Sensitivities and where to find them

Domain shift robustness, attacks, and training variations in visual learning

Wang, Z.

DOI
10.4233/uuid:21e5b9c2-3e58-495c-881c-634e67ebe645

Publication date
2024

Document Version
Final published version

Citation (APA)
Wang, Z. (2024). *Sensitivities and where to find them: Domain shift robustness, attacks, and training variations in visual learning*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:21e5b9c2-3e58-495c-881c-634e67ebe645

SENSITIVITIES AND WHERE TO FIND THEM

DOMAIN SHIFT ROBUSTNESS, ATTACKS, AND TRAINING VARIATIONS IN VISUAL 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. T.H.J.J. van der Hagen,
chair of the Board for Doctorates
to be defended publicly on Friday, 22, March, 2024 at 10 o'clock.

by

Ziqi WANG

Master of Science in Mechanical Engineering,
Delft University of Technology, Delft, Netherlands,
born in Baoding, Hebei, China.

This dissertation has been approved by the promotors.
Composition of the doctoral committee:

Rector Magnificus,	Chairperson
Prof. dr. ir. M.J.T. Reinders,	Delft University of Technology, promotor
Dr. J.C. van Gemert,	Delft University of Technology, promotor
Prof. dr. M. Loog,	Radboud University, promotor
Independent members:	
Prof. dr. M.M. de Weerdt	Delft University of Technology
Dr. J.F.P. Kooij,	Delft University of Technology
Prof. dr. T.H.W. Bäck	Leiden University
Dr. J. Vanschoren,	Eindhoven University of Technology
Reserve member:	
Prof. dr. A.E. Zaidman,	Delft University of Technology

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

Man has continued to evolve by acts of disobedience. Not only was his spiritual development possible only because there were men who dared to say no to the powers that be in the name of their conscience or their faith, but also his intellectual development was dependent on the capacity for being disobedient, disobedient to authorities who tried to muzzle new thoughts and to the authority of long-established opinions which declared a change to be nonsense.

Erich Fromm, On Disobedience and Other Essays

CONTENTS

SUMMARY

Machine learning aims to solve a task with a certain algorithm or statistical model that is trained on data, with or without labels. As a subcategory of machine learning, deep learning achieves good performance with its flexibility on end-to-end representation learning and architecture design. Despite the successes of deep learning, the output of which can be sensitive to various factors. This work visits three sensitivity factors: distribution shifts, attacks, and human impact.

One factor that can impair the performance of a deep net is a distribution shift between the training data and the test data. Depending on the availability of either data or label, some coping strategies for distribution shifts are domain adaptation, domain generalization, transfer learning and multi-domain learning. We first show how domain adaptation can help to mitigate the gap between historic and modern photos for visual place recognition. We show that this can be realized by focusing the network on the buildings rather than the background with an attention module. In addition, we introduce a domain adaptation loss to align the source domain and the target domain. We then move to domain generalization and show that learning domain invariant representations cannot lead to good performance for domain generalization. We suggest to relax the constraint of learning domain invariant representation by learning representations that guarantee a domain invariant posterior, but the resulting representations are not necessarily domain invariant. We coin this type of representation as hypothesis invariant representation. Finally, we study multi-domain learning and transfer learning with the application of deep learning to classify Parkinson's disease. We show that a temporal attention mechanism is key for transferring useful information from large non-medical public video datasets to Parkinson videos. Weights are learned for various tasks involved in this Parkinson dataset to decide a final score for each single patient.

A deep net is also sensitive to malicious attacks, *e.g.*, adversarial classification attacks or explanation attacks. Adversarial classification attacks manipulate the classification result while explanation attacks change the explanation heatmap but do not alter the original classification results. We notice that the robustness to an adversarial classification attack is linked to the shape of the softmax function and can be improved by using a polynomial softRmax, which is based on a Cauchy class conditional distribution. This also shows that the performance of deep learning is sensitive to the choice of class conditional distribution. Regarding the explanation attacks, we design several ways to attack the GradCAM explanation heatmap to become a predetermined target explanation which does not explain the classification result.

We further explore the influence of human trainers in hyperparameter tuning during the learning of deep nets. A user study is designed to explore the correlation between the performance of a network and the human trainer's experience of deep learning. Experience of deep learning is found to be correlated with the performance of the deep net.

SAMENVATTING

Machine learning lost taken op met een model dat is getraind met data. Deep learning is een subcategorie van machine learning, dat goed werkt met grote hoeveelheden data. Ondanks dat Deep learning vrij successvol is en veel toepassingen heeft gevonden, kan deep learning gevoelig zijn voor verschillende factoren. Dit proefschrift onderzoekt drie van zulke factoren: distribution shifts, adverserial attacks en menselijke invloed.

Distribution shift betekend dat de distributie waar de trainingsdata vandaan komt anders is dan de distributie van de test data. Afhankelijk van de setting, zijn er meerdere manieren om hier mee om te gaan, zoals domain adaptation, domain generalization, transfer-learning of multi-domain learning.

We passen domain adaptation toe op foto's van historische plekken, om deze distributie te laten lijken op de distributie van moderne foto's. De toepassing is het herkennen van de locatie waar de foto is genomen. Dit kan in een diep neuraal netwerk worden gedaan met een attentie-module die het netwerk laat letten op gebouwen in plaats van de rest van de foto. We introduceren ook een loss-functie om de source en target op elkaar te laten lijken.

We analyseren domain generalization, en we laten zien dat domein-invariante representaties niet goed kunnen werken voor domain generalization. We stellen voor dat domein-invariante representaties te strikt zijn, en dat alleen de posterior domein invariant moet zijn. We noemen deze representatie een hypothese invariante representatie.

We bestuderen ook multi-domain learning en transfer learning met een toepassing voor het classificeren van Parkinson. Een temporale attention module is nodig om informatie vergaart uit een grote collectie niet-medische videos effectief te gebruiken voor medische Parkinson videos.

Een diep neuraal netwerk is ook gevoelig voor vijandelijke aanvallen, zoals een classificatie aanval of uitleg aanval. Een classificatie aanval veranderd het voorspelde label, terwijl een uitleg aanval de uitleg van het neurale netwerk manipuleert terwijl het de classificatie onveranderd laat.

Ons werk laat zien dat classificatie aanvallen onder andere mogelijk zijn door de vorm van de activatie functie, de gebruikelijke softax functie. Door de softmax te vervangen met een polynomiale vervanging, dat we de softRmax noemen, worden aanvallen minder succesvol. Oftewel, de activatie is ook een aspect waar neurale netwerken gevoelig voor zijn.

In de context van uitleg aanvallen, ontwikkelen we een nieuwe aanval op het Grad-CAM uitleg-algoritme. Onze aanval veranderd de uitleg maar veranderd niet de classificaties.

Tenslotte onderzoeken we de invloed van de persoon die de hyperparameters tuned van een diep neuraal netwerk. Via een gebruikers onderzoek onderzoeken we de correlatie tussen de prestaties van de getunede diepe neurale netwerken en de kennis van deep learning van de gebruiker. We komen er achter dat deze twee gecorreleerd zijn.

1

INTRODUCTION

Machine learning differs from imperative computer programming in that it does not tell the machine *how* to do a task; instead it focuses on trying to specify *what* task to do by offering examples with for each data sample the expected outcome for the task at hand [1]. As a sub-field of machine learning, deep learning [2] differs itself from classic machine learning models by focusing on signals such as images, speech, text where its flexibility in representation learning and architecture design is shown to be an advantage. With the progress in the field of hardware, which offered greater computational power for algorithms, we observed successes of deep learning models on large image datasets that were not possible otherwise. The great success of the AlexNet deep network [3] on the ImageNet Large Scale Visual Recognition Challenge in 2012 is considered as the breakthrough of deep learning. AlexNet belongs to the family of Convolutional Neural Networks (CNNs) that was initially proposed by Yann LeCun *et al.* in 1989 [4]. AlexNet marks a milestone of the use of CNNs on computer vision tasks.

The representation learning characteristic of deep nets brings both simplicity for application and good performance for visual recognition. Unlike classic machine learning approaches where smart representations need to be extracted from images by experts, deep nets can learn discriminative high-dimensional representations. This has greatly lowered the barrier of the application of deep learning in fields outside computer science such as health, agriculture and business [5]–[7]. Such a way of representation learning also gives the network great flexibility to explore possible features that are beyond human understanding but can lead to state of the art performance, which further has led to the popularity of deep learning.

In particular, Convolutional Neural Networks (CNNs) [8], [9] have been successfully applied on many visual learning problems. No more handcrafted features are needed compared to classic computer vision approaches. Nevertheless, deep nets remain black-boxes [10], the performance of which can be overly sensitive to many factors. In this theses, I mainly focus on three of these factors: distribution shifts, malicious attacks, and human factors. These three factors cover possible disturbances to the performance of CNNs from three perspectives: intrinsic data sampling bias, extrinsic threats and the in-

teractive factor between the network and human beings. The three factors are explained below.

The performance of CNNs is sensitive to distribution shifts [11] between datasets, *i.e.* , when datasets follow different distributions. The training data and the test data are not guaranteed to be independent and identically distributed. This happens especially in real-world applications, such as differences in recording devices. A well trained CNN usually fails under distribution shifts. We explore various ways to cope with distribution shifts, including domain adaptation, domain generalization, transfer learning and multi-domain learning.

CNNs are also sensitive to malicious attacks. Two types of malicious attacks, classification attack [12], [13] and explanation attack [14], [15], are discussed in this thesis. Both of the two attacks can be categorized into adversarial attacks and architecture attacks. Adversarial classification attacks aim to add a small perturbation to the input to cause decrease in classification accuracy. We will present a simple way to increase the robustness of deep nets against adversarial classification attacks. An explanation attack changes the explanation heatmap without harming the classification results. An adversarial explanation attack adds a small perturbation in the input to change the explanation while architecture attack manipulates the network to change the explanation. We will focus on the architecture attack for explanation attacks and show multiple ways to achieve misleading explanation heatmaps.

Furthermore, the performance of deep learning is sensitive to the settings of its hyperparameters [16], [17], and the choice of hyperparameters highly depends on the human trainers [18]. We explore correlations between the human trainers' experience with deep learning and the final deep net performance. We show that the experience of a human trainer plays an important role in the final classification accuracy of the same network on the same task.

We will look into the three sensitivity factors, distribution shifts, malicious attacks and human factors and propose some solutions to remedy the sensitivities. The following sections in this chapter will present some background of these factors for interested readers.

1.1. PRELIMINARIES

CNNs are a nonlinear mapping from an input space, typically images, to a target output, typically a class label. The nonlinear mapping is constructed by several stacked layers consisting of multiple filter kernels. The kernel weights are learned by optimizing a loss function designed for a specific task. The architecture of a CNN is flexible (like LEGO) and can be reconstructed for different tasks, *e.g.*, number of layers, kernels, kernel size, can all be customized.

A classification task aims to assign a data sample to a certain category. Logistic regression is a classic machine learning model for classification that maps a data sample linearly to a category. A deep learning network resembles logistic regression but can learn a nonlinear mapping of the data samples to categories. The classification pipeline with CNNs can be dissected into a feature extractor and a classifier. The feature extractor g maps the input data $\mathbf{x}$ to a latent representation $g(\mathbf{x})$. The classifier f further maps the latent representation to embedding $\mathbf{z}$, the size of which is the number of classes m. With

the standard softmax activation ς, the embedding $\mathbf{z}$ is normalized to a vector of posteriors with $p(y_i|\mathbf{x}) = \varsigma_i(\mathbf{z}) = e^{\mathbf{z}_i} / \sum_k e^{\mathbf{z}_k}$, where y_i is the class number i. The predicted class label is then the class $\hat{y}$ with the highest posterior. Given n samples, with x_j denoting sample j, weights in deep nets are optimized by minimizing the cross entropy loss as follows

$$L = -\frac{1}{n}\sum_{j}^{n}\sum_{i}^{m} y_i \cdot \log(p(y_i|x_j)). \tag{1.1}$$

A regression task aims to learn a mapping from an input to a continuous value [19] and is performed in a similar way as a classification task in deep learning. It can use the same architecture as a classification network, but the last layer is changed to have a 1 dimensional output. As a result in a regression task, input $\mathbf{x}$ is mapped to a 1-D value $\hat{y}$. The basic loss function for regression is the mean square error. With y as the target value, the mean square error is:

$$L = -\frac{1}{n}\sum_{i}^{n}(\hat{y}_i - y_i)^2. \tag{1.2}$$

The main difference between classification and regression is that classification maps inputs to discrete categories while regression learns a mapping to a continuous value. Both the classification task and the regression task will be discussed in this work.

1.2. DISTRIBUTION SHIFT

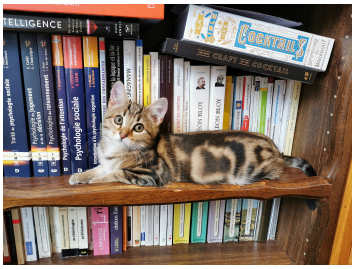
A standard assumption made in deep learning, or more general in machine learning, is that the training data and the test data are independent and identically distributed (i.i.d.). That means, all the data are sampled independently from the same distribution p. However, real world data often does not follow the i.i.d. sampling rule which leads to a distribution shift between the training data and the test data. For standard sets where the training and test data are nearly i.i.d. , a deep net usually transfers well from the training data to the test data. However, a trained deep net is likely to fail on test data if the training data and the test data are non-i.i.d. [20], [21]. This is defined as sensitivity to distribution shift in this thesis.

Consider a scenario in autonomous driving as an illustration. An autopilot system that is well trained on the data collected from Europe can recognize the pedestrian and cars when it is deployed in Europe. If the same autopilot system is directly operated in Asia without any tuning, its performance there will not be able to match what can be achieved in Europe due to the differences in infrastructure, environment and appearance *etc.* One solution is that we can label many data from Asia and use the data to retrain the model, which is expensive and requires a lot of human effort. Another solution is to transfer the knowledge learned from the European dataset to the Asian dataset. Several solutions for transferring the knowledge are discussed below.

1.2.1. DOMAIN ADAPTATION

Domain adaptation [22] is one of the approaches to tackle the sensitivity of a model to distribution shift by adapting the distribution of the training data p_{D_s} to approximate

1



(a) Source domain: photo of my cat Pepper.

(b) Possible target domain: painting, sketch, porcelain.

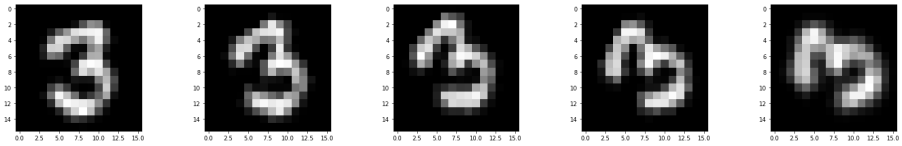
Figure 1.1: Illustration of domains. Figure 1.1a shows the source domain that consists of real world photo of cat. A model trained on the source domain for a classification task may not recognize the object 'cat' in a non-identical domain, which can be paintings or sketches of cats, as shown in Figure 1.1b. Domain adaptation is not constrained by classification tasks. The source domain and target domains can also form a cross-domain image retrieval task. 1.1a, credit to my cat, Pepper. 1.1b, reproduced (with altered formatting) with permission from Rijksmuseum, Amsterdam.

the distribution of the test data p_{D_t}. In domain adaptation, the training data is also referred as the source domain D_s, and the test data is named as the target domain D_t. Both the source and the target domain are available during training. The source domain is labelled while the target domain is usually unlabelled or partially labelled due to lack of experts or resources, which makes domain adaptation an unsupervised/semi-supervised task to obtain predictions on the target domain. Figure 1.1 presents an illustration of a source domain and target domain in domain adaptation.

Covariate shift is a common type of distribution shift, that assumes a constant labelling function across distributions of domains $p_{D_d}, d \in \{1,2,3,\ldots,d\}$, as the index of domain D. That means the same posterior and label will be obtained for an input $\mathbf{x}$, no matter from which distribution p_{D_d} it is sampled; $p(y_i|\mathbf{x}, D_d) = p(y_i|\mathbf{x})$. One classic domain adaptation approach [20] for covariate shift is weighting samples in the source domain by importance. Samples that are more likely belonging to the target domain are considered as more important thus receive higher weights w and vice versa. The likelihood that a sample belongs to a certain domain can be obtained by a density estimation. However, we show in this work that the choice of the distribution when estimating the density impacts the sensitivity of the model to outliers in the data.

Another domain adaptation approach aims to find a common space where the distributions of the source domain and the target domain are as similar as possible. Note that the true distribution of a domain is not available but can only be estimated by the sampled data. Due to the feature learning characteristic of deep learning, domain adaptation in deep learning relies on learning a domain invariant representation, $\mathbf{z}$, that satisfies $p_{z|D_s} = p_{z|D_t}$. The underlying assumption of learning domain invariant representations is that the source domain and the target domain can be mapped to a latent space where the distributions overlap.

1.2.2. DOMAIN GENERALIZATION



(a) 0° (b) 15° (c) 30° (d) 45° (e) 60°

Figure 1.2: The rotated MNIST dataset serves as an example for domain order, which is rotation in this case.

Another non-i.i.d. setting is domain generalization [23] where multiple labeled non-i.i.d. source domains $D_s, s \in \{1,2,3,\ldots,d\}$ are available while the target domain D_t is absent. It is not possible to perform adaptation due to the absence of the target domain so the focus is on generalization. The intuition of domain generalization is that domains share similar information and useful features can be learned from the source domains D_s which can generalize to the unseen target domain D_t. The hidden assumption about the underlying distribution shift, is that the distribution shift is a covariate shift [23], [24], which means the same posterior is obtained for a sample $\mathbf{x}$ no matter which domain it comes from.

