal{X}}} = \kappa_{\mathcal{X}\tilde{\mathcal{X}}} - \gamma\kappa_{\mathcal{X}'\tilde{\mathcal{X}}} \tag{6.31}$$

After some rearranging, one obtains the following expression for the covariance $\mathrm{Cov}(f^\infty_{\mathcal{X}_T})$

$$\mathrm{Cov}_{\theta_0}\big[f(\mathcal{X}_T, \theta_\infty)\big] = \\ \kappa_{\mathcal{X}_T\mathcal{X}_T} - (\Theta_{\mathcal{X}_T\mathcal{X}}\Delta^{-1}_{\mathcal{X}}\Lambda_{\mathcal{X}_T} + h.c.) + (\Theta_{\mathcal{X}_T\mathcal{X}}\Delta^{-1}_{\mathcal{X}}(\Lambda_{\mathcal{X}} - \gamma\Lambda_{\mathcal{X}'})\Delta^{-1^\top}_{\mathcal{X}}\Theta_{\mathcal{X}\mathcal{X}_T}).$$

□

Distribution of UVU Predictive Errors

We now aim to find an analytical description of the predictive errors as generated by our approach. For this, let $u(x, \vartheta_t)$ denote the predictive (online) network and $g(x; \psi_0)$ the fixed target network. We furthermore denote $\epsilon(x, \vartheta_t, \psi_0) = u(x, \vartheta_t) - g(x, \psi_0)$ the prediction error between online and target network.

Corollary 6.2. Under the conditions of Theorem 6.1, let $u(x, \vartheta_\infty)$ be a converged online predictor trained with synthetic rewards generated by the fixed target network $g(x, \psi_0)$ with $r^z_g = g(\mathcal{X}, \psi_0) - \gamma g(\mathcal{X}', \psi_0)$. Furthermore denote the variance of converged universal Q-functions $\mathbb{V}_{\theta_0}[Q(x, \theta_\infty)]$. Assume u, g, and Q are architecturally equal and parameters are drawn i.i.d. $\theta_0, \vartheta_0, \psi_0 \sim \mathcal{N}(0, 1)$. The expected squared prediction error coincides with Q-function variance

$$\mathbb{E}_{\vartheta_0, \psi_0}\big[\tfrac{1}{2}\epsilon(x, \vartheta_\infty, \psi_0)^2\big] = \mathbb{V}_{\theta_0}\big[Q(x, \theta_\infty)\big], \tag{6.11}$$

where the l.h.s. expectation and r.h.s. variance are taken over random initializations $\vartheta_0, \psi_0, \theta_0$.

Proof. Since our algorithm uses semi-gradient TD losses to train $u(x, \vartheta_t)$, the linearized dynamics of Theorem (6.1) apply. However, we consider a fixed target network $g(x; \psi_0)$ to produce synthetic rewards according to

$$r_g = g(x, \psi_0) - \gamma g(x', \psi_0). \tag{6.32}$$

With the post training function as described by Eq. 6.27, the post-training prediction error in a query point x for this reward is given by

$$u(x, \vartheta_\infty) - g(x, \psi_0) = u(x, \vartheta_0) - g(x, \psi_0) - \Theta_{x\mathcal{X}}\Delta^{-1}_{\mathcal{X}}(u(\mathcal{X}, \vartheta_0) \\ - (\gamma u(\mathcal{X}', \vartheta_0) + g(\mathcal{X}, \psi_0) - \gamma g(\mathcal{X}', \psi_0))). \tag{6.33}$$

We again use the shorthand $\epsilon^t = (\epsilon(\mathcal{X}_T, \vartheta_t, \psi_0), \epsilon(\mathcal{X}, \vartheta_t, \psi_0), \epsilon(\mathcal{X}', \vartheta_t, \psi_0))^\top$ and reusing the block matrix A from Eq. 6.29, we can write

$$\epsilon^\infty = A\epsilon^0. \tag{6.34}$$

By assumption, $u(x, \vartheta_0)$ and $g(x, \psi_0)$ are architecturally equivalent and initialized i.i.d., and ϵ^0 is simply the sum of two independent Gaussian vectors with covariance $\mathrm{Cov}[\epsilon^0] = 2K$. We conclude that prediction errors ϵ^∞ are Gaussian with distribution $\epsilon^\infty \sim \mathcal{N}(0, 2AKA^\top)$. Taking the diagonal of the covariance matrix $AKA_{11}^\top$, we obtain

$$\mathbb{E}_{\vartheta_0,\psi_0}\left[\tfrac{1}{2}\epsilon(x, \vartheta_\infty, \psi_0)^2\right] = \mathbb{V}_{\theta_0}\left[Q(x, \theta_\infty)\right], \tag{6.35}$$

where

$$\mathbb{V}_{\theta_0}\left[Q(x, \theta_\infty)\right] = \kappa_{xx} - (\Theta_{x\mathcal{X}}\Delta_{\mathcal{X}}^{-1}\Lambda_x + h.c.) + (\Theta_{x\mathcal{X}}\Delta_{\mathcal{X}}^{-1}(\Lambda_{\mathcal{X}} - \gamma\Lambda_{\mathcal{X}'})\Delta_{\mathcal{X}}^{-1^\top}\Theta_{\mathcal{X}x}). \tag{6.36}$$

□

6.9.2 Error Distribution with Multiheaded Architectures

We now show results concerning the equivalence of multiheaded UVU prediction errors and finite ensembles of Q-functions. We first outline proofs for two results by Lee et al. (2018a) and Jacot et al. (2018), which rely on in our analysis.

Neural Network Gaussian Process Propagation and Independence

Consider a deep neural network f with L layers. Let $z_i^l(x)$ denote the i-th output of layer $l = 1, \ldots, L$, defined recursively as:

$$z_i^l(x) = \sigma_b b_i^l + \frac{\sigma_w}{\sqrt{n_{l-1}}} \sum_{j=1}^{n_{l-1}} w_{ij}^l x_j^l(x), \quad x_j^l(x) = \phi(z_j^{l-1}(x)), \tag{6.37}$$

where n_l is the width of layer l with $n_0 = n_{\text{in}}$ and $x^0 = x$. Further, σ_w and σ_b are constant variance multipliers, weights w^l and biases b^l are initialized i.i.d. with $\mathcal{N}(0, 1)$, and ϕ is a Lipschitz-continuous nonlinearity. The i-th function output $f_i(x)$ of the NN is then given by $f_i(x) = z_i^L(x)$.

Proposition 6.6 (Lee et al. (2018a)). *At initialization and in the limit $n_1 \ldots, n_{L-1} \to \infty$, the i-th output at layer l, $z_i^l(x)$, converges to a GP with zero mean and covariance function κ_{ii}^l given by*

$$\kappa_{ii}^1(x, x') = \frac{\sigma_w^2}{n_0} x^\top x' + \sigma_b^2, \quad \text{and} \quad k_{ij}^1 = 0, \quad i \neq j. \tag{6.38}$$

$$\kappa_{ii}^l(x, x') = \sigma_b^2 + \sigma_w^2 \mathbb{E}_{z_i^{l-1} \sim \mathcal{GP}(0, \kappa_{ii}^{l-1})}[\phi(z_i^{l-1}(x))\phi(z_i^{l-1}(x'))]. \tag{6.39}$$

$$\tag{6.40}$$

and

$$\kappa_{ij}^{l}(x,x') = \mathbb{E}[z_i^l(x)z_j^l(x')] = \begin{cases} \kappa^l(x,x') & \text{if } i=j, \\ 0 & \text{if } i \neq j. \end{cases} \tag{6.41}$$

Proof. The proof is done by induction. The induction assumption is that if outputs at layer $l-1$ satisfy a GP structure

$$z_i^{l-1} \sim \mathcal{GP}(0, \kappa_{ii}^{l-1}), \tag{6.42}$$

with the covariance function defined as

$$\kappa_{ii}^{l-1}(x,x') = \mathbb{E}[z_i^{l-1}(x)z_i^{l-1}(x')] = k_{jj}^{l-1}(x,x'), \quad \forall i,j, \tag{6.43}$$

$$\kappa_{ij}^{l-1}(x,x') = \mathbb{E}[z_i^{l-1}(x)z_j^{l-1}(x')] = 0, \quad \text{for } i \neq j, \tag{6.44}$$

then, outputs at layer l follow

$$z_i^l(x) \sim \mathcal{GP}(0, \kappa_{ii}^l), \tag{6.45}$$

where the kernel at layer l is given by:

$$\kappa_{ii}^{l}(x,x') = \mathbb{E}[z_i^l(x)z_i^l(x')] = \kappa_{jj}^{l}(x,x'), \quad \forall i,j, \tag{6.46}$$

$$\kappa_{ij}^{l}(x,x') = \mathbb{E}[z_i^l(x)z_j^l(x')] = 0, \quad \text{if } i \neq j. \tag{6.47}$$

with the recursive definition

$$\kappa_{ii}^{l}(x,x') = \sigma_b^2 + \sigma_w^2 \mathbb{E}_{z_i^{l-1} \sim \mathcal{GP}(0, k_{ii}^{l-1})}[\phi(z_i^{l-1}(x))\phi(z_i^{l-1}(x'))]. \tag{6.48}$$

Base case ($l=1$). At layer $l=1$ we have:

$$z_i^1(x) = \frac{\sigma_w}{\sqrt{n_0}} \sum_{j=1}^{n_0} w_{ij}^1 x_j + \sigma_b b_i^1. \tag{6.49}$$

This is an affine transform of Gaussian random variables; thus, $z_i^1(x)$ is Gaussian distributed with

$$z_i^1(x) \sim \mathcal{GP}(0, \kappa_{ii}^1), \tag{6.50}$$

with kernel

$$\kappa_{ii}^1(x,x') = \frac{\sigma_w^2}{n_0} x^\top x' + \sigma_b^2, \quad \text{and} \quad \kappa_{ij}^1 = 0, \quad i \neq j. \tag{6.51}$$

Induction step $l > 1$. For layers $l > 1$ we have

$$z_i^l(x) = \sigma_b b_i^l + \frac{\sigma_w}{\sqrt{n_{l-1}}} \sum_{j=1}^{n_{l-1}} w_{ij}^l x_j^l(x), \quad x_j^l(x) = \phi(z_j^{l-1}(x)). \tag{6.52}$$

By the induction assumption, $z_j^{l-1}(x)$ are generated by independent GPs. Hence, $x_i^l(x)$ and $x_j^l(x)$ are independent for $i \neq j$. Consequently, $z_i^l(x)$ is a sum of independent random variables. By the CLT (as $n_1, \ldots, n_{L-1} \to \infty$) the tuple $\{z_i^l(x), z_i^l(x')\}$ tends to be jointly Gaussian, with covariance given by:

$$\mathbb{E}[z_i^l(x) z_i^l(x')] = \sigma_b^2 + \sigma_w^2 \mathbb{E}_{z_i^{l-1} \sim \mathcal{GP}(0, \kappa_{ii}^{l-1})}[\phi(z_i^{l-1}(x))\phi(z_i^{l-1}(x'))]. \tag{6.53}$$

Moreover, as z_i^l and z_j^l for $i \neq j$ are defined through independent rows of the parameters w^l, b^l and independent pre-activations $x^l(x)$, we have

$$\kappa_{ij}^l = \mathbb{E}[z_i^l(x) z_j^l(x')] = 0, \quad i \neq j, \tag{6.54}$$

completing the proof. □

6

Neural Tangent Kernel Propagation and Independence

We change notation slightly from the previous section to make the parametrization of $f_i(x, \theta^L)$ and $z_i^l(x; \theta^l)$ explicit with

$$z_i^l(x, \theta^l) = \sigma_b b_i^l + \frac{\sigma_w}{\sqrt{n_{l-1}}} \sum_{j=1}^{n_{l-1}} w_{ij}^l x_j^l(x), \quad x_j^l(x) = \phi(z_j^{l-1}(x; \theta^{l-1})), \tag{6.55}$$

where θ^l denotes the parameters $\{w^1, b^1, \ldots, w^l, b^l\}$ up to layer l and $f_i(x, \theta^L) = z_i^L(x; \theta^L)$. Let furthermore ϕ denote a Lipschitz-continuous nonlinearity with derivative $\dot{\phi}(x) = \frac{\mathrm{d}}{\mathrm{d}x}\phi(x)$.

Proposition 6.7 (Jacot et al. (2018)). *In the limit $n_1 \ldots, n_{L-1} \to \infty$, the neural tangent kernel $\Theta_{ii}^l(x, x')$ of the i-th output $z_i^l(x, \theta^l)$ at layer l, defined as the gradient inner product*

$$\Theta_{ii}^l(x, x') = \nabla_{\theta^l}^\top z_i^l(x, \theta^l) \nabla_{\theta^l} z_i^l(x', \theta^l), \tag{6.56}$$

is given recursively by

$$\Theta_{ii}^1(x, x') = \kappa_{ii}^1(x, x') = \frac{\sigma_w^2}{n_0} x^\top x' + \sigma_b^2, \quad \textit{and} \quad \Theta_{ij}^1(x, x') = 0, \quad i \neq j. \tag{6.57}$$

$$\Theta_{ii}^l(x, x') = \Theta_{ii}^{l-1}(x, x') \dot{\kappa}_{ii}^{l-1}(x, x') + \kappa_{ii}^l(x, x'), \tag{6.58}$$

$$\tag{6.59}$$

where

$$\dot{\kappa}^l_{ii}(x,x') = \sigma_w^2 \mathbb{E}_{z_i^{l-1}\sim\mathcal{GP}(0,\kappa_{ii}^{l-1})}[\dot{\phi}(z_i^{l-1}(x))\dot{\phi}(z_i^{l-1}(x'))] \tag{6.60}$$

and

$$\Theta^l_{ij}(x,x') = \nabla^\top_{\theta^l} z_i^l(x,\theta^l)\nabla_{\theta^l} z_j^l(x',\theta^l) = \begin{cases}\Theta^l(x,x') & \text{if } i=j,\\ 0 & \text{if } i\neq j.\end{cases} \tag{6.61}$$

Proof. We again proceed by induction. The induction assumption is that if gradients satisfy at layer $l-1$

$$\Theta^{l-1}_{ij}(x,x') = \nabla^\top_{\theta^{l-1}} z_i^{l-1}(x,\theta^{l-1})\nabla_{\theta^{l-1}} z_j^{l-1}(x',\theta^{l-1}) = \begin{cases}\Theta^{l-1}(x,x') & \text{if } i=j,\\ 0 & \text{if } i\neq j,\end{cases} \tag{6.62}$$

then at layer l we have

$$\Theta^l_{ii}(x,x') = \Theta^{l-1}_{ii}(x,x')\dot{\kappa}^l_{ii}(x,x') + \kappa^l_{ii}(x,x') \tag{6.63}$$

and

$$\Theta^l_{ij}(x,x') = \nabla^\top_{\theta^l} z_i^l(x,\theta^l)\nabla_{\theta^l} z_j^l(x',\theta^l) = 0 \quad \text{if } i\neq j. \tag{6.64}$$

Base Case ($l=1$). At layer $l=1$, we have

$$z_i^1(x) = \sigma_b b_i^1 + \frac{\sigma_w}{\sqrt{n_0}}\sum_j^{n_0} w_{ij}^1 x_j, \tag{6.65}$$

and the gradient inner product is given by:

$$\nabla^\top_{\theta^1} z_i^1(x,\theta^1)\nabla_{\theta^1} z_i^1(x',\theta^1) = \frac{\sigma_w^2}{n_0}x^\top x' + \sigma_b^2 = \kappa_{ii}^1(x,x'). \tag{6.66}$$

Inductive Step ($l>1$). For layers $l>1$, we split parameters $\theta^l = \theta^{l-1}\cup\{w^l,b^l\}$ and split the inner product by

$$\Theta^l_{ii}(x,x') = \underbrace{\nabla^\top_{\theta^{l-1}} z_i^l(x,\theta^l)\nabla_{\theta^{l-1}} z_i^l(x',\theta^l)}_{l.h.s} + \underbrace{\nabla^\top_{\{w^l,b^l\}} z_i^l(x,\theta^l)\nabla_{\{w^l,b^l\}} z_i^l(x',\theta^l)}_{r.h.s}. \tag{6.67}$$

Note in the expression above that the r.h.s involves gradients w.r.t. last-layer parameters, i.e. the post-activation outputs of the previous layer, and by the

same arguments as in the NNGP derivation of Proposition 6.6, this is a sum of independent post activations s.t. in the limit $n_{l-1} \to \infty$

$$\nabla^\top_{\{w^l,b^l\}} z^l_i(x,\theta^l) \nabla_{\{w^l,b^l\}} z^l_j(x',\theta^l) = \begin{cases} k^l_{ii}(x,x'), & i=j, \\ 0, & i \neq j. \end{cases} \tag{6.68}$$

For the *l.h.s.*, we first apply chain rule to obtain

$$\nabla_{\theta^{l-1}} z^l_i(x,\theta^l) = \frac{\sigma_w}{\sqrt{n_{l-1}}} \sum_j^{n_{l-1}} w^l_{ij} \dot{\phi}(z^{l-1}_j(x,\theta^{l-1})) \nabla_{\theta^{l-1}} z^{l-1}_j(x,\theta^{l-1}). \tag{6.69}$$

The gradient inner product of outputs i and j thus reduces to

$$\begin{aligned} &\nabla^\top_{\theta^{l-1}} z^l_i(x,\theta^l) \nabla_{\theta^{l-1}} z^l_j(x',\theta^l) = \\ &\quad \frac{\sigma_w^2}{n_{l-1}} \sum_k^{n_{l-1}} w^l_{ik} w^l_{jk} \dot{\phi}(z^{l-1}_k(x,\theta^{l-1})) \dot{\phi}(z^{l-1}_k(x',\theta^{l-1})) \Theta^{l-1}_{kk}(x,x'). \end{aligned} \tag{6.70}$$

By the induction assumption $\Theta^{l-1}_{kk}(x,x') = \Theta^{l-1}(x,x')$ and again by the independence of the rows w^l_i and w^l_j for $i \neq j$, the above expression converges in the limit $n_{l-1} \to \infty$ to an expectation with

$$\Theta^l_{ij}(x,x') = \begin{cases} \Theta^{l-1}(x,x') \dot{\kappa}^l_{ii}(x,x') + \kappa^l_{ii}(x,x') & i=j, \\ 0 & i \neq j. \end{cases} \tag{6.71}$$

This completes the induction. □

Multiheaded UVU: Finite Sample Analysis

We now define multiheaded predictor with M output heads $u_i(x,\vartheta_t)$ for $i = 1,\dots,M$ and a fixed multiheaded target network $g_i(x_t;\psi_0)$ of equivalent architecture as u with the corresponding prediction error $\epsilon_i(x,\vartheta_t,\psi_0)$ accordingly. Let $u_i(x,\vartheta_t)$ be trained such that each head runs the same algorithm as outlined in Section 6.3 independently.

Corollary 6.3. Under the conditions of Theorem 6.1, consider online and target networks with M independent heads u_i, g_i, $i = 1,\dots,M$, each trained to convergence with errors $\epsilon_i(x,\vartheta_\infty,\psi_0)$. Let $\frac{1}{2}\bar{\epsilon}(x,\vartheta_\infty,\psi_0)^2 = \frac{1}{2M}\sum_{i=1}^M \epsilon_i(x,\vartheta_\infty,\psi_0)^2$ be the sample mean squared prediction error over M heads. Moreover, consider $M+1$ independent converged Q-functions $Q_i(x;\theta_\infty)$ and denote their sample variance $\bar{\sigma}^2_Q(x,\theta_\infty) = \frac{1}{M}\sum_{i=1}^{M+1}(Q_i(x;\theta_\infty) - \bar{Q}(x;\theta_\infty))^2$, where $\bar{Q}$ is the sample mean. The

two estimators are identically distributed according to a scaled Chi-squared distribution

$$\frac{1}{2}\bar{\epsilon}(x,\vartheta_\infty,\psi_0)^2 \overset{D}{=} \bar{\sigma}_Q^2(x,\theta_\infty), \quad \bar{\sigma}_Q^2(x,\theta_\infty) \sim \frac{\sigma_Q^2}{M}\chi^2(M), \tag{6.12}$$

with M degrees of freedom and $\sigma_Q^2(x,\theta_\infty) = \mathbb{V}_{\theta_0}[Q(x,\theta_\infty)]$ is the analytical variance of converged Q-functions given by Theorem 6.1.

