Promises of Deep Kernel Learning for Control Synthesis

Reed, Robert; Laurenti, Luca; Lahijanian, Morteza

DOI
10.1109/LCSYS.2023.3340995

Publication date
2023

Document Version
Final published version

Published in
IEEE Control Systems Letters

Promises of Deep Kernel Learning for Control Synthesis

Robert Reed, *Member, IEEE*, Luca Laurenti, *Member, IEEE*, and Morteza Lahijanian, *Member, IEEE*

***Abstract*—Deep Kernel Learning (DKL) combines the representational power of neural networks with the uncertainty quantification of Gaussian Processes. Hence, it is potentially a promising tool to learn and control complex dynamical systems. In this letter, we develop a scalable abstraction-based framework that enables the use of DKL for control synthesis of stochastic dynamical systems against complex specifications. Specifically, we consider temporal logic specifications and create an end-to-end framework that uses DKL to learn an unknown system from data and formally abstracts the DKL model into an interval Markov decision process to perform control synthesis with correctness guarantees. Furthermore, we identify a deep architecture that enables accurate learning and efficient abstraction computation. The effectiveness of our approach is illustrated on various benchmarks, including a 5-D nonlinear stochastic system, showing how control synthesis with DKL can substantially outperform state-of-the-art competitive methods.**

***Index Terms*—Machine learning, robust control, stochastic systems.**

I. Introduction

DATA-DRIVEN control synthesis is emerging as a central research topic in recent years [1], [2], [3], [4], [5]. This is due to three main reasons: (i) increased complexity of modern systems, (ii) availability of data in large scale, and (iii) increased capability of machine learning (ML) techniques. There are however several challenges in data-driven approaches for control systems, especially in *safety-critical* applications where robustness guarantees are vital. Such guarantees are conditioned on quantification of the learning error and its propagation through the control synthesis procedure. While there exist ML techniques that supply information about the error [6], they are often empirical and lack necessary mathematical rigor. Those methods that do provide formal error analysis [7] suffer from scalability [8], [9]. This letter aims to provide a scalable data-driven synthesis framework with robustness guarantees.

Formal synthesis is a rigorous approach to providing guarantees on the performance of control systems against complex properties [10], [11]. In that approach, specifications are expressed in a formal language such as *linear temporal logic* (LTL) over *finite* behaviors (LTLf) [12] and the system progression is abstracted into a finite model called an *abstraction*. Then, automated model-checking-like algorithms are used on the abstraction to synthesize a controller. To ensure correctness, the abstraction must have a *simulation* relation with the system, which is often achieved by including all the uncertainties, e.g., errors due to discretization, stochasticity, and learning, in the abstraction. A popular model that allows that is Interval Markov Decision Process (IMDP) [13], which is shown to also enable scalability to high dimensional systems [14]. Constructing a scalable IMDP abstraction, however, requires tight uncertainty bounds, which is difficult to achieve in a data-driven setting.

A widely-used method for accurate representation of the latent control system from data is *Gaussian Process* (GP) regression [7], [8], [15]. Its power lies in rigorous uncertainty quantification, which comes at the expense of cubic computational complexity in the size of data. That makes GPs ideal for formal control synthesis, but they suffer in high dimensional spaces, where a massive amount of data is required to obtain small uncertainty. For high-dimensional systems, *neural networks* (NNs) are successfully used to learn the dynamics, called *NN dynamic models* (NNDMs) [16], with control synthesis methods [17], [18]. However, quantification of the learning error of NNDMs in a formal manner remains an open problem in spite of recent attempts to use confidence-based approaches [6], which cannot be propagated through the synthesis procedure.

In this letter, we bridge the gap by introducing a scalable synthesis framework that employs *deep kernel learning* (DKL) [19], [20], which uses NNs as informed priors for GPs while maintaining an analytical posterior, to efficiently construct (accurate) IMDP abstractions. We leverage recent techniques for linear relaxations of NNs [21] and provide bounds on the mean and variance of the GP. Critically, we show that the optimization problems that bound the probabilities in the IMDP construction reduce to evaluations of a finite set of points on an analytical function, resulting in computational efficiency. Then, we employ existing tools [11] to synthesize a strategy on the IMDP that maximizes the probability of satisfying a given LTLf specification and is robust against the learning error. We prove that this strategy

Manuscript received 14 September 2023; revised 13 November 2023; accepted 2 December 2023. Date of publication 8 December 2023; date of current version 10 January 2024. This work was supported in part by the Air Force Research Laboratory (AFRL) under Agreement FA9453-22-2-0050. Recommended by Senior Editor P. Tesi. *(Corresponding author: Robert Reed.)*

Robert Reed and Morteza Lahijanian are with the Department of Aerospace Engineering Sciences, University of Colorado Boulder, Boulder, CO 80304 USA (e-mail: Robert.Reed-1@colorado.edu; Morteza.Lahijanian@colorado.edu).

Luca Laurenti is with the Delft Center for Systems and Control, TU Delft, 2628 CN Delft, The Netherlands (e-mail: L.Laurenti@tudelft.nl).

Digital Object Identifier 10.1109/LCSYS.2023.3340995

can be mapped to the underlying latent system with correctness guarantees. We illustrate the efficacy of our framework on various benchmarks, which show control synthesis with DKL substantially outperforms state-of-the-art methods. We also identify a DKL architecture that results in high accuracy and efficient abstraction construction, promoting further scalability.

In summary, the contributions are: (i) a scalable data-driven framework for control synthesis with complex specifications and hard guarantees, (ii) an efficient finite abstraction technique for DKL models with correctness guarantees, (iii) a DKL architecture design for fast and accurate abstraction, and (iv) illustration of the efficacy and scalability of the framework via benchmarking against state-of-the-art methods on a set of rich case studies with complex nonlinear stochastic systems up to 5 dimensions.