Due to the similarities between domain adaptation and domain generalization, similar methods can be easily spotted and many focus on learning domain invariant representations based on the source domains [25]–[27]. To be noted, learning domain invariant representation is meaningful for domain adaptation given the fact that the target domain is available. It is questionable to learn source domain invariant representations with the hope to generalize to the unseen target domain for domain generalization as there is no guarantee that the target domain will also be mapped to the same representations.

We suggest to focus on the distribution of domains P_D for domain generalization so we can better model the possibilities of the target domain. We define domain order as the distribution P_D from which all domains involved in a task are sampled. Note that it is not possible to simulate the domain order for most inputs, except for some special cases, for example when domains are rotated versions of each other (as shown in Figure 1.2 in which numbers are drawn). In the early years of this field [24], [28], research focuses on developing an universal generalization approach without putting much attention on the prior domain order (distribution), such as rotation, light change [24]. The potential target domain does not follow any order presented in the source domains and the source domains do not have a clear order themselves. However, recent research starts to debate whether it is at all possible to generalize to any unseen target domain without assuming a domain order [29]. We show in this work that learning domain invariant representation is not necessary and may even hurt the generalization. We speculate that this is due to the discarding of domain order. We propose a way to relax the domain invariant representation learning approach, to give the network the possibility to keep the domain order.

1.2.3. TRANSFER LEARNING

Transfer learning [30] is an often encountered practical approach [31]–[33] for deep nets to use large scale public dataset to benefit specific tasks where the target domain D_t is labelled but of small size. Unlike domain adaptation and domain generalization, the same labelling space is not always assumed in transfer learning. The intuition behind this is that the large scale dataset, for example, ImageNet, contains information that can be useful for other tasks. However, not surprisingly, recent research has shown that such kind of transfer is not always successful [34], [35]. Due to potential large distribution shifts between public datasets and specialized datasets, *e.g.*, consumer photographs and medical images, it is hard to imagine how transfer learning would help. Surprisingly, in practice, we show that for a Parkinson's disease classification task that one can benefit from other public datasets.

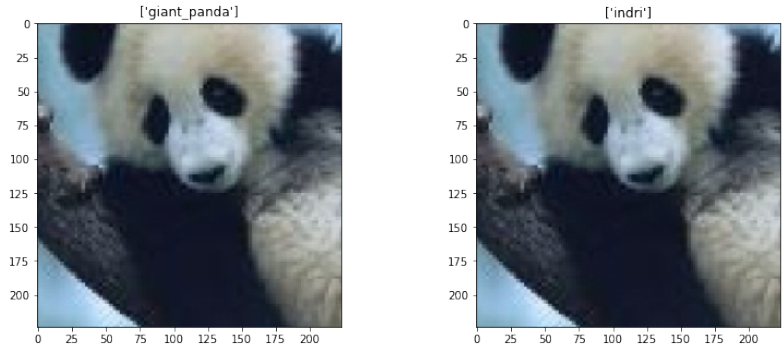
1.2.4. MULTI-DOMAIN LEARNING

Multi-domain learning [36], [37] is similar to domain generalization where multiple non-i.i.d. source domains are combined to make a final prediction on a target task. Different from domain generalization, the target task is known in multi-domain learning. The final prediction could be on the same task or on a different one. For example, data of Parkinson patients can be recorded into various tasks, finger tapping, hands movements, kinetic tremor *etc.* Those tasks can be combined to make a final prediction on a patient. The non-i.i.d. nature of domains make the challenge to be a domain combining strategy. It is not hard to imagine that some domains are more important for the final task while some are less. So the model is sensitive to the distribution of domains and correspondingly the combination strategy of domains is of uttermost importance.

1.3. MALICIOUS ATTACKS

Deep nets are sensitive to malicious attacks that are designed to harm the performance of a network. A typical malicious attack in deep learning is an adversarial attack which aims to change the prediction result of a model by adding tiny perturbations in the input. The manipulated input $\mathbf{x}'$ and the original input $\mathbf{x}$ are visually not distinguishable by a human. But the manipulated input can fool the network to give a wrong prediction result. Usually the wrongly predicted results are also accompanied with high confidence, which makes the attack even harder to be detected. Adversarial attack can be dangerous in some scenarios where a high level of safety is required, for instance, autonomous driving and the medical field. With adversarial attacks, a hacker with malicious intention can change the prediction of a pedestrian to be a road for example, or a patient who is healthy to be ill.

We present another type of attack in this thesis, explanation attack. An explanation attack is a relatively new type of attack that has not been extensively explored yet. The high sensitivity of deep net to adversarial attack is partially attributed to the black-box magic in deep learning. Given this background, how to obtain an explanation of a trained deep net has become an important topic with the hope to understand the mechanism of deep learning better, thus reduce the sensitivities of deep nets to attacks. Unintentionally, the existing explanation approaches themselves are sensitive to attacks as



(a) Confidence: 95.05%

(b) Confidence: 79.03%

Figure 1.3: Input image and predicted class by ResNet-50 pretrained on ImageNet. The image on the left is the original ImgeNet image $\mathbf{x}$ and the one on the right is the manipulated image $\mathbf{x}'$ generated by FGSM at attack level $\epsilon = 0.01$. The two images are not visually different but the manipulated one is predicted to be a wrong class 'Indri' with a high confidence of 79.03%. Note that ImageNet contains 1000 classes so a posterior of 79.03% is very high.

well. This leaves the doubt whether the explanation results are trustworthy or not.

1.3.1. ADVERSARIAL ATTACK

Adversarial attacks can be categorized into two types, white-box attack and black-box attack [38]. White-box attacks have access to the deep net architecture and are usually gradient based, *e.g.*, Fast Gradient Sign Method (FGSM) [12] computes the gradient of the loss function to the input and adds a small value in the input following the gradient direction to increase the loss, therefore changes the prediction result. The manipulated image is not visually different from the original image, as shown in Figure 1.3. The perturbation is decided by a coefficient ϵ multiplied by the sign of the loss gradient $\nabla_x L(\mathbf{w}, \mathbf{x}, y)$ where L is the loss function used to optimize the network weights $\mathbf{w}$ with $\mathbf{x}$ as the input and y as the label. The perturbed input becomes:

$$\mathbf{x}' = \mathbf{x} + \epsilon \operatorname{sign}(\nabla_x L(\mathbf{w}, \mathbf{x}, y)) \tag{1.3}$$

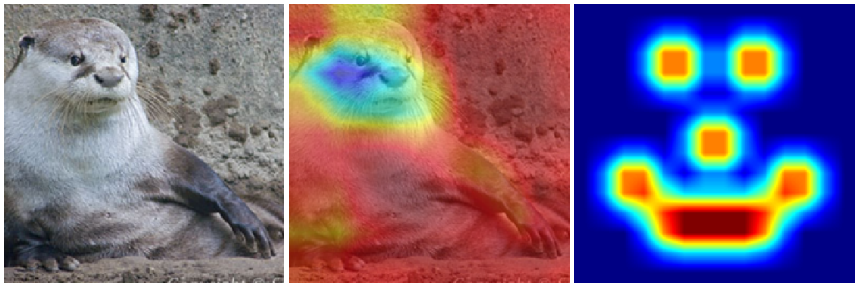
A black-box attack does not have direct access to the architecture but can train a substitute network that mimics the decision of the target architecture. The ability of a network defending adversarial attacks is defined as robustness, which usually requires adversarial training.

1.3.2. EXPLANATION ATTACK

For computer vision by deep learning tasks, a heatmap based explanation aims to generate a heatmap that highlights the informative region of the input regarding the task [39], [40]. Take the famous GradCAM [39] as an example, the importance is determined by the gradient of the posterior $p(y_i|\mathbf{x})$ to the feature maps, which are the outputs of a certain layer in the network. The heatmap is a linear combination of all the feature maps (channels) followed by a ReLU function that removes negative values. This explanation

is shown to be effective on the public dataset ImageNet for classification task. For example in Figure 1.4, if an otter sits on the sand, the heatmap highlights the otter instead of the background for the class 'otter'. However, a real explanation is still far from giving clear reasoning on how the deep net makes a certain decision but it brings intuitive understandings.

Unfortunately the explanation approaches also suffer from attacks, like the prediction result itself as shown in Figure 1.4. The aim of an explanation attack is to change the explanation heatmap to be a certain targeted explanation map or random map that cannot explain the classification result, without harming the classification performance. Attacks for explanations can be categorized into adversarial input attacks, which changes the inputs, and architecture attacks, which manipulates the network itself. The latter does not change the input but alternates the architecture. We will focus on architecture attack in this thesis.



(a) Input (b) Original explanation (c) attacked explanation

Figure 1.4: Illustration of explanation and explanation attack. The original GradCAM explanation highlights the otter itself while the attacked explanation is forced to be a target smiley face. No perturbation is added in the input. The classification result is not changed but the explanation heatmap cannot explain the classification decision anymore.

1.4. HUMAN IMPACT

Another factor, the human role, to which a deep net is sensitive is often neglected. After all, it is still a human who does the training before the true machine intelligence emerges. Given a fixed deep net architecture, the interaction between human trainers and deep nets is mainly on the hyperparameters, such as learning rate, batch size, weight decay, *etc.* There exist many studies regarding the impact of hyperparameters on the weights optimization of the network [41]–[43] but few sheds light on the impact of human trainer. How are the hyperparameters chosen? Whether trainers from different education levels have different training strategies? Or is it the experience with deep learning that decides the strategies of choosing hyperparameters? We will study the correlation between human experience of deep learning and the final performance of a deep net.

1.5. CONTRIBUTIONS

The structure of this thesis is presented in table 1.1 for the readers' convenience.

Table 1.1: Sensitivities in deep learning

Sensitivity	**Chapter**
Sensitive to distribution shift	
Domain adaptation	Chapter 2, image retrieval
Domain generalization	Chapter 3, preserve domain order
Transfer learning and multi-domain learning	Chapter 4, Parkinson disease analysis
Sensitive to distribution	Chapter 5, softRmax
Sensitive to attack	
Adversarial attack	Chapter 5, softRmax
Explanation attack	Chapter 6, GradCAM attack
Sensitive to human trainer	
Human impact	Chapter 7 black magic

Chapter 2 presents a domain adaptation solution for visual place recognition across different time periods. The visual place recognition task is designed to be image retrieval with the goal to collect images from the database that contain the same building as shown in the query image. Query images are archived old Amsterdam architectures while the database consists of modern Amsterdam cityscapes. We show that the deep net is sensitive to the distribution shift between the query and the database caused by time change.

Chapter 3 discusses whether it is possible to generalize to unseen target domains and shows that learning domain invariant representations cannot achieve this goal. To the contrary, learning domain invariant representation may further discard the information about domain distribution or domain order. We propose to relax the constraints on representation learning by learning hypothesis invariant representations, which in practice, is learning posterior invariant representations. Hypothesis invariant representation does not force representations from source domains to be invariant thus can potentially preserve the domain order. If the target domain follows the same domain sampling strategy, the model can better generalize to this unseen target domain by keeping the domain order.

Chapter 4 shows that transfer learning from large public dataset is a good solution when the dataset is very small, but may need extra design for the medical applications. We show that Parkinson disease quantification based on videos can benefit from public video datasets. Guiding the network to focus on temporal correlations between frames leads to an even better transfer results. We speculate this is because the actions in the public video dataset are of large scale while the tremor in the Parkinson videos are subtle. A temporal attention mechanism can learn the frequency discrepancy between the public dataset and the medical dataset better. We also present a learned weighting scheme

for various non-i.i.d. medical tasks/domains to quantify Parkinson disease at patient-level.

Chapter 5 proposes an adversarial defense by linking the standard softmax activation function $\varsigma_i^S(\mathbf{z})$ to Linear Discriminant Analysis (LDA). Following Bayes' rule, the posterior of class y_i with LDA, under equal priors, is

$$p(y_i|\mathbf{x}) = \frac{p(y_i)p(\mathbf{x}|y_i)}{\sum_k p(y_k)p(\mathbf{x}|y_k)} = \frac{p(\mathbf{x}|y_i)}{\sum_k p(\mathbf{x}|y_k)}. \tag{1.4}$$

To obtain class conditional probability $p(\mathbf{x}|y_i)$, the density estimation is often based on a Gaussian distribution. We show that the exponential behavior of the Gaussian distribution leads to overconfident prediction for samples in the distribution tail and sensitivity to adversarial attacks. We substitute the Gaussian distribution with a polynomial Cauchy distribution and propose a polynomial softRmax function. Deep nets with softRmax as the final layer activation function give more conservative predictions for samples in the tail and show improved robustness against adversarial attacks. This further shows deep nets are sensitive to the choice of distributions.

Chapter 6 presents four different attacks on the popular explanation method, GradCAM, on ImageNet. We show that it is possible to manipulate the explanation to be random, constant color, constant image, or targeted sticker for all the images, while maintaining the original classification performance. An example is presented in Figure 1.4. We do not imply the GradCAM method is useless by any means, but simply raise the awareness that it is also sensitive to attacks, like the network itself.

Chapter 7 analyzes statistically how sensitive the performance of a deep net is to human experience with deep learning. We try to answer the questions about human impacts on the choice of hyperparameters and how this can further influence the training of the network.

BIBLIOGRAPHY

[1] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*, 4. Springer, 2006, vol. 4.

[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[4] Y. LeCun, B. Boser, J. S. Denker, *et al.*, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[5] D. Ravì, C. Wong, F. Deligianni, *et al.*, "Deep learning for health informatics," *IEEE Journal of Biomedical and Health Informatics*, vol. 21, no. 1, pp. 4–21, 2017. DOI: 10.1109/JBHI.2016.2636665.

[6] A. Kamilaris and F. X. Prenafeta-Boldú, "Deep learning in agriculture: A survey," *Computers and electronics in agriculture*, vol. 147, pp. 70–90, 2018.

[7] J. B. Heaton, N. G. Polson, and J. H. Witte, "Deep learning for finance: Deep portfolios," *Applied Stochastic Models in Business and Industry*, vol. 33, no. 1, pp. 3–12, 2017.

[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[9] 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*, 2016, pp. 770–778.

[10] R. Shwartz-Ziv and N. Tishby, "Opening the black box of deep neural networks via information," *arXiv preprint arXiv:1703.00810*, 2017.

[11] J. G. Moreno-Torres, T. Raeder, R. Alaiz-RodrıGuez, N. V. Chawla, and F. Herrera, "A unifying view on dataset shift in classification," *Patt.Recog.*, vol. 45, no. 1, pp. 521–530, 2012.

[12] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *stat*, vol. 1050, p. 20, 2015.

[13] A. Kurakin, I. Goodfellow, S. Bengio, *et al.*, *Adversarial examples in the physical world*, 2016.

[14] A.-K. Dombrowski, M. Alber, C. Anders, M. Ackermann, K.-R. Müller, and P. Kessel, "Explanations can be manipulated and geometry is to blame," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[15] J. Heo, S. Joo, and T. Moon, "Fooling neural network interpretations via adversarial model manipulation," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[16] F. Hutter, H. Hoos, and K. Leyton-Brown, "An efficient approach for assessing hyperparameter importance," in *Proceedings of the 31st International Conference on Machine Learning*, PMLR, 2014, pp. 754–762.

[17] A. Koutsoukas, K. J. Monaghan, X. Li, and J. Huan, "Deep-learning: Investigating deep neural networks hyper-parameters and comparison of performance to shallow methods for modeling bioactivity data," *Journal of cheminformatics*, vol. 9, no. 1, p. 42, 2017.

[18] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, pp. 148–175, 2016.

[19] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: Data mining, inference, and prediction*. 2009.

[20] M. Sugiyama, M. Krauledat, and K.-R. Müller, "Covariate shift adaptation by importance weighted cross validation.," *Journal of Machine Learning Research*, vol. 8, no. 5, 2007.

[21] Z. Wang, J. Li, S. Khademi, and J. van Gemert, "Attention-aware age-agnostic visual place recognition," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019.

[22] W. M. Kouw and M. Loog, "A review of domain adaptation without target labels," *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[23] K. Zhou, Z. Liu, Y. Qiao, T. Xiang, and C. C. Loy, "Domain generalization: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[24] D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales, "Deeper, broader and artier domain generalization," in *ICCV*, 2017, pp. 5542–5550.

[25] S. Motiian, M. Piccirilli, D. A. Adjeroh, and G. Doretto, "Unified deep supervised domain adaptation and generalization," in *ICCV*, 2017, pp. 5715–5725.

[26] M. Ghifary, D. Balduzzi, W. B. Kleijn, and M. Zhang, "Scatter component analysis: A unified framework for domain adaptation and domain generalization," *IEEE TPAMI*, vol. 39, no. 7, pp. 1414–1430, 2016.

[27] M. Ghifary, W. Bastiaan Kleijn, M. Zhang, and D. Balduzzi, "Domain generalization for object recognition with multi-task autoencoders," in *ICCV*, 2015, pp. 2551–2559.

[28] K. Muandet, D. Balduzzi, and B. Schölkopf, "Domain generalization via invariant feature representation," in *ICML*, 2013, pp. 10–18.

[29] T. Galstyan, H. Harutyunyan, H. Khachatrian, G. V. Steeg, and A. Galstyan, "Failure modes of domain generalization algorithms," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 19 077–19 086.