Proof. By Collorary. 6.2, the prediction error of a single headed online and target network $\epsilon(x,\vartheta_t,\psi_0) = u(x,\vartheta_t) - g(x,\psi_0)$ converges in the limit $n_1 \ldots, n_{L-1} \to \infty$ and $t \to \infty$ to a Gaussian with zero mean and variance $\epsilon(x,\vartheta_\infty,\psi_0) \sim \mathcal{N}(0, 2\sigma_Q^2)$ where

$$\begin{aligned}\sigma_Q^2 &= \mathbb{V}_{\theta_0}\big[Q(x,\theta_\infty)\big] \\ &= \kappa_{xx} - (\Theta_{x\mathcal{X}}\Delta_{\mathcal{X}}^{-1}\Lambda_x + h.c.) + (\Theta_{x\mathcal{X}}\Delta_{\mathcal{X}}^{-1}(\Lambda_{\mathcal{X}} - \gamma\Lambda_{\mathcal{X}'})\Delta_{\mathcal{X}}^{-1^\top}\Theta_{\mathcal{X}x}).\end{aligned} \tag{6.72}$$

By Propositions 6.6 and 6.7, the NNGP and NTK associated with each online head $u_i(x,\vartheta_\infty)$ in the infinite width and time limit are given by

$$\kappa_{ij}(x,x') = \mathbb{E}[u_i(x,\vartheta_\infty)u_j(x',\vartheta_\infty)] = \begin{cases}\kappa(x,x') & \text{if } i=j,\\ 0 & \text{if } i\neq j,\end{cases} \tag{6.73}$$

$$\Theta_{ij}(x,x') = \nabla_\vartheta^\top u_i^l(x,\vartheta_\infty)\nabla_\vartheta u_j^l(x',\vartheta_\infty) = \begin{cases}\Theta(x,x') & \text{if } i=j,\\ 0 & \text{if } i\neq j.\end{cases} \tag{6.74}$$

Due to the independence of the NNGP and NTK for different heads u_i, prediction errors $\epsilon_i(x_t;\vartheta_\infty,\psi_0)$ are i.i.d. draws from a zero mean Gaussian with variance equal as given in Eq. 6.72. Note that this is despite the final feature layer being shared between the output functions. The empirical mean squared prediction errors are thus Chi-squared distributed with M degrees of freedom

$$\frac{1}{M}\sum_{i=1}^{M}\frac{1}{2}\epsilon_i(x_t;\vartheta_\infty,\psi_0)^2 \sim \frac{\sigma_Q^2}{M}\chi^2(M) \tag{6.75}$$

Now, let $\{Q_i(x;\theta_t)\}_{i=1}^{M+1}$ be a deep ensemble of $M+1$ Q-functions from independent initializations. By Corollary 6.2, these Q-functions, too, are i.i.d. draws from a Gaussian, now with mean $\Theta_{x\mathcal{X}}\Delta_{\mathcal{X}}^{-1}r$ and variance as given in Eq. 6.72. The sample variance of this ensemble thus also follows a Chi-squared distribution with M degrees of freedom

$$\frac{1}{M}\sum_{i=1}^{M+1}\frac{1}{2}\big(Q_i(x;\theta_\infty) - \bar{Q}(x;\theta_\infty)\big)^2 \sim \frac{\sigma_Q^2}{M}\chi^2(M), \tag{6.76}$$

where $\bar{Q}(x;\theta_\infty) = \frac{1}{M+1}\sum_i^{M+1} Q_i(x;\theta_\infty)$ is the sample mean of $M+1$ independently initialized universal Q-functions, completing the proof. □

7

Discussion and Outlook

The research presented in the preceding chapters constitutes the core technical contribution of this dissertation, detailing a suite of novel algorithms and theoretical analyses for uncertainty quantification in deep reinforcement learning. This final chapter serves to synthesize these findings and to situate them within a broader context of future research. We begin by revisiting and answering the research questions that guided this work. Following this, we discuss several promising and important research directions that arise from the findings of this thesis. Our agenda here progresses from the need for a comprehensive deep reinforcement learning theory to more application-driven research such as the incorporation of reinforcement learning in generative discovery. The chapter, and the dissertation as a whole, then closes with a final, summary conclusion.

7.1 Answers to Research Questions

This dissertation was guided by a principal research question concerning the development of efficient and principled uncertainty quantification methods for deep RL, which we investigated through three specific lines of inquiry. Having presented our detailed findings in the preceding chapters, we now revisit these questions to provide an answer to each.

On enhancing ensemble diversity (RQ1). Our first research question concerned the quality of uncertainty estimates derived from deep ensembles:

RQ1: *Can member-specific architectural choices in deep ensembles promote diverse generalization behaviors and thereby improve the quality of uncertainty estimates?*

The research presented in Chapter 3 answers this question in the affirmative. Our work was motivated by the empirical finding that different architectural elements within distributional RL algorithms —specifically, their underlying projection operators — induce distinct generalization behaviors in the learned value functions. Based on this insight, we developed *diverse projection ensembles*, which are constructed from a mixture of different projection operators. We demonstrated both analytically and empirically that this form of structural diversity leads to more reliable uncertainty signals, enabling smaller ensembles to achieve comparable or superior results to larger, homogenous deep ensembles. This was particularly evident in challenging exploration tasks, where our method achieved superior sample efficiency, suggesting that the quality of diversity is a more critical factor for effective exploration than the mere quantity of ensemble members.

On emulating ensembles with a single models (research question (RQ)2 & RQ3). Our second line of inquiry focused on bridging the gap between computationally expensive, principled methods and more efficient, less understood single-model approaches.

RQ2: *Can the predictive variance of supervised deep ensembles be approximated directly and accurately by a single neural network in the limit of infinite width?*

In response to RQ2, Chapter 4 presented a constructive proof-of-concept with a new method we term contextual similarity distillation (CSD). We showed that by framing the direct prediction of ensemble variance as a contextualized kernel regression problem, the task becomes amenable to a gradient-based, single-model training pipeline. Our theoretical analysis, grounded in the infinite-width limit of neural networks, demonstrates that this approach can, in principle, exactly emulate the predictive variance of an infinite-member ensemble. While this result assumes access to relevant context data and the idealized dynamics of infinitely wide networks, we evaluated CSD empirically and found its practical efficacy to be on par with, or superior to, standard deep ensembles at a fraction of the computational cost.

The third research question sought a more rigorous understanding of existing single-model uncertainty quantification methods, taking random network distillation (RND) as a prominent example (Burda et al., 2019b):

RQ3: *What is the theoretical nature of the uncertainty captured by random network distillation, as a prominent example of single-model heuristic methods, when analyzed in the infinite-width limit?*

Our analysis in Chapter 5 provided a clear answer: in the infinite-width limit, the RND error signal is not merely a heuristic proxy for uncertainty but is formally equivalent to the predictive variance of a deep ensemble, both in expectation and for finite-sample sizes (i.e., finite-member ensembles). This correspondence revealed that RND is amenable to prior shaping techniques previously developed for Bayesian deep ensembles (He et al., 2020). Specifically, we showed that deliberate shaping of the RND target function facilitates learning dynamics that equate the predictive RND error with the true posterior predictive variance of an infinitely wide Bayesian neural network (BNN). These results provide a strong theoretical justification for RND's empirical success and introduce a novel modification that rigorously situates this popular method within the principled framework of Bayesian inference.

On emulating ensembles of value functions with a single model (RQ4). Our final line of investigation addressed whether single-model methods for uncertainty

quantification could be successfully designed for the more complex setting of temporal difference learning with neural value functions.

RQ4: *Can the predictive variance of an ensemble of deep value functions be approximated directly and accurately by a single neural network in the limit of infinite width?*

In Chapter 6, we developed a method labeled universal value-function uncertainty (UVU) that demonstrates that this is indeed possible. Our method relies on a self-predictive training process akin to the methods treated in the Chapters 4 and 5. A key distinction in this novel method, however, is that it uses a temporal difference (TD) training objective based on a synthetic reward function. This allows UVU to directly estimate cumulative value uncertainties rather than the myopic uncertainties predicted by CSD or RND (or most other existing baselines). Our theoretical analysis of UVU established the connection between this single-model method and its ensemble counterpart in the neural tangent kernel (NTK) limit, showing that its self-predictive error signals are equivalent to the variance of neural universal value functions. Our empirical results furthermore showed that UVU can be employed in challenging practical offline RL settings as a reliable estimator of task-capability, that is, whether the available dataset is sufficient to learn certain tasks. This is achieved with the computational efficiency of a single-model method.

Taken together, the answers to these questions form an investigative trajectory from enhancing existing multi-model approaches to the development of theoretically grounded, efficient single-model alternatives. They collectively provide an affirmative answer to our principal research question, demonstrating that it is indeed possible to develop algorithms that reconcile computational efficiency with principled foundations for quantifying long-term, cumulative uncertainty in deep reinforcement learning. These contributions not only address specific gaps in the current literature but also lay the groundwork for future research directions, some of which are discussed in the following.

7.2 Future Research

The results presented in this dissertation, while addressing several key challenges in uncertainty quantification in the field of deep reinforcement learning, also illuminate a number of promising and important avenues for future research. This final section outlines four such directions, organized in a progression from foundational theory to more application-oriented areas. We begin by discussing the need for a more comprehensive deep reinforcement learning theory. We then explore connections between uncertainty quantification and representation learning. Building upon this, our discussion consid-

ers uncertainty-aware agentic behaviors, outlining a vision for agents that can *learn* adaptive strategies in response to their own state of knowledge. Finally, we discuss an exciting application of these ideas: leveraging uncertainty-aware RL to guide generative models towards genuine scientific and creative discovery. Collectively, these ideas form an agenda that points towards a future of more principled, reliable, and self-aware autonomous systems.

7.2.1 Towards a Deep Reinforcement Learning Theory

A central theme of this dissertation is the significant gap between the empirical successes of deep reinforcement learning and a formal theoretical understanding of its underlying mechanisms. While deep learning theory has made substantial strides, its application to the dynamic setting of RL remains relatively scarce. This thesis has made extensive use of results from deep learning theory, and the NTK in particular, as a primary analytical tool for providing principled motivations for uncertainty quantification methods in deep RL (Chapters 4, 5 and 6). Our work on universal value-function uncertainties (Chapter 6), for instance, represents one of few applications of NTK theory to temporal difference learning, establishing conditions under which the tangent kernel remains constant throughout training even in this non-stationary setting.

Our analysis provides a theoretical baseline but also highlights the limitations of applying current theoretical tools to the full complexity of reinforcement learning. This motivates several immediate compelling directions for future work, which can be framed by systematically relaxing the simplifying assumptions made in our analyses. It is possible that in these more realistic settings, the constancy of the tangent kernel — a core property enabling the tractability of NTK learning dynamics — breaks even at infinite width, presenting new theoretical challenges:

- *Dynamics of policy improvement:* A core component of most RL algorithms is policy improvement, where an agent's policy is updated greedily (or related operations) with respect to a value function estimate. This update is a highly non-linear operator that is fundamentally different from the fixed-target regression settings typically analyzed in the NTK literature. A critical open question is how evolving policies and the associated non-stationary action distributions influence the training dynamics of neural value functions and under what conditions they might disrupt or maintain the stationarity of the tangent kernel.

- *Dynamics of online exploration:* Our analyses, in line with most existing NTK literature, consider training on a fixed dataset (exceptions are for example the works by Tsilivis and Kempe (2022) and Bennani et al. (2020)). However, online exploration in the typical RL setting leads to a non-stationary data

distribution that is endogenously determined by the agent's evolving policy and uncertainty estimates. Characterizing the learning dynamics under self-generated online data streams is a major theoretical hurdle for current frameworks.

- *Extension to other RL paradigms:* Another valuable extension of this line of theoretical analysis may investigate the learning dynamics of neural dynamics models in a model-based RL context or more complex hybrid algorithms. Methods such as successor representations may be of particular interesting in this regard (Barreto et al., 2017; Dayan, 1993). Because successor representations can be learned in a fashion akin to temporal difference learning, their analysis may even offer a tractable path towards analyzing feature learning within the infinite-width regime, a possibility previously demonstrated in supervised settings (Yang and Hu, 2020).

More broadly, the intermediate goal for a deep reinforcement learning theory must be to account for representation learning. While the NTK provides a strong kernel-based perspective, alternative theoretical frameworks aim to model the evolution of learned features. These include approaches based on mean-field theory, higher-order Taylor expansions of the learning dynamics, or kernel alignment dynamics (Bai and Lee, 2020; Bordelon and Pehlevan, 2022; Hanin and Nica, 2020; Mei et al., 2018). The application of these advanced theoretical frameworks to an RL setting represents a significant long-term research direction. In summary, the gap with which theory lags behind empirical practice in the field of deep reinforcement learning signifies a vast and fertile ground for future research. Theoretical advancements are essential for making future autonomous agents more robust, reliable, and understandable. The work presented in this dissertation aims to be a definitive step in that direction.

7.2.2 Towards Uncertainty-Driven Representation Learning

There exists a subtle but consequential tension between the objectives of epistemic uncertainty estimation and representation learning. On one hand, reliable uncertainty quantification requires a model to remain sensitive to out-of-distribution inputs — which often implies preserving fine-grained distinctions between seemingly minor variations. On the other hand, representation learning typically aims to discard precisely such variations: by mapping semantically similar inputs to a shared representation, models become largely invariant to task-irrelevant features (Bengio et al., 2013). This trade-off becomes particularly pronounced in the online setting of reinforcement learning. Representations learned from early, limited data are liable to suppress features that appear uninformative in early stages of training but may in fact be crucial

for the discovery of novel strategies or generalization to future tasks. The central question, then, is: how can a model learn compact, useful representations while remaining sensitive to potential novelty?

This motivates a direction we refer to as uncertainty-driven representation learning. The goal is not to treat uncertainty estimation as an additive step to learning a black-box model, but instead to learn feature spaces that are intrinsically amenable to efficient and effective uncertainty quantification. A guiding question might be: What properties must a representation $\phi(x)$ possess such that lightweight uncertainty estimation techniques — such as Bayesian linear regression or ensembling applied atop a final linear layer — are reliable? This shifts attention from the algorithmic implementation of uncertainty estimation to shaping the feature space itself.

Recent methods in self-supervised learning may already be using tools related to this purpose. For instance, models like Barlow twins explicitly penalize feature redundancy and encourage decorrelated latent dimensions (Zbontar et al., 2021). Such objectives counteract the rank collapse often induced by task-focused training, potentially yielding feature spaces that support a broader range of downstream predictive functions — and, by extension, better support uncertainty quantification. Our own work on contextual similarity distillation (Chapter 4) likewise provides a perspective towards learning feature spaces that support direct uncertainty prediction (e.g., a variance function), derived from the kernel-regression formulation based representations that express predictive variance as a function of pairwise similarity to previously observed inputs. Still, the more general problem of learning representations that explicitly support uncertainty quantification — in a task-agnostic, data-driven, and computationally tractable manner — remains largely open.

A particularly intriguing perspective arises in the context of self-predictive learning, which surprisingly underlies both many recent uncertainty quantification methods (e.g., RND (Burda et al., 2019b) or Chapters 4 and 6) and state-of-the-art self-supervised learning algorithms (e.g., BYOL (Grill et al., 2020), SimSiam (Chen and He, 2021), or Dino-V2 (Oquab et al., 2024)). In this general setup, a trainable predictor network is trained to match the output of a separate target network — but with different dynamics depending on the application:

- *For uncertainty estimation (e.g., RND)*: The predictor learns to match the output of a fixed, randomly initialized target network on the same input. Here, high prediction error signals epistemic uncertainty, as the predictor only succeeds on familiar inputs from the training distribution.

- *For representation learning (e.g., BYOL)*: The predictor is trained to match the output of a slowly-moving target network, typically on augmented views of

the same input. The objective is to induce invariance across augmentations and learn stable, semantic features.

While their objectives differ, the structural similarity between these setups is striking — especially early in training, when target networks are close to their initialization. This raises the hypothesis that the behavior of such self-predictive mechanisms may lie on a spectrum: at one end, fixed targets preserve sensitivity to raw novelty encoded by inductive priors of randomly initialized networks (i.e., prior features); at the other, moving targets guide the model toward invariance over increasingly abstract transformations (i.e., posterior features). The moving rate and nature of the target function may thus act as a tunable control over the granularity of uncertainty being captured — from local surprise to more semantic novelty. Understanding this continuum could offer new pathways toward learning representations that are not only robust and compact but also flexible enough to support both fine-grained and high-level uncertainty estimation downstream.

7.2.3 Towards Truly Uncertainty-Aware Agents

As reinforcement learning agents are increasingly deployed in real-world applications — from autonomous driving to robotic assistance in household or surgery — their ability to reason reliably under uncertainty becomes critical. While several contemporary algorithms are labeled "uncertainty-aware", this awareness is often implemented as a fixed, hard-coded response to a quantified uncertainty signal. For example, an agent employing upper confidence bound exploration follows a non-adaptive rule of optimism by adding a scaled uncertainty bonus to its value estimates (Auer, 2002). In contrast, we advocate for a framework where agents become truly uncertainty-aware by explicitly *learning* strategies for how to behave given their current state of knowledge. Such agents would learn when it is beneficial to be optimistic (e.g., in a safe, exploratory context) and when it is necessary to be conservative (e.g., when approaching potentially catastrophic hazards).

The principle of acting optimally under epistemic uncertainty is already subject of the literature and is arguably captured most accurately by the BAMDP framework (Duff, 2002; Martin, 1965). In a BAMDP, states are augmented with a belief distribution over all possible environment dynamics, and optimal policies plan within this belief space. In theory, such agents can act optimally by taking into account their uncertainty — for example, choosing to explore or act cautiously depending on their current belief. However, maintaining and planning over an analytical belief state is computationally intractable for all but the simplest of problems. As a consequence, most existing implementations of BAMDP algorithms do not use learned policies that

condition directly on rich representations of their epistemic state.

Our proposal can be viewed as a data-driven, learnable approximation of the ideal envisioned by BAMDPs. A central challenge herein is the representation of an agent's epistemic state in a tractable yet informative way. We draw inspiration from the analytical properties of kernel regression, as previously explored in Chapter 4. The predictive variance of a kernel-based model at a test input x, given training data $\mathcal{X}$, can be expressed as a quadratic form of kernel similarities $v(x) = \kappa(x,x) - \kappa(x,\mathcal{X})A\kappa(\mathcal{X},x)$ with some matrix A. This variance can be reformulated as a linear model in a feature space $\phi(x)$ defined by the pairwise similarities between the input and the data, for instance $\phi(x) = \mathrm{vec}\big(\kappa(\mathcal{X},x)\kappa(x,\mathcal{X})\big)$. Conceptually, this vectorized "similarity matrix" resembles modern attention mechanisms, where queries retrieve similarity scores between a sequence of observations to produce a contextual embedding of relations (Vaswani et al., 2017)[1].

We propose to use such a representation — an "epistemic context vector" c — as an explicit conditioning variable in the agent's decision-making. Policies and value functions would take the form $\pi(\cdot \mid s, c)$ or $V(s, c)$, where c summarizes the agent's current state of knowledge with respect to its past experiences $\mathcal{X}$. Crucially, such policies are learned, not fixed and can thus in principle learn from data when a particular context c (e.g., one indicating low similarity to past data) warrants optimistic exploration, or when a different context warrants caution. Existing approaches from the POMDP literature or the meta-RL literature indeed incorporate the idea of stochastic latent variables as representations of knowledge (e.g., variBAD (Zintgraf et al., 2020) or PEARL (Rakelly et al., 2019)). However, these approaches typically learn policies that condition on variables representing task-uncertainty, or episodic state-uncertainty due to partial observability (Kaelbling et al., 1998) and do not represent the agent's full body of experience, nor do they directly model epistemic uncertainty.