II. Problem Formulation

Consider the following discrete-time stochastic system:

$$\mathbf{x}(k+1) = f(\mathbf{x}(k), \mathbf{u}(k)) + \mathbf{v}(k), \tag{1}$$

where $\mathbf{x}(k) \in \mathbb{R}^n$, $\mathbf{u}(k) \in U$, $U = \{a_1, \ldots, a_{|U|}\}$ is a finite set of actions or control laws, $\mathbf{v}(k) \in \mathbb{R}^n$ is a Gaussian random variable $\mathbf{v}(k) \sim \mathcal{N}(0, \mathcal{V})$ with zero mean and covariance $\mathcal{V} \in \mathbb{R}^{n \times n}$, and $f : \mathbb{R}^n \times U \to \mathbb{R}^n$ is an *unknown* function. Without loss of generality, we assume covariance $\mathcal{V}$ is diagonal.[1] Intuitively, System (1) represents a switched stochastic systems with additive noise and unknown dynamics.

We define a *finite trajectory* of length $N \in \mathbb{N}$ of System (1) as $\omega_{\mathbf{x}}^N = x_0 \xrightarrow{\mathbf{u}_0} x_1 \xrightarrow{\mathbf{u}_1} \cdots \xrightarrow{\mathbf{u}_{N-1}} x_N$, where each $x_k \in \mathbb{R}^n$ is a sample from System (1). We denote the i-th element of $\omega_{\mathbf{x}}^N$ by $\omega_{\mathbf{x}}^N(i)$ and the set of all finite trajectories by X^*. A *control strategy* $\pi : X^* \to U$ is a function that chooses the next action $u \in U$ given a finite trajectory. Under π and initial condition $x_0 \in \mathbb{R}^n$, System (1) defines a unique probability measure P^{x_0} over X^* [22].

We impose a standard smoothness (well-behaved) assumption on f. Namely, we assume f is a sample from a Gaussian process (GP)[2] (see Section III for details). Since f is unknown, we aim to reason about System (1) solely from a set of input-output data. Specifically, we assume $D = \{(x_i, u_i, x_i^+)\}_{i=0}^m$ is a set of identically and independently distributed (i.i.d.) data, where x_i^+ is a sample of a one time-step evolution of System (1) from $x_i \in \mathbb{R}^n$ under action $u_i \in U$.

We are interested in the temporal properties of $\mathbf{x}$ in a compact set $X \subset \mathbb{R}^n$ w.r.t. a set of regions $R = \{r_1, \ldots, r_l\}$, where $r_i \subseteq X$. To this end, we define a set of atomic proposition $\Pi = \{\mathrm{p}_1, \ldots, \mathrm{p}_l\}$, where p_i is true iff $\mathbf{x} \in r_i$. Let $L : X \to 2^\Pi$ be a labeling function that assigns to each state the set of atomic propositions that are true at that state. Then, the observation trace of trajectory $\omega_{\mathbf{x}}^N$ is $\rho = \rho_0 \rho_1 \ldots \rho_N$, where $\rho_i = L(\omega_{\mathbf{x}}^N(i))$ for all $0 \le i \le N$. To express the temporal properties of System (1), we use LTLf [12].

Definition 1 (LTLf): Given a set of atomic propositions Π, an LTLf formula is defined recursively as

$$\varphi = \mathrm{p} \mid \neg\varphi \mid \varphi \wedge \varphi \mid \bigcirc\varphi \mid \varphi \mathcal{U} \varphi \mid \mathcal{F}\varphi \mid \mathcal{G}\varphi$$

where $\mathrm{p} \in \Pi$, and $\bigcirc$, $\mathcal{U}$, $\mathcal{F}$, and $\mathcal{G}$ are the 'next', 'until', 'eventually', and 'globally' temporal operators, respectively.

The semantics of LTLf are defined over finite traces [12]. We say trajectory $\omega_{\mathbf{x}} \in X^*$ satisfies formula φ, denoted by $\omega_{\mathbf{x}} \models \varphi$, if a prefix of its observation trace satisfies φ.

Problem 1 (Control Synthesis): Given a dataset $D = \{(x_i, u_i, x_i^+)\}_{i=1}^m$ of i.i.d. samples of System (1), compact set X, and LTLf formula φ, find control strategy π^* that maximizes the probability of satisfying φ without existing X, i.e., for every $x_0 \in X$,

$$\pi^* = \arg\max_{\pi} P^{x_0}(\omega_{\mathbf{x}} \models \varphi \mid D, \pi) \tag{2}$$

There are three main challenges in Problem 1: (i) the dynamics of System (1) are unknown, can be nonlinear, and its evolution is stochastic, (ii) guarantees are required for the underlying system to satisfy complex specifications, and (iii) scalability to higher dimensions is necessary, which is an additional challenge that we impose. In our approach, we show that challenge (i) can be successfully addressed by utilizing the power of DKL to approximate f. For challenges (ii) and (iii), we draw inspirations from formal methods literature and construct a discrete abstraction of the dynamics as an IMDP. With an LTLf specification, we can then use off-the-shelf tools for synthesizing provably correct strategies.

III. Modelling Dynamical Systems Using Deep Kernel Learning

To describe how we learn f in System (1), we first need to introduce GPs. Then, we present DKL in the GP framework.

Gaussian Process Models: A GP is a collection of random variables, such that any finite collection of those random variables are jointly Gaussian [15]. Because of the favorable analytical properties of Gaussian distributions, GPs are widely employed to learn unknown functions, such as f in System (1), from observations of the system [7], [23]. In particular, given a prior GP, $GP(\mu, k_\gamma)$, where $\mu : \mathbb{R}^n \to \mathbb{R}$ is the mean function and $k_\gamma : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is a positive semi-definite covariance function (or kernel) with hyper-parameters γ, the assumption is that for each $a \in U$ and for each $j \in \{1, \ldots, n\}$, $f^{(j)}(\cdot, a)$, the j-th component of $f(\cdot, a)$, is a sample from $GP(\mu, k_\gamma)$. Then, given dataset $D = \{(x_i, u_i, x_i^+)\}_{i=1}^m$ of samples of System (1), which we partition in $|U|$ subsets $D_a = \{(x, u, x^+) \in D \mid u = a\}$, we obtain that, at every point $x^* \in \mathbb{R}^n$, the posterior predictive distribution of $f^{(j)}(x^*, a)$ given D is still Gaussian with mean and variance:

$$\mathbb{E}\Big(f^{(j)}(x^*, a) \mid D\Big) = \mu\big(x^*\big) + \beta(Y - \mu(\mathcal{X})), \tag{3}$$

$$cov\Big(f^{(j)}(x^*, a) \mid D\Big) = K_{x^*,x^*} - \beta K_{\mathcal{X},x^*}, \tag{4}$$

where $\mathcal{X} = (x_1, \ldots, x_{|D_a|})$, $Y = (x_1^{(j)+}, \ldots, x_{|D_a|}^{(j)+})$, $\beta = K_{x^*,\mathcal{X}}(K_{\mathcal{X},\mathcal{X}} + \sigma^2 I)^{-1}$, and $K_{\mathcal{X},\mathcal{X}} \in \mathbb{R}^{|D_a| \times |D_a|}$ is a matrix whose i-th row and l-th column is $k_\gamma(x_i, x_l)$.