[30] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big data*, vol. 3, no. 1, pp. 1–40, 2016.

[31] M. Huh, P. Agrawal, and A. A. Efros, "What makes imagenet good for transfer learning?" *arXiv preprint arXiv:1608.08614*, 2016.

[32] M. Raghu, C. Zhang, J. Kleinberg, and S. Bengio, "Transfusion: Understanding transfer learning for medical imaging," *Advances in neural information processing systems*, vol. 32, 2019.

[33] A. R. Zamir, A. Sax, W. Shen, L. J. Guibas, J. Malik, and S. Savarese, "Taskonomy: Disentangling task transfer learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2018.

[34] B. Zoph, G. Ghiasi, T.-Y. Lin, *et al.*, "Rethinking pre-training and self-training," *Advances in neural information processing systems*, vol. 33, pp. 3833–3845, 2020.

[35] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Transfer learning for time series classification," in *2018 IEEE international conference on big data (Big Data)*, IEEE, 2018, pp. 1367–1376.

[36] M. Joshi, M. Dredze, W. Cohen, and C. Rose, "Multi-domain learning: When do domains matter?" In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 2012, pp. 1302–1312.

[37] M. Dredze, A. Kulesza, and K. Crammer, "Multi-domain learning by confidence-weighted parameter combination," *Machine Learning*, vol. 79, no. 1, pp. 123–149, 2010.

[38] N. Papernot, P. McDaniel, A. Sinha, and M. Wellman, "Towards the science of security and privacy in machine learning," *arXiv preprint arXiv:1611.03814*, 2016.

[39] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 618–626.

[40] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning Deep Features for Discriminative Localization," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[41] S. Jasper, L. Hugo, and P. A. Ryan, "Practical bayesian optimization of machine learning algorithms," *Bartlett et al. [8], pp. 2960–2968*, 2012.

[42] M. Lucic, K. Kurach, M. Michalski, S. Gelly, and O. Bousquet, "Are GANs Created Equal? A Large-Scale Study," *arXiv e-prints*, 2017.

[43] G. Melis, C. Dyer, and P. Blunsom, "On the State of the Art of Evaluation in Neural Language Models," *arXiv e-prints*, 2017.

2

DOMAIN ADAPTATION

*A cross-domain visual place recognition (VPR) task is proposed in this work, i.e., matching images of the same architectures depicted in different domains. VPR is commonly treated as an image retrieval task, where a query image from an unknown location is matched with relevant instances from geo-tagged gallery database. Different from conventional VPR settings where the query images and gallery images come from the same domain, we propose a more common but challenging setup where the query images are collected under a new unseen condition. The two domains involved in this work are contemporary street view images of Amsterdam from the Mapillary dataset (source domain) and historical images of the same city from Beeldbank dataset (target domain). We tailored an age-invariant feature learning CNN that can focus on domain invariant objects and learn to match images based on a weakly supervised ranking loss. We propose an attention aggregation module that is robust to domain discrepancy between the train and the test data. Further, a multi-kernel maximum mean discrepancy (MK-MMD) domain adaptation loss is adopted to improve the cross-domain ranking performance. Both attention and adaptation modules are unsupervised while the ranking loss uses weak supervision. Visual inspection shows that the attention module focuses on built forms while the dramatically changing environment are less weighed. Our proposed CNN achieves state of the art results (*99% *accuracy) on the single-domain VPR task and* 20% *accuracy at its best on the cross-domain VPR task, revealing the difficulty of age-invariant VPR.*

This chapter has been published as:
Z. Wang, J. Li, S. Khademi, J. van Gemert, "Attention-Aware Age-Agnostic Visual Place Recognition", Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops, 2019 [1].

2.1. INTRODUCTION

Recently, there has been interest among the computer vision researchers to solve the visual place recognition (VPR) task in the form of image retrieval [2]–[8]. In [9], the discriminative visual cues learned for visual place classification task are investigated. Interestingly, CNN filters learn human-like discriminative visual cues to recognize a place, including built forms, signs or vegetation. Among these discriminative attributes, buildings are the most robust that remain, more or less, invariant during the changes in day and night lighting, different seasons and even years. However, CNNs are still influenced by irrelevant objects like roads and the sky. In this work, we introduce a CNN model with attention aggregation module to focus on domain invariant features, i.e. buildings, for the cross-domain VPR task. We will demonstrate that our work can be further combined with multi-kernel Maximum Mean Discrepancy (MK-MMD) loss to obtain better domain adaptation results. The images from the two domains with a large time lag are depicted in Fig.2.1, being historical images (queries) and current street view images (gallery) of Amsterdam.

Figure 2.1: Correctly retrieved images with our proposed method. The top row illustrate the general place recognition in the same domain: both the query (left) and gallery image (right) are from the same dataset. The bottom row shows the cross-domain place recognition task where the query (left) is from the *Beeldbank* dataset and the galley image (right) is from the *Mapillary* dataset.

The VPR task is commonly formulated as content based image retrieval (CBIR), i.e., sorting the geo-tagged gallery images by their distances to the unknown query image. The query is then labeled based on its best matching image in the gallery. Deep image representation learning is currently state of the art for almost all CBIR settings. Among the deep feature learning methods, distance learning CNNs are the most popular ones [10], [11]. Nevertheless, supervised deep distance learning requires similar and dissimilar image pairs for training. In this work, image pair labels are not available and we only have access to geo-tagged images from the *Mapillary* street view imagery and thus a weakly supervised deep feature learning is used, similar to the work of NetVLAD[2].

Different from [2], our queries are historical images which are not geo-tagged and exhibit a domain discrepancy between training data and test data. Age-agnostic place

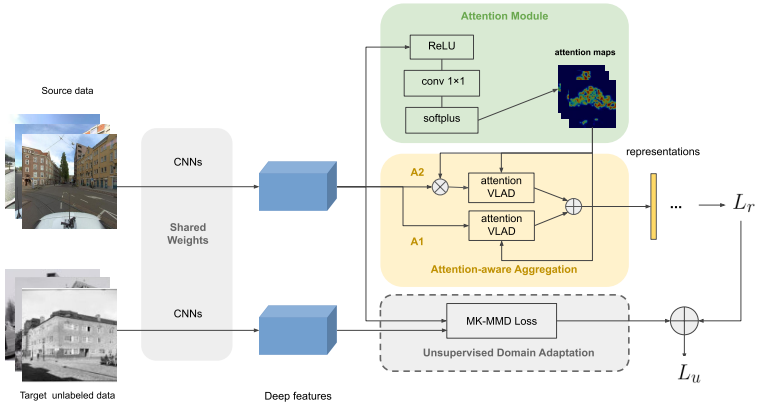


Figure 2.2: Our proposed CNN model include three modules: an attention module, am attention-aware VLAD module and a domain adaptation module. The attention-aware VLAD module uses the attention scores to weigh both the deep features and the global descriptors with two streams, A_1 and A_2, which are explained in Section 2.3.2.

recognition that is addressed in this paper is a more challenging problem firstly due to the lack of image pair labels for training, secondly due to the domain shift between the gallery and query images caused by the change of scenery over a large time gap and thirdly due to the outliers in target domain. Different technologies of photography, equipment and processes used in the production of photos in the past also contribute to this domain shift. Fig2.1 shows the general and the age-agnostic place recognition task.

We are inspired by [12], which introduces an attention module into NetVLAD for the classification task to address the unequal importance of local features in VLAD feature aggregation layer. In our work, a new attention aggregation technique is proposed to weigh both global VLAD descriptors and local descriptors. A domain adaptation loss based on MK-MMD is additionally introduced to achieve better cross-domain performance. Note that both the attention and the domain adaption modules are unsupervised and thus no labels are required.

Our attention-aware architecture is depicted in Fig.2.2 which consists of three modules and a shared convolutional neural network for feature extraction (AlexNet cropped before $conv5$). The attention module is a single convolutional layer followed by softplus activation function, transforming the feature map to a heatmap. This heatmap contains attention scores for the deep features. The VLAD module aggregates deep features in the attention-aware scheme by assigning attention scores to both local and global descriptors. The unsupervised domain adaptation module is additionally used to learn domain-invariant features. Our oblation studies show that both modules are important to reach state of the art results. Our speculation is that MK-MMD loss aligns the photo styles while attention module focuses on domain invariant contents.

Our contributions are summarized as:

- To the best of our knowledge, this is the first large scale (40k) image database for age-invariant visual place recognition task. We manually annotated 104 histori-

cal images and their corresponding matched current street view images only for evaluation purpose.

- A new attention aggregation scheme is proposed to combine both the local and global image descriptors (Section 2.3.2).
- We combined the MK-MMD domain adaptation loss with the ranking loss to learn domain-invariant features for cross-domain VPR task. (Section 2.3.3).

We tested our proposed model on conventional VPR task and our experiments show the state of the art results on Mapillary dataset compared to other competitors. Detailed results and ablation studies will be presented in Section 2.4.5. The comparison of single-domain and cross-domain results reveals the difficulty of age-agnostic place recognition task.

2.2. RELATED WORK

The performance of VPR as an image retrieval problem depends on the ranking accuracy w.r.t. a similarity metric. The query location is suggested based on the top M similar images (annotated with geo-tags). To extract good features for indexing, traditional works focus on hand-crafted features such as SIFT[13]) and SURF[14]. Some other efficient methods are based on the aggregation of local gradient-based descriptors like Fisher Vectors [15] and VLAD[16]. [5] is a SURF based model which improves the performance by detecting and removing 'confusing objects'. [7] uses SIFT to detect the repetitive patterns in the image which is representative for buildings. [17] focuses on matching images that have large view point changes by generating artificial views of a scene for the training process.

Recent works suggest that a CNN trained on a large scale dataset as a feature extractor outperforms hand crafted features on various tasks [3], [18]–[20]. In turn, [21] shows that features in the early layers of a CNN trained for image classification can be effectively used as visual descriptors for image retrieval. LIFT [22] is a learning pipeline for feature extraction which introduces an end-to-end unified network for detection, orientation estimation, and feature description. [23] proposes a global image representation by the regional maximum activation of convolutional layers (R-MAC) well-suited for place recognition. [3] proposes novel CNN-based features designed for place recognition by detecting salient regions and extracting regional representations as descriptors. NetVLAD [2] introduces a novel triplet ranking loss together with a VLAD aggregation layer that can learn powerful representations for the VPR task in an end-to end manner. A known disadvantage of NetVLAD lies in its global feature aggregation. [24] proposes a region proposal network to learn which regions should be pooled to form the final global descriptor. Similar to [2] , we use current geo-location tags for weakly supervised feature learning using triplet distance learning network. However, we do not have access to matched image pairs from the two domains for supervised training, i.e., matched historical and contemporary images. To address this domain mismatch between the test and train data, we need to promote domain-invariant feature learning.

We tailor an attention aggregation model that can boost the cross-domain performance for our specific task, age-agnostic urban scene matching. Attention model is

broadly used in natural language processing [25], [26] and computer vision tasks [8], [19], [27]–[31]. [32] shows that attention model can also be adopted to benefit metric learning. [19] proposes an attention mechanism to select key points for matching. Attention model is considered to be effective for domain adaptation as well [33], [34]. Our attention model is implemented in an unsupervised way which means no ground truth score maps are available for training. The learning process of the attention module is guided by the image retrieval ranking loss.

Given two different domains, unsupervised deep domain adaptation schemes [35], [36] are mostly used to enhance the performance of CNNs on target domain by using labels only from the source domain. Among the vast amount of literature on deep domain adaption for classification tasks, the Maximum Mean Discrepancy (MMD) loss is introduced by [37] to minimize the domain discrepancy by projecting data into a kernel space. Later [38] proposed multi-kernel MMD (MK-MMD) which uses linear combination of multiple kernels. We adopt MK-MMD loss as an additional domain adaptation module for our attention aggregation model. Similarly, we feed untagged historical images of Amsterdam to the adaptation layer in an unsupervised manner.

2.3. METHOD

Our proposed model consists of three modules for feature extraction, namely a weakly supervised image retrieval module with a triplet ranking loss (Section 2.3.1), an attention aggregation module(Section 2.3.2) and an unsupervised domain adaptation module with MK-MMD loss (Section 2.3.3). MK-MMD loss constrains the feature maps after the last convolution layer ($conv\_5$). The final loss function for training, L_u, can be expressed as:

$$L_u = L_r + \alpha M(\mathscr{D}_s, \mathscr{D}_t) \tag{2.1}$$

where $M(\mathscr{D}_s, \mathscr{D}_t)$ is the MK-MMD loss term, $\mathscr{D}_s$ and $\mathscr{D}_t$ denote the source domain and target domain, L_r is the triplet ranking loss used in NetVLAD [2], α is the weight that trades off the image retrieval loss and the domain adaptation loss.

2.3.1. IMAGE RETRIEVAL WITH WEAK SUPERVISION

We use NetVLAD [2] as our baseline model which tackles the weakly supervised image retrieval task with a triplet ranking loss. NetVLAD considers the generated $H \times W \times D$ feature maps as a set of $N(H \times W) \times D$ local descriptors where N is the number of local descriptors and D is the dimension. Latter, a soft clustering is used to store the residual information contained in the descriptors to form $K \times D$ final descriptors denoted as V where K is the number of cluster centers. $V(j,k)$ can be expressed as:

$$V(j,k) = \sum_{i=1}^{N} a_k(x_i)(x_i(j) - c_k(j)), \tag{2.2}$$

where $j \in \{1,\ldots,D\}$ is the j-th dimension of a descriptor $\{x_i\}$, $k \in \{1,\ldots,K\}$ is the k-th cluster center, and $a_k(x_i)$ is the soft assignment of the descriptor x_i to k-th cluster center c_k. In Eq.2.1, A weakly supervised triplet ranking loss L_r is used to govern the learning process of descriptors that ensures the Euclidean distance between the query image and

the best potential positive images are smaller than the Euclidean distance between the query image and all the negative pairs (based on geo-tags).

$$L_r = \sum_j l\,(\min_i d_\theta^2(q, p_i^q) + m - d_\theta^2(q, n_j^q)), \tag{2.3}$$

where, q denotes the query image and p_i^q are potential positive images. $\min_i d_\theta^2(q, p_i^q)$ denotes the best matching pair with shortest distance d_θ. In turn, n_j^q are all negative image pairs and m is the distance margin to be maintained. The function l is the hinge loss which penalizes the pairs that violate the margin.

2.3.2. ATTENTION MODULE

The triplet network for image retrieval task produces feature maps with the dimension of $H \times W \times D$. The inserted attention module consists of a 1×1 convolutional layer with coefficients $\mathbf{w}_a \in \mathbb{R}^{D\times 1}$ and a softplus activation function. This convolutional layer will produce an attention score map H_a with spatial size $H \times W$, which could be interpreted as the weight $\{w_i\}$ for each descriptor $\{x_i\}$. [12] proposed an attention aware aggregation scheme A_1 as:

$$V(j,k)_{A_1} = \sum_{i=1}^{N} w_i a_k(x_i)(x_i(j) - c_k(j)), \tag{2.4}$$

where $w_i \in \mathbf{w}_a$. Note that the VLAD module first assigns the local descriptors $\{x_i\}$ to K cluster centers $\{c_k\}$, then computes the residuals of each descriptor $x_i - c_k$ to its cluster center and assigns the weight a_k of descriptor x_i to cluster c_k proportional to their proximity.

In Eq.2.4, the global descriptors (residuals) are weighed after clustering. However, the VLAD descriptor is very sensitive to cluster centers [39] since it defines the origin of coordinates system to a cluster. Under this circumstance, we propose to weigh the local descriptors according to attention scores before performing clustering. The soft-assignment term a_k is re-calculated based on the newly weighed descriptors. Our proposed aggregation scheme A_2 can be formulated as

$$V(j,k)_{A_2} = \sum_{i=1}^{N} w_i a_k(x_i w_i)(x_i(j) w_i - c_k(j)). \tag{2.5}$$

The difference between A_1 and A_2 is that A_1 assigns the attention scores after clustering the descriptors to multiple centers so the attention scores are only used to weigh the residuals but A_2 first uses the attention scores to filter out uninteresting regions in the individual local descriptors and then performs the same step as A_1. Considering that the reweighing of individual descriptors may remove information that are useful for global descriptor generation, we aggregate the two attention schemes linearly:

$$V(j,k)_{our} = V(j,k)_{A_1} + V(j,k)_{A_2}. \tag{2.6}$$

2.3.3. DOMAIN ADAPTATION MODULE

We use MK-MMD loss [38] with five Gaussian kernels of different bandwidths for unsupervised domain adaptation. The loss minimizes the distance between the expectation

of the kernel mappings $\phi(.)$ of the descriptors in the source domain x_i^s and the target domain x_i^t.

$$M(\mathscr{D}_s, \mathscr{D}_t) = \sum_{i}^{N} ||\mathbb{E}(\phi(x_i^s)) - \mathbb{E}(\phi(x_i^t))||_2. \tag{2.7}$$

The MK-MMD loss guides the CNN to learn a latent space where the two domains are not distinctive, i.e., the gap between the statistical means of these two domains are closed in the reproducing kernel Hilbert space (RKHS).

2.4. EXPERIMENT

2.4.1. DATASET

We construct a cross-domain dataset with two sources of data to evaluate our proposed method, namely the street view panorama images of Amsterdam city from the *Mapillary* dataset[40] and the *Beeldbank* dataset[41] containing historical images from Amsterdam city archives.