Of course, a primary reason that agents do not typically condition on their full state of knowledge is that such a variable would be intractably large in almost all problems. To make full epistemic conditioning practical, we propose incorporating retrieval mechanisms, analogous to those in RAG for large language models (Lewis et al., 2020). Rather than conditioning on its entire memory, the agent would here use its current state s as a query to retrieve a compact, salient subset of past experiences. The epistemic context vector c would then be approximated from this retrieved subset, for instance via an attention mechanism. Interestingly, reinforcement learning algorithms may be used as subroutines themselves in implementing such retrieval mechanisms (Kulkarni et al., 2024). The process of autonomously constructing a context to

[1]The connection between kernel machines and attention mechanisms is indeed itself a recent subject of theoretical analysis, see for example work by Tsai et al. (2019) and Chen et al. (2023).

support a decision also shares a conceptual link with chain-of-thought prompting, where language models auto-regressively condition on self-generated intermediate reasoning steps to improve their final output (Wei et al., 2022).

Framed this way, we treat uncertainty-awareness as a learnable strategy, not by analytically planning over beliefs or sampling from posteriors, but by learning how to construct and utilize epistemic representations from experience. We believe this offers a promising path toward agents that not only act intelligently but do so with calibrated confidence, grounded in their own accumulated knowledge.

7.2.4 Towards Generative Discovery with Reinforcement Learning

In recent years, generative models have made extraordinary strides across a wide range of domains: they can synthesize high-resolution images, write fluent text, design molecular structures, and generate executable code (Askr et al., 2023; Goodfellow et al., 2016; Kingma and Welling, 2014; Li et al., 2022; Rombach et al., 2022). Yet, many current generative models are designed for what is in essence replication; they learn to reproduce a given data distribution rather than to uncover novel, useful modes outside of it. Discovery, in contrast, must go beyond generating variation through sampling. It requires structured, goal-oriented exploration of a typically vast solution space. In this section, we argue that modeling generation as a sequential decision-making problem opens new paths toward this goal and illustrate how ideas from reinforcement learning, uncertainty estimation, and unsupervised representation learning may be leveraged for truly diverse generative discovery.

From generative sampling to constructive decision-making. One central premise of our proposal is that many generative tasks can be fruitfully recast as sequential decision-making problems. Rather than designing models that directly sample complete objects (e.g., images, molecules, or machines), one may instead model the process of constructing these objects in a way that is grunded in physical generation processes. This framing bears two primary advantages: first, it allows models to learn a more structured representation of objects by explicitly modeling the dynamics of their sequential construction; and second, it invites the algorithmic tools of reinforcement learning to shape an exploratory generative process aimed at discovering realistic and *novel* artifacts.

As an example, consider the challenge of machine design. A standard generative model could be trained to sample complete blueprints of functional engines from an extensive dataset. Alternatively, we might provide an RL agent with access to a library of machining and assembly tools and train it to con-

struct an engine step-by-step — drilling holes, machining parts, and assembling components. We argue that the second approach provides richer learning signals, as it inherently captures the structure and physical constraints of the generative process itself. A similar duality arises in molecular discovery. While standard models may generate molecules atom-by-atom to reproduce statistically likely structures, a model that engages with a generative process grounded in chemically valid reactions[2] may acquire a more fundamental knowledge of chemical synthesis. This not only enables more realistic generation but also supports better representation learning and more actionable outcomes, such as producing molecules that are not only novel but also readily synthesizable.

Representation learning via generative dynamics. Phrasing generation as a sequential process naturally raises the question of how to learn useful representations within this framework. Ideally, modeling the generative dynamics as outlined above provides a rich signal for representation learning even before specifying reward functions. In many scientific and engineering problems, evaluating a design's utility is costly; analyzing the chemical properties of a molecule or the failure modes of a machine prototype may require expensive computation or physical experimentation. This motivates the idea of reward-free pre-training — a paradigm aimed at extracting knowledge purely from the generative dynamics.

In this context, we can draw on work in unsupervised and reward-free reinforcement learning, such as methods based on the SR or forward-backward representations (Barreto et al., 2017; Dayan, 1993; Touati and Ollivier, 2021). These frameworks focus on features corresponding to future state-occupancies that capture the distribution of reachable future states from any given state[3]. Such representations are powerful because any value function — for any downstream reward — can then be expressed as a linear model in this feature space and can, in principle, be learned without requiring a reward function during the representation learning phase. Translated to the generative processes we envision, an agent could be pre-trained on purely simulated generative processes (e.g., valid chemical reactions) to learn a feature space that reflects the dynamics of construction: what generative actions are feasible, what substruc-

[2]We acknowledge that the computational modeling of chemical synthesis is itself a major undertaking and an active area of research (see, for example, the open reaction database (Kearnes et al., 2021)).

[3]The forward-backward representation can here be understood as a low-rank factorization of the successor representation, i.e. the discounted state-occupancy matrix. The left side of such a factorization, the forward representation, thus summarizes which future states are likely to follow given a state, action, and policy (independent of any specific reward).

tures exist, and how final objects can be composed, all without the need for a potentially expensive reward oracle.

Exploration for generative discovery. Once generation is formulated as a decision process, a natural path towards discovery becomes accessible: exploration. We argue that most current generative models explore only in a limited sense, akin to injecting stochasticity into a learned policy. While this produces variation, there exist problems for which it is unlikely that this procedure will discover truly novel and superior strategies. Instead, such problems may require policies that actively seek to reduce epistemic uncertainty by exploring novel regions of the solution space. By adopting the RL framework, advanced exploration strategies — such as those based on intrinsic rewards from uncertainty estimates or posterior sampling (Burda et al., 2019b; Osband and Van Roy, 2017) — become natural tools for strategically seeking out surprising or rare constructions. A molecular generation agent, for instance, could be incentivized not just to generate valid molecules, but to explore entire classes of compounds that are structurally distant from known families. Conversely, such methods could also be used to explore novel synthesis strategies for existing materials. The epistemic uncertainty quantification methods developed in this thesis are natural candidates for guiding such a principled exploration process.

Several existing research areas already relate to this vision. Quality-diversity algorithms from evolutionary computation share the goal of generating diverse, high-quality objects (Pugh et al., 2016). More recently, a trend at the forefront of generative artificial intelligence (AI) incorporates reinforcement learning into *generative flows* (GFlowNets) (Bengio et al., 2021). Our proposal complements these approaches by emphasizing the role of the generative dynamics — grounded in the physical synthesis process — to yield more structured, dynamics-oriented representations that can support more meaningful and efficient exploration.

7.3 Conclusion

This dissertation has addressed a central challenge in modern AI: the development of reliable and scalable uncertainty quantification methods for agents that engage in sequential decision-making problems in complex, high-dimensional environments. The research herein follows a cohesive progression, beginning with the enhancement of multi-model ensembles and concluding with the development of efficient single-model approximations, first for immediate predictions and ultimately for the long-term, cumulative uncertainties inherent in reinforcement learning. Our development of novel computational algorithms was accompanied by theoretical analyses that attend to the idealized learning

dynamics of the involved neural function approximators. In doing so, we designed methods that are grounded in, and seek to leverage, the generalization properties of deep neural networks, rather than obscuring these mechanisms through a black-box treatment.

Ultimately, this work represents a definitive step towards creating more efficient, reliable, and truly uncertainty-aware autonomous agents, thereby establishing a foundation for their responsible deployment in the real world.

References

M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya, V. Makarenkov, and S. Nahavandi. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *arXiv:2011.06225*, 2021.

J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

R. Agarwal, D. Schuurmans, and M. Norouzi. An optimistic perspective on offline reinforcement learning. In *International conference on machine learning*, pages 104–114. PMLR, 2020.

T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019.

D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané. Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*, 2016.

G. An, S. Moon, J.-H. Kim, and H. O. Song. Uncertainty-based offline reinforcement learning with diversified Q-ensemble. *Advances in neural information processing systems*, 34:7436–7447, 2021.

S. Arora, S. S. Du, W. Hu, Z. Li, R. R. Salakhutdinov, and R. Wang. On exact computation with an infinitely wide neural net. *Advances in neural information processing systems*, 32, 2019.

H. Askr, E. Elgeldawi, H. Aboul Ella, Y. A. Elshaier, M. M. Gomaa, and A. E. Hassanien. Deep learning in drug discovery: An integrative review and future challenges. *Artificial intelligence review*, 56(7):5975–6037, 2023.

P. Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of machine learning research*, 3, 2002.

P. Auer, T. Jaksch, and R. Ortner. Near-optimal regret bounds for reinforcement learning. *Advances in neural information processing systems*, 21, 2008.

J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

F. Bach. *Learning theory from first principles*. MIT press, 2024.

Y. Bai and J. D. Lee. Beyond linearization: On quadratic and higher-order approximation of wide neural networks. In *International conference on learning representations*, 2020.

A. Barreto, W. Dabney, R. Munos, J. J. Hunt, T. Schaul, H. P. van Hasselt, and D. Silver. Successor features for transfer in reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

M. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, and R. Munos. Unifying count-based exploration and intrinsic motivation. *Advances in neural information processing systems*, 29, 2016.

M. G. Bellemare, W. Dabney, and R. Munos. A distributional perspective on reinforcement learning. In *International conference on machine learning*. PMLR, 2017.

M. G. Bellemare, W. Dabney, and M. Rowland. *Distributional reinforcement learning*. MIT Press, 2023.

R. Bellman. A Markovian decision process. *Journal of mathematics and mechanics*, 6, 1957.

E. Bengio, M. Jain, M. Korablyov, D. Precup, and Y. Bengio. Flow network based generative models for non-iterative diverse candidate generation. *Advances in neural information processing systems*, 34:27381–27394, 2021.

Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

M. A. Bennani, T. Doan, and M. Sugiyama. Generalisation guarantees for continual learning with orthogonal gradient descent. *arXiv preprint arXiv:2006.11942*, 2020.

D. Blackwell. Discrete dynamic programming. *The annals of mathematical statistics*, pages 719–726, 1962.

D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical association*, 112(518):859–877, 2017.

C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural network. In *International conference on machine learning*, pages 1613–1622. PMLR, 2015.

B. Bordelon and C. Pehlevan. Self-consistent dynamical field theory of kernel evolution in wide neural networks. *Advances in neural information processing systems*, 35:32240–32256, 2022.

J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/jax-ml/jax`.

Y. Burda, H. Edwards, D. Pathak, A. J. Storkey, T. Darrell, and A. A. Efros. Large-scale study of curiosity-driven learning. In *International conference on learning representations*, 2019a.

Y. Burda, H. Edwards, A. J. Storkey, and O. Klimov. Exploration by random network distillation. In *International conference on learning representations*, 2019b.

Q. Cai, Z. Yang, J. D. Lee, and Z. Wang. Neural temporal-difference learning converges to global optima. *Advances in neural information processing systems*, 32, 2019.

S. Calvo-Ordoñez, K. Palla, and K. Ciosek. Epistemic uncertainty and observation noise with the neural tangent kernel. *arXiv preprint arXiv:2409.03953*, 2024.

Y. Cao and Q. Gu. Generalization bounds of stochastic gradient descent for wide and deep neural networks. *Advances in neural information processing systems*, 32, 2019.

M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

R. Y. Chen, S. Sidor, P. Abbeel, and J. Schulman. UCB exploration via Q-ensembles. *arXiv preprint arXiv:1706.01502*, 2017.

T. Chen, E. Fox, and C. Guestrin. Stochastic gradient Hamiltonian Monte Carlo. In *International conference on machine learning*, pages 1683–1691. PMLR, 2014.

T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

X. Chen and K. He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15750–15758, 2021.

X. Chen, C. Wang, Z. Zhou, and K. Ross. Randomized ensembled double Q-learning: Learning fast without a model. *arXiv preprint arXiv:2101.05982*, 2021.

Y. Chen, Q. Tao, F. Tonin, and J. Suykens. Primal-attention: Self-attention through asymmetric kernel SVD in primal representation. *Advances in Neural Information Processing Systems*, 36:65088–65101, 2023.

M. Chevalier-Boisvert, B. Dai, M. Towers, R. Perez-Vicente, L. Willems, S. Lahlou, S. Pal, P. S. Castro, and J. Terry. Minigrid & Miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. In *Advances in neural information processing systems 36*, December 2023.

L. Chizat and F. Bach. On the global convergence of gradient descent for over-parameterized models using optimal transport. *Advances in neural information processing systems*, 31, 2018.

H. Choi, E. Jang, and A. A. Alemi. Waic, but why? generative ensembles for robust anomaly detection. *arXiv preprint arXiv:1810.01392*, 2018.

A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun. The loss surfaces of multilayer networks. In *Artificial intelligence and statistics*, pages 192–204. PMLR, 2015.

Y. Chow, M. Ghavamzadeh, L. Janson, and M. Pavone. Risk-constrained reinforcement learning with percentile risk criteria. *Journal of machine learning research*, 18(167):1–51, 2018.

P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

K. Chua, R. Calandra, R. McAllister, and S. Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems*, 31, 2018.

T. Clanuwat, M. Bober-Irizar, A. Kitamoto, A. Lamb, K. Yamamoto, and D. Ha. Deep learning for classical Japanese literature. *CoRR*, abs/1812.01718, 2018.

W. R. Clements, B. Van Delft, B.-M. Robaglia, R. B. Slaoui, and S. Toth. Estimating risk and uncertainty in deep reinforcement learning. *arXiv preprint arXiv:1905.09638*, 2019.

F. Cuzzolin. *The Geometry of Uncertainty: The Geometry of Imprecise Probabilities*. Artificial Intelligence: Foundations, Theory, and Algorithms. Springer International Publishing, Cham, 2021.

G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

W. Dabney, G. Ostrovski, D. Silver, and R. Munos. Implicit quantile networks for distributional reinforcement learning. In *International conference on machine learning*. PMLR, 2018a.

W. Dabney, M. Rowland, M. Bellemare, and R. Munos. Distributional reinforcement learning with quantile regression. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018b.

F. D'Angelo and V. Fortuin. Repulsive deep ensembles are Bayesian. *Advances in Neural Information Processing Systems*, 34:3451–3465, 2021.

Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *Advances in neural information processing systems*, 27, 2014.

P. Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural computation*, 5(4):613–624, 1993.

R. Dearden, N. Friedman, S. Russell, et al. Bayesian Q-learning. *Proceedings of the AAAI conference on artificial intelligence*, 1998:761–768, 1998.

J. Degrave, F. Felici, J. Buchli, M. Neunert, B. Tracey, F. Carpanese, T. Ewalds, R. Hafner, A. Abdolmaleki, D. de Las Casas, et al. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, 602(7897):414–419, 2022.

Delft Artificial Intelligence Cluster (DAIC), 2024.

Delft High Performance Computing Centre (DHPC). DelftBlue Supercomputer (Phase 1), 2022.

L. Deng. The MNIST database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

S. Depeweg, J. M. Hernández-Lobato, F. Doshi-Velez, and S. Udluft. Learning and policy search in stochastic dynamical systems with Bayesian neural networks. In *International conference on learning representations*, 2017.

A. Der Kiureghian and O. Ditlevsen. Aleatory or epistemic? Does it matter? *Structural safety*, 31, 2009.

T. G. Dietterich. Ensemble methods in machine learning. In *Multiple classifier systems: First international workshop, MCS*. Springer, 2000.

L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density estimation using real NVP. *arXiv:1605.08803 [cs, stat]*, Feb. 2017.

P. D'Oro, M. Schwarzer, E. Nikishin, P.-L. Bacon, M. G. Bellemare, and A. Courville. Sample-efficient reinforcement learning by breaking the replay ratio barrier. In *International Conference on Learning Representations, ICLR*, 2023.

M. O. Duff. *Optimal Learning: Computational procedures for Bayes-adaptive Markov decision processes*. University of Massachusetts Amherst, 2002.

G. Dulac-Arnold, N. Levine, D. J. Mankowitz, J. Li, C. Paduraru, S. Gowal, and T. Hester. Challenges of real-world reinforcement learning: Definitions, benchmarks and analysis. *Machine Learning*, 110(9):2419–2468, 2021.

R. Durrett. *Probability: Theory and examples*, volume 49. Cambridge university press, 2019.

A. Ecoffet, J. Huizinga, J. Lehman, K. O. Stanley, and J. Clune. Go-explore: A new approach for hard-exploration problems. *arXiv preprint arXiv:1901.10995*, 2019.

B. Efron. *The jackknife, the bootstrap and other resampling plans*. SIAM, 1982.

Y. Engel, S. Mannor, and R. Meir. Reinforcement learning with Gaussian processes. In *International conference on machine learning*, pages 201–208, 2005.

H. Eriksson, D. Basu, M. Alibeigi, and C. Dimitrakakis. Sentinel: Taming uncertainty with ensemble based distributional reinforcement learning. In *Uncertainty in artificial intelligence*. PMLR, 2022.

L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, Y. Doron, V. Firoiu, T. Harley, I. Dunning, et al. Impala: Scalable distributed deep-RL with importance weighted actor-learner architectures. In *International conference on machine learning*. PMLR, 2018.

M. Fellows, K. Hartikainen, and S. Whiteson. Bayesian Bellman operators. *Advances in neural information processing systems*, 34, 2021.

A. Filos, E. Vértes, Z. Marinho, G. Farquhar, D. Borsa, A. Friesen, F. Behbahani, T. Schaul, A. Barreto, and S. Osindero. Model-value inconsistency as a signal for epistemic uncertainty. *arXiv preprint arXiv:2112.04153*, 2021.

S. Flennerhag, J. X. Wang, P. Sprechmann, F. Visin, A. Galashov, S. Kapturowski, D. L. Borsa, N. Heess, A. Barreto, and R. Pascanu. Temporal difference uncertainties as a signal for exploration. *arXiv preprint arXiv:2010.02255*, 2020.

S. Fort, H. Hu, and B. Lakshminarayanan. Deep ensembles: A loss landscape perspective. *arXiv preprint arXiv:1912.02757*, 2019.

S. Fujimoto, H. Hoof, and D. Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*. PMLR, 2018.

S. Fujimoto, W.-D. Chang, E. Smith, S. S. Gu, D. Precup, and D. Meger. For sale: State-action representation learning for deep reinforcement learning. *Advances in neural information processing systems*, 36:61573–61624, 2023.

Y. Gal and Z. Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International conference on machine learning*, pages 1050–1059. PMLR, 2016.

M. Gallici, M. Fellows, B. Ellis, B. Pou, I. Masmitja, J. N. Foerster, and M. Martin. Simplifying deep temporal difference learning. *arXiv preprint arXiv:2407.04811*, 2024.

A. Garriga-Alonso and V. Fortuin. Exact Langevin dynamics with stochastic gradients. *arXiv preprint arXiv:2102.01691*, 2021.

A. Gelman and C. R. Shalizi. Philosophy and the practice of Bayesian statistics. *British journal of mathematical and statistical psychology*, 66(1):8–38, 2013.

S. Gerschgorin. Uber die abgrenzung der eigenwerte einer matrix. *Izvestija Akademii Nauk SSSR, Serija Matematika*, 7(3):749–754, 1931.

Z. Ghahramani. Probabilistic machine learning and artificial intelligence. *Nature*, 521(7553):452–459, 2015.

M. Ghavamzadeh, S. Mannor, J. Pineau, and A. Tamar. Bayesian reinforcement learning: A survey. *Foundations and trends in machine learning*, 8, 2015.

E. Goan and C. Fookes. Bayesian neural networks: An introduction and survey. *Case studies in applied Bayesian data science: CIRM Jean-Morlet chair, Fall 2018*, pages 45–87, 2020.

F. Gogianu, T. Berariu, M. C. Rosca, C. Clopath, L. Busoniu, and R. Pascanu. Spectral normalisation for deep reinforcement learning: an optimisation perspective. In *International conference on machine learning*, pages 3734–3744. PMLR, 2021.

I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio. *Deep learning*. MIT press Cambridge, 2016.

A. Graves. Practical variational inference for neural networks. *Advances in neural information processing systems*, 24, 2011.

S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu. A survey of deep learning techniques for autonomous driving. *Journal of field robotics*, 37(3):362–386, 2020.