A widely-used kernel function is the squared exponential $k_{\gamma_{se}}(x, x') = \sigma_s \exp(-\|x - x'\| / 2l^2)$, with the set of hyper-parameters $\gamma_{se} = \{\sigma_s, l\}$, where σ_s and l are the output scale and length scale, respectively. These hyper-parameters are generally learned by minimizing the negative marginal log-likelihood of the data [15].

[1] There always exists a linear transformation, namely the Mahalanobis transformation, that enables diagonalization of the covariance matrix.

[2] Note that the restrictions that this assumption poses on f depends on the choice of the covariance (kernel) function for the GP, and there exist universal kernels, such as the squared exponential, that allow for a GP to approximate *any* continuous f arbitrarily well.

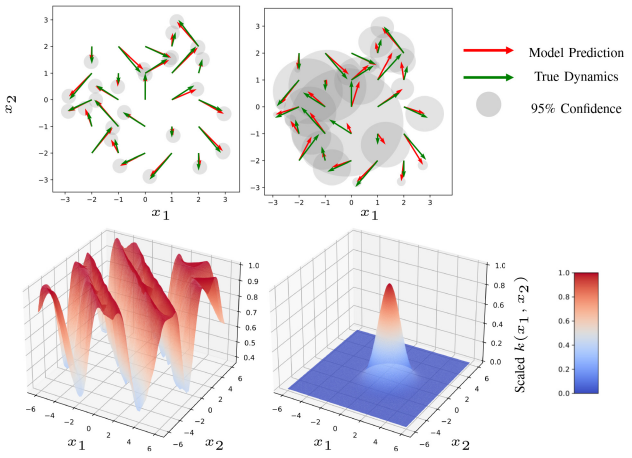

Fig. 1. Results on learning a 2D vector field with (left) DKL and (right) GP. Top: Vector Field. Bottom: scaled correlation function for the first dimension at one point. 1000 samples were used to pre-train the NN. Both methods use 100 samples for predictions.

Deep Kernel Learning: The squared exponential kernel, like most commonly-employed kernels for GP regression [15], only depends on few hyper-parameters. This limits the flexibility of GPs in learning complex representations of data [24], often resulting in predictions with large uncertainty (variance). One can reduce this uncertainty with more data, but that leads to computational intractability since the time complexity of GP regression is $\mathcal{O}(|D|^3)$ [15]. DKL aims to address this issue by considering a kernel that is composed with a NN. The underlying idea is that a fully connected NN $g_a^w : \mathbb{R}^n \to \mathbb{R}^s$, parameterized by weights and biases vector w over action a, is employed to map the input into an s-dimensional feature space, where GP regression is performed. Specifically, starting with a base kernel k_γ (which we always assume $k_\gamma = k_{\gamma_{se}}$), we define a deep kernel as

$$k_{dkl}(x, x') = k_\gamma\left(g_a^w(x), g_a^w(x')\right). \tag{5}$$

Then, with k_{dkl}, predictions still use GP's mean and covariance equations in (3)–(4), but the number of hyper-parameters (i.e., γ and w) are drastically increased. This significantly improves the flexibility and representational power of GPs.

The learning of the parameters in γ and w can be achieved by either minimizing the negative marginal log-likelihood or considering a fully Bayesian approach [20]. Furthermore, the NN portion of DKL models can be pre-trained and its parameters fixed. This minimizes the number of parameters being optimized through the marginal log-likelihood and mitigates the possibility of DKL over-fitting the data [20].

The combination of flexibility (NNs) and principled uncertainty quantification (GPs) in DKL is particularly important for problems that require learning complex dynamics with robustness analysis. The former often requires large amounts of data and the latter reasoning about uncertainty. The power of DKL is illustrated in Figure 1, where we consider learning a 2D vector field $f(\mathbf{x}) = (\sin(x_1 + x_2), \cos(x_1 - x_2))^T$ with noise distribution $\mathcal{N}(0, 0.01)$ by using both a standard GP and DKL. We observe that the k_{dkl} learns the oscillatory behavior of the data, while the GP with $k_{\gamma_{se}}$ only correlates nearby points. As a consequence, with the same data, the predictions of DKL are more accurate and less uncertain compared to the ones of the GP. Particularly, in our synthesis framework, which relies on abstraction-based techniques, the lower uncertainty associated with DKL can lead to less conservative abstractions and probabilistic guarantees.

IV. IMDP Abstraction

DKL allows one to predict the one-step evolution of System (1) from a given $x \in X$ and $u \in \mathcal{U}$. To analyze LTLf properties of System (1), however, we need to reason over finite trajectories (with arbitrary lengths) of System (1) and consequently perform multi-step predictions of arbitrary length. Unfortunately, such analysis is intractable for standard GPs even for a fixed finite horizon [25]. To address this problem we rely on finite abstractions, which in turn allows one to use existing LTLf control synthesis tools [11], [26]. Specifically, we use an IMDP [13] as the abstraction model.

Definition 2 (IMDP): An interval Markov Decision Process (IMDP) is a tuple $\mathcal{I} = (Q, A, \hat{P}, \check{P}, \Pi, L)$, where Q is a finite set of states, A is a finite set of actions, $\check{P}, \hat{P} : Q \times A \times Q \to [0, 1]$ are functions that define the lower and upper bounds, respectively, of the transition probability from state $q \in Q$ to state $q' \in Q$ under action $a \in A$, Π is a set of atomic propositions, and $L : Q \to 2^\Pi$ is a labeling function that assigns to each state $q \in Q$ a subset of Π.