Mapillary40k is a subset of *Mapillary250k* dataset collected from the source domain. The source domain contains panoramic images with high resolution collected from the *Mapillary*, Amsterdam area. Each image is annotated with a geotag. The cylindrical panorama is converted to 6 cubmaps (all share the same geotag): 'top', 'down', 'left', 'right', 'front' and 'back' textures with 512 × 512 resolution. The 'top' and 'down' textures are discarded since they usually contain sky and the vehicle that carries the camera. 40k gallery images and 4k query images are collected in total which are then divided into two roughly equal parts for training and testing when tested for single-domain VPR task, each containing around 20k gallery images and 2k queries. The two sub-datasets are geographically disjoint.

Beeldbank, the target domain, contains historical images of Amsterdam with low resolution and random size (height and weight are around 100 pixels). This dataset not only depicts Amsterdam street view in the past but also contains outliers including people, sketches and indoor scenes.

Mapillary40k - Beeldbank dataset is introduced in this work for the cross-domain VPR task (Table 2.1). The cross-domain test set contains 104 labeled queries from the target domain and 2,469 gallery images from the source domain. In the cross-domain test set, each target query has around 10 corresponding matched images in source domain. 30k unlabeled *Beeldbank* images are used during training for domain adaptation.

2.4.2. SINGLE-DOMAIN AND CROSS-DOMAIN VPR TASKS

Single-domain VPR task ($S \rightarrow S$) In the single-domain VPR setup, we train the network with only weakly labeled source domain images. The network is tested on the test set of the same domain. *Mapillary40k* is used for this single-domain VPR experiment as shown in Tab.2.1. This is the common setting for VPR task as there is no domain mismatch between train and test data. We use single-domain VPR as a pilot experiment to evaluate the performance of the proposed model on conventional VPR task.

Cross-domain VPR task ($S \rightarrow T$) The domain discrepancy between train and test data makes the cross-domain VPR task more challenging. This is the core of our ex-

Dataset		Gallery	Query
Source	*Mapillary40k*-train	20,884	2,320
	single-domain-test	18,980 (**M**)	2,108 (**M**)
Target	*Beeldbank*-train	29,726	-
	cross-domain-test	2,469 (**M**)	104 (**B**)

Table 2.1: *Mapillary40k→Beeldbank dataset*, the source domain is *Mapillary40k* and the target domain is *Beeldbank* denoted by **M** and **B**, respectively. *Beeldbank-train* is only used for unsupervised domain adaptation. Cross-domain VPR requires matching query images from *Beeldbank* to gallery images from *Mapillary40k*.

periments in this work which aims at labeling the images from beeldbank dataset with correct geo-location. We train the MK-MMD layer with weakly labeled source data and unlabeled target data for the cross-domain VPR task. Labeled data with matching pairs from beeldbank and Mapilary dataset is only used for evaluation of the model. We use queries from the target domain to retrieve relevant gallery image(s) collected from the source domain.

We made the hypothesis that our attention module itself can improve the cross-domain VPR task to some extent without the MK-MMD loss compared to vanilla NetVLAD. An experiment was carried out to examine the function of the attention module later in Section 2.3.2. Further ablation study of the attention module and the MK-MMD loss will be presented in section 2.4.5 and 2.4.5.

Baseline work We compare our attention-aware framework with 'off-the-shelf' CNNs for both single-domain VPR and cross-domain VPR tasks. The baseline work used AlexNet pretrained on ImageNet cropped before $conv5$ as feature extractor. Features are then sub-sampled by either max pooling (f_{max}), average pooling (f_{avg}), vanilla VLAD pooling without attention(f_{VLAD}) and VLAD with attention-aware A_1 method ($f_{A1-VLAD}$) [12].

2.4.3. EVALUATION METRICS

We follow the standard place recognition evaluation metric in [2] where the query image is considered as correctly matched if at least one of the retrieved top N images is located within 25 meters away from the ground truth query location. The Recall@N evaluates the percentage of correctly localized queries at different N matching levels. For cross-domain place recognition, since the *Beeldbank* dataset contains labeled positive pairs, the Recall@N will be directly calculated using these labels.

2.4.4. IMPLEMENTATION DETAILS

The attention module starts with a ReLU activation, followed by a 1×1 convolutional layer and softplus activation to produce attention scores. In the VLAD layer, the number of cluster centers used is $K = 64$. *Mapillary* images were cropped with a random proportion to the original size between (0.3 1.0) for data augmentation before training.

We froze the layers before $conv4$ and fine-tuned the weights of all the other layers afterwards with the optimizer ADAM. We used the following hyperparameters: learning rate $lr = 1e\text{-}5$, batch size = 2 tuples (each tuple contains 24 images, including query, positive and negative pairs), epochs = 25. The hard negatives mining uses the same tech-

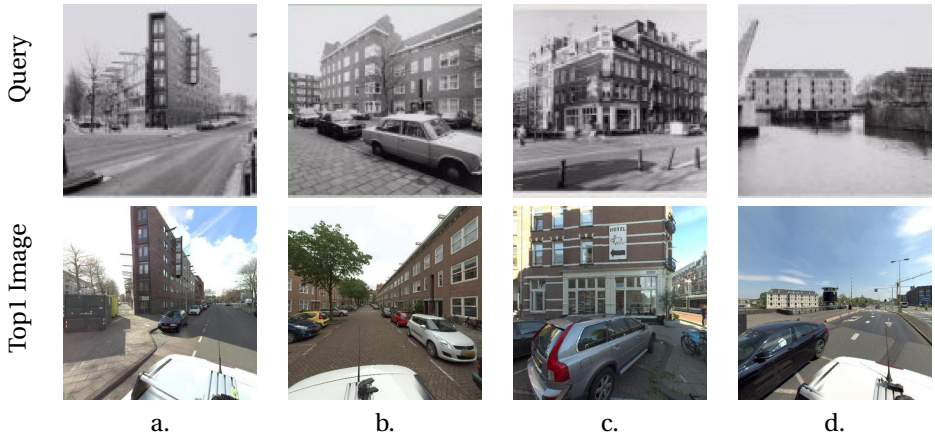


Figure 2.3: Correctly retrieved top1 image from our framework trained with unsupervised domain adaptation, (top) queries are from the *Beeldbank* dataset, (bottom) retrieved images are from the *Mapillary* dataset. Our model can retrieve images not only depicting a similarly scene (a.), but also images from a different perspective (c.) and images captured further away from the query (b., d.). (b.) is correctly retrieved by matching the features of the building like the window and the unique shape of the door on the right side.

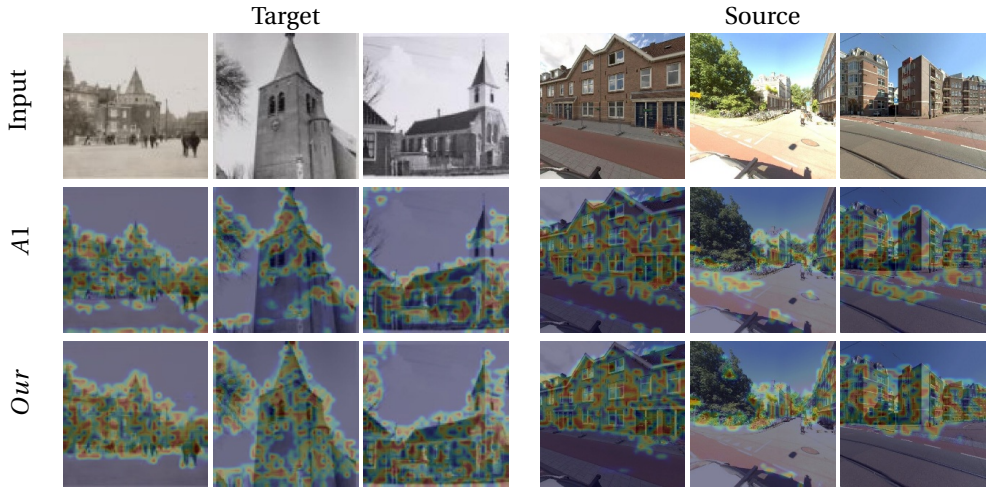


Figure 2.4: Visualization of attention score maps for source and target images. The top row shows the input images. The middle row is the heatmaps obtained by using [12], defined in Eq.2.4. The bottom row presents the results from our proposed method defined in Eq.2.6. It shows that our proposed attention module can generate accurate attention score maps with higher density on domain invariant objects for both source images and target images.

nique as NetVLAD[2]: it first caches all the training queries and gallery images for a time and then randomly selects 1000 negatives (image away from 25 meters). It keeps the top 10 hardest negatives from the cached gallery image features. The cache is updated every 1000 training queries.

We center cropped and reshaped all target *Beeldbank* images to 512 × 512 pixels in the cross-domain VPR experiment. The MK-MMD loss is calculated after $conv5$. The weight α in Eq.2.1 is 0.99. The margin m in Eq.2.3 is set as 0.1.

	$S \rightarrow T$				$S \rightarrow S$			
	R@1	R@5	R@10	R@20	R@1	R@5	R@10	R@20
$f_{max}{}^{+}$	0.0096	**0.0577**	0.0769	0.1058	0.6347	0.8226	0.8800	0.9203
$f_{avg}{}^{+}$	0.0000	0.0096	0.0481	0.0769	0.7884	0.9284	0.9535	0.9730
f_{max}	0.0000	0.0000	0.0577	0.1250	0.7410	0.9108	0.9431	0.9639
f_{avg}	0.0096	0.0192	0.0481	0.0577	0.7984	0.9269	0.9564	0.9725
f_{VLAD}	0.0096	0.0192	0.0192	0.0577	0.8843	0.9687	0.9782	0.9853
$f_{A1\text{-}VLAD}$	0.0096	0.0481	0.1058	0.1538	0.8819	0.9649	0.9801	0.9877
$f_{our\text{-}VLAD}$	**0.0192**	**0.0577**	**0.1154**	**0.2019**	**0.9132**	**0.9753**	**0.9815**	**0.9900**

Table 2.2: The $^{+}$ denotes that the 'off-the shelf' model is pretrained on ImageNet[42] for classification task. The others are trained on *Mapillary40k* for place recognition from scratch, and directly tested on the cross-domain dataset.

2.4.5. RESULTS

This section presents the results of the experiments with a detailed ablation study for the attention module (Section 2.4.5) and the domain adaptation module (Section 2.4.5) separately on both single-domain and cross-domain VPR tasks. Visual inspection of retrieval results and attention heatmaps are shown in Fig.2.3 and Fig.2.4.

ATTENTION MODULE

To evaluate the performance of our attention aggregation module on both single and cross-domain VPR tasks, we first trained the model on the source domain (*Mapillary40k*) and directly tested it on the source test set and the target test set without MK-MMD loss. Tab.2.2 shows the retrieval results where our attention aggregation method consistently outperforms the model without attention on both $S \rightarrow T$ and $S \rightarrow S$ tasks. A possible explanation could be that the VLAD descriptors are easily affected by the irrelevant objects. By not focusing on representative details that describe unique features of each building, it may retrieve an image that has a similar road or sky etc.

To inspect whether our attention module can produce reasonable attention scores for each descriptor, we visualize the attention maps of different attention-aware schemes in Fig.2.4. Our attention aggregation method generates heatmaps with higher densities on representative features and better robustness against irrelevant objects. Most attention is assigned to the architectures and less attention is assigned to non representative regions such as road and sky as expected. Note that in Tab.2.2, the performance of $f_{A1\text{-}VLAD}$ is worse than f_{VLAD} and $f_{our\text{-}VLAD}$ achieves the best results. We conclude that an insufficient attention map will deteriorate the performance.

DOMAIN ADAPTATION MODULE

The additional domain adaptation loss (MK-MMD) is added to our model and all baseline works in this section. The MK-MMD loss is adopted to further minimize the domain

discrepancy in this experiment. We applied it on the vanilla NetVLAD (f_{VLAD}-DA), A_1 attention model ($f_{A1\text{-}VLAD\text{-}DA}$) and our attention aggregation model ($f_{our-VLAD}$-DA). The performance of different models with and without MK-MMD loss are examined on both source and target test test. The results are visualized at different recall rates in Fig.2.5.

When trained with the MK-MMD loss for the $S \rightarrow T$ cross-domain VPR task, both f_{VLAD}-DA and $f_{our\text{-}VLAD}$-DA benefit from domain adaptation, while no significant improvement of $f_{A1\text{-}VLAD\text{-}DA}$ is observed. Detailed results are presented in Tab.2.3.

	with DA				without DA			
	R@1	R@5	R@10	R@20	R@1	R@5	R@10	R@20
f_{VLAD}	0.0096	0.0481	0.0769	0.1635	0.0096	0.0192	0.0192	0.0577
$f_{A1\text{-}VLAD}$	0.0096	0.0769	0.1058	0.1442	0.0096	0.0481	0.1058	0.1538
$f_{our\text{-}VLAD}$	**0.0577**	**0.1346**	**0.1731**	**0.2788**	**0.0192**	**0.0577**	**0.1154**	**0.2019**

Table 2.3: Comparison of different models' performance on the cross-domain $S \rightarrow T$ VPR task under two conditions: with or without domain adaptation using the MK-MMD loss. DA stands for domain adaptation using MK-MMD loss.

In addition, we also examined the performance of the model trained for the cross-domain VPR task on the source domain due to the reason that extra data from the target doamin may also help with retrieval in source domain if the model is robust to the outliers in the *Beeldbank* dataset. f_{VLAD}-DA does not show much power in the original source domain compared to f_{VLAD}. The retrieval accuracy of $f_{A1\text{-}VLAD\text{-}DA}$ in the source domain decreases after domain adaptation. Our proposed model gets better retrieval result even on the source domain as shown in Fig.2.5. This experiment proves that the domain specific features and outliers are reduced while more domain invariant features are captured by our proposed attention aggregation model which further facilitates the domain adaptation procedure.

Overall, we show that our attention aggregation model can achieve more accurate retrieval results on both single-domain $S \rightarrow S$ and cross-domain $S \rightarrow T$ VPR tasks even without domain adaptation and it can further facilitate unsupervised domain adaptation to achieve better performance on both source and target test sets.

2.5. DISCUSSION

Usually we assume that the training data and test data are sampled from an identical distribution which is violated in our cross-domain setting. We designed an attention-aware adaptive network to tackle the existing distribution shift. The results indicate that both the attention and adaptation modules contribute to the accurate retrieval of visual information. We speculate that the attention module mainly helps with focusing on domain invariant objects and the domain adaptation module aligns the depiction styles between the two different domains. Our dual experiments on both conventional and cross-domain VPR tasks admit the difficulty of learning age-invariant features when there is no cross-domain pairing labels available for directly training CNNs.

Besides the large domain shift, our *Beeldbank* target dataset contains various classes

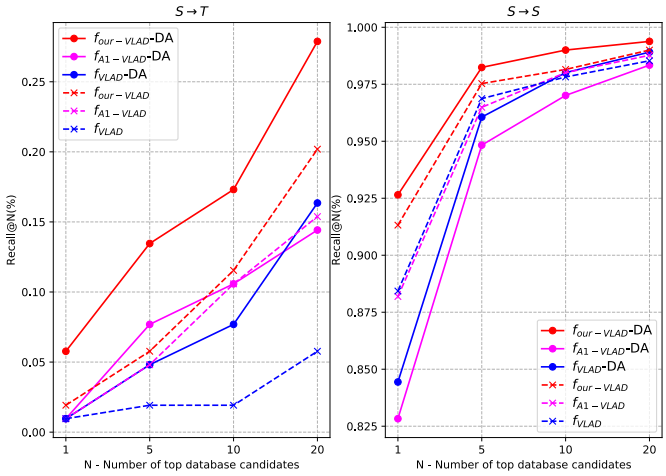


Figure 2.5: Comparison of the models trained with or without the MK-MMD loss on both single-domain ($S \rightarrow S$) and cross-domain ($S \rightarrow T$) tasks. DA denotes that the MK-MMD loss is added during training

of images like people, indoor scenes, sketches and ground plans of buildings. These outliers are not contained in source dataset *Mapillary40k* rendering the task more difficult. Domain adaptation with more classes or outliers in the target domain compared to the source domain can be considered as open-set domain adaptation problem [43]–[46]. Some other works refer to this as outlier detection problem [47], [48]. We speculate that the attention module can filter out the outliers by weighing them less with the heatmaps.

2.6. CONCLUSION

We proposed a specially-designed CNN for automatic annotation of historical images with their location. This is helpful specifically for museum curators and historians to retrieve the location information of a historical urban scene or architecture. This task is more challenging than single-domain (conventional) location retrieval due to the domain discrepancy caused by the large time lag between depicted scenes. A cross-domain dataset is collected accordingly with *Mapillary40k* used as source domain and *Beedldbank*, as target domain. To tackle this challenge, an attention aggregation module with a domain adaptation layer is designed, the performance of which is demonstrated by detailed experiments and ablation studies. Our attention aggregation model achieves state of the art results on both single and cross-domain VPR tasks by focusing more on domain invariant objects. It can be further combined with an extra domain adaptation module using the MK-MMD loss to achieve higher retrieval accuracy not only on the target domain but also on the source domain. Moreover, we believe our methods can achieve promising results on open-set domain adaptation tasks where unseen classes or outliers are not involved during training.

BIBLIOGRAPHY

[1] Z. Wang, J. Li, S. Khademi, and J. van Gemert, "Attention-aware age-agnostic visual place recognition," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019.