J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

Z. Guo, S. Thakoor, M. Pîslar, B. Avila Pires, F. Altché, C. Tallec, A. Saade, D. Calandriello, J.-B. Grill, Y. Tang, et al. BYOL-Explore: Exploration by bootstrapped prediction. *Advances in neural information processing systems*, 35:31855–31870, 2022.

T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

B. Hanin and M. Nica. Finite depth and width corrections to the neural tangent kernel. In *International conference on learning representations*, 2020.

L. K. Hansen and P. Salamon. Neural network ensembles. *IEEE transactions on pattern analysis and machine intelligence*, 12(10):993–1001, 1990.

H. Hasselt. Double Q-learning. *Advances in neural information processing systems*, 23, 2010.

B. He, B. Lakshminarayanan, and Y. W. Teh. Bayesian deep ensembles via the neural tangent kernel. *Advances in neural information processing systems*, 33, 2020.

K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on Imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, 2015.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International conference on learning representations*, 2017.

M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

C.-J. Hoel, K. Wolff, and L. Laine. Ensemble quantile networks: Uncertainty-aware reinforcement learning with applications in autonomous driving. *IEEE Transactions on intelligent transportation systems*, 2023.

L. Hoffmann and C. Elster. Deep ensembles from a Bayesian perspective. *arXiv preprint arXiv:2105.13283*, 2021.

S. C. Hora. Aleatory and epistemic uncertainty in probability elicitation with an example from hazardous waste management. *Reliability engineering & system safety*, 54, 1996.

K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

R. Houthooft, X. Chen, Y. Duan, J. Schulman, F. De Turck, and P. Abbeel. Vime: Variational information maximizing exploration. *Advances in neural information processing systems*, 29, 2016.

R. A. Howard. *Dynamic programming and Markov processes.* John Wiley, 1960.

E. Hüllermeier and W. Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *arXiv:1910.09457*, Sept. 2020.

O. Ibe. *Fundamentals of Applied Probability and Random Processes*. Elsevier Science, 2014.

A. Immer, M. Korzepa, and M. Bauer. Improving predictions of Bayesian neural nets via local linearization. In *International conference on artificial intelligence and statistics*, pages 703–711. PMLR, 2021.

H. Ishfaq, Q. Cui, V. Nguyen, A. Ayoub, Z. Yang, Z. Wang, D. Precup, and L. Yang. Randomized exploration in reinforcement learning with general value function approximation. In *International conference on machine learning*, pages 4607–4616. PMLR, 2021.

P. Izmailov, S. Vikram, M. D. Hoffman, and A. G. G. Wilson. What are Bayesian neural network posteriors really like? In *International conference on machine learning*, pages 4629–4640. PMLR, 2021.

A. Jacot, F. Gabriel, and C. Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

D. Janz, J. Hron, P. Mazur, K. Hofmann, J. M. Hernández-Lobato, and S. Tschiatschek. Successor uncertainties: Exploration and uncertainty in temporal difference learning. *Advances in neural information processing systems*, 32, 2019.

E. T. Jaynes. *Probability theory: The logic of science*. Cambridge university press, 2003.

Y. Jiang, J. Z. Kolter, and R. Raileanu. On the importance of exploration for generalization in reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.

C. Jin, Z. Allen-Zhu, S. Bubeck, and M. I. Jordan. Is Q-learning provably efficient? *Advances in neural information processing systems*, 31, 2018.

C. Jin, Z. Yang, Z. Wang, and M. I. Jordan. Provably efficient reinforcement learning with linear function approximation. In *Conference on learning theory*, pages 2137–2143. PMLR, 2020.

J. Kaddour, J. Harris, M. Mozes, H. Bradley, R. Raileanu, and R. McHardy. Challenges and applications of large language models. *arXiv preprint arXiv:2307.10169*, 2023.

L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101, 1998.

D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on robot learning*, pages 651–673. PMLR, 2018.

J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

S. M. Kearnes, M. R. Maser, M. Wleklinski, A. Kast, A. G. Doyle, S. D. Dreher, J. M. Hawkins, K. F. Jensen, and C. W. Coley. The open reaction database. *Journal of the American chemical society*, 143(45):18820–18826, 2021.

M. Kempka, M. Wydmuch, G. Runc, J. Toczek, and W. Jaśkowski. ViZDoom: A Doom-based AI research platform for visual reinforcement learning. In *IEEE Conference on computational intelligence and games*. IEEE, 2016.

A. Kendall and Y. Gal. What uncertainties do we need in Bayesian deep learning for computer vision? *Advances in neural information processing systems*, 30, 2017.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In Y. Bengio and Y. LeCun, editors, *International conference on learning representations, ICLR*, 2015.

D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In Y. Bengio and Y. LeCun, editors, *International conference on learning representations*, 2014.

S. Kobayashi, P. Vilimelis Aceituno, and J. Von Oswald. Disentangling the predictive variance of deep ensembles through the neural tangent kernel. *Advances in Neural Information Processing Systems*, 35:25335–25348, 2022.

R. Koenker and K. F. Hallock. Quantile regression. *Journal of economic perspectives*, 15, 2001.

V. Konda and J. Tsitsiklis. Actor-critic algorithms. *Advances in neural information processing systems*, 12, 1999.

M. Kulkarni, P. Tangarajan, K. Kim, and A. Trivedi. Reinforcement learning for optimizing rag for domain chatbots. *arXiv preprint arXiv:2401.06800*, 2024.

A. Kumar, A. Zhou, G. Tucker, and S. Levine. Conservative Q-learning for offline reinforcement learning. *Advances in neural information processing systems*, 33:1179–1191, 2020.

S. Lahlou, M. Jain, H. Nekoei, V. I. Butoi, P. Bertin, J. Rector-Brooks, M. Korablyov, and Y. Bengio. Deup: Direct epistemic uncertainty prediction. *arXiv preprint arXiv:2102.08501*, 2021.

B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

J. Lee, J. Sohl-dickstein, J. Pennington, R. Novak, S. Schoenholz, and Y. Bahri. Deep neural networks as Gaussian processes. In *International conference on learning representations*, 2018a.

J. Lee, S. Schoenholz, J. Pennington, B. Adlam, L. Xiao, R. Novak, and J. Sohl-Dickstein. Finite versus infinite neural networks: an empirical study. *Advances in Neural Information Processing Systems*, 33:15156–15172, 2020a.

J. Lee, L. Xiao, S. S. Schoenholz, Y. Bahri, R. Novak, J. Sohl-Dickstein, and J. Pennington. Wide Neural Networks of Any Depth Evolve as Linear Models Under Gradient Descent. *Journal of Statistical Mechanics: Theory and Experiment*, 2020, Dec. 2020b.

K. Lee, H. Lee, K. Lee, and J. Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *International conference on learning representations*, 2018b.

K. Lee, M. Laskin, A. Srinivas, and P. Abbeel. Sunrise: A simple unified framework for ensemble learning in deep reinforcement learning. In *International Conference on Machine Learning*, pages 6131–6141. PMLR, 2021.

S. Lee, Y. Seo, K. Lee, P. Abbeel, and J. Shin. Offline-to-online reinforcement learning via balanced replay and pessimistic Q-ensemble. In *Conference on Robot Learning*, pages 1702–1712. PMLR, 2022.

S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *Journal of machine learning research*, 17(39):1–40, 2016.

S. Levine, A. Kumar, G. Tucker, and J. Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in neural information processing systems*, 33:9459–9474, 2020.

Y. Li, D. Choi, J. Chung, N. Kushman, J. Schrittwieser, R. Leblond, T. Eccles, J. Keeling, F. Gimeno, A. Dal Lago, et al. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097, 2022.

B. Lindenberg, J. Nordqvist, and K.-O. Lindahl. Distributional reinforcement learning with ensembles. *Algorithms*, 13, 2020.

C. Liu, L. Zhu, and M. Belkin. On the linearity of large non-linear models: When and why the tangent kernel is constant. *Advances in Neural Information Processing Systems*, 33:15954–15964, 2020.

Q. Liu and D. Wang. Stein variational gradient descent: A general purpose Bayesian inference algorithm. *Advances in neural information processing systems*, 29, 2016.

C. E. Luis, A. G. Bottero, J. Vinogradska, F. Berkenkamp, and J. Peters. Model-based uncertainty in value functions. In *International Conference on Artificial Intelligence and Statistics*, pages 8029–8052. PMLR, 2023.

B. Lütjens, M. Everett, and J. P. How. Safe reinforcement learning with model uncertainty estimates. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8662–8668. IEEE, 2019.

C. Lyle, M. Rowland, W. Dabney, M. Kwiatkowska, and Y. Gal. Learning dynamics and generalization in reinforcement learning. *arXiv preprint arXiv:2206.02126*, 2022.

E. Mariucci and M. Reiß. Wasserstein and total variation distance between marginals of Lévy processes. *Electronic journal of statistics*, 12, 2018.

J. J. Martin. *Some Bayesian decision problems in a Markov chain.* PhD thesis, Massachusetts Institute of Technology, 1965.

M. Matthews, M. Beukman, B. Ellis, M. Samvelyan, M. Jackson, S. Coward, and J. Foerster. Craftax: A lightning-fast benchmark for open-ended reinforcement learning. *arXiv preprint arXiv:2402.16801*, 2024.

B. Mavrin, H. Yao, L. Kong, K. Wu, and Y. Yu. Distributional reinforcement learning for efficient exploration. In *International conference on machine learning*. PMLR, May 2019.

S. Mei, A. Montanari, and P.-M. Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the national academy of sciences*, pages E7665–E7671, 2018.

V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

T. M. Moerland, J. Broekens, and C. M. Jonker. Efficient exploration with double uncertain value networks. *arXiv:1711.10789 [cs, stat]*, Nov. 2017.

T. Morimura, M. Sugiyama, H. Kashima, H. Hachiya, and T. Tanaka. Nonparametric return distribution approximation for reinforcement learning. In *International conference on machine learning*. PMLR, 2010.

M. S. A. Nadeem, J.-D. Zucker, and B. Hanczar. Accuracy-rejection curves (arcs) for comparing classification methods with a reject option. In *Machine Learning in Systems Biology*, pages 65–81. PMLR, 2009.

E. Nalisnick, A. Matsukawa, Y. Teh, D. Gorur, and B. Lakshminarayanan. Do deep generative models know what they don't know? In *International conference on learning representations*, 2019.

R. M. Neal. *Bayesian Learning for Neural Networks*. Springer-Verlag, 1996.

G. Neustroev and M. M. de Weerdt. Generalized optimistic Q-Learning with provable efficiency. In *International conference on autonomous agents and multi-agent systems*, pages 913–921, 2020.

T. Nguyen-Tang, S. Gupta, and S. Venkatesh. Distributional reinforcement learning via moment matching. In *Proceedings of the AAAI conference on artificial intelligence*, pages 9144–9152, 2021.

N. Nikolov, J. Kirschner, F. Berkenkamp, and A. Krause. Information-directed exploration for deep reinforcement learning. In *International conference on learning representations, ICLR*, 2019.

A. Nikulin, V. Kurenkov, D. Tarasov, and S. Kolesnikov. Anti-exploration by random network distillation. In *International Conference on Machine Learning*, pages 26228–26244. PMLR, 2023.

A. Nitanda and T. Suzuki. Optimal rates for averaged stochastic gradient descent under neural tangent kernel regime. In *International conference on learning representations*, 2021.

I. K. Nti, A. F. Adekoya, B. A. Weyori, and O. Nyarko-Boateng. Applications of artificial intelligence in engineering and manufacturing: a systematic review. *Journal of Intelligent Manufacturing*, 33(6):1581–1601, 2022.

M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. HAZIZA, F. Massa, A. El-Nouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P.-Y. Huang, S.-W. Li, I. Misra, M. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jegou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on machine learning research*, 2024. ISSN 2835-8856.

I. Osband and B. Van Roy. Why is posterior sampling better than optimism for reinforcement learning? In *International conference on machine learning*, pages 2701–2710. PMLR, 2017.

I. Osband, D. Russo, and B. Van Roy. (more) efficient reinforcement learning via posterior sampling. *Advances in neural information processing systems*, 26, 2013.

I. Osband, C. Blundell, A. Pritzel, and B. Van Roy. Deep exploration via bootstrapped DQN. *Advances in neural information processing systems*, 29, 2016.

I. Osband, B. Van Roy, D. J. Russo, Z. Wen, et al. Deep exploration via randomized value functions. *Journal of machine learning research*, 20, 2019.

I. Osband, Y. Doron, M. Hessel, J. Aslanides, E. Sezener, A. Saraiva, K. McKinney, T. Lattimore, C. Szepesvári, S. Singh, B. V. Roy, R. S. Sutton, D. Silver, and H. van Hasselt. Behaviour suite for reinforcement learning. In *International conference on learning representations, ICLR*, 2020.

L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

B. O'Donoghue, I. Osband, R. Munos, and V. Mnih. The uncertainty Bellman equation and exploration. In *International conference on machine learning*. PMLR, 2018.

D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*. PMLR, 2017.

T. Pearce, F. Leibfried, and A. Brintrup. Uncertainty in neural networks: Approximately Bayesian ensembling. In *International conference on artificial intelligence and statistics*, pages 234–244. PMLR, 2020.

J. K. Pugh, L. B. Soros, and K. O. Stanley. Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI*, 3:40, 2016.

M. L. Puterman. *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

C. Qin, Z. Wen, X. Lu, and B. Van Roy. An analysis of ensemble sampling. *Advances in Neural Information Processing Systems*, 35:21602–21614, 2022.

M. Raghu and E. Schmidt. A survey of deep learning for scientific discovery. *arXiv preprint arXiv:2003.11755*, 2020.

K. Rakelly, A. Zhou, C. Finn, S. Levine, and D. Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International conference on machine learning*, pages 5331–5340. PMLR, 2019.

T. Rashid, B. Peng, W. Böhmer, and S. Whiteson. Optimistic exploration even with a pessimistic initialisation. *Proceedings of ICLR 2020*, 2020.

C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*. MIT Press, 2006.

D. Rezende and S. Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

D. A. Roberts, S. Yaida, and B. Hanin. *The principles of deep learning theory*, volume 46. Cambridge University Press Cambridge, MA, USA, 2022.

R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

M. Rowland, M. Bellemare, W. Dabney, R. Munos, and Y. W. Teh. An analysis of categorical distributional reinforcement learning. In *International conference on artificial intelligence and statistics*. PMLR, 2018.

M. Rowland, R. Dadashi, S. Kumar, R. Munos, M. G. Bellemare, and W. Dabney. Statistics and samples in distributional reinforcement learning. In *International conference on machine learning*. PMLR, 2019.

T. G. Rudner, Z. Chen, Y. W. Teh, and Y. Gal. Tractable function-space variational inference in bayesian neural networks. *Advances in neural information processing systems*, 35:22686–22698, 2022.

G. A. Rummery and M. Niranjan. *Online Q-learning using connectionist systems*, volume 37. University of Cambridge, Department of Engineering Cambridge, UK, 1994.

D. J. Russo, B. Van Roy, A. Kazerouni, I. Osband, Z. Wen, et al. A tutorial on Thompson sampling. *Foundations and trends® in machine learning*, 11(1): 1–96, 2018.

M. Samarin, V. Roth, and D. Belius. On the empirical neural tangent kernel of standard finite-width convolutional neural network architectures. *arXiv preprint arXiv:2006.13645*, 2020.

A. M. Saxe, J. L. McClelland, and S. Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

R. E. Schapire. The strength of weak learnability. *Machine learning*, 5:197–227, 1990.

T. Schaul, D. Horgan, K. Gregor, and D. Silver. Universal value function approximators. In *International conference on machine learning*, pages 1312–1320. PMLR, 2015.

T. Schaul, J. Quan, I. Antonoglou, and D. Silver. Prioritized experience replay. In Y. Bengio and Y. LeCun, editors, *International conference on learning representations, ICLR*, 2016.

D. Schmidt and T. Schmied. Fast and data-efficient training of rainbow: An experimental study on Atari. *arXiv preprint arXiv:2111.10247*, 2021.

S. Schmitt, J. Shawe-Taylor, and H. van Hasselt. Exploration via epistemic value estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 2023.

J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

M. Schwarzer, A. Anand, R. Goel, R. D. Hjelm, A. Courville, and P. Bachman. Data-efficient reinforcement learning with self-predictive representations. *arXiv preprint arXiv:2007.05929*, 2020.

M. Seleznova and G. Kutyniok. Analyzing finite neural networks: Can we trust neural tangent kernel theory? In *Mathematical and Scientific Machine Learning*, pages 868–895. PMLR, 2022.

M. Sensoy, L. Kaplan, and M. Kandemir. Evidential deep learning to quantify classification uncertainty. *Advances in neural information processing systems*, 31, 2018.

H. Sheikh, M. Phielipp, and L. Boloni. Maximizing ensemble diversity in deep reinforcement learning. In *International conference on learning representations*, 2022.

D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

S. Singh, T. Jaakkola, M. L. Littman, and C. Szepesvári. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine learning*, 38:287–308, 2000.

S. P. Singh and R. S. Sutton. Reinforcement learning with replacing eligibility traces. *Machine learning*, 22(1):123–158, 1996.

J. Smit, C. Ponnambalam, M. T. J. Spaan, and F. A. Oliehoek. PEBL: Pessimistic ensembles for offline deep reinforcement learning. In *Robust and reliable autonomy in the wild workshop at the 30th international joint conference of artificial intelligence*, 2021.

N. Srinivas, A. Krause, S. Kakade, and M. Seeger. Gaussian process optimization in the bandit setting: no regret and experimental design. In *International conference on machine learning*, pages 1015–1022, 2010.

A. L. Strehl, L. Li, E. Wiewiora, J. Langford, and M. L. Littman. Pac model-free reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 881–888, 2006.

M. J. Strens. A Bayesian framework for reinforcement learning. In *International conference on machine learning*, pages 943–950, 2000.

S. Suganyadevi, V. Seethalakshmi, and K. Balasamy. A review on deep learning in medical image analysis. *International Journal of Multimedia Information Retrieval*, 11(1):19–38, 2022.

R. S. Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM Sigart Bulletin*, 2(4):160–163, 1991.

R. S. Sutton, A. G. Barto, et al. *Reinforcement learning: An introduction*. MIT press Cambridge, 1998.

R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

G. Tesauro et al. Temporal difference learning and TD-Gammon. *Communications of the ACM*, 38(3):58–68, 1995.

J. Thiyagalingam, M. Shankar, G. Fox, and T. Hey. Scientific machine learning benchmarks. *Nature Reviews Physics*, 4(6):413–420, 2022.

W. R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25, 1933.

S. B. Thrun. *Efficient exploration in reinforcement learning*. Carnegie Mellon University, 1992.

A. Touati and Y. Ollivier. Learning one representation to optimize all rewards. *Advances in Neural Information Processing Systems*, 34:13–23, 2021.

Y.-H. H. Tsai, S. Bai, M. Yamada, L.-P. Morency, and R. Salakhutdinov. Transformer dissection: A unified understanding of transformer's attention via the lens of kernel. *arXiv preprint arXiv:1908.11775*, 2019.

N. Tsilivis and J. Kempe. What can the neural tangent kernel tell us about adversarial robustness? *Advances in Neural Information Processing Systems*, 35:18116–18130, 2022.

J. N. Tsitsiklis. Asynchronous stochastic approximation and Q-learning. *Machine learning*, 16:185–202, 1994.

J. Van Amersfoort, L. Smith, Y. W. Teh, and Y. Gal. Uncertainty estimation using a single deep deterministic neural network. In *International conference on machine learning*, pages 9690–9700. PMLR, 2020.

P. R. Van der Vaart, M. T. J. Spaan, and N. Yorke-Smith. Epistemic Bellman operators. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need, 2017.

O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *nature*, 575(7782):350–354, 2019.

H.-T. Wai, Z. Yang, Z. Wang, and M. Hong. Provably efficient neural GTD for off-policy learning. *Advances in Neural Information Processing Systems*, 33: 10431–10442, 2020.

C. J. Watkins and P. Dayan. Q-learning. *Machine learning*, 8:279–292, 1992.

J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

M. Welling and Y. W. Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688. Citeseer, 2011.

Y. Wen, G. Jerfel, R. Muller, M. W. Dusenberry, J. Snoek, B. Lakshminarayanan, and D. Tran. Combining ensembles and data augmentation can harm your calibration. *arXiv preprint arXiv:2010.09875*, 2020.