It holds for all $q, q' \in Q$ and $a \in A(q)$ that $\check{P}(q, a, q') \leq \hat{P}(q, a, q')$ and $\sum_{q' \in Q} \check{P}(q, a, q') \leq 1 \leq \sum_{q' \in Q} \hat{P}(q, a, q')$. A *finite path* of $\mathcal{I}$, denoted by $\omega_\mathcal{I} \in Q^*$, is a finite sequence of states in Q. A *strategy* of $\mathcal{I}$ is a function $\pi_\mathcal{I} : Q^* \to A$ that maps $\omega_\mathcal{I}$ to an action in A. We describe how to efficiently build $\mathcal{I}$ for System (1) from its DKL model below.

States and Actions: First, we partition X into a set of convex regions $\bar{Q} = \{q_1, \ldots, q_{|\bar{Q}|}\}$, e.g., by using a grid. We consider an additional region $q_u = \mathbb{R}^n \setminus X$ and call $Q = \bar{Q} \cup \{q_u\}$ the set of IMDP states. We assume that the discretization Q of X respects the regions of interest in R, i.e., $\forall r \in R$, $\exists Q_r \subseteq \bar{Q}$ such that $\cup_{q \in Q_r} q = r$. With an abuse of notation, we use q to denote both a state in the IMDP and its corresponding region, i.e., $q \in Q$ and $q \subset \mathbb{R}^n$. Note that for every $x, x' \in q$, $L(x) = L(x')$; accordingly, we set the IMDP labeling function as $L(q) = L(x)$. The set of IMDP actions A is given by the set of actions U and all actions are allowed to be available at each state $q \in Q$.

Probability Bounds: The key step to building an IMDP abstraction of System (1) is the computation of the transition probability functions $\hat{P}$ and $\check{P}$. Given $q \subset \mathbb{R}^n$, $a \in U$, and $x \in X$, we define the *transition kernel* $T_a(q \mid x)$ as:

$$T_a(q \mid x) = \int_q \mathcal{N}(v \mid \mathbb{E}(f(x, a) \mid D), cov(f(x, a) \mid D) + \mathcal{V}) dv \tag{6}$$

That is, $T_a(q \mid x)$ is the probability that, given the data D, our Gaussian prior assumption on f, and an initial state x, System (1) transitions to q under a in one time step. Note that $T_a(q \mid x)$ is defined by marginalizing the DKL predictive distribution for f over the dynamics of System (1) and the resulting kernel is still Gaussian due to the closure of Gaussian random variables under linear combinations [15]. As we show in Theorem 2 in Section V, this marginalization guarantees that our abstraction accounts for the uncertainty coming from the DKL predictions.

Consequently, for $q, q' \in \bar{Q}$, it follows that

$$\check{P}(q,a,q') = \min_{x \in q} T_a(q' \mid x), \tag{7}$$

$$\hat{P}(q,a,q') = \max_{x \in q} T_a(q' \mid x), \tag{8}$$

and for the unsafe region q_u, it holds that $\check{P}(q,a,q_u) = 1 - \max_{x\in q} T_a(X \mid x)$ and $\hat{P}(q,a,q_u) = 1 - \min_{x\in q} T_a(X \mid x)$. Lastly, since reaching q_u violates the requirement of not leaving X, we set q_u to be a sink state, i.e., $\forall a \in A$, $\check{P}(q_u,a,q_u) = \hat{P}(q_u,a,q_u) = 1$.

In the remainder of this section, we show how to efficiently compute the bounds in (7)-(8). We note that existing results for GPs [27, Propositions 4 and 7] allow for output bounding of standard kernel functions (e.g., the squared exponential) given a compact input set. We abstract the NNs in DKL via local linear relaxations, which can be built in constant time by utilizing algorithms in [21], to provide a compact input set to the base kernel for bounding. That is, for NN g_a^w and region $q \subset \mathbb{R}^n$, [21] finds matrices $\check{A}_q, \hat{A}_q \in \mathbb{R}^{s\times n}$ and $\check{b}_q, \hat{b}_q \in \mathbb{R}^s$ such that $\forall x \in q$, $\check{A}_q x + \check{b}_q \le g_a^w(x) \le \hat{A}_q x + \hat{b}_q$.

We use such relaxations to propagate q through the NN prior and produce compact set $Z_{q,a}$ that contains the output of $g_a^w(x)$ for every $x \in q$. Then, $Z_{q,a}$ is propagated through the base kernel using results from [27] (i.e., four convex optimization programs), obtaining the ranges of posterior mean and variance for all $x \in q$. Then, we obtain mean bounds $\underline{M}_{q,a}, \overline{M}_{q,a} \in \mathbb{R}^n$ and variance bounds $\underline{\Sigma}_{q,a}, \overline{\Sigma}_{q,a} \in \mathbb{R}^n_{\ge 0}$ such that, for every $x \in q$ and every $j \in \{1,\dots,n\}$,

$$\mathbb{E}\Big(f^{(j)}(x,a) \mid D_a\Big) \in \Big[\underline{M}^{(j)}_{q,a}, \overline{M}^{(j)}_{q,a}\Big], \tag{9}$$

$$cov\Big(f^{(j)}(x,a) \mid D_a\Big) \in \Big[\underline{\Sigma}^{(j)}_{q,a}, \overline{\Sigma}^{(j)}_{q,a}\Big]. \tag{10}$$

Theorem 1 (Efficient Computation for Tran. Prob. Bounds): For $\mu \in \mathbb{R}$ and $\sigma \in \mathbb{R}_{\ge 0}$ and closed interval $\theta = [\underline{\theta}, \overline{\theta}] \subset \mathbb{R}$, define function $h(\theta,\mu,\sigma) = \frac{1}{2}\Big(\mathrm{erf}\Big(\frac{\overline{\theta}-\mu}{\sqrt{2}\sigma}\Big) - \mathrm{erf}\Big(\frac{\underline{\theta}-\mu}{\sqrt{2}\sigma}\Big)\Big)$. Further, given region $q \in \bar{Q}$, let $[\underline{M}_{q,a}, \overline{M}_{q,a}]$ and $[\underline{\Sigma}_{q,a}, \overline{\Sigma}_{q,a}]$ be the poster mean and variance of DKL as reported in (9)–(10). Additionally, for region $q' \in \bar{Q}$, denote its centroid by $c_{q'}$ and define points

$$\underline{z} = \operatorname*{arg\,min}_{z \in \left[\underline{M}_{q,a}, \overline{M}_{q,a}\right]} \|z - c_{q'}\|, \quad \bar{z} = \operatorname*{arg\,max}_{z \in \left[\underline{M}_{q,a}, \overline{M}_{q,a}\right]} \|z - c_{q'}\|.$$