[2] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "Netvlad: Cnn architecture for weakly supervised place recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 6, pp. 1437–1451, Jun. 2018, ISSN: 2160-9292. DOI: 10.1109/tpami.2017.2711011. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.2017.2711011.

[3] Z. Chen, F. Maffra, I. Sa, and M. Chli, "Only look once, mining distinctive landmarks from convnet for visual place recognition," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2017, pp. 9–16.

[4] A. Iscen, G. Tolias, Y. Avrithis, T. Furon, and O. Chum, "Panorama to panorama matching for location recognition," *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval - ICMR '17*, 2017. DOI: 10.1145/3078971.3079033. [Online]. Available: http://dx.doi.org/10.1145/3078971.3079033.

[5] J. Knopp, J. Sivic, and T. Pajdla, "Avoiding confusing features in place recognition," in *European Conference on Computer Vision*, Springer, 2010, pp. 748–761.

[6] M. Lopez-Antequera, R. Gomez-Ojeda, N. Petkov, and J. Gonzalez-Jimenez, "Appearance-invariant place recognition by discriminatively training a convolutional neural network," *Pattern Recognition Letters*, vol. 92, pp. 89–95, 2017.

[7] A. Torii, J. Sivic, T. Pajdla, and M. Okutomi, "Visual place recognition with repetitive structures," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 883–890.

[8] Y. Zhu, J. Wang, L. Xie, and L. Zheng, "Attention-based pyramid aggregation network for visual place recognition," *2018 ACM Multimedia Conference on Multimedia Conference - MM '18*, 2018. DOI: 10.1145/3240508.3240525. [Online]. Available: http://dx.doi.org/10.1145/3240508.3240525.

[9] X. Shi, S. Khademi, and J. van Gemert, *Deep visual city recognition visualization*, 2019. arXiv: 1905.01932 [cs.CV].

[10] S. Chopra, R. Hadsell, Y. LeCun, *et al.*, "Learning a similarity metric discriminatively, with application to face verification," in *CVPR (1)*, 2005, pp. 539–546.

[11] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *International Workshop on Similarity-Based Pattern Recognition*, Springer, 2015, pp. 84–92.

[12] K. K. Nakka and M. Salzmann, "Deep attentional structured representation learning for visual recognition," *arXiv preprint arXiv:1805.05389*, 2018.

[13] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004, ISSN: 1573-1405. DOI: 10.1023/B:VISI.0000029664.99615.94. [Online]. Available: https://doi.org/10.1023/B:VISI.0000029664.99615.94.

[14] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *European conference on computer vision*, Springer, 2006, pp. 404–417.

[15] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *European conference on computer vision*, Springer, 2010, pp. 143–156.

[16] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, IEEE, 2010, pp. 3304–3311.

[17] A. L. Majdik, Y. Albers-Schoenberg, and D. Scaramuzza, "Mav urban localization from google street view data," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, IEEE, 2013, pp. 3979–3986.

[18] Z. Chen, A. Jacobson, N. Sünderhauf, *et al.*, "Deep learning features at scale for visual place recognition," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2017, pp. 3223–3230.

[19] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han, "Large-scale image retrieval with attentive deep local features," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3456–3465.

[20] P.-E. Sarlin, C. Cadena, R. Siegwart, and M. Dymczyk, "From coarse to fine: Robust hierarchical localization at large scale," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 716–12 725.

[21] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky, "Neural codes for image retrieval," *Lecture Notes in Computer Science*, pp. 584–599, 2014, ISSN: 1611-3349. DOI: 10.1007/978-3-319-10590-1_38. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-10590-1_38.

[22] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua, "Lift: Learned invariant feature transform," *Lecture Notes in Computer Science*, pp. 467–483, 2016, ISSN: 1611-3349. DOI: 10.1007/978-3-319-46466-4_28. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-46466-4_28.

[23] G. Tolias, R. Sicre, and H. Jégou, *Particular object retrieval with integral max-pooling of cnn activations*, 2015. arXiv: 1511.05879 [cs.CV].

[24] A. Gordo, J. Almazán, J. Revaud, and D. Larlus, "Deep image retrieval: Learning global representations for image search," in *European Conference on Computer Vision*, Springer, 2016, pp. 241–257.

[25] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Advances in neural information processing systems*, 2015, pp. 577–585.

[26] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[27] Y. Gu, C. Li, and J. Xie, *Attention-aware generalized mean pooling for image retrieval*, 2018. arXiv: 1811.00202 [cs.CV].

[28] X. Long, C. Gan, G. de Melo, J. Wu, X. Liu, and S. Wen, "Attention clusters: Purely attention based local feature integration for video classification," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2018. DOI: 10.1109/cvpr.2018.00817. [Online]. Available: http://dx.doi.org/10.1109/cvpr.2018.00817.

[29] Z. Seymour, K. Sikka, H.-P. Chiu, S. Samarasekera, and R. Kumar, *Semantically-aware attentive neural embeddings for image-based visual localization*, 2018. arXiv: 1812.03402 [cs.CV].

[30] J. Song, Q. Yu, Y.-Z. Song, T. Xiang, and T. M. Hospedales, "Deep spatial-semantic attention for fine-grained sketch-based image retrieval," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 5551–5560.

[31] T. Xiao, Y. Xu, K. Yang, J. Zhang, Y. Peng, and Z. Zhang, "The application of two-level attention models in deep convolutional neural network for fine-grained image classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 842–850.

[32] W. Kim, B. Goyal, K. Chawla, J. Lee, and K. Kwon, "Attention-based ensemble for deep metric learning," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 736–751.

[33] G. Kang, L. Zheng, Y. Yan, and Y. Yang, "Deep adversarial attention alignment for unsupervised domain adaptation: The benefit of target expectation maximization," *Lecture Notes in Computer Science*, pp. 420–436, 2018, ISSN: 1611-3349. DOI: 10.1007/978-3-030-01252-6_25. [Online]. Available: http://dx.doi.org/10.1007/978-3-030-01252-6_25.

[34] X. Wang, L. Li, W. Ye, M. Long, and J. Wang, "Transferable attention for domain adaptation," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2019.

[35] M. Long, Y. Cao, J. Wang, and M. I. Jordan, "Learning transferable features with deep adaptation networks," *arXiv preprint arXiv:1502.02791*, 2015.

[36] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," *arXiv preprint arXiv:1412.3474*, 2014.

[37] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola, "Integrating structured biological data by kernel maximum mean discrepancy," *Bioinformatics*, vol. 22, no. 14, e49–e57, 2006.

[38] A. Gretton, D. Sejdinovic, H. Strathmann, *et al.*, "Optimal kernel choice for large-scale two-sample tests," in *Advances in neural information processing systems*, 2012, pp. 1205–1213.

[39] R. Arandjelovic and A. Zisserman, "All about vlad," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2013, pp. 1578–1585.

[40] "Https://www.mapillary.com." ().

[41] "Https://beeldbank.amsterdam.nl/beeldbank." ().

[42] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, Ieee, 2009, pp. 248–255.

[43] M. Baktashmotlagh, M. Faraki, and T. Drummond, "Open-set domain adaptation," 2019.

[44] A. Bendale and T. E. Boult, "Towards open set deep networks," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016. DOI: 10.1109/cvpr.2016.173. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2016.173.

[45] P. Oza and V. M. Patel, *Deep cnn-based multi-task learning for open-set recognition*, 2019. arXiv: 1903.03161 [cs.CV].

[46] K. Saito, S. Yamamoto, Y. Ushiku, and T. Harada, "Open set domain adaptation by backpropagation," *Lecture Notes in Computer Science*, pp. 156–171, 2018, ISSN: 1611-3349. DOI: 10.1007/978-3-030-01228-1_10. [Online]. Available: http://dx.doi.org/10.1007/978-3-030-01228-1_10.

[47] M. Yamaguchi, Y. Koizumi, and N. Harada, "Adaflow: Domain-adaptive density estimator with application to anomaly detection and unpaired cross-domain translation," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2019, pp. 3647–3651.

[48] L. Luo, L. Chen, S. Hu, *et al.*, "Discriminative label consistent domain adaptation," *arXiv preprint arXiv:1802.08077*, 2018.

3

DOMAIN GENERALIZATION

In domain generalization, multiple labeled non-independent and non-identically distributed source domains are available during training while neither the data nor the labels of target domains are. Currently, learning so-called domain invariant representations *(DIRs) is the prevalent approach to domain generalization. In this work, we define DIRs employed by existing works in probabilistic terms and show that by learning DIRs, overly strict requirements are imposed concerning the invariance. Particularly, DIRs aim to perfectly align representations of different domains, i.e. their input distributions. This is, however, not necessary for good generalization to a target domain and may even dispose of valuable classification information. We propose to learn so-called* hypothesis invariant representations *(HIRs), which relax the invariance assumptions by merely aligning posteriors, instead of aligning representations. We report experimental results on public domain generalization datasets to show that learning HIRs is more effective than learning DIRs. In fact, our approach can even compete with approaches using prior knowledge about domains.*

This work has been published as:
Z. Wang, M. Loog, J. van Gemert, "Respecting Domain Relations: Hypothesis Invariance for Domain Generalization", International Conference on Pattern Recognition, 2020 [1].

3.1. INTRODUCTION

A standard assumption for many machine learning algorithms is that training and test data are *independent and identically distributed* (i.i.d.). In practice, however, training data may come from one source domain, while test data may come from another differently distributed target domain; violating the i.i.d. assumption [2]–[5]. Domain generalization is the setting where we have access to labeled data from multiple source domains during training while neither the data nor the labels of the target domain are available [6]–[9]. The absence of target data makes it impossible to infer distributional shifts between target and source domains, which significantly complicates the generalization to the target domain.

In some cases, strong prior knowledge about the relation between different domains that can be exploited in the building of a target classifier is available. A well-known example is the artificially created rotated MNIST images [10], where every domain—as the name eludes to—is a version of MNIST with all images rotated over a specific angle. To make domain generalization broadly applicable, however, one cannot rely on such very specific prior information, which in most realistic cases, will simply not be available. Approaches that indeed do not rely on such prior knowledge often draw inspiration from the field of domain *adaptation* [5], [11] where learning domain invariant representations is prevalent. In domain adaptation (rather than domain generalization as considered in this work), the availability of input target data is assumed, which enables one to directly relate this input distribution to those of the various source domains.

Approaches that learn so-called domain invariant representations (DIRs) for domain adaptation inspire domain generalization. DIRs aim to remove those parts of the representation that are domain specific in an attempt to generalize better to the target domain [6], [8], [10], [12], [13]. In general supervised learning, to achieve invariance of representations, minimal information of the input data should be kept, which refers to the mutual information between the learned representations and the input [14]. This, however, can compromise the sufficiency of representations for classification tasks if the learned invariance discards too much information. We transfer the definition for sufficiency of representations [14] to the setting of domain generalization and define the invariance for domain generalization. We show that learning domain invariant representations is too strict for the invariance because DIRs force the source domains to be at the same location in the representation space, which we will show is unnecessary for generalizing to the target domain. So by learning DIRs, the relation between different domain distributions can be distorted. However, this relation between domains is useful in the way that it can be used to infer the target domain.

In this work, we propose to learn hypothesis invariant representations (HIRs) to relax the invariance requirement of representation learning where DIRs align data sample representations between domains, our proposed HIRs merely align classifier predictions between domains. If there is any useful relation between domains, like the rotation in rotated MNIST, learning hypothesis invariant representations differs from DIRs in that HIRs can preserve the relative location of domains in the representation space without having prior domain knowledge. As it turns out, HIRs can even compete with approaches using prior knowledge. We demonstrate that hypothesis invariant representations can solve prior shifts as well. This setting is usually not considered by domain

invariant representations. Moreover, when the network is trained on augmented data [15], we show that learning hypothesis invariant representations can improve the performance over the baseline where various types of corruptions are aggregated and trained only with a classification loss.

All in all, this work makes the following contributions:

- We introduce the notions of sufficiency and invariance of representations to domain generalization.
- We present a probabilistic formulation to categorize and evaluate DIRs in terms of sufficiency and invariance.
- We introduce hypothesis invariant representations (HIRs). HIRs relax the invariance demands in DIRs by allowing to preserve useful relations in representation space while aligning predictions instead of representations.
- We compare HIRs to DIRs and against using prior knowledge about the domain distributions.

3.2. RELATED WORK

A similar setting to domain generalization is domain adaptation [5], [11], where both the data and label of the source domain are available while only the data from the target domain is accessible during training. When multiple labeled source domains and unlabeled target domains are available, it is referred to as multi-source domain adaptation [11], [16]–[19]. For both domain adaptation and domain generalization, the challenge is the distribution shift between source domains and target domains. Three types of distribution shifts are usually involved: covariate shift [20], concept drift [21], and prior shift [22]. We refer the reader to the study in [2] for a comprehensive treatment of these three shifts. In this work, both covariate shift and prior shift are considered.

Some domain generalization approaches can exploit strong prior knowledge of domains to infer the target domain distribution. LG [23] assumes a continuous representation space for domains with a specific order and applies perturbations on training domains to generalize to the unseen domain. DIVA [24] designs a generative model to decompose the representations of domain, class and variations. Due to the generative function of the network, it is possible to simulate unseen domains, especially if the domain is generated in an order. The disadvantage of these approaches is that the prior knowledge of domains is not always available. HIR learning does not require prior knowledge of domains.

When prior knowledge is not available, several approaches draw inspiration from domain adaptation to learn domain invariant representations for domain generalization. Guided by theoretical proof [25]–[27], learning domain invariant representations for domain adaptation by aligning distributions of source domains and target domains is prevalent [28]–[31]. KL divergence [32] and JS divergence [33] are used to measure the divergence between distributions. In this work, we formulate domain invariant representation in a probabilistic setting and extend the discussion of invariance and sufficiency [14] into domain generalization. We show that there is a trade-off between sufficiency and invariance learning, which aims to keep minimal information of domains.

Learning a domain invariant representation (DIR) for domain generalization can be achieved with kernel machines, e.g., DICA [34], MDA [35], SCA [6] and deep learning [8], [10], [12], [36]. DA [28] uses a gradient reverse layer for the domain classifier prevent the network from distinguishing different domains. HEX [37] proposes to separate the domain specific representations of different domains from the domain invariant representations of all possible domains. D-MTAE [10] designs a multi-task denoising autoencoder to reconstruct each domain with the goal to learn DIRs that are robust to noise. CIDDG [38] designs a new architecture to solve the problem of prior shift in domain generalization. MMD-AAE [36] applies *Maximum Mean Discrepancy (MMD)* [39] loss in the latent space of an encoder-decoder network to align the representations of different domains. CCSA [8] aligns the representations of different domains by minimizing the Euclidean distance in the feature space. DBADG [9] introduces a low-rank constraint to the weights of the network to not learn domain specific information. Different from learning DIRs, our hypothesis invariant representation learning does not strictly align domains in the latent space but rather keeps the relative positions between domains.

Meta-learning for domain generalization introduces a way to train a more robust model. MLDG [40] introduces meta-learning into the training procedure by treating part of the training data as from the target domain to learn a model that can generalize well to the unseen domain. Epi-FCR [41] proposes to learn a domain agnostic model by shuffling the feature extractors of different domains. We also compare our approach to meta-learning.

3.3. METHOD

We first give a probabilistic analysis of domain generalization. The analysis details sufficiency and invariance to show the trade-off between DIR learning and sufficiency of representations. Existing DIR approaches can be categorized in two conditions: class-agnostic and class-conditional. We formulate probabilistic definitions for DIRs under these two conditions and define hypothesis invariant representations to compare DIRs and HIRs. We show that (1) learning HIRs is less strict than learning DIRs; (2) HIRs can tackle prior shift and (3) a loss to learn HIRs.

3.3.1. PRELIMINARIES

Inspired by [42] and [43], we examine the relationship between distributions of domains and classes. We formalize domain generalization in the setting of classification. The label space and the domain space are denoted as $\mathscr{Y}$ and $\mathscr{D}$ respectively where class label Y and domain D are sampled from distributions P_Y and P_D. Seen source domains $D_s \in \mathscr{D}$ and unseen target domains $D_t \in \mathscr{D}$ are all in the space $\mathscr{D}$. The distribution of X is in input space $\mathscr{X} = \mathbb{R}^d$ conditioned on $\mathscr{Y} \times \mathscr{D}$ and is denoted as $P_{X|Y,D}$. For simplicity, we consider discrete domains and labels.

We introduce a latent space $\mathscr{Z}$ for representations to facilitate the discussion about DIRs and HIRs later. We denote the mapping function from the input space to the latent space as a *representation function* $r : \mathscr{X} \to \mathscr{Z}$ and a *hypothesis function* from the latent space to the predicted labels $h : \mathscr{Z} \to \hat{\mathscr{Y}}$.

Prior shift is caused by the unbalance of classes across domains, $P_{Y|D} \neq P_Y$. If the

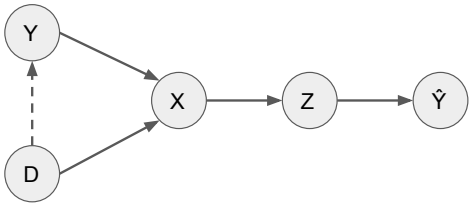


Figure 3.1: The graphical model shows the relationship from domain D, class Y to the input data X. A latent representation Z is learned to predict $\hat{Y}$. If Y is independent of D, $P_{Y|D} = P_Y$. Else if Y is dependent on D, $P_{Y|D} \neq P_Y$, which is denoted as an arrow pointing from D to Y. This dependency causes prior shift.

prior of the target domain $P_{Y|D_t}$ varies significantly from the prior of the source domain $P_{Y|D_s}$, this prior shift will lead to the failure of an approach which does not consider this type of distribution shift. To cover the possible prior shift in domain generalization, we set our graphical model in two different conditions: Y is independent of D or not. If the distribution of Y is independent of domains, $P_{Y|D} = P_Y$, it means there is no prior shift as shown in the graphical model in Fig. 3.1. Else if Y is dependent on D, $P_{Y|D} \neq P_Y$, prior shift exists.