D. J. White. Mean, variance, and probabilistic criteria in finite Markov decision processes: A review. *Journal of optimization theory and applications*, 56, 1988.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.

A. G. Wilson and P. Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. *Advances in neural information processing systems*, 33:4697–4708, 2020.

J. Wilson, C. van der Heide, L. Hodgkinson, and F. Roosta. Uncertainty quantification with the empirical neural tangent kernel. *arXiv preprint arXiv:2502.02870*, 2025.

J. Wu, Z. Huang, and C. Lv. Uncertainty-aware model-based reinforcement learning: Methodology and application in autonomous driving. *IEEE Transactions on intelligent vehicles*, 8(1):194–203, 2022.

L. Wu and S. A. Williamson. Posterior uncertainty quantification in neural networks using data augmentation. In *International Conference on Artificial Intelligence and Statistics*, pages 3376–3384. PMLR, 2024.

C. Xiao, B. Dai, J. Mei, O. A. Ramirez, R. Gummadi, C. Harris, and D. Schuurmans. Understanding and leveraging overparameterization in recursive value estimation. In *International Conference on Learning Representations*, 2021.

H. Xiao, K. Rasul, and R. Vollgraf. Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms. *CoRR*, abs/1708.07747, 2017.

D. Yang, L. Zhao, Z. Lin, T. Qin, J. Bian, and T.-Y. Liu. Fully parameterized quantile function for distributional reinforcement learning. *Advances in neural information processing systems*, 32, 2019.

G. Yang. Scaling limits of wide neural networks with weight sharing: Gaussian process behavior, gradient independence, and neural tangent kernel derivation. *arXiv preprint arXiv:1902.04760*, 2019.

G. Yang and E. J. Hu. Feature learning in infinite-width neural networks. *arXiv preprint arXiv:2011.14522*, 2020.

Z. Yang, C. Jin, Z. Wang, M. Wang, and M. Jordan. Provably efficient reinforcement learning with kernel and neural function approximations. *Advances in Neural Information Processing Systems*, 33:13903–13916, 2020.

Y. Yue, R. Lu, B. Kang, S. Song, and G. Huang. Understanding, predicting and better resolving Q-value divergence in offline-rl. *Advances in Neural Information Processing Systems*, 36:60247–60277, 2023.

M. A. Zanger, W. Böhmer, and M. T. J. Spaan. Diverse projection ensembles for distributional reinforcement learning. In *International conference on learning representations*, 2024.

M. A. Zanger, P. R. Van der Vaart, W. Böhmer, and M. T. J. Spaan. Contextual similarity distillation: Ensemble uncertainties with a single model. *arXiv preprint arXiv:2503.11339*, 2025a.

M. A. Zanger, M. Weltevrede, Y. Oren, P. R. Van der Vaart, C. Horsch, W. Böhmer, and M. T. J. Spaan. Universal value-function uncertainties. *arXiv preprint arXiv:2505.21119*, 2025b.

J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International conference on machine learning*, pages 12310–12320. PMLR, 2021.

C. Zheng, R. Salakhutdinov, and B. Eysenbach. Contrastive difference predictive coding. *arXiv preprint arXiv:2310.20141*, 2023.

Q. Zhou, H. Li, and J. Wang. Deep model-based reinforcement learning via estimated uncertainty and conservative policy optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

L. Zintgraf, K. Shiarlis, M. Igl, S. Schulze, Y. Gal, K. Hofmann, and S. Whiteson. VariBAD: A very good method for Bayes-adaptive deep rl via meta-learning. *arXiv:1910.08348*, Feb. 2020.

Curriculum Vitæ

Moritz Akiya Zanger

born in Basel, Switzerland on January 25th, 1994

Education

2004-2012	*General education* Gymnasium Korntal-Münchingen, Korntal, Germany
2012-2017	*Bachelor of science, mechanical engineering* Karlsruhe Institute of Technology, Karlsruhe, Germany
2015-2016	*Visiting student, computer science* Tohoku University, Sendai, Japan
2017-2020	*Master of science, mechanical engineering* Karlsruhe Institute of Technology, Karlsruhe, Germany

Experience

2016	*Intern, software engineering* Robert Bosch GmbH, Bühlertal, Germany
2016-2019	*Working student, software engineering* Robert Bosch GmbH, Bühlertal, Germany
2019-2020	*Research assistant, natural language processing* Research Center of Information Technology, Karlsruhe, Germany
2020-2021	*Research assistant - reinforcement learning* Research Center of Information Technology, Karlsruhe, Germany

2021-2025 *Doctoral researcher, reinforcement learning*
Delft University of Technology, Delft, The Netherlands

List of Publications

1. ***Moritz A. Zanger***, *Karam Daaboul, and J. Marius. Zöllner*: Safe continuous control with constrained model-based policy optimization, *IEEE/RSJ International conference on intelligent robots and systems (IROS), 2021.*

2. ***Moritz A. Zanger***, *Wendelin Böhmer, and Matthijs T. J. Spaan*: Diverse projection ensembles for distributional reinforcement learning, *International conference on learning representations (ICLR), 2024.*

3. *Yaniv Oren,* ***Moritz A. Zanger***, *Pascal R. van der Vaart, Matthijs T. J. Spaan and Wendelin Böhmer*: Value improved actor critic algorithms, *Advances in Neural Information Processing Systems (NeurIPS), 2025.*

4. *Max Weltevrede,* ***Moritz A. Zanger***, *Matthijs T. J. Spaan and Wendelin Böhmer*: How ensembles of distilled policies improve generalisation in reinforcement learning, Advances in Neural Information Processing Systems (NeurIPS), 2025.

5. ***Moritz A. Zanger***, *Pascal R. van der Vaart, Wendelin Böhmer, and Matthijs T. J. Spaan*: Contextual similarity distillation: Ensemble uncertainties with a single model, to appear in *International conference on learning representations (ICLR), 2026.*

6. ***Moritz A. Zanger***, *Max Weltevrede, Yaniv Oren, Pascal R. Van der Vaart, Caroline Horsch, Wendelin Böhmer, Matthijs T. J. Spaan*: Universal value-function uncertainties, to appear in *International conference on learning representations (ICLR), 2026.*

7. ***Moritz A. Zanger***, *Yijun Wu, Pascal R. Van der Vaart, Wendelin Böhmer, Matthijs T. J. Spaan*: On the Equivalence of Random Network Distillation, Deep Ensembles, and Bayesian Inference, under review 2026.

8. *Guopeng Li,* ***Moritz A. Zanger***, *Matthijs T. J. Spaan and Julian F. P. Kooij*: Cholesky ordered projection Q-learning (COP-Q): Guiding safety-first exploitation-exploration by multi-objective uncertainty, under review, 2026.

Included in this thesis.

Acknowledgments

First and foremost, I would like to thank my promotor, Matthijs Spaan, and my copromotor, Wendelin Böhmer. Given the many challenges of pursuing a Ph.D., what I am perhaps most thankful for is that my supervision was never one of them. Matthijs, thank you for believing in my potential and taking me on as a Ph.D. student. I deeply appreciate the time and effort you invested in me, through the frequent discussions we had and by the guidance you provided in my research trajectory. Wendelin, I am equally grateful to you: for your keen eye, for providing a crucial, critical perspective that always improved my ideas. Thank you, Matthijs and Wendelin, for always being a source of support during this time.

I would also like to thank the members of my defense committee—Bart De Schutter, Andreas Krause, Ann Nowé, and Thomas Moerland—for their interest in my research, their valuable feedback, and for making the defense an enjoyable academic event.

This research was made possible by the Epistemic AI project and the EU Horizon program. I am especially indebted to Fabio Cuzzolin for initiating this project and affording me the opportunity to work on such a fascinating topic. I also wish to thank Noah Schutte, Shireen Manchingal, Maryam Sultana, Julian Kooij, Neil Yorke-Smith, Keivan Shariatmadar, Andrew Bradley, and Kaizheng Wang, to name just a few, who collectively made the project a success.

During this journey, I had the pleasure of collaborating with excellent coauthors. I am grateful to Yaniv Oren, Pascal van der Vaart, Max Weltevrede, Caroline Horsch, Guopeng Li, and Yijun Wu for the seamless teamwork and the enjoyable work we did together.

More broadly, I want to thank the entire Algorithmics and Sequential Decision Making Lab for providing a welcoming environment that I genuinely enjoyed coming to every day, when our building would allow it. Special thanks go to Sophie den Hartog, Sofia Suarez, and Vanessa Kestel for always supporting me in administrative matters. I also want to highlight Canmanie Ponnambalam, Thiago Simão, and Qisong Yang for giving me a warm welcome to the group in the beginning, as well as my good friends and colleagues Yaniv, Pascal, Junhan, Caroline, and Max.

Finally, my deepest appreciation goes to my closest friends and family. To David Mackie and Marije de Groot: thank you for being great friends and providing much-needed distractions from the struggles of a Ph.D., through

evenings filled with board games and wine. To my parents, Uli and Kyoko: none of this, obviously, would have been possible without you. I am aware how much you paved the way for me, and having parents who truly understand the struggles of academic research made your support all the more meaningful. And finally, to my amazing girlfriend, Loes. You gave me love, stability, an escape, and supported me selflessly even in times of struggle. Thank you; I love you.

Appendices

A

Distributional Projection Ensembles

This appendix provides additional material, experimental results, and implementation details for Chapter 3.

A.1 Experimental Details

We provide a detailed exposition of our experimental setup, including the hyperparameter search procedure, hyperparameter settings, algorithmic details, and the full bsuite experimental results.

A.1.1 Hyperparameter settings

In our experiments, we aimed to keep most hyperparameters between different implementations equal to maintain comparability between the analyzed methods. Hyperparameters specific to algorithms were optimized over a search space of hyperparameters using Optuna (Akiba et al., 2019). The total search space for bsuite and VizDoom are given in Table A.1 and Table A.2 respectively, where the *Heads K* parameter only applies to distributional algorithms. categorical Q-network (C51) requires us to define return ranges, which we defined manually and can be found in the online code repository. All algorithms use the Adam optimizer (Kingma and Ba, 2015).

Bsuite. For bsuite, the hyperparameter search was conducted on a subselection of environments of the bsuite, as shown in Table A.3. For each environment, we evaluate a set of hyperparameters by means of a scoring function. A particular set of hyperparameters is evaluated every $T/5$ episodes with a maximum training horizon of T episodes. The "continuous" scoring functions make

the hyperparameter search more amenable to pruning, for which we use the median pruner of Optuna, reducing the computational burden of the combinatorial search space significantly.

Here, $\sum_{(s,a)} \mathbb{1}_{\text{visited}}(s,a)$ is the count of visited state-action tuples and $\sum_0^t(-1)$ is simply the negative number of total environment interactions. For every hyperparameter configuration ζ_i, the scores $f(\zeta_i)$ are calibrated to facilitate a meaningful comparison between different environments. The calibrated score function we use is given by

$$f_c(\zeta_i) = \exp\left(0.693 \frac{f(\zeta_i) - \mu_\zeta}{\sup_i f(\zeta_i) - \mu_\zeta}\right), \tag{A.1}$$

where μ_ζ is the average score of all hyperparameter configurations $\mu_\zeta = \sum_i^N 1/N f(\zeta_i)$, and $\sup_i f(\zeta_i)$ is the maximal score achieved. The calibration function in Eq. (A.1) was chosen heuristically to have an intuitive interpretation: it assigns a score of 1 to the best-performing hyperparameter configuration, 0.5 to configurations that achieve exact average performance, and decays exponentially according to score. The final score assigned to a hyperparameter configuration ζ_i is the sum of all scores of the tested environments. Table A.4 shows the full set of hyperparameters used for every algorithm.

VizDoom. For the VizDoom domain, the hyperparameter search was conducted on the *MyWayHomeSparse-v0* variation with a training budget of 5 million frames, where final configurations were chosen by achieved return at the end of training. Due to the sparsity of the problem, we did not make use of a pruning algorithm. The specific difference between the different variations of the VizDoom environment *MyWayHome* are shown in Fig. A.1, where the *sparsity* of the problem is increased by changing the agents spawning location to a room further from the goal position. The network architecture is based to a large extent on the rainbow network proposed by Schmidt and Schmied (2021) who in turn base their architecture

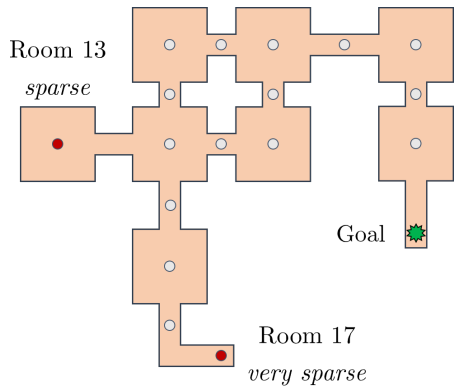


Figure A.1: Map for the VizDoom *MyWayHome* environment. Agents are spawned in the *sparse* and *very sparse* locations to vary the exploration difficulty.

on IMPALA (Espeholt et al., 2018). The specific algorithm configuration for VizDoom is given in Table A.5 with a schematic of the network architecture shown in Fig. A.3. Table A.6 shows our preprocessing pipeline used for the VizDoom environments.

A.1.2 Implementation details

Parametric model. Our parametric model is given by a mixture distribution $\eta_{E,\theta}$, parametrized by θ. We construct $\eta_{E,\theta}$ as an equal mixture between a quantile and a categorical representation, each parametrized through a neural network (NN) with K output logits where we use the notation θ_{ik} to mean the k-th logit of the network parametrized by the parameters θ_i of the i-th model in the ensemble. We consider a sample transition (s,a,r,s',a') where a' is chosen greedily according to $\mathbb{E}_{Z\sim\eta_{E,\theta}(s',a')}[Z]$. Dependencies on (s,a) are dropped for conciseness by writing $\theta_{ik}(s,a)=\theta_{ik}$ and $\theta_{ik}(s',a')=\theta'_{ik}$. The full mixture model $\eta_{E,\theta}$ is then given by

$$\eta_{E,\theta}=\frac{1}{2}\sum_{i=1}^{M=2}\sum_{k=1}^{K}p(\theta_{ik})\delta_{z(\theta_{ik})},\quad\text{with}\quad\begin{array}{l}p(\theta_{1k})=\frac{1}{K},\,z(\theta_{1k})=\theta_{1k},\\ p(\theta_{2k})=\sigma(\theta_{2k}),\,z(\theta_{2k})=z_k,\end{array}\tag{A.2}$$

where $\sigma(x_i)=e^{x_i}/\sum_j e^{x_j}$ is the softmax transfer function. Consequently, this representation comprises a total of $2K$ atoms, K of which parametrize locations in the quantile model, and the remaining K parametrizing probabilities in the categorical representation. The losses used for each projection method are as provided in the main text.

Distributional estimation of bonuses. For the parametric bonus estimate $b_\vartheta(s,a)$ we use the same procedure for learning a distributional projection ensemble as with extrinsic rewards. Note that it is not necessary for our method to learn a distributional estimate of the bonus but we find that diverse projection ensembles are good value learners in general and simply reuse the existent function approximation machinery for an intrinsic reward instead of the extrinsic reward. We thus have a model of parameters ϑ trained with an alternate tuple $(s,a,w_{\text{avg}},s',a'_\epsilon)$, where we replaced the immediate reward with the ensemble disagreement w_{avg} and a'_ϵ is an exploratory action chosen according to the rule

$$a'_\epsilon=\operatorname*{arg\,max}_{a\in\mathcal{A}}\left(\mathbb{E}_{Z\sim\eta_{E,\theta}(s,a)}[Z]+\beta\, b_\vartheta(s,a)\right),\quad\text{where}\quad b_\vartheta(s,a)=\mathbb{E}_{B\sim\eta_{E,\vartheta}(s,a)}[B].\tag{A.3}$$

Here, β is a hyperparameter to control the policy's drive towards exploratory actions.

Table A.1: Hyperparameter search space for bsuite

Hyperparameter	Values
Neural net architecture	$[[64,64],[128,128],[512]]$
Learning rate	$[5\times10^{-5}, 1\times10^{-4}, 5\times10^{-4}, 1\times10^{-3}]$
Prior function scale	$[0.0, 5.0, 20.0]$
Heads K	[51, 101]
Initial bonus β	[0.5, 5.0, 50.0]

Table A.2: Hyperparameter search space for VizDoom

Hyperparameter	Values
Learning rate	$[1.25\times10^{-5}, 2.5\times10^{-5}, 3.75\times10^{-5}, 5\times10^{-5}, 6.25\times10^{-5}, 7.5\times10^{-5}]$
Prior function scale	$[1.0, 3.0, 5.0]$
Initial bonus β	[0.05, 0.1, 0.5, 1.0, 5.0]

Table A.3: Hyperparameter search environments

Environment ID	Horizon in no. of episodes	Scoring function f
deep_sea/20	500	$\sum_{(s,a)} \mathbb{1}_{\text{visited}}(s,a)$
deep_sea_stochastic/20	1500	$\sum_{(s,a)} \mathbb{1}_{\text{visited}}(s,a)$
mountain_car/19	100	$\sum_0^t(-1)$

Table A.4: Hyperparameter settings bsuite

Hyperparameter	BDQNP	DLTV	IDS	PE-DQN
Net architecture	$[64,64]$	$[512]$	$[64,64]$ / $[512]$	$[512]$
Adam Learning rate	10^{-3}	10^{-3}	10^{-3} / 5×10^{-4}	5×10^{-4}
Prior function scale	5.0	20.0	20.0 / 5.0	20.0 / 0.0
Heads K	1	101	1 / 101	101/101
Ensemble size	20	1	20/1	2/2
Initial bonus β_{init}	n/a	5.0	5.0	5.0
Final bonus β_{final}	n/a	n/a	5.0	5.0
Bonus decay (in eps)	n/a	$10^3/N_{\text{eps}}$	$0.33\times N_{\text{eps}}$	$0.33\times N_{\text{eps}}$
Discount	0.99			
Buffer size	10,000			
Adam epsilon	0.001/batch size			
Initialization	He truncated normal (He et al., 2015)			
Update frequency	1			
Target update step size	1.0			
Target update frequency	4			
Batch size	128			

Table A.5: Hyperparameter settings VizDoom

Hyperparameter	BDQNP	DLTV	IDS	PE-DQN
Adam Learning rate	2.5×10^{-5}	7.5×10^{-5}	2.5×10^{-5}	6.25×10^{-5}
Prior function scale	1.0	3.0	1.0	3.0
Heads K	1	101	1 / 101	101/101
Ensemble size	10	1	10/1	2/2
Initial bonus β_{init}	n/a	0.5	0.1	5.0
Final bonus β_{final}	n/a	n/a	0.01	0.01
Bonus decay (in frames)	n/a	$10^3/N_{\text{frames}}$	$0.33 \times N_{\text{frames}}$	$0.33 \times N_{\text{frames}}$
Loss function	Huber	QR-Huber	Huber/C51	QR-Huber/C51
Initial ϵ in ϵ-greedy	1.0			
Final ϵ in ϵ-greedy	0.01			
ϵ decay time	500,000			
Training starts	100,000			
Discount	0.997			
Buffer size	1,000,000			
Batch size	512			
Parallel Envs	32			
Adam epsilon	0.005/batch size			
Initialization	He uniform (He et al., 2015)			
Gradient clip norm	10			
Regularization	spectral normalization			
Double DQN	Yes			
Update frequency	1			
Target update step size	1.0			
Target update frequency	8000			
PER β_0	0.45			
n-step returns	10			

Table A.6: VizDoom Preprocessing

Parameter	Value
Grayscale	Yes
Frame-skipping	No
Frame-stacking	6
Resolution	42×42
Max. Episode Length	2100

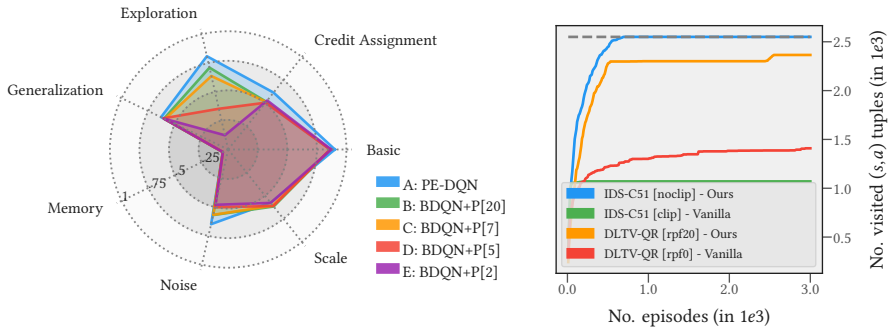


Figure A.2: (a) Summary of bsuite experiments. Comparison between bootstrapped deep Q-network + priors (BDQNP) with different ensemble sizes and PE-DQN (total ensemble size 4). (b) Deep sea comparison between our implementations and vanilla implementations of baseline algorithms. Shown are median state-action visitation counts over number of episodes on the deep sea environment with size 50. Shaded regions represent the interquartile range of 10 seeds. Higher is better.