Then, denoting the closed interval obtained by projecting $q' \subset \mathbb{R}^n$ onto the j-th dimension by $q'^{(j)} \subset \mathbb{R}$, it holds that

$$\min_{x\in q} T_a(q' \mid x) \ge \prod_{j=1}^{n} \min_{\tau \in \{\underline{\Sigma}^{(j)}_{q,a}, \overline{\Sigma}^{(j)}_{q,a}\}} h\Big(q'^{(j)}, \bar{z}^{(j)}, \tau + \mathcal{V}^{(j,j)}\Big),$$

$$\max_{x\in q} T_a(q' \mid x) \le \prod_{j=1}^{n} \max_{\tau \in \{\underline{\Sigma}^{(j)}_{q,a}, \overline{\Sigma}^{(j)}_{q,a}\}} h\Big(q'^{(j)}, \underline{z}^{(j)}, \tau + \mathcal{V}^{(j,j)}\Big),$$

where $\mathcal{V}^{(j,j)}$ is the j,j element of the noise covariance $\mathcal{V}$.

Proof: In the proof, we consider the min case; the max case follows similarly. Note that $h(\theta,\mu,\sigma)$ is the integral of $\mathcal{N}(\mu,\sigma)$ over θ. Then, under the assumption of diagonal $cov(f^{(j)}(x,a) \mid D_a)$ and $\mathcal{V}$, it holds that for every $x \in q$, $T_a(q' \mid x) = \prod_{j=1}^n h(q'^{(j)}, \mathbb{E}(f^{(j)}(x,a) \mid D), cov(f^{(j)}(x,a) \mid D) + \mathcal{V}^{(j,j)}) \ge \prod_{j=1}^n \min_{\mu \in [\underline{M}^{(j)}_{q,a}, \overline{M}^{(j)}_{q,a}],\, \tau \in [\underline{\Sigma}^{(j)}_{q,a}, \overline{\Sigma}^{(j)}_{q,a}]} h(q'^{(j)}, \mu, \tau + \mathcal{V}^{(j,j)})$. Consequently, what is left to show is how to place mean μ and variance τ of a uni-dimensional Gaussian to minimize its integral over the respective dimension of q'. Each of these is minimized by first maximizing the distance of $z^{(j)}$ from $c^{(j)}_{q'}$, hence z can be chosen according to $\arg\max_{z \in [\underline{M}_{q,a}, \overline{M}_{q,a}]} \|z - c_{q'}\|$. Then, there are two cases: $z^{(j)} \in q'^{(j)}$ and $z^{(j)} \notin q'^{(j)}$. In the first case, T is minimized if we minimize the probability mass in q', which results in $\tau = \overline{\Sigma}^{(j)}_{q,a}$. In the second case, with a similar reasoning we obtain $\tau = \underline{\Sigma}^{(j)}_{q,a}$ or $\tau = \overline{\Sigma}^{(j)}_{q,a}$. ■

Theorem 1 shows that we can compute transition bounds for (7)–(8) by simply evaluating an error function at $4n$ points, thus guaranteeing efficient abstraction construction.

V. Control Synthesis and Refinement

Once we obtain IMDP abstraction $\mathcal{I}$, our goal is to synthesize a strategy $\pi_{\mathcal{I}}$ that maximizes the probability of satisfying specification φ on $\mathcal{I}$ and then map it back to System (1) to obtain control strategy π.

Let $\mathcal{D}(Q)$ be the set of all probability distributions over Q. We define an *adversary* $\nu_{\mathcal{I}} : Q^* \times A \to \mathcal{D}(Q)$ to be a function that maps a finite path $\omega_{\mathcal{I}} \in Q^*$ and an action $a \in A$ to a transition probability distribution such that, $\forall q' \in Q$, $\check{P}(\text{last}(\omega_{\mathcal{I}}), a, q') \le \nu_{\mathcal{I}}(\omega_{\mathcal{I}}, a)(q') \le \hat{P}(\text{last}(\omega_{\mathcal{I}}), a, q')$, where $\text{last}(\omega_{\mathcal{I}})$ is the last state in $\omega_{\mathcal{I}}$. Given $\pi_{\mathcal{I}}$ and $\nu_{\mathcal{I}}$, a probability measure Pr over paths in Q^* is induced [11]. Our objective can be translated as finding an optimal $\pi^*_{\mathcal{I}}$ that is robust to all uncertainties induced by abstraction, i.e., $\pi^*_{\mathcal{I}} = \arg\max_{\pi_{\mathcal{I}}} \min_{\nu_{\mathcal{I}}} Pr(\omega_{\mathcal{I}} \models \varphi \mid \pi_{\mathcal{I}}, \nu_{\mathcal{I}}, \omega_{\mathcal{I}}(0) = q)$. Then, $\pi^*_{\mathcal{I}}$ can be computed using off-the-shelf tools with a time complexity polynomial in Q [11].

We can then define π according to $\pi^*_{\mathcal{I}}$ by using a mapping between trajectories of System (1) and paths of $\mathcal{I}$. Let $\mathbb{M} : X \to \bar{Q}$ be a mapping such that $\mathbb{M}(x) = q$ for all $x \in q$. With an abuse of notation, for a finite trajectory $\omega_{\mathbf{x}} \in X^*$ with length N, we define $\mathbb{M}(\omega_{\mathbf{x}}) = \mathbb{M}(\omega_{\mathbf{x}}(0)) \dots \mathbb{M}(\omega_{\mathbf{x}}(N)) \in Q^*$. Then, the control strategy of System (1) is given by: $\pi(\omega_{\mathbf{x}}) = \pi^*_{\mathcal{I}}(\mathbb{M}(\omega_{\mathbf{x}}))$. Furthermore, for $\pi^*_{\mathcal{I}}$, we also obtain lower bound probabilities of satisfaction of φ from every $q \in \bar{Q}$ as $\check{p}(q) = \min_{\nu_{\mathcal{I}}} Pr(\omega_{\mathcal{I}} \models \varphi \mid \pi^*_{\mathcal{I}}, \nu_{\mathcal{I}}, \omega_{\mathcal{I}}(0) = q)$, the upper bound $\hat{p}(q)$ is then found on $\max_{\nu_{\mathcal{I}}}$. The following theorem shows that these bounds also hold for System (1).