3.3.2. SUFFICIENCY OF REPRESENTATIONS

According to the work of Achille and Soatto [14], a representation Z is considered as sufficient for the classification task if $P_{Y|X,Z} = P_{Y|Z}$ in a Markov chain $X \rightarrow Z \rightarrow Y$ for general supervised machine learning setting. That implies the mutual information between the input X and the label Y equals the mutual information between the latent representation Z and the label Y, $I(X;Y) = I(Z;Y)$ because the representation Z distills all the useful information for the classification task from the input X. Here we extend the definition of sufficiency into domain generalization. If we assume that the representation of each domain is sufficient, the posterior can be defined to satisfy:

$$\begin{aligned} &P_{Y|X,Z,D} = P_{Y|Z,D}, \\ &\forall D \in \mathscr{D}. \end{aligned} \tag{3.1}$$

3.3.3. INVARIANCE OF REPRESENTATIONS

A concept that recurs in many domain generalization works [34], [36], [43], is the so-called domain invariant representations which are supposed to be the essential representations of all domains and invariant to different domains, not only source domains but also the target domain. However, many approaches aim to learn domain invariant representations without clear definitions. Therefore we formulate two mostly adopted conditions of DIRs as below.

Class-agnostic DIRs refer to representations Z that are independent of the domain D, irrespective of labels Y. We formulate it as:

$$\begin{aligned} &P_{Z|D} = P_Z, \\ &\forall D \in \mathscr{D}. \end{aligned} \tag{3.2}$$

Table 3.1: Categorization of approaches. Existing approaches that learn domain invariant representations for domain generalization are sorted according to the definitions (3.2), (3.3).

Class-agnostic DIRs	Class-conditional DIRs	both
DA [28]	CCSA [8]	SCA [6]
MMD-AAE [36]	CIDDG [38]	MDA [35]
D-MTAE [10]		DICA [34]
HEX [37]		

Approaches that fall in this category align representations from multiple source domains without considering the labels.

Class-conditional DIRs are conditioned on both the domains D and the labels Y, so it is referred to as class-conditional DIR:

$$\begin{aligned} &P_{Z|Y,D} = P_{Z|Y}, \\ &\forall D \in \mathscr{D}. \end{aligned} \tag{3.3}$$

Different from class-agnostic DIRs, approaches that aim to learn class-conditional DIRs align representations of the same class but different domains. Some works adopt both class-agnostic and class-conditional domain invariant representation learning. We sort existing approaches in Table 3.1.

Learning domain invariant representations adds constraints on the representation invariance. However, if the algorithm focuses on learning DIRs, the useful domain specific information in the input X may be discarded so the sufficiency is compromised. If a method is forced to learn DIRs, a degenerate example of a trivial representation can be a vector full of zeros, which is invariant to all domains or hypotheses, but, such a vector is not sufficient for the classification task. Furthermore, by the definitions of DIRs, all domains are aligned with each other, and the relation between domains is no longer available. Thus, if the domains are sampled according to a specific order, this order information is subsequently removed by learning DIRs. Instead, we propose to learn HIRs to relax the constraints on the invariance so the relation between domains is retained.

Hypothesis invariant representations make domains invariant to the *prediction hypotheses* instead of DIRs that make the *feature representations* invariant to domains. Thus, the aim of HIRs is to align the predictions for representations from different domains, where a prediction label for two domains a, b is aligned with the hypothesis function: $\arg\max_Y P_{Y|Z,D=a} = \arg\max_Y P_{Y|Z,D=b}$. The requirement of DIRs for aligning feature representations from different domains is too strict, as only the prediction hypothesis needs to be domain invariant. HIRs should satisfy:

$$\begin{aligned} &\arg\max_Y P_{Y|Z,D} = \arg\max_Y P_{Y|Z}, \\ &\forall D \in \mathscr{D}. \end{aligned} \tag{3.4}$$

What matters for the final classification result is not DIRs, as in (3.2) or (3.3), but $\arg\max_Y P_{Y|Z,D}$, which is more relaxed because it can still be satisfied even if the representations differ across domains. See for example Fig. 3.2 where neither the distributions of representations $P_{Z|Y,D}$ nor the posterior $P_{Y|Z,D}$ is domain invariant for all the

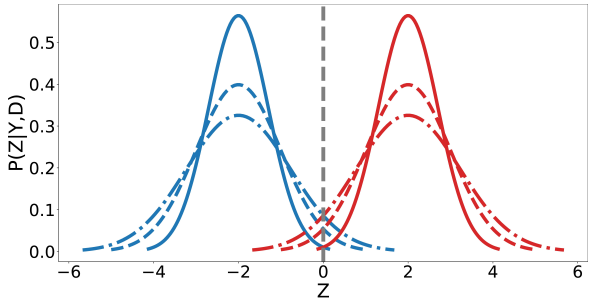


Figure 3.2: Representations of three domains. Colors represent the two classes of the three domains. This shows that learning DIRs is too strict compared to learning HIRs because representations Z of the three domains are hypothesis invariant as in (3.4) but not domain invariant as in (3.2) and (3.3). A threshold function on Z can serve as a hypothesis function h with low error for the binary classification task on all the three domains.

domains presented, but the hypothesis function h can give labels that satisfy (3.4). So for HIR, $P_{Z|Y,D=a} \neq P_{Z|Y,D=b}$ can hold.

Note that (3.4) in itself does not imply correct classification performance but only guarantees that the samples of all domains share the same prediction hypothesis function $h : \mathcal{Z} \to \hat{\mathcal{Y}}$, which can be completely different from the ground truth hypothesis function. If the hypothesis function h is inappropriate, then the samples could all be wrongly classified. The classification result is related to the sufficiency of the representation. So in practice, both the HIR learning and the classification learning are required.

3.3.4. HIRs AND DIRs COMPARISON

To further examine the relationship between HIRs, class-agnostic DIRs and class-conditional DIRs, we expand the posterior $P_{Y|Z,D}$ as:

$$P_{Y|Z,D} = \frac{P_{Z|Y,D} P_{Y|D}}{P_{Z|D}}. \tag{3.5}$$

If each of the three items on the right-hand side of (3.5) is independent of D, we will get class-agnostic DIRs as in (3.2), class-conditional DIRs as in (3.3), and domain invariant priors separately. If all the three items are independent of domain, then the posterior $P_{Y|Z,D}$ is domain invariant by construction, that is, $P_{Y|Z,D} = P_{Y|Z}$. So if there is no prior shift, $P_{Y|D} = P_Y$, the DIR is sufficient for the HIR. To the contrary, if $P_{Y|Z,D} = P_{Y|Z}$ holds, DIRs cannot be guaranteed. This expansion shows first, learning DIRs cannot tackle the prior shift when Y is dependent on D, and second, DIRs are sufficient but not necessary for HIRs. Therefore, learning HIRs is a more relaxed regularization on invariance and can align the priors of different domains.

3.3.5. ALIGNING HYPOTHESES: THE HIR LOSS

Learning domain invariant posteriors is an approximation to learn HIRs, as in (3.4), because it regularizes the invariance aspect of posteriors to generalize to the unseen target domain. Based on the analyses above on (3.5), learning HIRs by aligning the posteriors of

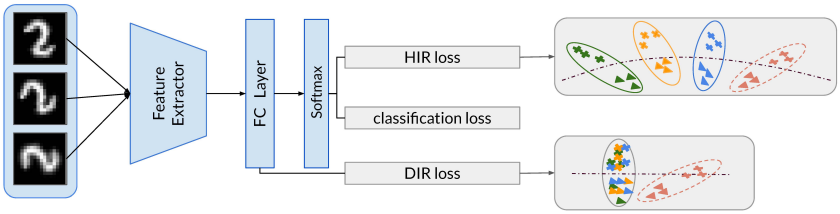


Figure 3.3: Comparison between HIR and DIR losses. The inputs are three domains of digits with different rotated angles. In the latent space, different colors represent domains and classes are marked by different shapes. The dash-line circles the unseen target domain. For learning DIRs, the loss is applied after the feature extractor which aims to align the representations of samples from different domains. HIR loss is applied after the Softmax layer to align the distributions of posteriors from different domains. With HIR loss, representations may keep the global structure without being strictly aligned.

domains $P_{Y|Z,D}$ can tackle the prior shift in practice and it is a more relaxed invariance regularization for representation learning. However, the distribution $P_{Y|Z,D}$ is usually not available. To avoid arbitrary density estimation of $P_{Y|Z,D}$ and guide the network to learn HIRs, we propose to align the domain-agnostic class-conditional posteriors of training data by minimizing the KL divergence as below:

$$L_h = \sum_{i=1}^{n} \sum_{j=i+1}^{n} \sum_{c=1}^{m} P(\hat{Y}_i|Z_i, Y_i = c) \cdot \log(\frac{P(\hat{Y}_i|Z_i, Y_i = c)}{P(\hat{Y}_j|Z_j, Y_j = c)}), \tag{3.6}$$

where n is the number of samples for class c, i and j are indices of samples. The KL divergence is computed for all m classes separately and summed up. The comparison between HIR loss and general DIR loss is presented in Fig. 3.3. We choose KL divergence to align the posteriors because it is a measurement for the difference between two distributions, but not only a distance between distributions. KL divergence is asymmetric and in this work we only calculate one direction between two samples. That is because there is no target distribution in our case. Instead, posteriors of both the two samples can change during training. So calculating the symmetric version of KL divergence or JS divergence is not very different from calculating the asymmetric KL divergence in our setting. The difference between the two approaches is presented in Fig. 3.4.

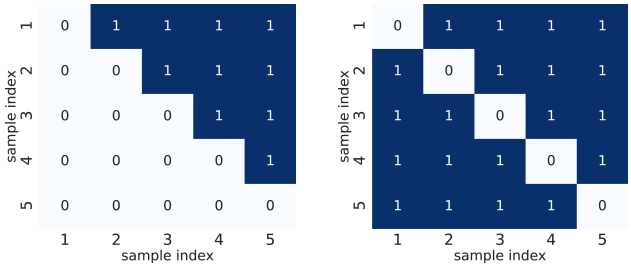
The sufficiency of representations is guaranteed by the cross-entropy loss which is computed as:

$$L_c = -\frac{1}{n} \sum_{i}^{n} \sum_{c}^{m} Y_i \cdot \log(P(\hat{Y}_i = c|Z_i)). \tag{3.7}$$

The trade-off between the sufficiency and the invariance learning is regulated by a coefficient α:

$$L = L_c + \alpha \cdot L_h, \tag{3.8}$$

where α is a tunable parameter during training. Different α values should be selected according to the scale of the HIR loss L_h to match the scale of L_c.



(a) KL asymmetric (b) KL symmetric

Figure 3.4: Asymmetric and symmetric KL divergence between 5 samples. The two matrices show all the possible matches among 5 samples. If one match is denoted as 1, then the KL divergence is computed, otherwise not. We use the asymmetric KL divergence in the HIR loss.

3.4. EXPERIMENTS

We show empirically that our approach respects the relations between domains without using prior knowledge of domains because it does not align the representations as strict as learning DIRs. We also show that our approach can do well on datasets which lack an obvious relation or global structure among domains. We compare our results with other existing approaches, especially the approaches that focus on learning DIRs and the approaches that can exploit prior knowledge of domain distributions. In addition, we also demonstrate the effectiveness of HIR learning on the data augmentation task. The hold-one-domain-out domain generalization setting is adopted for all experiments, that is, neither the label nor the data of the test domain is available during training. The trained network is applied on the unseen domain for evaluation without any adaptation or fine tuning.

3.4.1. DATASETS

We examine the results of learning HIRs on three datasets, (1) rotated MNIST dataset with clear prior knowledge about the global structure of domains, (2) VLCS where there is no order for domains and (3) tiny ImageNet-C which consists of 7 types of corruptions, where each corruption is a domain.

ROTATED MNIST

Rotated MNIST dataset consists of 6 domains with the original domain $\mathscr{M}_{0^\circ}$ and it rotated by 15°, 30°, 45°, 60° and 75°. Each domain has 10 classes of hand written digits from 0 to 9, and 100 images for each class. This dataset has a specific rotation order for domains so it is usually used to test approaches where prior domain information is used.

VLCS

VLCS dataset [44] has four domains, each domain is a different dataset collected under different backgrounds, namely PASCAL VOC2007 (V) [45], LabelMe (L) [46], Caltech-101 (C) [47] and SUN09 (S) [48] with 5 common classes, bird, car, chair, dog and person. To be consistent with other approaches [8], [10], [41], we also use DeCAF features in the

Table 3.2: Results on rotated MNIST dataset. Rotated MNIST is a dataset with a global structure for the domains, where the domains are rotated by a fixed angle. AGG is the baseline setting with only a classification loss without using HIR loss. We show that on the ordered dataset, approaches using prior knowledge perform the best. Moreover, our HIR learning can compete with these approaches without using the prior knowledge.

	Methods	$\mathcal{M}_{0^\circ}$	$\mathcal{M}_{15^\circ}$	$\mathcal{M}_{30^\circ}$	$\mathcal{M}_{45^\circ}$	$\mathcal{M}_{60^\circ}$	$\mathcal{M}_{75^\circ}$	Avg.
prior	LG	89.7	97.8	98.0	97.1	96.6	92.1	95.3
knowledge	DIVA	93.5	99.3	99.1	99.2	99.3	93.0	**97.2**
no	D-MTAE	82.5	96.3	93.4	78.6	94.2	80.5	87.5
prior	CCSA	84.6	95.6	94.6	82.9	94.8	82.1	89.1
knowledge	MMD-AAE	83.7	96.9	95.7	85.2	95.9	81.2	89.8
	DA	86.7	98.0	97.8	97.4	96.9	89.1	94.3
	HEX	90.1	98.9	98.9	98.8	98.3	90.0	95.8
	AGG	89.87	99.41	98.98	95.14	98.63	91.13	95.53
Ours	HIR	90.34 ±0.88	99.75 ±0.18	99.40 ±0.21	96.17 ±0.71	99.25 ±0.26	91.26 ±0.66	**96.03**

experiments. The domains in VLCS do not follow an obvious order, so the approaches using prior knowledge cannot be applied on this dataset.

TINY IMAGENET-C

The Tiny ImageNet dataset is a subset of ImageNet with 200 selected classes and 500 images per class. Tiny ImageNet-C has 7 domains, where each domain is one type of corruption of the original Tiny ImageNet dataset. The 7 types of corruptions, Gaussian noise, Impulse noise, JPEG compression, Defocus blur, Motion blur, Zoom blur and Glass blur are selected by us. We deployed the corruption methods from [15] and used the severest level 5 corruption. We designed this dataset to evaluate the effectiveness of HIR learning on data augmentation task.

3.4.2. RESULTS

ROTATED MNIST

As we expect that there is a global structure for the order of rotated angles, as shown in Fig. 3.3, the decision boundaries of domains in the middle of this global structure can be interpolated by the domains from both sides, e.g., $\mathcal{M}_{15^\circ}$ can be inferred from $\mathcal{M}_{0^\circ}$ and $\mathcal{M}_{30^\circ}$. For the same reason, domain $\mathcal{M}_{0^\circ}$ and $\mathcal{M}_{75^\circ}$ are significantly more difficult to be generalized to, compared to the other domains. These two domains are located at the two ends of this global structure of all the domains, so the decision boundary can only be inferred from all the domains at only one side of the global structure.

We adopt the same network architecture of CCSA [8], which has two convolutional layers with 32 kernels each and three fully connected layers. We report the average results of 10 repetitions for both the aggregation training setting (AGG) and the HIR setting in Table 3.2. For AGG, the network is trained with only (3.7) and no domain information is used as a baseline. For HIR the KL divergence is regularized as in (3.8). For HIR, a batch size of 250 is used, with 5 samples from each class and each domain. We use Adam for optimization with a learning rate of 1e-3. The coefficient α is set to be 1e-3. We can see that AGG with only classification loss can already give much better results compared to CCSA with the same architecture. After imposing the HIR loss, the performance can be further improved. We compare our HIR learning with approaches using

Table 3.3: Pair/unpair experiment results on rotated MNIST. The AGG setting uses only the classification loss and gives similar performance for both the paired and unpaired inputs. Our HIR learning is more effective on unpaired inputs for this dataset.