Pseudocode. We provide pseudocode for a basic version of projection ensemble deep Q-network (PE-DQN) where we have simplified details such as the previously described distributional estimation of $b_\phi(s,a)$, prioritized replay, double Q-learning, and prior functions for clarity.

Randomized prior functions. Randomized prior functions are added to all baselines and PE-DQN. Specifically, we add the output of a fixed, randomly initialized NN of the same architecture as the main net, scaled by a hyperparameter, to the main network's logits. In the case of C51, the prior function is added pre softmax. To the best of our knowledge, decaying left-truncated variance (DLTV)-quantile regression (QR) does not use prior functions in its original formulation but we find it to be crucial in improving exploration performance. Fig. A.2 (b) shows an experiment assessing the exploration performance of DLTV-QR with randomized prior functions and prior scale 20 (*DLTV [rpf20]*) compared to the vanilla implementation without priors (*DLTV [rpf0]*).

Information-gain. Information-gain in our information-directed sampling (IDS) implementation for bsuite is computed in a slightly modified way compared to the vanilla version. Nikolov et al. (2019) compute the information gain function $I(s,a)$ with

$$I(s,a) = \log\left(1 + \frac{\sigma^2(s,a)}{\rho^2(s,a)}\right) + \epsilon_2\,,$$

Algorithm 1 PE-DQN

initialize quantile parameters θ_1, target parameters $\tilde{\theta}_1$, and K heads
initialize categorical parameters θ_2, target parameters $\tilde{\theta}_2$, K heads
initialize grid $[z_1, \ldots, z_K]$
initialize bonus parameters ϕ, and target parameters $\tilde{\phi}$
initialize exploration rate β, learning rate α
initialize Buffer $\mathcal{B}$
sample initial state s_0
for $t = 0, \ldots, T$ **do**
    predict locations $[\theta_{11}, \ldots, \theta_{1K}](s_t, a)$ and probabilities $[\theta_{21}, \ldots, \theta_{2K}](s_t, a)$
    $Q(s_t, a) := \frac{1}{2} \sum_{k=1}^{K} \theta_{1k}(s_t, a) \frac{1}{K} + \theta_{2k}(s_t, a) z_k$
    predict bonus $b_\phi(s_t, a)$
    $a_t \leftarrow \arg\max_{a \in \mathcal{A}} \{Q(s_t, a) + \beta b_\phi(s_t, a)\}$
    for $j = 0, \ldots, N_{\text{trainsteps}}$ **do**
        sample transition tuple $(s_j, a_j, r_j, s'_j) \sim \mathcal{B}$
        predict locations $[\theta_{11}, \ldots, \theta_{1K}](s_j, a_j)$
        predict probabilities $[\theta_{21}, \ldots, \theta_{2K}](s_j, a_j)$
        predict target locations $[\tilde{\theta}_{11}, \ldots, \tilde{\theta}_{1K}](s'_j, a)$
        predict target probabilities $[\tilde{\theta}_{21}, \ldots, \tilde{\theta}_{2K}](s'_j, a)$
        $Q(s'_j, a) := \frac{1}{2} \sum_{k=1}^{K} \theta_{1k}(s'_j, a) \frac{1}{K} + \theta_{2k}(s'_j, a) z_k$
        $a'_j \leftarrow \arg\max_{a \in \mathcal{A}} \{Q(s'_j, a)\}$
        compute mixture target $\tilde{\eta}'_M \leftarrow \frac{1}{2} \sum_{k=1}^{K} \frac{1}{K} \delta_{r_j + \gamma \tilde{\theta}_{1k}(s'_j, a'_j)} + \tilde{\theta}_{2k}(s'_j, a'_j) \delta_{r_j + \gamma z_k}$
        compute quantile loss $l_1 \leftarrow \mathcal{L}_Q(\theta_1, \tilde{\eta}'_M)$
        compute categorical loss $l_2 \leftarrow \mathcal{L}_C(\theta_2, \tilde{\eta}'_M)$
        compute intrinsic reward $r_{\text{intr}} \leftarrow w_1(\sum_{k=1}^{K} \frac{1}{K} \delta_{\theta_{1k}(s_j, a_j)}, \sum_{k=1}^{K} \theta_{2k}(s_j, a_j)) \delta_{z_k}$
        compute bonus target $\tilde{b}' \leftarrow r_{\text{intr}} + \gamma b_{\tilde{\phi}}(s'_j, a'_j)$
        compute bonus loss $l_3 \leftarrow MSE(b_\phi(s_j, a_j), \tilde{b}')$
        $[\theta_1, \theta_2, \phi]^T \leftarrow [\theta_1, \theta_2, \phi]^T + \alpha \nabla_{\theta_1, \theta_2, \phi} (l_1 + l_2 + l_3)$
    end for
    execute a_t and store (s_t, a_t, r_t, s_{t+1}) in $\mathcal{B}$
end for

where $\sigma^2(s,a)$ is the empirical variance of BDQNP predictions, $\epsilon_2 = 1\times10^{-5}$ is a zero-division protection, and $\rho^2(s,a)$ is the clipped action-space normalized return variance

$$\rho(s,a)^2 = \max\left(\frac{\mathrm{Var}(Z(s,a))}{\frac{1}{|\mathcal{A}|}\sum_{a\in\mathcal{A}}\mathrm{Var}(Z(s,a))}, 0.25\right). \tag{A.4}$$

$\mathrm{Var}(Z(s,a))$ here is the variance of the distributional estimate provided by C51. We replace the clipping in Eq. (A.4) by adding a small constant $\epsilon_1 = 1\times10^{-4}$ to $\mathrm{Var}(Z(s,a))$, s.t.

$$\rho_\epsilon(s,a)^2 = \frac{\mathrm{Var}(Z(s,a)) + \epsilon_1}{\epsilon_1 + \frac{1}{|\mathcal{A}|}\sum_{a\in\mathcal{A}}\mathrm{Var}(Z(s,a))}.$$

Fig. A.2 (b) shows the effect of clipping as in the vanilla version (*IDS-C51 [clip]*) compared to our variation (*IDS-C51 [noclip]*) on the deep sea environment.

Intrinsic reward priors. Intrinsic reward priors are a computational method we implement with PE-DQN, which leverages the fact that we can compute the one-step uncertainty estimate $w_{\mathrm{avg}}(s,a)$ deterministically from a parametric ensemble given a state-action tuple. This obviates the need to learn it explicitly in the bonus estimation step. We thus add $w_{\mathrm{avg}}(s,a)$ automatically to the forward pass of the bonus estimator $b_\vartheta(s,a)$ as a sort of "prior" mechanism according to

$$b_\vartheta(s,a) := b_\vartheta^{\mathrm{raw}}(s,a) + w_{\mathrm{avg}}(s,a),$$

where $b_\vartheta^{\mathrm{raw}}$ is the raw output of the bonus estimator NN of parameters ϑ. In the VizDoom environment, we follow the default pipeline suggested by Burda et al. (2019b) and subsequent works (Burda et al., 2019a) that normalize intrinsic rewards by a running estimate of its marginal standard deviation.

Bonus decay. Bonus decay is the decaying of the exploratory bonus during action selection. It is well-known that the factor β is a sensitive parameter for UCB-type exploration algorithms, enabling efficient exploration when chosen correctly but simultaneously preventing proper convergence when chosen wrongly. Due to the variety of tasks included in the bsuite and VizDoom, we opted for a fixed schedule by which β is linearly decayed to 0.0 over one third of the total training horizon. In the bsuite experiments, we apply this schedule to all tested baselines where applicable and chose the initial β_{init} value according to the hyperparameter search. Since the decay rate is a central part of the DLTV algorithm, we here do not use our linearly deacying schedule but adopt the original decay rate of $\beta = \beta_0 * \sqrt{\log(\alpha t)/\alpha t}$ where α is a scaling parameter.

Ensembles. Ensembles and their size are a central parameter in IDS and BDQNP. For the bsuite experiments, we used a size of 20 as in the implementation by Osband et al. (2020), who find that increasing the ensemble size beyond 20 did not lead to significant performance improvements on the bsuite. Fig. A.2 (a) shows a comparison of the influence of ensemble size in BDQNP compared to PE-DQN. For VizDoom, we used 10 models in accordance with Nikolov et al. (2019) for their Atari experiments. Here, we follow the original implementations and let the ensembles used in BDQNP and IDS-C51 (and also PE-DQN) share a network body for feature extraction to save computation.

Replay buffer In the VizDoom environment, all our algorithms make use of prioritized experience replay (Schaul et al., 2016).

Computational resources. The **computational resources** we used to conduct the bsuite experiments were supplied by the Delft High Performance Computing Centre (DHPC) and the Delft Artificial Intelligence Cluster (DAIC). We deployed bsuite environments in 16 parallel jobs to be executed on 8 NVIDIA Tesla V100S 32GB GPUs, 16 Intel XEON E5-6248R 24C 3.0GHz CPUs, and 64GB of memory in total. In this setup, the execution of one seed on the entire suite experiment took approximately 38 hours for DLTV, 72 hours for PE-DQN, and 80 hours for IDS. Due to the narrower network architecture of BDQNP, we in this case parallelized environments over 64 Intel XEON E5-6248R 24C 3.0GHz CPUs, taking approximately 76 hours wall-clock time for the entire suite. In the VizDoom environments, we deployed 32 parallel environments for each agent on the same hardware. In this case, computation for 10×10^6 took approximately 24 hours per seed per environment and did not differ significantly between any of the tested methods. Table A.7 shows the average wall clock time for the VizDoom experiments.

A.1.3 Additional experimental results

Fig. A.4 illustrates a comparison of the uncertainty estimates used in PE-DQN for the deep sea environment. Every plot shows the entire state-space of the deep sea environment. In *deep sea*, the agent starts at the top left entry in a matrix and, depending on his action, moves to the left or right column while descending one row. The upper right triangular matrix above the diagonal is thus not reachable to the agent. The goal, i.e., the rewarding final state is located at the bottom right of the matrix.

For different time steps t (total environment interactions) during training, we evaluate the entire state-space and compare three quantities:

- *Inverse counts* are the inverse of visitations to each state-action $\frac{1}{N(s,a)+0.1}$. For every state, we plot the maximum of both actions.

Table A.7: VizDoom wall clock time comparisons

Environment	BDQNP	DLTV	IDS	PE-DQN
MyWayHome - Dense	14h 35m	14h 22m	16h 49m	17h 3m
MyWayHome - Sparse	14h 29m	13h 49m	16h 11m	16h 11m
MyWayHome - Very Sparse	21h 27m	21h 12m	23h 3m	23h 3m

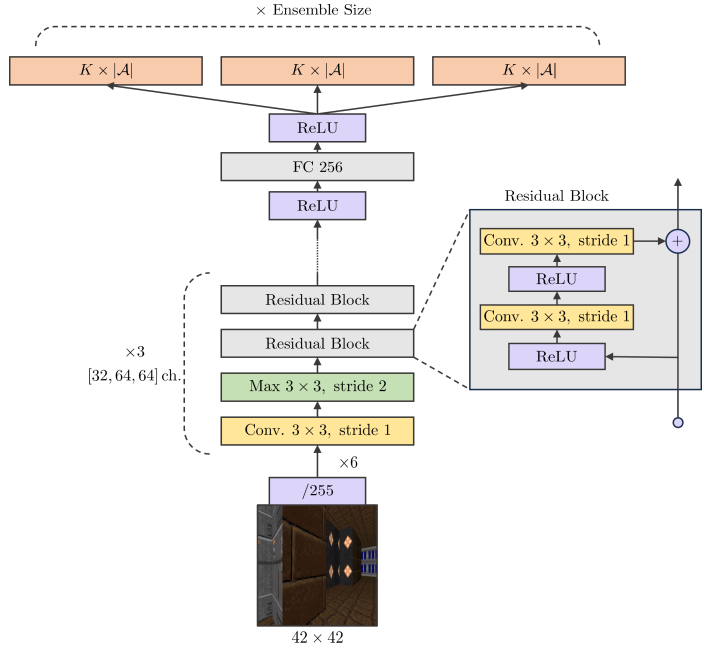


Figure A.3: Schematic of the architecture used for VizDoom environments. Based on the architecture used by Espeholt et al. (2018).

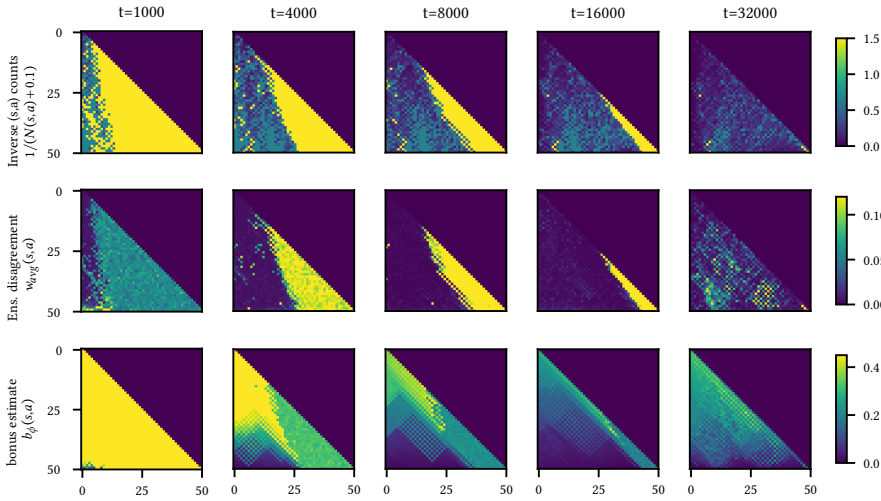


Figure A.4: A comparison of inverse counts (top row), ensemble disagreement (mid row), and bonus estimates (bottom row) on the deep sea environment. t indicates total environment interactions. Each image depicts the state-space of deep sea, where only the lower triangle (including the diagonal) is reachable. For each state, the plotted values indicate the maximum of two actions. At $t = 32000$, the agent has discovered the goal-state at the bottom right.

- *Ensemble disagreement*, with

$$w_{\text{avg}}(s,a) = 1/(M(M-1)) \sum_{i,j=1}^{M} w_1(\eta_{\theta_i}, \eta_{\theta_j})(s,a).$$

  For every state, we plot the maximum of both actions.

- *Bonus estimates* $b_\vartheta(s,a)$ as defined in Section 3.4. For every state, we plot the maximum of both actions.

In the top row, the agent has explored an increasing fraction of the state space with increasing time. The number of states with high inverse counts thus decreases. The ensemble disagreement $w_{\text{avg}}(s,a)$ behaves similarly to inverse counts, a result in line with the notion that $w_{\text{avg}}(s,a)$ serves as an estimate of the myopic, local TD error $w_1(\eta_{E,\theta}, \Omega_M \hat{\mathcal{T}}^\pi \eta_{E,\theta})(s,a)$, which is expected to decrease with number of visits. In contrast to this, we expect bonus estimates $b_\vartheta(s,a)$ to quantify errors w.r.t the true value, that is $w_1(\hat{\eta}, \eta^\pi)(s,a)$. As a result, $b_\vartheta(s,a)$ should not, for example, vanish prematurely for the initial state at the top left, even after many visitations, since its value can only be assessed upon having explored the entire state space. The bottom row of Fig. A.4 is closely in line with this intuition. At $t = 32000$, the agent has discovered the reward at the bottom right.

A.1.4 Full results of bsuite experiments

Fig. A.5 shows the averaged undiscounted episodic return for all bsuite tasks. Each curve represents the average over approximately 20 variations of the same task (Osband et al. (2020) provide a detailed account of the task variations) where results were taken from a separate evaluation episode using a greedy action-selection rule. In the "scale" environments, evaluation results were rescaled to the original reward range to maintain a sensible average. Bold titles indicate environments tagged as hard exploration tasks.

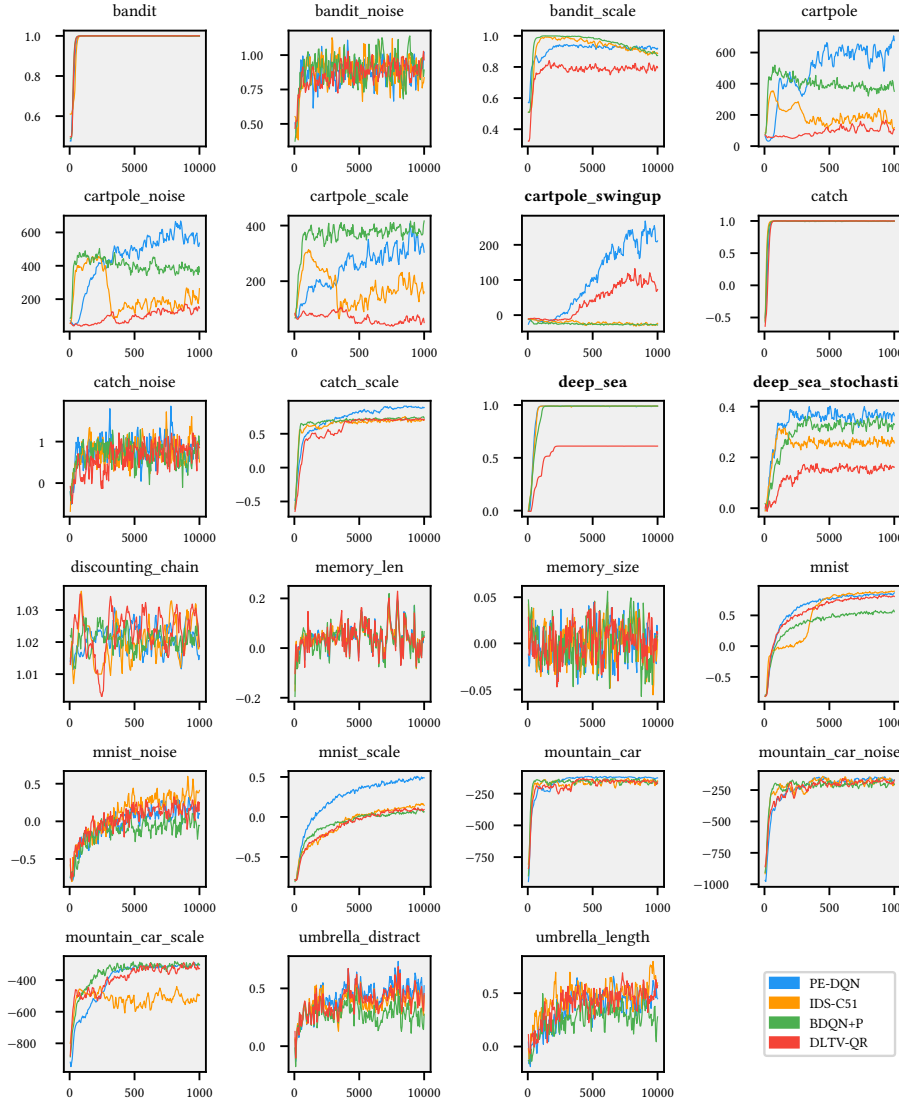


Figure A.5: Averaged episodic return for all 23 bsuite tasks.

B

Contextual Similarity Distillation

This appendix provides additional experimental results and implementation details for Chapter 4.