Theorem 2 (Correctness): For $q \in Q$, let $\check{p}(q)$ and $\hat{p}(q)$ the lower- and upper-bound probabilities of satisfying φ from q. Then, it holds that

$$P^{x_0}(\omega_{\mathbf{x}} \models \varphi | D, \pi, x_0 \in q) \in \big[\check{p}(q), \hat{p}(q)\big].$$

Proof: $\mathbf{x}^{(j)}(k+1) = f^{(j)}(x,a) + \mathbf{v}^{(j)}(k)$ is a Gaussian process with zero mean and covariance $k_{dkl}(x,x) + \mathcal{V}^{(j,j)}$. Consequently, for $x_1, \dots, x_l \in D_a$ the joint distribution of $f(x_1,a) + \mathbf{v}(k), \dots, f(x_l,a) + \mathbf{v}(k)$ is still Gaussian. Then the transition kernel $T_a(q \mid x)$ in (6) defines the one step dynamics of System (1). Then, for any strategy π, the upper and lower bound probabilities returned by the IMDP from initial region q as built in Section IV contains $P^{x_0}(\omega_{\mathbf{x}} \models \varphi \mid D, \pi, x_0 \in q)$ as follows from [8, Th. 2]. ■

Refinement: The uncertainty induced by the discretization of X may result in undesirable results where much of the space has a large gap between $\check{p}$ and $\hat{p}$. We consider a refinement strategy similar to that in [17], [28] to efficiently reduce this

TABLE I
CONSIDERED SYSTEMS WITH DIMENSION DIM., ACTION SET U, DATASET PER ACTION D_a, DATASET USED FOR POSTERIOR PREDICTIONS $D_a^{\text{PRED}} \subset D_a$, NUMBER OF LAYERS #L AND #N/L NEURONS PER LAYER OF THE NNS. DATASETS WERE GENERATED BY UNIFORMLY SAMPLING STATES IN X

System Type	Dim.	$\lvert U\rvert$	$\lvert D_a\rvert$	$\lvert D_a^{\text{pred}}\rvert$	#L	#N/L
Non-linear [3], [17]	2D	4	1,000	100	2	64
Dubin's Car [17]	3D	7	10,000	400	2	128
2nd-order Car [17]	5D	3	50,000	250	3	64

conservatism. In particular, to decide on which states to refine, we define a scoring function $\eta : \bar{Q} \to \mathbb{R}_{\geq 0}$ as $\eta(q) = (\hat{p}(q) - \check{p}(q)) \sum_{a \in U} \sum_{q' \in q} (\hat{P}(q,a,q') - \check{P}(q,a,q'))$. η gives higher score to states that have the most uncertainty associated with satisfying φ and states with conservative outgoing transition probabilities. We refine the n_{ref} states with the highest score and for each state we only split in half the dimension that minimizes the volume of $Z_{q,a}$ (i.e., conservatism induced by the NN linear relaxation).

VI. CASE STUDIES

We evaluate our DKL control synthesis framework on various nonlinear systems and cases studies. First, we assess the learning performance of DKL under different NN architectures against other GP-based methods. Then, we show the efficacy of our control synthesis framework in various environments and specifications. Experiments were run on an Intel Core i7-12700K CPU at 3.60GHz with 32 GB of RAM limited to 10 threads. Our tool is available on GitHub [29].

Setup and Training: We consider three nonlinear systems from [3], [17] as shown in Table I. To learn their dynamics, we use four learning models: GP: the squared exponential kernel, NN-GP: joint NN and GP model where the NN is trained as a predictor of the dynamics on D_a and a GP is regressed to predict the error of the NN from truth, DKL$^{\text{F}}$: DKL with a NN that is trained on D_a and the full output of the NN is provided as an input to the base kernel, DKL$^{\text{S}}$: similar to DKL$^{\text{F}}$ but only the corresponding output dimension of the NN is provided as an input to the base kernel.

All NNs use the ReLU activation function ($\text{ReLU}(t) = \max(0,t)$) and all base kernels use D_a^{pred} for predictions. We train the NN priors via stochastic mini-batching as a scaled predictor of the dynamics (i.e., $s = n$) and fix the parameters before learning the kernel parameters via maximum log likelihood. Details on the models are in Table I. The number of parameters in the NNs are kept small to produce tighter linear relaxations.

Accuracy of Deep Kernel Learning: We first demonstrate the advantages of DKL by comparing the predictive accuracy of the learning models. We define the predictive mean error and uncertainty of each model at a point x under action a as $\text{err}_\mu(x,a) = \|\mathbb{E}(f(x,a) \mid D) - f(x,a)\|_2$ and $\text{err}_\sigma(x,a) = \text{trace}(cov(f(x,a) \mid D))^{\frac{1}{2}}$, respectively. Table II shows the maximum values over 100,000 test points for a fixed action.

In all cases, GP has the worse performance in mean error (err_μ) and compensates with large uncertainty (err_σ). For low dimensional systems, NN-GP performs well in mean error but retains a large uncertainty due to poor correlation between data points in the GP. In higher dimensions DKL$^{\text{S}}$ has the best performance. This is mainly due to the NN used in the prior for the kernel, which improves both mean and variance accuracy, unlike the NN-GP. We note that an ill-formed prior may result in uncertainty being underestimated hence, care must be taken when training the NN prior and stochastic mini-batching is shown to be effective [20].

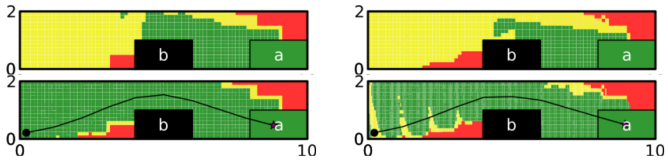

Fig. 2. 3D system. left: DKL$^{\text{S}}$, right: GP, top: initial abstraction, bottom: 2 refinements. Green: Q^{yes}, yellow: $Q^{?}$, red: Q^{no}.