Domains	unpaired AGG	paired AGG	unpaired HIR	paired HIR
$\mathcal{M}_{0^\circ}$	45.83±2.67	44.41±2.86	79.99±3.78	57.30±4.05
$\mathcal{M}_{15^\circ}$	65.83 ± 3.08	66.33 ± 3.47	94.83 ± 4.14	70.44 ± 4.55
$\mathcal{M}_{30^\circ}$	71.30 ± 4.86	70.50 ± 4.56	94.32 ± 4.07	72.74 ± 3.65
$\mathcal{M}_{45^\circ}$	63.76 ± 3.94	64.02 ± 3.51	85.54 ± 5.34	63.63 ± 1.96
$\mathcal{M}_{60^\circ}$	60.37 ± 3.05	62.46 ± 4.94	89.62 ± 4.57	67.63 ± 4.91
$\mathcal{M}_{75^\circ}$	44.91 ± 2.65	44.46 ± 3.30	76.37 ± 5.64	53.51 ± 2.49
Avg.	58.67	58.70	86.78	64.21

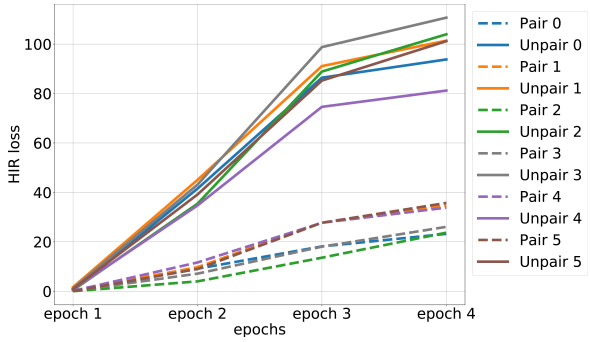


Figure 3.5: HIR losses of all 6 domains for the pair/unpair experiment on rotated MNIST dataset. The scale of the unpaired HIR loss is much larger than that of the paired inputs. So there is more room for HIR loss to contribute to the posterior alignment. For the paired inputs, the posteriors are similar within the pair so the HIR loss is low. Further regularizing HIR loss will lead the network to overfit to each pair of images.

prior knowledge and DIR learning without using prior knowledge. The results show that HIR learning can achieve better results than DIR learning and can compete with methods using prior knowledge, despite the fact that prior knowledge is unavailable in HIR learning.

The uniqueness of RMNIST dataset is that it contains paired images across domains which is the same image with different rotated angles. We investigate the influence of our HIR loss with paired and unpaired training batch. We set the batchsize to be 50 to contain one sample per domain per class. The samples from different domains are paired images in the paired setting and vice versa. The results are compared in Table 3.3. The baseline results of paired and unpaired settings are close while the HIR loss makes significant difference. This is because the posteriors of paired images are similar and further regularizing the HIR loss leads to overfit on each pair of images. The HIR losses of paired and unpaired experiments are visualized in Fig. 3.5 for each domain. HIR works better in the unpaired setting where the divergences between posteriors are larger.

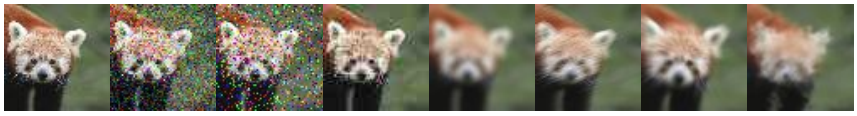
Table 3.4: Results on VLCS dataset. The domains in VLCS dataset do not follow a specific order or distribution, so prior knowledge cannot be used on this dataset. AGG is the baseline setting with only a classification loss without HIR loss. Our HIR learning performs better than the approaches that learn domain invariant representations.

Domains	D-MTAE	CIDDG	DBADG	MMD-AAE	MLDG	Epi-FCR	CCSA	**AGG**	**HIR**
V	63.90	64.38	69.99	67.70	67.7	67.1	67.10	65.4	69.10 ± 1.8
L	60.13	63.06	63.49	62.60	61.3	64.3	62.10	60.6	62.22 ± 1.7
C	89.05	88.83	93.63	94.40	94.4	94.1	92.30	93.1	95.39 ± 0.9
S	61.33	62.10	61.32	64.40	65.9	65.9	59.10	65.8	65.71 ± 1.6
Avg	68.60	69.59	72.11	72.28	72.3	72.9	70.15	71.2	**73.10**

VLCS

Unlike rotated MNIST, VLCS is consisted of four independent datasets where the global structure of domains is not obvious. This experiment further demonstrates the effectiveness of HIR loss on datasets without any order in the domains. The number of samples varies across domains and classes and we do stratified sampling. Samples of each training domain are split into 80 folds with balanced classes in each fold. One training batch is consisted of a fold from each domain. We use Adam with a learning rate of 1e-4. The coefficient α is 1e-6 to match the scale of the empirical loss.

For VLCS, we use the CCSA architecture [8] with two fully connected layers of dimension 1024 and 128. We adopt the same experiment setting as in CCSA where the dataset is randomly split into 0.7 for training and 0.3 for testing. Results are averaged across 20 repetitions. We initiate an individual random split for each repetition, which causes the relatively high standard deviations. The results of AGG and HIR are presented together with other existing approaches in Table 3.4.



(a) uncorrupted (b) Gauss. noise (c) Impulse noise (d) JPEG (e) Defocus blur (f) Motion blur (g) Zoom blur (h) Glass blur

Figure 3.6: Corruptions in Tiny ImageNet-C dataset. All images are corrupted at the highest severity [15].

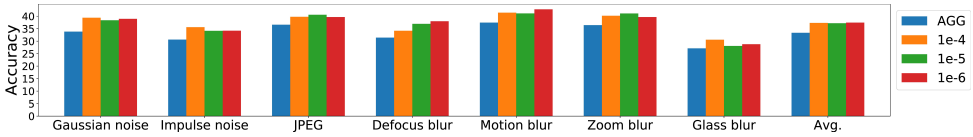


Figure 3.7: Results on Tiny ImageNet-C dataset. Different colors denote the values of α, which regulates the strength of HIR loss w.r.t. the classification loss. AGG is the baseline setting with only a classification loss without using HIR loss.

TINY IMAGENET-C

We show that data augmentation, especially when the augmented corruption types are divergent, fits well in a domain generalization setting. Our HIR loss can help with aligning the divergence between different corruptions.

Paired images of the 7 domains in Tiny ImageNet-C are visualized in Fig. 3.6 together with the uncorrupted image. We adopt ResNet50 pretrained on ImageNet for this experiment, so the uncorrupted images are not included as one domain in this setting. This dataset is challenging in the way that the corruptions are severe and both blurring and noise corruptions are presented. Training on one type of corruption cannot help but may deteriorate the performance on an unseen corruption.

We use the Adam optimizer with learning rate 1e-5. A small batch is consisted of one image and all its paired images from all the training domains and a large batch contains 20 shuffled small batches. The large batch is used as one batch during training. Unlike the rotated MNIST dataset, due to the large domain shifts in this dataset, even posteriors of paired images have large variation so the HIR loss of paired images is high enough to contribute to the posterior alignment. We show the impact of coefficient α for all the seven domains in Fig. 3.7. For domains with noise corruptions, larger α works better while smaller α is more suitable for blurring corruptions.

3.5. CONCLUSION

This work summarizes existing approaches for domain generalization in probabilistic expressions and shows that learning DIRs is too strict for representation learning so useful domain information is discarded. We proposed to learn HIRs instead of DIRs aiming to keep possible global structure of the domains without prior knowledge of domains, thus the target domain can be inferred from the relation between domains.

In our work, to avoid arbitrary density estimation of the posterior of each domain, we approximated it by aligning the posteriors of samples from each domain. Future work can explore how to reliably estimate the distribution of domain posteriors to further relax the invariance learning.

BIBLIOGRAPHY

[1] Z. Wang, M. Loog, and J. van Gemert, "Respecting domain relations: Hypothesis invariance for domain generalization," *arXiv preprint arXiv:2010.07591*, 2020.

[2] J. G. Moreno-Torres, T. Raeder, R. Alaiz-RodrıGuez, N. V. Chawla, and F. Herrera, "A unifying view on dataset shift in classification," *Patt.Recog.*, vol. 45, no. 1, pp. 521–530, 2012.

[3] J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, *Dataset shift in machine learning*. MIT Press, 2009.

[4] B. Zadrozny, "Learning and evaluating classifiers under sample selection bias," in *ICML*, 2004, p. 114.

[5] W. M. Kouw and M. Loog, "A review of domain adaptation without target labels," *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[6] M. Ghifary, D. Balduzzi, W. B. Kleijn, and M. Zhang, "Scatter component analysis: A unified framework for domain adaptation and domain generalization," *IEEE TPAMI*, vol. 39, no. 7, pp. 1414–1430, 2016.

[7] D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales, "Deeper, broader and artier domain generalization," in *ICCV*, 2017, pp. 5542–5550.

[8] S. Motiian, M. Piccirilli, D. A. Adjeroh, and G. Doretto, "Unified deep supervised domain adaptation and generalization," in *ICCV*, 2017, pp. 5715–5725.

[9] Z. Xu, W. Li, L. Niu, and D. Xu, "Exploiting low-rank structure from latent domains for domain generalization," in *ECCV*, Springer, 2014, pp. 628–643.

[10] M. Ghifary, W. Bastiaan Kleijn, M. Zhang, and D. Balduzzi, "Domain generalization for object recognition with multi-task autoencoders," in *ICCV*, 2015, pp. 2551–2559.

[11] H. Daumé III, "Frustratingly easy domain adaptation," *arXiv preprint arXiv:0907.1815*, 2009.

[12] Z. Ding and Y. Fu, "Deep domain generalization with structured low-rank constraint," *IEEE TIP*, vol. 27, no. 1, pp. 304–313, 2017.

[13] C. V. Dinh, R. P. Duin, I. Piqueras-Salazar, and M. Loog, "Fidos: A generalized fisher based feature extraction method for domain shift," *Patt.Recog.*, vol. 46, no. 9, pp. 2510–2518, 2013.

[14] A. Achille and S. Soatto, "Emergence of invariance and disentanglement in deep representations," *JMLR*, vol. 19, no. 1, pp. 1947–1980, 2018.

[15] D. Hendrycks and T. G. Dietterich, "Benchmarking neural network robustness to common corruptions and surface variations," *arXiv:1807.01697*, 2018.

[16] M. Dredze and K. Crammer, "Online methods for multi-domain learning and adaptation," in *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, 2008, pp. 689–697.

[17] J. Hoffman, B. Kulis, T. Darrell, and K. Saenko, "Discovering latent domains for multisource domain adaptation," in *ECCV*, Springer, 2012, pp. 702–715.

[18] Y. Mansour, M. Mohri, and A. Rostamizadeh, "Domain adaptation with multiple sources," in *NeurIPS*, 2009, pp. 1041–1048.

[19] K. Zhang, M. Gong, and B. Schölkopf, "Multi-source domain adaptation: A causal view," in *AAAI*, 2015.

[20] H. Shimodaira, "Improving predictive inference under covariate shift by weighting the log-likelihood function," *Journal of statistical planning and inference*, vol. 90, no. 2, pp. 227–244, 2000.

[21] G. Widmer and M. Kubat, "Learning in the presence of concept drift and hidden contexts," *Machine learning*, vol. 23, no. 1, pp. 69–101, 1996.

[22] G. I. Webb and K. M. Ting, "On the application of roc analysis to predict classification performance under varying class distributions," *Machine learning*, vol. 58, no. 1, pp. 25–32, 2005.

[23] S. Shankar, V. Piratla, S. Chakrabarti, S. Chaudhuri, P. Jyothi, and S. Sarawagi, "Generalizing across domains via cross-gradient training," *arXiv: 1804.10745*, 2018.

[24] M. Ilse, J. M. Tomczak, C. Louizos, and M. Welling, "Diva: Domain invariant variational autoencoders," *arXiv:1905.10427*, 2019.

[25] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, "Analysis of representations for domain adaptation," in *NeurIPS*, 2007, pp. 137–144.

[26] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Machine learning*, vol. 79, no. 1-2, pp. 151–175, 2010.

[27] S. Ben-David, T. Lu, T. Luu, and D. Pál, "Impossibility theorems for domain adaptation," in *International Conference on Artificial Intelligence and Statistics*, 2010, pp. 129–136.

[28] Y. Ganin, E. Ustinova, H. Ajakan, *et al.*, "Domain-adversarial training of neural networks," *JMLR*, vol. 17, no. 1, pp. 2096–2030, 2016.

[29] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars, "Unsupervised visual domain adaptation using subspace alignment," in *ICCV*, 2013, pp. 2960–2967.

[30] B. Bhushan Damodaran, B. Kellenberger, R. Flamary, D. Tuia, and N. Courty, "Deepjdot: Deep joint distribution optimal transport for unsupervised domain adaptation," in *ECCV*, 2018, pp. 447–463.

[31] N. Courty, R. Flamary, A. Habrard, and A. Rakotomamonjy, "Joint distribution optimal transportation for domain adaptation," in *NeurIPS*, 2017, pp. 3730–3739.

[32] D. Yu, K. Yao, H. Su, G. Li, and F. Seide, "Kl-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition," in *ICASSP*, IEEE, 2013, pp. 7893–7897.

[33] D. M. Endres and J. E. Schindelin, "A new metric for probability distributions," *IEEE IT*, vol. 49, no. 7, pp. 1858–1860, 2003.

[34] K. Muandet, D. Balduzzi, and B. Schölkopf, "Domain generalization via invariant feature representation," in *ICML*, 2013, pp. 10–18.

[35] S. Hu, K. Zhang, Z. Chen, and L. Chan, "Domain generalization via multidomain discriminant analysis," in *Uncertainty in artificial intelligence: proceedings of the... conference. Conference on Uncertainty in Artificial Intelligence*, NIH Public Access, vol. 35, 2019.

[36] H. Li, S. Jialin Pan, S. Wang, and A. C. Kot, "Domain generalization with adversarial feature learning," in *CVPR*, 2018, pp. 5400–5409.

[37] H. Wang, Z. He, Z. C. Lipton, and E. P. Xing, "Learning robust representations by projecting superficial statistics out," *arXiv:1903.06256*, 2019.

[38] Y. Li, X. Tian, M. Gong, *et al.*, "Deep domain generalization via conditional invariant adversarial networks," in *ECCV*, 2018, pp. 624–639.

[39] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola, "Integrating structured biological data by kernel maximum mean discrepancy," *Bioinformatics*, vol. 22, no. 14, e49–e57, 2006.

[40] D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales, "Learning to generalize: Meta-learning for domain generalization," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[41] D. Li, J. Zhang, Y. Yang, C. Liu, Y.-Z. Song, and T. M. Hospedales, "Episodic training for domain generalization," in *ICCV*, 2019, pp. 1446–1455.

[42] Q. Xie, Z. Dai, Y. Du, E. Hovy, and G. Neubig, "Controllable invariance through adversarial feature learning," in *NeurIPS*, 2017, pp. 585–596.

[43] K. Akuzawa, Y. Iwasawa, and Y. Matsuo, "Adversarial invariant feature learning with accuracy constraint for domain generalization," *arXiv:1904.12543*, 2019.

[44] C. Fang, Y. Xu, and D. N. Rockmore, "Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias," in *ICCV*, 2013, pp. 1657–1664.

[45] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *IJCV*, vol. 88, no. 2, pp. 303–338, 2010.

[46] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, "Labelme: A database and web-based tool for image annotation," *IJCV*, vol. 77, no. 1-3, pp. 157–173, 2008.

[47] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories," in *CVPR workshop*, IEEE, 2004, pp. 178–178.

[48] M. J. Choi, J. J. Lim, A. Torralba, and A. S. Willsky, "Exploiting hierarchical context on a large database of object categories," in *CVPR*, IEEE, 2010, pp. 129–136.

4

NON-I.I.D. VIDEO

*Parkinson's disease (*PD*) diagnosis is based on clinical criteria, i.e., bradykinesia, rest tremor, rigidity, etc. Assessment of the severity of PD symptoms with clinical rating scales, however, is subject to inter-rater variability. In this paper, we propose a deep learning based automatic PD diagnosis method using videos to assist the diagnosis in clinical practices. We deploy a 3D Convolutional Neural Network (*CNN*) as the baseline approach for the PD severity classification and show the effectiveness. Due to the lack of data in clinical field, we explore the possibility of transfer learning from non-medical dataset and show that PD severity classification can benefit from it. To bridge the domain discrepancy between medical and non-medical datasets, we let the network focus more on the subtle temporal visual cues, i.e., the frequency of tremors, by designing a Temporal Self-Attention (*TSA*) mechanism.*

*Seven tasks from the Movement Disorders Society - Unified PD rating scale (*MDS-UPDRS*) part III are investigated, which reveal the symptoms of bradykinesia and postural tremors. Furthermore, we propose a multi-domain learning method to predict the patient-level PD severity through task-assembling. We show the effectiveness of TSA and task-assembling method on our PD video dataset empirically. We achieve the best MCC of 0.55 on binary task-level and 0.39 on three-class patient-level classification.*

This work has been published as:
Z. Yin, V.J. Geraedts, Z. Wang, M.F. Contarino, H. Dibeklioglu, J. van Gemert, "Assessment of Parkinson's Disease Severity from Videos using Deep Architectures", IEEE Journal of Biomedical and Health Informatics, 2021 [1]. As a joint first author, I designed the TSA and task-assembling methods for the transfer learning.

4.1. INTRODUCTION

Parkinson's disease (PD) is a chronic, progressive neurological disorder, affecting over 10 million people around the world according to the American Parkinson Disease Association (APDA) [2]. Individuals with Parkinson's disease typically present with characteristic motor symptoms, including bradykinesia (i.e. slowness of movement), rigidity (stiffness), and rest tremor [3]. These symptoms are progressive over time, subsequently leading to an increase in their severity.