B.1 Experimental Details

In the following, we outline details on our experimental setup. This includes hyperparameter settings, hyperparameter search procedures, algorithmic and experimental details, and dataprocessing details.

B.1.1 Hyperparameter Settings

In order to facilitate comparable results, our experiments are conducted using a central codebase and follow similar modeling choices such as architectures, optimizer, etc. where sensible. All experiments use a resnet-based model (He et al., 2016) following the IMPALA architecture by Espeholt et al. (2018). We optimized essential and algorithm-specific hyperparameters through a search on a selected subset of experiments.

Distribution shift detection. In the supervised distribution shift detection, we use the IMPALA architecture with 2 residual blocks and channels widths 32 and 64. Hyperparameters were searched on the FashionMNIST dataset as the in-distribution set and the remaining datasets as out-of-distribution sets. Each dataset is normalized to zero-mean and standard deviation 1 using the training set statistics. For the main classifier we apply random horizontal flips (p=0.5), random vertical flips (p=0.5) and random sized crops (zoom range between 1.0 and 1.3) to training data in all experiments. Learning rate and algorithm-specific hyperparameters were optimized independently, meaning we first performed a search for learning rates, which we used in the (if applicable) sub-

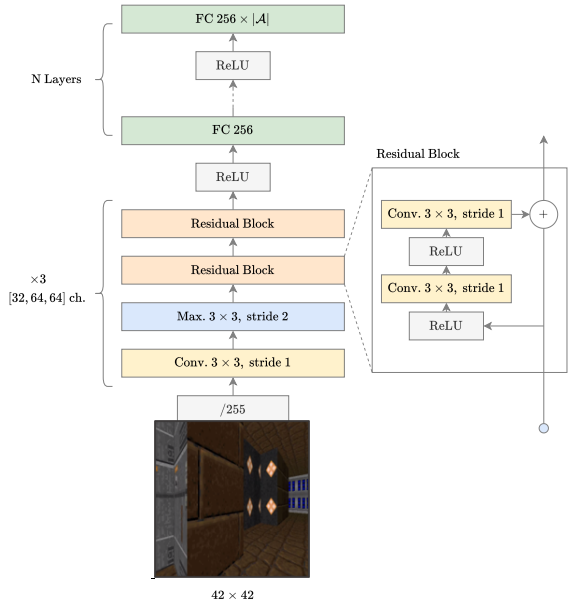


Figure B.1: Illustration of the architecture for VizDoom environments. Based on the architecture used by Espeholt et al. (2018).

sequent algorithm-specific parameter searches. Table B.1 contains lists of all searched parameters, with parenthesis indicating algorithm-specific parameters and italics indicating the parameter used during the learning rate search. The final hyperparameters were chosen based on the average AUROC metric and are reported in Table B.2.

VizDoom. In the RL experiments, we conducted a full grid search on the *MyWayHomeSparse* variation of the environment and chose parameters based on performance after $5 \cdot 10^6$ steps. Our basic network architecture is based on the rainbow (Hessel et al., 2018) network proposed by Schmidt and Schmied (2021) who in turn base their architecture on IMPALA (Espeholt et al., 2018) (see also Fig. B.1). We use 3 residual blocks with channel widths according to Table B.5. Detailed final hyperparameter settings are given in Table B.4. All agents furthermore use a data preprocessing pipeline as outlined in Table B.5.

B.1.2 Implementation Details

In this section, we briefly outline implementation details concerning CSD and the tested baselines.

Table B.1: Searched hyperparameters for distribution shift experiments.

Hyperparameter	Values
Learning rate (All)	$[10^{-4}, 3\cdot10^{-4}, 10^{-3}, 3\cdot10^{-3}, 10^{-2}, 3\cdot10^{-2}, 10^{-1}]$
Dropout probability (MCD)	[0.05, 0.1, *0.15*, 0.25, 0.5]
RND Learning rate (RND)	$[10^{-4}, 3\cdot10^{-4}, \mathit{10^{-3}}, 3\cdot10^{-3}, 10^{-2}, 3\cdot10^{-2}, 10^{-1}]$
CSD Learning rate (CSD)	$[10^{-4}, 3\cdot10^{-4}, \mathit{10^{-3}}, 3\cdot10^{-3}, 10^{-2}, 3\cdot10^{-2}, 10^{-1}]$

Table B.2: Hyperparameter settings for distribution shift experiments.

Hyperparameter	MCMC	Laplace	MCD	ENS	RND	CSD
Main Classifier Network						
Learning rate	10^{-3}	10^{-3}	$3\cdot10^{-4}$	10^{-3}	10^{-3}	10^{-3}
MLP hidden layers	2					
MLP layer width	256					
Channel Widths	32, 64					
RND/CSD Network						
Learning rate	n/a				$3\cdot10^{-3}$	10^{-2}
MLP hidden layers	n/a				2	2
MLP layer width	n/a				256	256
Channel Widths	n/a				16	32
Target hidden layers	n/a				1	1
Output dimensions	n/a				256	256
Ensemble size	n/a	n/a	n/a	3, 15	n/a	n/a
Dropout rate	n/a	n/a	0.1		n/a	
Prior Precision	n/a	100	n/a		n/a	
Posterior Temperature	1.0	1.0	n/a		n/a	
Posterior Samples	30	30	100		n/a	
Epochs per sample	2	n/a	n/a		n/a	
Burn-In Epochs	10	n/a	n/a		n/a	
Adam epsilon	n/a	10^{-5}	10^{-5}		10^{-5}	
Learning rate anneal	Linear					
Batch size	256					
Initialization	Orthogonal (Saxe et al., 2013)					

Table B.3: Searched hyperparameters for VizDoom

Hyperparameter	Values
Learning rate (all)	$[1.25\cdot10^{-4}, 2.5\cdot10^{-4}, 3.75\cdot10^{-4}, 5\cdot10^{-4}, 6.25\cdot10^{-4}, 7.5\cdot10^{-4}]$
Loss (all)	[Huber, C51]
Prior function scale (BDQNP, IDS)	[1.0, 3.0, 5.0]
Initial bonus β (RND, CSD)	[0.05, 0.1, 0.5, 1.0, 5.0, 10.0]
RND Learning rate (RND)	$[1.25\cdot10^{-4}, 2.5\cdot10^{-4}, 3.75\cdot10^{-4}, 5\cdot10^{-4}, 6.25\cdot10^{-4}, 7.5\cdot10^{-4}]$
CSD Learning rate (CSD)	$[1.25\cdot10^{-4}, 2.5\cdot10^{-4}, 3.75\cdot10^{-4}, 5\cdot10^{-4}, 6.25\cdot10^{-4}, 7.5\cdot10^{-4}]$

Table B.4: Hyperparameter settings for VizDoom experiments.

Hyperparameter	DQN	BDQNP	RND	IDS	CSD
Adam Learning rate	$2.5 \cdot 10^{-4}$	$2.5 \cdot 10^{-4}$	$6.25 \cdot 10^{-4}$	$2.5 \cdot 10^{-4}$	$6.25 \cdot 10^{-4}$
Prior function scale	n/a	1.0	n/a	1.0	n/a
Heads K	1	1	101	1 / 101	101/101
Ensemble size	n/a	10	n/a	10/1	n/a
Initial bonus β_{init}	n/a	n/a	1.0	0.1	0.1
Final bonus β_{final}	n/a	n/a	0.01	0.01	0.01
Bonus decay frames	n/a	n/a	$3.3 \cdot 10^6$	$3.3 \cdot 10^6$	$3.3 \cdot 10^6$
Loss function	Huber	Huber	C51	Huber/C51	C51
Channel Widths			32, 32, 64		
MLP hidden layers			1		
MLP layer width			256		
RND / CSD Network Parameters					
Adam Learning rate	n/a	n/a	$2.5 \cdot 10^{-4}$	n/a	$2.5 \cdot 10^{-4}$
Channel Widths	n/a	n/a	16, 16, 32	n/a	16, 16, 32
MLP hidden layers	n/a	n/a	1	n/a	1
MLP layer width	n/a	n/a	256	n/a	256
Target hidden layers	n/a	n/a	1	n/a	1
Output dimensions	n/a	n/a	256	n/a	256
Initial ϵ in ϵ-greedy			1.0		
Final ϵ in ϵ-greedy			0.01		
ϵ decay frames			500,000		
Training starts			100,000		
Discount			0.997		
Buffer size			1,000,000		
Batch size			256		
Parallel Envs			16		
Adam epsilon			0.005/batch size		
Initialization			He uniform (He et al., 2015)		
Gradient clip norm			10		
Regularization			spectral normalization (Gogianu et al., 2021)		
Double DQN			Yes (Hasselt, 2010)		
Update frequency			1		
Target lambda			1.0		
Target frequency			8000		
PER β_0			0.45 (Schaul et al., 2016)		
n-step returns			10		

Table B.5: VizDoom Preprocessing

Parameter	Value
Grayscale	Yes
Frame-skipping	No
Frame-stacking	6
Resolution	42×42
Max. Episode Length	2100

Data augmentations For both the distribution shift detection experiments (CSD-Aug.) and the VizDoom experiments, we add data augmentation to obtain additional context variables in CSD. In both experiments, we apply augmentations with a probability of $p = 0.25$ and specific augmentations are listed in Table B.6.

Data and context sampling. To compute the loss 4.15, we sample minibatches $\mathcal{X}_{mb}$ from a buffer or data set. Context minibatches $\mathcal{C}_{mb}$ either simply reuse $\mathcal{X}_{mb}$, are generated by applying data augmentations as outlines above, or by sampling from a context data set. We compute inner products over all pairings of the two batches with $\phi(\mathcal{X}_{mb}, \tilde{\theta}_f)^\top \psi(\mathcal{C}_{mb}, \tilde{\theta}_c) \in \mathbb{R}^{N_{mb} \times N_{mb}}$ and compute loss 4.15 elementwise. Finally, we sum the average diagonal loss and the average off-diagonal loss.

Normalization. During training, we normalize prior features by

$$\bar{\varphi}(x, \theta_0^{1:L-1}) = \frac{\varphi(x, \theta_0^{1:L-1})}{\|\varphi(x, \theta_0^{1:L-1})\|_2}, \tag{B.1}$$

feature vectors by

$$\bar{\phi}(x, \tilde{\theta}_f) = \frac{\phi(x, \tilde{\theta}_f)}{\|\phi(x, \tilde{\theta}_f)\|_2}, \tag{B.2}$$

and context vectors by

$$\bar{\psi}(c, \tilde{\theta}_c) = frac\psi(c, \tilde{\theta}_c)\|\psi(c, \tilde{\theta}_c)\|_2 . \tag{B.3}$$

When computing predictive variances at inference time, we rescale by

$$\mathbb{V}[f(x, \theta_\infty)] \approx \|\varphi(x, \theta_0^{1:L-1})\|_2^2 \big(\bar{\varphi}(x, \theta_0^{1:L-1})^\top \bar{\varphi}(x, \theta_0^{1:L-1}) - \bar{\phi}(x, \tilde{\theta}_f)^\top \bar{\psi}(c, \tilde{\theta}_c)\big), \tag{B.4}$$

to obtain predictions in the original scale again.

Small function initialization. While our theoretical suggests using small function initialization with $g(x, \tilde{\theta}_0) \approx 0, \forall x$, preliminary experiments with a reparametrization $\hat{g}(x, \tilde{\theta}_t) := g(x, \tilde{\theta}_t) - g(x, \tilde{\theta}_0)$ showed no significant differences. In our main implementation we thus refrain from using this reparametrization in favor of simplicity.

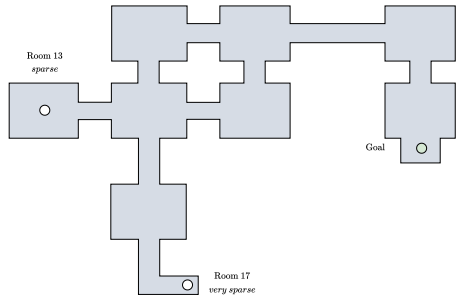


Figure B.2: Map for the VizDoom *MyWayHome* environment. Agents are spawned in the *sparse* and *very sparse* locations to vary the exploration difficulty.

Environment details. We conduct experiments on three variations of the VizDoom VizDoom environment *MyWayHome*. A top-down view of environment map is shown in Fig. B.2. In the *dense* setting, at the beginning of each episode agents are spawned in random positions of the map, such that the goal position is encountered stochastically without requiring coordinated exploration. The *sparsity* of the problem is increased by changing the agents spawning location deterministically to a room further from the goal position, that is *Room 13* for the *sparse* setting and *Room 17* for the *very sparse* setting. As described in Section 4.4, the reward function is sparse. A constant reward of $-1 * 10^{-4}$ is given every timestep and a reward of 1 is given for reaching the goal. Episodes are limited to a length of 2100 timesteps.

Reinforcement learning implementation. We outline the basic implementation of our deep Q-network (DQN)-based RL agent. The agent follows the same algorithmic flow as the established DQN-algorithm (Mnih et al., 2015) and subsequent variations (Hessel et al., 2018; Schmidt and Schmied, 2021). The agent maintains a replay buffer of transitions, from which we sample minibatches of transition $\mathcal{X}_{mb} = \{s_i, a_i, r_i, s_i', T_i\}_{i=1}^{N_{mb}}$, where T_i are terminations. Q-networks are then updated at a fixed frequency using the sampled minibatch. As is established, we use target networks with slow-moving parameters for value learning.

We provide intrinsic rewards as generated by CSD to the DQN agent to incentivize exploration. For all our experiments including intrinsic rewards (CSD and RND), we use separate value functions for the intrinsic reward and employ *intrinsic reward priors*, a mechanism suggested by Zanger et al. (2024) which includes intrinsic rewards to the forward pass of the value network. This addresses a common issue with intrinsic reward learning as described previously by Rashid et al. (2020) by preventing underestimation of unseen actions. Specifically, intrinsic reward priors redefine the forward pass of the intrinsic

Q-function according to

$$\hat{Q}_{\text{in}}(s,a,\theta,\theta_{\text{in}}) = Q_{\text{in}}(s,a,\theta) + \frac{1}{2} r_{\text{in}}(s,a,\theta_{\text{in}}),$$

where $r_{\text{in}}(s,a,\theta_{\text{in}})$ denotes an intrinsic reward term, in our experiments generated by either RND or CSD with parameters θ_{in}. The altered function $\hat{Q}_{\text{in}}(s,a,\theta,\theta_{\text{in}})$ is then used as a drop-in replacement for the Q-function in the used algorithm.

Pseudocode for reinforcement learning experiments. We provide pseudocode for a DQN agent with CSD in Algorithm 2. For clarity, we omit standard algorithmic details such as double Q-learning, distributional value functions, prioritized experience replay, separate value functions for intrinsic reward, and intrinsic reward priors.

Algorithm 2 CSD-DQN

initialize CSD model $g(s,a,s_c,a_c,\tilde{\theta}_t) = \phi(s,a,\tilde{\theta}_t)^\top \psi(s_c,a_c,\tilde{\theta}_t)$ with $\tilde{\theta}_0$.
initialize CSD prior $\Theta^L(s,a,s_c,a_c,c,\theta_p) = \varphi(s,a,\theta_p)^\top \varphi(s_c,a_c,\theta_p)$ with $\tilde{\theta}_p$.
initialize Q-function $Q(s,a,\theta_t)$ with θ_0 and target parameters $\bar{\theta}_0$.
sample initial state s_0 from the environment.
for $t = 1,\dots,T$ **do**
    take action $a \leftarrow \arg\max_{a' \in \mathcal{A}} \{Q(s,a')\}$
    obtain observations (s_t, r_t, T_t) from the environment.
    store samples $(s_{t-1}, a_{t-1}, r_t, s_t, T_t)$.
    sample transition tuple $\{s_i, a_i, r_i, s'_i, T_i\}_{i=1}^{N_{mb}} \sim \mathcal{B}$ from buffer
    sample context tuple $\{\hat{s}_i, \hat{a}_i, \hat{r}_i, \hat{s}'_i, \hat{T}_i\}_{i=1}^{N_{mb}} \sim \mathcal{B}$ from buffer
    generate intrinsic reward $r_{\text{in}} := \Theta^L(s_i,a_i,s_i,a_i,\tilde{\theta}_p) - g(s_i,a_i,s_i,a_i,\tilde{\theta}_t)$.
    generate next action $a'_i := \arg\max_{a' \in \mathcal{A}} \{Q(s'_i,a',\theta_t)\}$.
    update Q-function $\theta_t \leftarrow \theta_t - \nabla_{\theta_t} \mathcal{L}(\theta_t)$ with

$$\mathcal{L}(\theta_t) = \frac{1}{2N_{mb}} \sum_i^{N_{mb}} \left(r_i + \beta r_{\text{in}} + Q(s_i,a_i,\bar{\theta}_t) - Q(s'_i,a'_i,\theta_t) \right)^2.$$

    update CSD model $\tilde{\theta}_t \leftarrow \tilde{\theta}_t - \nabla_{\tilde{\theta}_t} \mathcal{L}(\tilde{\theta}_t)$ with

$$\mathcal{L}(\tilde{\theta}_t) = \frac{1}{2N_{mb}} \sum_i^{N_{mb}} \left(g(s_i,a_i,\hat{s}_i,\hat{a}_i,\tilde{\theta}_t) - \Theta^L(s_i,a_i,\hat{s}_i,\hat{a}_i,\tilde{\theta}_p) \right)^2.$$

    if t % freq == 0 **then**
        update target parameters $\bar{\theta}_t \leftarrow \lambda\theta_t + (1-\lambda)\bar{\theta}_t$
    end if
end for

B.1.3 Additional Experimental Results

We report the detailed results of our distribution shift detection tasks. Tables B.7 to B.10 show OOD detection metrics for the datasets FashionMNIST,

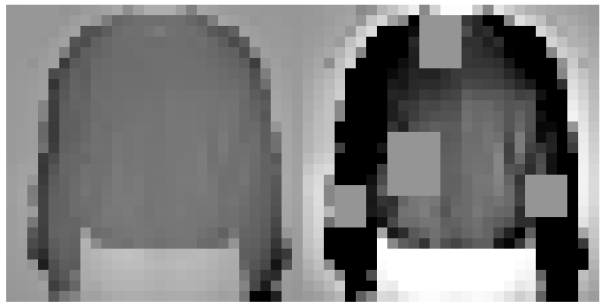


Figure B.3: Left: Original Image. Right: Perturbed OOD Image.

MNIST, NotMNIST, and KMNIST. Each table shows the test accuracy and average AUROC, AUPR-IN and AUPR-OUT scores against the remaining three training datasets and an additional perturbed dataset. The perturbed dataset is constructed by applying data augmentations to the ID dataset. In our experiments, we use random brightness changes ($p = 1.0, r = [-1.0, 1.0]$), random contrast changes($p = 1.0, r = [-1.0, 1.0]$), and randomly set patches of an image to zero ($p = 1.0, r = [-1.0, 1.0]$). Fig. B.3 shows an example of this.

Table B.6: Data augmentations for context data.

Distribution Shift	VizDoom
RandomHorizontalFlip($p = 0.25$)	RandomPerspective($p = 0.5$)
RandomVerticallFlip($p = 0.25$)	RandomHorizontalFlip($p = 0.5$)
Rotate($p = 0.25$)	RandomResizedCrop(r = $[0.75, 1.0]$)
GaussianBlur($\sigma = 1.0$, $p = 0.25$)	
RandomResizedCrop(r = $[0.75, 1.0]$)	
RandomBrightness(r = $[-1.0, 1.0]$, $p = 0.5$)	
RandomContrast(r = $[-1.0, 1.0]$, $p = 0.5$)	

Table B.7: Distribution Shift Detection. FashionMNIST as ID dataset.