TABLE II
MAXIMUM PREDICTIVE MEAN (ERR_μ) AND VARIANCE ERRORS (ERR_σ) OVER 100,000 TEST POINTS FOR A FIXED ACTION OF EACH SYSTEM

Model	2D System		3D System		5D System	
	err_μ	err_σ	err_μ	err_σ	err_μ	err_σ
GP	0.6777	0.3473	1.8517	0.5261	0.7479	0.7825
NN-GP	**0.0591**	0.3215	0.1158	0.5209	0.2856	0.7797
DKL$^{\text{F}}$	0.2575	0.1817	0.2141	0.1643	0.1909	0.2626
DKL$^{\text{S}}$	0.1716	**0.1276**	**0.0856**	**0.1545**	**0.1294**	**0.1778**

TABLE III
3D SYSTEM RESULTS. PERCENT VOLUME OF SPACE FOR $Q^{yes/no}$ AND TIME TAKEN FOR NN RELAXATION (NN R.), KERNEL BOUNDING (KERNEL B.), TRANSITION PROBABILITY BOUNDING (PROB.), AND SYNTHESIS (SYN.) FOR 0 & 2 REFINEMENTS (#R). ALL TRAN. PROBS. WERE RECALCULATED FOR #R=2

Model	# R.	$\lvert Q\rvert$	Volume (%)		Time (min.)				
			Q^{yes}	Q^{no}	NN R.	Kernel B.	Prob.	Syn.	Total
GP	0	20,482	17.67	40.86	–	4,866.46	5.87	3.53	4,875.86
	2	40,482	36.67	43.40	–	7.00	17.00	2.24	26.24
DKL$^{\text{F}}$	0	20,482	18.60	36.27	418.8	63.80	4.98	1.76	489.34
	2	40,490	38.06	39.04	–	1.42	16.72	1.92	20.06
DKL$^{\text{S}}$	0	20,482	20.05	36.46	418.8	8.80	4.73	1.31	433.64
	2	38,885	**42.65**	**44.08**	–	**0.11**	**15.37**	**1.38**	**16.86**

Synthesis Results: Here, we illustrate the efficacy of our control synthesis framework. Our metrics are the *computation time* and *percent volume* of the space from which the system under the synthesized control strategy is guaranteed to satisfy the specification with probability $\check{p} \geq 0.95$ called Q^{yes}, $\hat{p} < 0.95$ called Q^{no}, and the remaining states called $Q^{?}$. All results are validated using 1000 MC simulations.

1) Refinement and Computation Time: We consider the 3D Dubin's car system, with the state space representing position and orientation, and synthesize a strategy for a static overtaking scenario as shown in Figure 2. We label the stationary car as b and the goal region as a. The LTLf specification $\varphi_1 = \mathcal{G}(\neg b) \wedge \mathcal{F}(a)$ then defines the task. The control synthesis results can be seen in Table III and a visualization is shown in Figure 2. Simulated trajectories under the strategy are shown in black lines, starting from the black dot and ending at the purple star. Synthesis for the NN-GP model timed out after 5000 min.

We see that DKL$^{\text{S}}$ has the best performance, leaving only 13.27% of the state space volume as undetermined ($Q^{?}$). This is expected as the DKL$^{\text{S}}$ model has the highest accuracy as shown in Table II and follows the prediction that the lower uncertainty would result in a less conservative abstraction. DKL$^{\text{F}}$ provides guarantees on a lower volume of space than GP but achieves similar results in one tenth the time with a larger volume of Q^{yes}. The computational bottleneck for GP

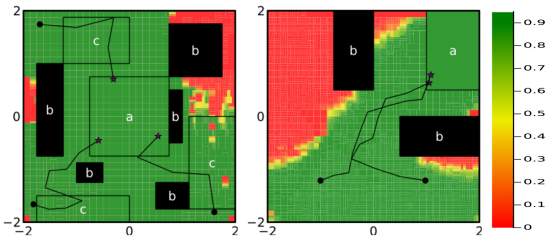

Fig. 3. Lower bound satisfaction probabilities $\check{p}(q)$ for Experiments 2 (left: 2D system) and 3 (right: 5D system).

abstractions comes from bounding the kernel outputs, and the DKL models vastly outperform the GP here due to the NN producing a more informative input to the base kernel.

2) Control Synthesis With Complex Specifications: We perform control synthesis on the 2D system given the same labeling considered in [3], [17] and complex LTLf specification $\varphi_2 = \mathcal{G}(\neg b) \wedge \mathcal{F}(a) \wedge \mathcal{F}(c)$. In this low dimensionality, there is little difference between the synthesis results for the 4 learning models. We show results for the $\texttt{DKL}^{\texttt{F}}$. After two refinements, which take 12.57 minutes, a control strategy where 73.48% of the space is in Q^{yes} and 0.07% in $Q^{?}$ is synthesized; results are shown in Figure 3.

3) Scalability to Higher Dimensions: We synthesize a control strategy for the 5D system on the environment described in [17] and specification $\varphi_1 = \mathcal{G}(\neg b) \wedge \mathcal{F}(a)$. Here we only show results for the $\texttt{DKL}^{\texttt{S}}$, as the GP model cannot scale. After two refinements, which take 344 minutes of which only 6 minutes are used to bound the kernel, we synthesize a control strategy where 44.36% is Q^{yes} and 39.46% is $Q^{?}$ producing comparable results to [17], which assumes given dynamics, but ours is from data. The final abstraction has roughly one tenth the states a uniform discretization produces; results are shown in Figure 3.

VII. Conclusion

We introduced an abstraction framework for unknown, stochastic dynamics via DKL which can be used to synthesize strategies with guarantees on the behavior of the system. This letter shows that the NN prior enables more accurate predictions, easier computation of posterior bounds, and faster synthesis times than standard kernels; enabling scalable data-driven synthesis. DKL models can effectively use large data sets by optimizing the NN prior over all the data and using a subset for posterior predictions. This is promising for systems with millions of data points available, as this allows for a computationally tractable form of uncertainty quantification. Our method relies on discrete actions, but recent works have provided methods for IMDP synthesis over continuous action spaces [30]. We plan to expand our method to this domain in future work.

References

[1] S. Dutta, S. Jha, S. Sankaranarayanan, and A. Tiwari, "Learning and verification of feedback control systems using feedforward neural networks," *IFAC PapersOnLine*, vol. 51, no. 16, pp. 151–156, 2018.

[2] S. Haesaert, P. M. J. Van den Hof, and A. Abate, "Data-driven and model-based verification via bayesian identification and reachability analysis," *Automatica*, vol. 79, pp. 115–126, May 2017.

[3] J. Jackson, L. Laurenti, E. Frew, and M. Lahijanian, "Strategy synthesis for partially-known switched stochastic systems," in *Proc. 24th Int. Conf. HSCC*, 2021, pp. 1–11.

[4] A. Nejati and M. Zamani, "Data-driven synthesis of safety controllers via multiple control barrier certificates," *IEEE Control Syst. Lett.*, vol. 7, pp. 2497–2502, 2023.

[5] S. Haesaert, A. Abate, and P. M. J. Van den Hof, "Data-driven and model-based verification: A bayesian identification approach," in *Proc. 54th IEEE CDC*, 2015, pp. 6830–6835.

[6] C. Knuth, G. Chou, N. Ozay, and D. Berenson, "Planning with learned dynamics: Probabilistic guarantees on safety and reachability via lipschitz constants," *IEEE Robot. Autom. Lett.*, vol. 7, no. 1, p. 381, Jan. 2022.

[7] A. Lederer, J. Umlauft, and S. Hirche, "Uniform error bounds for Gaussian process regression with application to safe control," in *Proc. 33rd NeurIPS*, 2019, pp. 1–11.

[8] J. Jackson, L. Laurenti, E. Frew, and M. Lahijanian, "Formal verification of unknown dynamical systems via Gaussian process regression," 2021, *arXiv:2201.00655*.

[9] R. Wajid, A. U. Awan, and M. Zamani, "Formal synthesis of safety controllers for unknown stochastic control systems using Gaussian process learning," in *Proc. Learn. Dyn. Control Conf.*, 2022, pp. 624–636.

[10] P. Tabuada, *Verification and Control of Hybrid Systems: A Symbolic Approach*. New York, NY, USA: Springer, 2009.

[11] M. Lahijanian, S. B. Andersson, and C. Belta, "Formal verification and synthesis for discrete-time stochastic systems," *IEEE Trans. Autom. Control*, vol. 60, no. 8, pp. 2031–2045, Aug. 2015.

[12] G. De Giacomo and M. Y. Vardi, "Linear temporal logic and linear dynamic logic on finite traces," in *Proc. 23rd IJCAI'13*, 2013, pp. 854–860.

[13] R. Givan, S. Leach, and T. Dean, "Bounded-parameter Markov decision processes," *Artif. Intell.*, vol. 122, nos. 1–2, pp. 71–109, 2000.

[14] N. Cauchi, L. Laurenti, M. Lahijanian, A. Abate, M. Kwiatkowska, and L. Cardelli "Efficiency through uncertainty: Scalable formal synthesis for stochastic hybrid systems," in *Proc. 22nd ACM HSCC*, 2019, pp. 240–251.

[15] C. E. Rasmussen, and C. K. Williams, *Gaussian Processes for Machine Learning*, vol. 1, Berlin, Germany, Springer, 2006.

[16] T. Wei and C. Liu, "Safe control with neural network dynamic models," in *Proc. Learn. Dyn. Control Conf.*, 2022, pp. 1–12.

[17] S. Adams, M. Lahijanian, and L. Laurenti, "Formal control synthesis for stochastic neural network dynamic models," *IEEE Control Syst. Lett.*, vol. 6, pp. 2858–2863, 2022.

[18] R. Mazouz, K. Muvvala, A. R. Babu, L. Laurenti, and M. Lahijanian, "Safety guarantees for neural network dynamic systems via stochastic barrier functions," in *Proc. NeurIPS*, 2022, pp. 9672–9686.

[19] A. G. Wilson, Z. Hu, R. Salakhutdinov, and E. P. Xing, "Deep kernel learning," in *Proc. Artif. Intell. Statist.*, 2016, pp. 370–378.

[20] S. W. Ober, C. E. Rasmussen, and M. van der Wilk, "The promises and pitfalls of deep kernel learning," in *Proc. Uncertainty Artif. Intell.*, 2021, pp. 1206–1216.

[21] S. Wang et al., "Beta-crown: Efficient bound propagation with per-neuron split constraints for complete and incomplete neural network verification," in *Proc. NeurIPS*, 2021, pp. 29909–29921.

[22] D. Bertsekas, P. Bertsekas, and S. E. Shreve, *Stochastic Optimal Control: The Discrete-Time Case*, vol. 5, Nashua, NH, USA: Athena Sci., 1996.

[23] Y. Sui, A. Gotovos, J. Burdick, and A. Krause, "Safe exploration for optimization with Gaussian processes," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 997–1005.

[24] R. Calandra, J. Peters, C. E. Rasmussen, and M. P. Deisenroth, "Manifold Gaussian processes for regression," in *Proc. Int. Joint Conf. Neural Netw.*, 2016, pp. 3338–3345.

[25] M. Deisenroth and C. E. Rasmussen, "PILCO: A model-based and data-efficient approach to policy search," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 465–472.

[26] L. Laurenti, M. Lahijanian, A. Abate, L. Cardelli, and M. Kwiatkowska, "Formal and efficient synthesis for continuous-time linear stochastic hybrid processes," *IEEE Trans. Autom. Control*, vol. 66, no. 1, pp. 17–32, Jan. 2021.

[27] A. Patane, A. Blaas, L. Laurenti, L. Cardelli, S. Roberts, and M. Kwiatkowska, "Adversarial robustness guarantees for Gaussian processes," *J. Mach. Learn. Res.*, vol. 23, no. 1, pp. 6524–6578, 2022.

[28] M. Dutreix and S. Coogan, "Efficient verification for stochastic mixed monotone systems," in *Proc. Int. Conf. Cyber-Phys. Syst.*, 2018, pp. 150–161.

[29] R. Reed, "Formal deep kernel synthesis." 2023. [Online]. Available: https://github.com/aria-systems-group/Formal-Deep-Kernel-Synthesis

[30] G. Delimpaltadakis, M. Lahijanian, M. Mazo. Jr., and L. Laurenti, "Interval Markov decision processes with continuous action-spaces," in *Proc. 26th ACM Int. Conf. Hybrid Syst. Comput. Control* , 2023, pp. 1–10.