Method	**Acc.**	**AUROC**	**AUPR-IN**	**AUPR-OUT**
MCD	89.24 ± 0.36	82.23 ± 0.48	79.88 ± 0.75	83.01 ± 0.34
BNN-MCMC	85.73 ± 0.24	85.01 ± 0.62	85.16 ± 0.68	83.38 ± 0.62
BNN-Laplace	88.57 ± 0.80	86.50 ± 0.67	86.32 ± 0.75	85.95 ± 0.75
RND	91.90 ± 0.15	$\mathbf{93.93} \pm 0.73$	$\mathbf{93.45} \pm 1.12$	$\mathbf{93.64} \pm 0.52$
ENS(3)	92.90 ± 0.09	88.90 ± 0.20	89.63 ± 0.19	88.16 ± 0.20
ENS(15)	$\mathbf{93.33} \pm 0.06$	91.93 ± 0.12	92.83 ± 0.11	91.09 ± 0.12
CSD	91.93 ± 0.17	$\mathbf{96.18} \pm 0.67$	$\mathbf{96.49} \pm 0.74$	$\mathbf{95.74} \pm 0.62$
CSD-Aug.	91.92 ± 0.16	$\mathbf{97.84} \pm 0.30$	$\mathbf{98.24} \pm 0.27$	$\mathbf{97.34} \pm 0.31$
CSD-OOD.	91.96 ± 0.13	$\mathbf{97.35} \pm 0.50$	$\mathbf{97.87} \pm 0.45$	$\mathbf{96.72} \pm 0.56$

Table B.8: Distribution Shift Detection. MNIST as ID dataset.

Method	**Acc.**	**AUROC**	**AUPR-IN**	**AUPR-OUT**
MCD	98.97 ± 0.06	90.03 ± 0.23	87.70 ± 0.38	89.01 ± 0.32
BNN-MCMC	94.29 ± 0.39	80.24 ± 2.19	80.20 ± 2.05	77.33 ± 2.56
BNN-Laplace	94.17 ± 1.01	74.05 ± 1.70	72.24 ± 1.90	74.39 ± 1.73
RND	99.85 ± 0.02	94.66 ± 0.52	93.83 ± 0.95	$\mathbf{94.25} \pm 0.35$
ENS(3)	99.95 ± 0.01	94.03 ± 0.24	95.09 ± 0.22	92.32 ± 0.31
ENS(15)	$\mathbf{99.97} \pm 0.00$	$\mathbf{95.33} \pm 0.06$	$\mathbf{96.31} \pm 0.06$	93.79 ± 0.10
CSD	99.88 ± 0.01	$\mathbf{96.78} \pm 0.58$	$\mathbf{96.96} \pm 0.72$	$\mathbf{96.25} \pm 0.57$
CSD-Aug.	99.87 ± 0.02	$\mathbf{98.39} \pm 0.17$	$\mathbf{98.63} \pm 0.20$	$\mathbf{97.94} \pm 0.19$
CSD-OOD.	99.87 ± 0.02	$\mathbf{99.37} \pm 0.08$	$\mathbf{99.51} \pm 0.07$	$\mathbf{99.14} \pm 0.11$

Table B.9: Distribution Shift Detection. NotMNIST as ID dataset.

Method	**Acc.**	**AUROC**	**AUPR-IN**	**AUPR-OUT**
MCD	95.17 ± 0.14	83.21 ± 0.45	75.86 ± 0.89	85.73 ± 0.18
BNN-MCMC	90.20 ± 0.44	87.05 ± 0.80	85.93 ± 1.10	87.68 ± 0.63
BNN-Laplace	95.29 ± 0.52	86.38 ± 1.46	82.99 ± 2.36	87.55 ± 1.04
RND	96.25 ± 0.12	$\mathbf{95.49} \pm 0.82$	$\mathbf{95.81} \pm 0.97$	$\mathbf{95.23} \pm 0.74$
ENS(3)	97.12 ± 0.08	92.37 ± 0.26	92.11 ± 0.30	91.93 ± 0.27
ENS(15)	$\mathbf{97.47} \pm 0.05$	94.04 ± 0.16	94.26 ± 0.17	93.29 ± 0.17
CSD	96.48 ± 0.08	$\mathbf{96.98} \pm 0.41$	$\mathbf{97.26} \pm 0.44$	$\mathbf{96.86} \pm 0.36$
CSD-Aug.	96.45 ± 0.09	$\mathbf{98.51} \pm 0.22$	$\mathbf{98.70} \pm 0.24$	$\mathbf{98.31} \pm 0.21$
CSD-OOD.	96.49 ± 0.10	$\mathbf{98.49} \pm 0.35$	$\mathbf{98.78} \pm 0.29$	$\mathbf{98.21} \pm 0.42$

Table B.10: Distribution Shift Detection. KMNIST as ID dataset.

Method	**Acc.**	**AUROC**	**AUPR-IN**	**AUPR-OUT**
MCD	94.18 ± 0.26	87.22 ± 0.75	83.48 ± 0.74	88.00 ± 0.77
BNN-MCMC	80.57 ± 1.29	80.40 ± 1.46	79.31 ± 1.93	80.75 ± 1.31
BNN-Laplace	85.39 ± 1.79	78.58 ± 2.66	76.18 ± 3.11	79.47 ± 2.49
RND	96.73 ± 0.21	93.50 ± 1.17	93.58 ± 1.45	92.93 ± 1.05
ENS(3)	97.68 ± 0.10	93.88 ± 0.24	94.49 ± 0.26	93.05 ± 0.24
ENS(15)	**97.96** ± 0.06	**94.68** ± 0.11	**95.39** ± 0.12	**93.81** ± 0.11
CSD	96.89 ± 0.13	**96.57** ± 0.73	**97.05** ± 0.74	**95.90** ± 0.74
CSD-Aug.	96.90 ± 0.19	**98.12** ± 0.46	**98.45** ± 0.41	**97.61** ± 0.53
CSD-OOD.	96.86 ± 0.12	**99.06** ± 0.19	**99.30** ± 0.14	**98.71** ± 0.25

C

Universal Value-Function Uncertainties

C.1 Experimental Details

We provide details on our experimental setup, implementations and additional results. This includes architectural design choices, algorithmic design choices, hyperparameter settings, hyperparameter search procedures, and environment details.

C.1.1 Implementation Details

All algorithms are self-implemented and tuned in JAX (Bradbury et al., 2018). A detailed exposition of our design choices and parameters follows below.

Environment setup. We use a variation of the GoToDoor environment of the minigrid suite (Chevalier-Boisvert et al., 2023). As our focus is not on partially observable settings, we use fully observable 35-dimensional state descriptions with $\mathcal{S} = \mathbb{R}^{35}$. Observation vectors comprise the factors:

$$o = \left(o_{\text{agent-pos}}^\top, o_{\text{agent-dir}}^\top, o_{\text{door-config}}^\top, o_{\text{door-pos}}^\top\right)^\top, \tag{C.1}$$

where $o_{\text{agent-pos}} \in \mathbb{R}^2$ is the agent position in x, y-coordinates, $o_{\text{agent-dir}} \in \mathbb{R}$ is a scalar integer indicating the agent direction (takes on values between 1 and 4), $o_{\text{door-config}} \in \mathbb{R}^{24}$ is the door configuration, comprising 4 one-hot encoded vectors indicating each door's color, and $o_{\text{door-pos}} \in \mathbb{R}^8$ is a vector containing the x, y-positions of the four doors. The action space is discrete and

four-dimensional with the following effects

$$a_{\text{effect}} = \begin{cases} \text{turn left} & \text{if } a = 0, \\ \text{turn right} & \text{if } a = 1, \\ \text{go forward} & \text{if } a = 2, \\ \text{open door} & \text{if } a = 3. \end{cases} \tag{C.2}$$

Tasks are one-hot encodings of the target door color, that is $z \in \mathbb{R}^6$ and in the online setting are generated such that they are achievable. The reward function is an indicator function of the correct door being opened, in which case a reward of 1 is given to the agent and the agent position is reset to a random location in the grid. Episodes terminate only upon reaching the maximum number of timesteps (50 in our experiments).

In the task rejection setting described in our evaluation protocol, an agent in a start state s_0 is presented a list of tasks, which may or may not be attainable, and is allowed to reject a fixed number of tasks from this list. In our experiments, the agent is allowed to reject 4 out of 6 total tasks at the beginning of each episode.

Data collection. Our offline datasets are recorded replay buffers from a DQN-agent deployed to the GoToDoor environment with an ϵ-greedy exploration strategy and a particular policy: When the door indicated by the task encoding z provided by the environment lies at the south or west wall, the regular policy by the online DQN agent is executed. If the target door lies at the north or east wall, however, actions are generated by a fixed random Q-network. This mixture policy emulates a policy that exhibits expert performance on certain combinations of tasks and states, but suboptimal behavior for other combinations. The replay buffer does, however, contain most combinations of states and tasks, albeit some with trajectories from suboptimal policies. Hyperparameter details of the online agent are provided in section C.1.2.

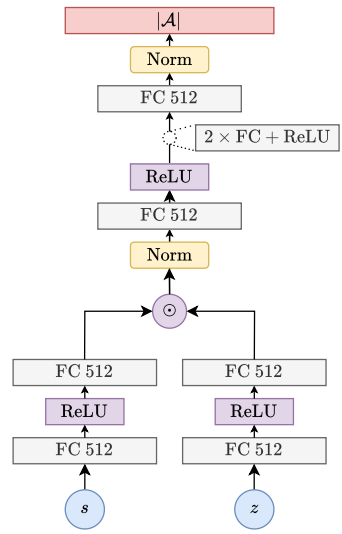


Figure C.1: Illustration of the used architecture. ⊙ indicates elementwise multiplication.

Algorithmic details. All tested algorithms and experiments are based on DQN agents (Mnih et al., 2015) which we adapted for the task-conditioned universal value function (Schaul et al., 2015) setting. While our theoretical analysis considers full-batch gradient descent, in practice we sample minibatches from offline datasets with $\mathcal{X}_{mb} = \{(s_i, a_i, z_i)\}_{i=1}^{N_{mb}}$, $\mathcal{X}'_b = \{(s'_i, a'_i, z_i)\}_{i=1}^{N_b}$, where next-state actions are generated by the policy $a'_i = \text{argmax}_{a \in \mathcal{A}} Q(s'_i, a, z_i, \theta_t)$ and rewards are $r = \{r_i\}_{i=1}^{N_{mb}}$. Moreover, we deviate from our theoretical analysis and use target networks in place of the stop-gradient operation. Here, a separate set of parameters $\tilde{\theta}_t$ is used to generate bootstrap targets in the TD loss which is in practice given by

$$\mathcal{L}(\theta_t) = \frac{1}{2} \| \gamma Q(\mathcal{X}'_{mb}, \tilde{\theta}_t) + r - Q(\mathcal{X}_{mb}, \theta_t) \|_2^2. \tag{C.3}$$

The parameters $\tilde{\theta}_t$ are updated towards the online parameters θ_t at fixed intervals through polyak updating, as is common. We use this basic algorithmic pipeline for all tested algorithms, including the online agent used for data collection.

Architectural details. We use a hypernetwork MLP architecture adapted to the DQN setting, as depicted in Fig. C.1. Specifically, this means we pass states s and task encodings z through single-layer encoders, which are then joint by elementwise multiplication. The resulting vector is normalized by its l^2 norm, $x' = \frac{x}{\|x\|_2}$. This joint vector is passed thorugh a 3-layer MLP with network width 512, again normalized by its l^2 norm and finally passed through a fully-connected layer to obtain a vector of dimension $\mathbb{R}^{|\mathcal{A}|}$. Although our experiments are conducted in the offline RL setting, preliminary experiments showed no benefits of using ensemble-based pessimism (An et al., 2021) or conservative Q-updates (Kumar et al., 2020). Instead, our normalization pipeline appears to sufficiently address overestimation issues as is suggested by several recent works (Gallici et al., 2024; Yue et al., 2023).

Independent bootstrapping. For the ensemble-based BDQNP baseline and our UVU model, we perform independent bootstrapping in the TD loss computation. By this, we mean that both the bootstrapped value and actions are generated by individual Q-functions. In the case of BDQNP, this means we compute Loss C.3 for each model Q_k, indexed by $k \in [1, \dots, K]$ with $\mathcal{X}_{mb,k} = \mathcal{X}_{mb}$ and bootstraps are generated as

$$\mathcal{X}'_{mb,k} = \{(s'_i, a'_{ik}, z_i)\}_{i=1}^{N_{mb}}, \quad \text{and} \quad a'_{ik} = \text{argmax}_{a \in \mathcal{A}} Q_k(s'_i, a, z_i, \theta_t). \tag{C.4}$$

Note, that this procedure is established (Osband et al., 2016) and serves the purpose of maintaining independence between the models in the ensemble. In

order to conduct the same procedure in our UVU method, where we have access to only one Q-function, we generate K distinct Q-estimates by computing

$$Q_k^{UVU}(s,a,z,\theta_t) := Q(s,a,z,\theta_t) + \epsilon_k(s,a,z,\vartheta_t,\psi_0), \tag{C.5}$$

that is, by adding the UVU error of the k-th output head. Bootstraps are then generated according to Eq. C.4.

Intrinsic reward priors. Intrinsic reward priors are a trick suggested by Zanger et al. (2024) to address a shortcoming of propagation methods used for intrinsic reward methods like RND(Burda et al., 2019b; O'Donoghue et al., 2018). The issue is that while learning a Q-function with intrinsic rewards can, with the right choice of intrinsic reward, provide optimistic estimates of the value function, but only for state-action regions covered in the data. A potential underestiation of the optimistic bound, however, counteracts its intention, a phenomenon also described by Rashid et al. (2020). Intrinsic reward priors are a heuristic method to address this issue by adding local, myopic uncertainty estimates automatically to the forward pass of the intrinsic Q-function, leading to a "prior" mechanism that ensures a

$$\hat{Q}_{intr}(s,a,z,\theta_t) = Q_{intr}(s,a,z,\theta_t) + \frac{1}{2}\epsilon_{rnd}(s,a,z,\theta_{rnd})^2$$

where $\epsilon_{rnd}(s,a,z,\theta_{rnd})$ denotes a local RND error as an example. The altered function $\hat{Q}_{intr}(s,a,z,\theta_t)$ is trained as usual with Loss C.3 and intrinsic rewards $\frac{1}{2}\epsilon_{rnd}(s,a,z,\theta_{rnd})^2$.

C.1.2 Hyperparameter Settings

To ensure a consistent basis for comparison across our findings, all experimental work was carried out using a shared codebase. We adopted standardized modeling approaches, including uniform choices for elements like network architectures and optimization algorithms, where appropriate. Specifically, every experiment employed the same architecture as detailed in Appendix C.1.1. Key hyperparameters, encompassing both foundational and algorithm-specific settings, were tuned through a grid search on the 10×10 variation of the *GoToDoor* environment. The search grid and final hyperparamters are provided in Tables C.1 and C.2 respectively. DQN in Table C.2 refers to the online data collection agent.

C.1.3 Additional Experimental Results

We report additional results of the illustrative experiment shown in Section 6.3. In Fig. C.2, we show different uncertainty estimates in the described chain

Table C.1: Hyperparameter search space

Hyperparameter	Values
Q Learning rate (all)	$[1 \cdot 10^{-6}, 3 \cdot 10^{-6}, 1 \cdot 10^{-5}, 3 \cdot 10^{-5}, 1 \cdot 10^{-4}, 3 \cdot 10^{-4}, 1 \cdot 10^{-3}]$
Prior function scale (BDQNP)	$[0.1, 0.3, 1.0, 3.0, 10.0]$
RND Learning rate (RND, RND-P)	$[1 \cdot 10^{-6}, 3 \cdot 10^{-6}, 1 \cdot 10^{-5}, 3 \cdot 10^{-5}, 1 \cdot 10^{-4}, 3 \cdot 10^{-4}, 1 \cdot 10^{-3}]$
UVU Learning rate (UVU)	$[1 \cdot 10^{-6}, 3 \cdot 10^{-6}, 1 \cdot 10^{-5}, 3 \cdot 10^{-5}, 1 \cdot 10^{-4}, 3 \cdot 10^{-4}, 1 \cdot 10^{-3}]$

Table C.2: Hyperparameter settings for *GoToDoor* experiments.

Hyperparameter	DQN	BDQNP	DQN-RND	DQN-RND+P	UVU
Adam Q-Learning rate	$3 \cdot 10^{-4}$	$3 \cdot 10^{-4}$	$3 \cdot 10^{-4}$	$3 \cdot 10^{-4}$	$3 \cdot 10^{-4}$
Prior function scale	n/a	1.0	n/a	n/a	n/a
N-Heads 1	1	1	1 / 512	1 / 512	1 / 512
Ensemble size	n/a	3 / 15	n/a	n/a	n/a
MLP hidden layers	3				
MLP layer width	512				
Discount γ	0.9				
Batch size	512				
Adam epsilon	0.005/batch size				
Initialization	He uniform (He et al., 2015)				
Double DQN	Yes (Hasselt, 2010)				
Update frequency	1				
Target lambda	1.0				
Target frequency	256				

Table C.3: GoToDoor Environment Settings

Parameter	Value
State space dim	35
Action space dim	3
Task space dim	6
N Task Rejections	4
Max. Episode Length	50

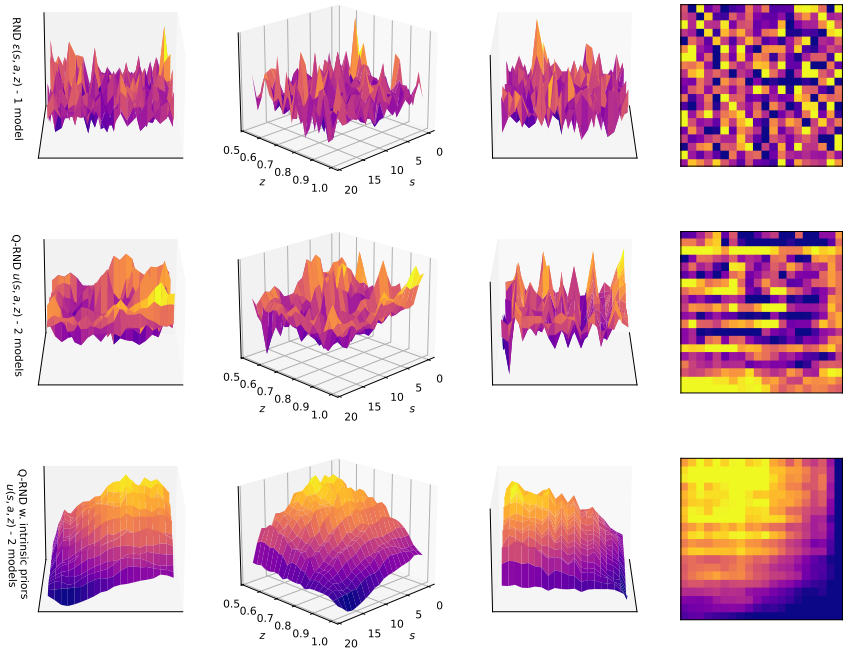


Figure C.2: *Top Row*: RND errors. *2nd Row*: Value uncertainty as measured by an intrinsic Q-function. *3rd Row*: Value uncertainty as measured by an intrinsic Q-function with intrinsic reward priors.

environment. The first row depicts *myopic* uncertainty estimates or, equivalently, RND errors. The second and third row show propagated local uncertainties with and without the intrinsic reward prior mechanism respectively. This result shows clearly the shortcoming of the standard training pipeline for intrinsic rewards: in a standard training pipeline, the novelty bonus of RND is given only for transitions (s_i, a_i, z_i, s_i') already present in the dataset and is never evaluated for OOD-actions. To generate reliable uncertainty estimates, RND requires, in addition to the RND network and the additional intrinsic Q-function, an algorithmic mechanism such as the *intrinsic reward priors* or even more sophisticated methods as described by Rashid et al. (2020).

Efficient Uncertainty Quantification in Deep Reinforcement Learning

As artificial intelligence is increasingly integrated into high-stakes domains, from autonomous driving to medical diagnostics, its predictions remain inherently subject to uncertainty. To develop safe behaviors and to generate actionable predictions, a system must know what it does not know.

This dissertation develops novel techniques that allow for efficient and reliable uncertainty quantification. The herein conducted research focuses on the field of reinforcement learning, where the long-term consequences of sequential decision-making further complicate accurate uncertainty estimation. To this end, this thesis draws on techniques from deep learning theory to explicitly account for the behaviors of deep neural networks as the central model type.

TU Delft