At present, the Movement Disorder Society - Unified Parkinson's Disease Rating Scale (MDS-UPDRS), containing four parts: I for non-motor experiences of daily living, II for motor experiences of daily living, III for motor examination and IV for motor complications, has been widely used as a validated tool to quantify PD severity [4], [5]. MDS-UPDRS is the revised and more comprehensive version of the original UPDRS [6] and they are highly correlated on the motor sections [7]. This study uses the MDS-UPDRS part III (MDS-UPDRS-III) as the measurement for analysis, which contains 18 tasks and 33 scores, with some tasks pertaining to either left or right extremities. Each task, tied to a symptom, has five responses linked to symptom-severity: 0-normal, 1-slight, 2-mild, 3-moderate, and 4-severe, providing consistency across tasks. The clinical scores are assessed by a single examinator, that is either a nurse specialized in Parkinson's Disease or a physician. Both have the certification to rate the MDS-UPDRS III. Collapsing all the scores to provide the patient with a composite total score is not recommended by [4] but can still be applicable given the minimal clinically important difference threshold values [8] and is often used in clinical practice to monitor disease progression. Although MDS-UPDRS-III is currently the gold standard to quantify the severity, it still has the potential to cause less reliable ratings due to the intrinsic inter-rater variability caused by the non-identical inter-rater protocols and inexperienced examiners [9], [10]. Besides, the presence of the specialist is mandatory when giving the rating decisions. These difficulties make the manual rating inefficient and urge for automatic quantification method. In this work, we propose a deep learning based PD severity quantification approach using videos. Fig. 4.1 shows the overall pipeline.

The goal of PD severity quantification is that, given an individual patient's video performing a specific task, the corresponding severity level can be predicted by the machine learning algorithm to assist ratings of examiners. As the task performed by the patient in the video is a kind of action, we naturally think of the human action recognition method to solve the identification of Parkinson's severity. Recently, many action recognition architectures [11]–[13] achieved promising performance on public human action datasets and one of the mostly used architecture is the inflated 3D CNN (I3D) [12], which is a 3D CNN with 3D kernels inflated from a 2D CNN with an additional temporal dimension. Therefore, we opt to use I3D as the base model for this work.

Due to the small size of our PD dataset, directly training I3D from scratch is inefficient and prone to overfitting; thus, we use transfer learning to pre-train the network on large datasets to make the training process more stable. However, public datasets we pre-train on have noticeable motion differences while the motion difference in our PD dataset is subtle. Such large domain discrepancy makes it difficult to transfer knowledge between domains, so we need a solution to focus on exploring the temporal motion changes. Besides, the video in our dataset is a repeating task with periodic actions, where

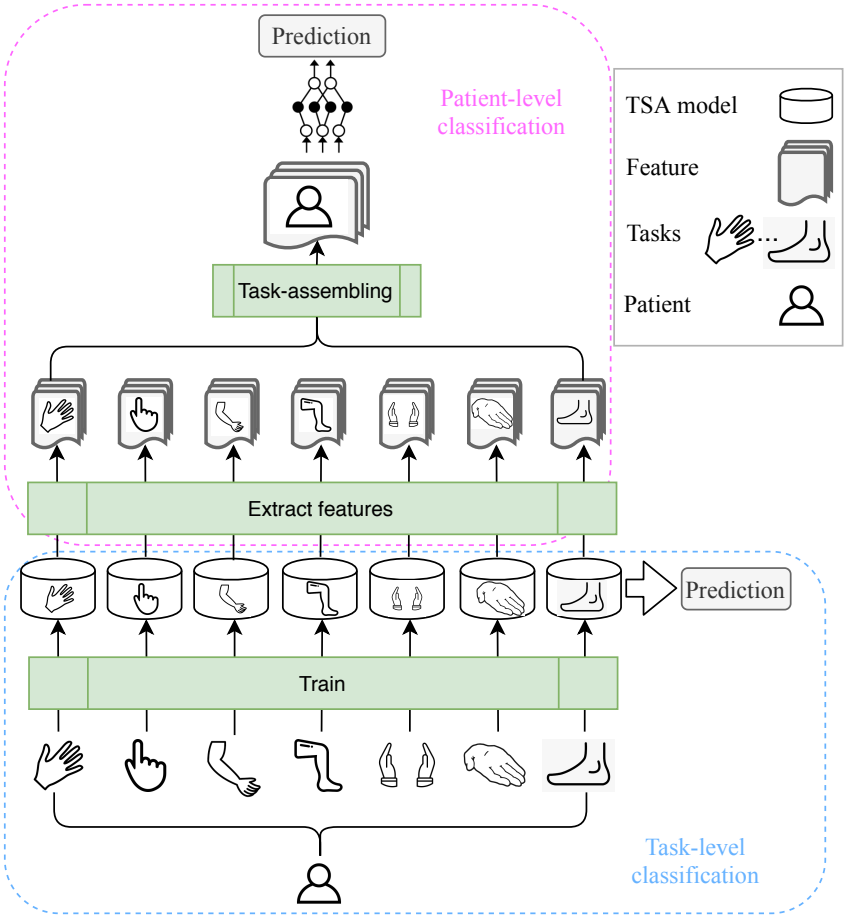


Figure 4.1: The flowchart of the automatic PD severity quantification. The task symbols from left to right denote task *finger tapping*, *hand movements*, *kinetic tremor*, *leg agility*, *postural tremor*, *pronation*, and *toe tapping*.

the model should learn the repeating frequency or the starting and ending point. Thus, we need another solution to assign different weights for the frames of the video. Additionally, as stated in [14], [15], not all frames are equally crucial for action recognition, so we propose to use temporal self-attention to assign the weights for frames as well as solve the domain discrepancy issue. The benefit is not only for such a repeating dataset but also for other datasets because it holds for other datasets as well that not all frames are equally important.

Once we can predict each task's severity, each patient will have a separate severity score for each task. However, it is more clinically interesting to give a summary severity for the patient rather than multiple ones, so we propose to apply a novel task-assembling method to combine the predictions of different tasks from the patient to predict a single score.

The contributions of this work are:

1. we perform automatic task-level PD severity classification using I3D from videos of our PD dataset, based on seven tasks in MDS-UPDRS-III;
2. we show that I3D can benefit from non-medical datasets with transfer learning;
3. we propose TSA to focus on the temporal visual clues and overcome the large discrepancy of motion difference between non-medical datasets and our PD dataset during transfer learning;
4. we propose a task-assembling method to combine models of different tasks to produce a single concluding severity score for a patient.

4.2. RELATED WORK

4.2.1. MACHINE/DEEP LEARNING BASED APPROACHES

Machine/deep learning based PD motor assessment and analysis has been intensively researched in recent years. For instance, the K-nearest neighbors (KNN) AdaBoost classifier and support vector machines (SVM) with RBF kernel were used to classify between PD patients and controls based on the features extracted from individual handwriting [16]. Butt et al. [17] applied machine learning based methods to investigate the significance of PD motor features. For signal-based analysis, signals acquired from the gyroscope attached to the subject's finger were extracted to feed into multiple classifiers [18]. In [19], glottal flow features were used as input for SVM classifier to detect PD with an accuracy of 75.3%. Ferraris et al. [20] used data from optial RGB-Depth devices, which tracks hands and body movements, to train classifiers for PD motor severity rating. Apart from the signal-based analysis, the video was also used as an input data type for PD quantification [21], [22]. Lu et al. [23] designed a pose-based estimation system for assessing Parkinson's disease motor severity. However, to the best of our knowledge, apart from [24] in which freezing of gait videos were used to feed the 3D network, most researchers extracted the feature from videos as the final input for classifiers without fully utilizing the video resource. Based on machine/deep learning approaches, our work applies action recognition method to quantify PD severity using RGB video data.

4.2.2. TRANSFER LEARNING

Transfer learning is a research problem in machine learning that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem [25]. It is widely used as a pre-training approach to offer the model a better starting point instead of training from scratch. In the work of [26], CNN layers trained from ImageNet is reused to transfer visual recognition tasks to learn mid-level representations for small datasets. In action recognition, researchers apply transfer learning to pre-train the model on a large dataset to make the training process faster, more efficient, and less prone to overfitting with a significant performance improvement [11], [12]. Most related research shows that transfer learning can be a useful tool to make the network work on small datasets, and thus we use transfer learning in this work to help improve the performance on our PD dataset.

4.2.3. CAPTURE TEMPORAL INFORMATION

FOR GENERAL VIDEO DATASET

In action recognition, researchers apply various methods to capture the temporal information crucial in video data. In the work of [27] (*C3D*), 3D CNN is used as a spatiotemporal feature extractor for videos, and the extracted features are used as inputs for simple linear classifiers. Based on the 3D CNN, an I3D is introduced to take advantage of pre-trained 2D models [12]. Similar to 3D CNN, I3D performs 3D convolution on both spatial and temporal dimensions simultaneously. However, in I3D, pre-trained 2D filters are repeated or inflated multiple times to form 3D filters. Therefore, I3D can benefit from successful image (2D) classification models trained on large datasets such as ImageNet [28]. Besides 3D CNN, a combination of a stack of CNNs and Long Short-Term Memory (LSTM [29]) networks is applied to exploit the temporal information [30], [31] as well. These methods apply either 3D CNN or 2D CNN with fusion methods such as LSTM on the video data to capture the temporal information. We use I3D as our base model because of its decent performance on public datasets, including Kinetics-400 experimented in [11].

FOR PERIODIC- AND SUBTLE-MOTION VIDEO DATASET

The spatiotemporal template of motion features is used to recognize and segment the repetitive motion by template matching [32]. In [33], CNN is used to count the number of repetitions, and circle length in periodic-motion videos. Besides the task of action recognition, the estimation of repeating frequency is studied in [34], using a Lagrangian approach and an Eulerian approach as the frequency estimators. In periodic-motion videos, we need to focus on the repeating frequency, starting, and ending points to make the model work.

In medical datasets such as movement disorder dataset, videos usually have subtle motion changes, which are hard for architectures to work because subtle motion information is difficult to capture and can not even be seen with bare eyes. The subtle motions can be magnified using a steerable pyramid [35], [36]. In the work of [37], motion frequency is used to estimate material properties. Similarly, signal analysis in the Fourier domain is employed to estimate the tremor frequency of subtle motions [34]. In subtle-motion videos, we need to focus on magnifying the subtle motion or directly estimating the frequency.

4.2.4. SELF-ATTENTION

Attention module is widely used in natural language processing[38] and computer vision [39], [40] fields by allowing the network to focus on key words or pixels. Self-attention mechanism is proposed to capture the relative relationship between words or pixels. Self-attention is extensively explored since the Transformer network is introduced for machine translation [41] where the self-attention is used to compute the interactions between words. In recent work, the QANet [42] architecture uses self-attention in cooperation with convolutions for machine-reading and question answering tasks, where the convolution computes local interactions and self-attention computes global interactions. In image tasks, self-attention with relative positional embeddings is usually used to compute the interactions among pixels in the same image and allows the model to

learn which part of the image is of more importance [43]. In the non-local network [44], self-attention can be used in convolutional architectures to learn the long-range interactions among pixels in images or videos for object detection and video classification. In general, self-attention is used in architectures for modeling sequences as it can capture long-distance interactions. In this paper, we propose a new method, temporal self-attention model, for PD quantification, which involves I3D and the self-attention mechanism, attempting to detect the periodic and subtle motion in the video data.

4.2.5. MULTI-DOMAIN LEARNING

Different non-i.i.d. Parkinson tasks can be treated in a multi-domain setting [45]–[47] with each task being one domain. Multiple similar domains can be learned to let the model work on a new target domain using parameter combination from multiple classifiers [48]. In [49], perceptron-based algorithms are employed for multi-task binary classification problem with the similarity estimation among tasks. Multi-domain learning aims at exploring the relationship between tasks or domains and integrating them to solve a common task. In this work, we combine the features from multiple domains (i.e., tasks from MDS-UPDRS-III) to predict patient-level PD severity classification.

4.3. METHODS

The overall flow of the algorithm is described as follows. Initially, each video is preprocessed to have the same spatial and temporal size. At the same time, we use network-based transfer learning to transfer knowledge from non-medical datasets to the medical one, i.e., reusing the network trained on large datasets as the pre-trained model to replace model initialization. Then, the pre-trained model is fine-tuned on the collected Parkinson's dataset to learn the underlying patterns. After fine-tuning, the model can be used as the classifier for task-level classification. By combining the features extracted by the deep models from different tasks and training a shallow neural network using those features, patient-level analysis can be further made.

4.3.1. INFLATED 3D CONVOLUTIONAL NEURAL NETWORK (I3D)

In this paper, we use I3D as the base network with Residual Networks (ResNet) as the backbone (currently 18, 34, 50, 101, 152-layer variations are available) and its pre-trained models are already available [11]. Furthermore, rather than using two streams (RGB frames and optical flow), we use RGB frames as the only input because computing optical flow is time-consuming, which is not feasible if the real-time prediction is required.

The model is optimized using gradient descent by minimizing the empirical loss with class-balanced focal loss [50]:

$$J(\omega) = \frac{1}{N}\sum_{i=1}^{N}\Big(-\frac{1-\beta}{1-\beta^{n_y}}\sum_{c=1}^{C}(1-p_{i,c}^{t})^{\gamma}\log(p_{i,c}^{t})\Big) + \lambda\,\|\omega\|_2^2, \tag{4.1}$$

where C, N, ω and γ denote the number of classes, number of samples, learned parameters and *focusing* parameter, and $\beta = (N-1)/N$. n_y stands for the number of samples

in the ground-truth class y and p^t is defined as

$$p^t = \begin{cases} p & \text{if } y = c \\ 1-p & \text{otherwise.} \end{cases} \tag{4.2}$$

4.3.2. SELF-ATTENTION REPLACING CONVOLUTION

We describe the proposed temporal self-attention block for video classification following the symbol styles of [43].

TEMPORAL SELF-ATTENTION OVER VIDEO VOLUME

We first transpose and flatten the input of shape (C, T, H, W)[1] from the previous layer to the shape of $HW \times T \times C$ and then perform multi-head-attention on the temporal dimension

$$O_h = \text{Softmax}\Big(\frac{QK^T}{\sqrt{d_k^h}}\Big)V, \tag{4.3}$$

where queries $Q = XW_q$, keys $K = XW_k$ and values $V = XW_v$ and W_q, $W_k \in \mathbb{R}^{C \times d_k^h}$ and $W_v \in \mathbb{R}^{C \times d_v^h}$ are learned linear transformations[2]. d_k^h and d_v^h stand for the dimension of each head of K and V. Note that we transpose the last two dimensions of V to correctly multiply with Q. Concatenating the outputs from all heads we get

$$O = \big[O_1, \ldots, O_{N_h}\big]. \tag{4.4}$$

The shape of O is $(HW \times T \times d_k^h)$ and is transformed with $W^O \in \mathbb{R}^{d_v \times d_v}$ to

$$\text{MultiHead}(Q, K, V) = OW^O, \tag{4.5}$$

where $\text{MultiHead}(Q, K, V)$ is of shape $(HW \times T \times d_v^h)$. After reshaping back to the original spatial and temporal dimension, we have the final output $\text{MultiHead}(Q, K, V) \in \mathbb{R}^{T \times H \times W \times d_v}$ of our temporal self-attention block if relative postional embeddings [43] (see Section 4.3.2) not applied.

The novelty of our temporal self-attention block is applying the self-attention mechanism solely on the temporal dimension, leaving the spatial dimension untouched. The advantage is that self-attention can capture the long-range temporal changes while keeping standard CNN there, capturing the necessary visual patterns simultaneously. As such, the abilities of both self-attention and CNN be retained and incorporated in the temporal self-attention block, which effectively makes up the drawback of I3D.

Fig. 4.2 illustrates the temporal self-attention mechanism. The temporal sequence of feature points (red ones) that share the same spatial position is the atomic unit, on top of which the temporal self-attention applies. We have HW sequences/units located at all spatial positions, and each of them is independent of others when performing the temporal self-attention.

[1] The number of channels, time or frames, height and width.
[2] Bias terms are ignored when we mention linear transformations.

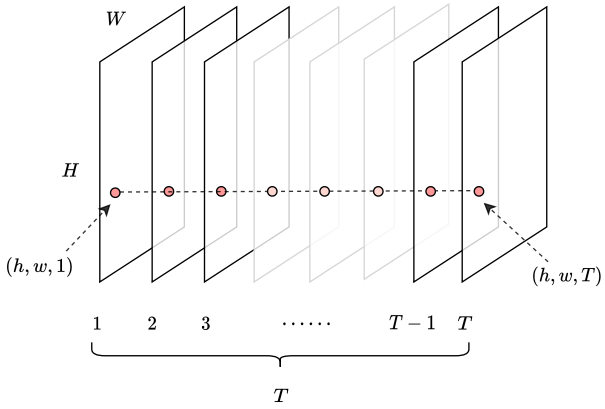


Figure 4.2: An example of temporal self-attention. Assume the stack of those rectangles is a feature map (or more intuitively for 3D data, feature volume) from one channel. Each rectangle represents the spatial visual patterns at a specific temporal position. Our temporal self-attention is performed on the feature points colored in red, which share the same spatial position along the temporal dimension. It can be seen as self-attention through time.

RELATIVE POSITIONAL EMBEDDINGS

The only difference between 1D and 2D relative positional embeddings is the dimensions involved in the algorithm. Thus we refer to [43] for the details of 2D relative positional embeddings, and we do not discuss the 1D variation anymore in this paper. To implement temporal relative self-attention, we add relative temporal information to the temporal self-attention block's output. The output is now changed from Equation 4.3 to

$$O_h = \text{Softmax}\Big(\frac{QK^T + S_T^{rel}}{\sqrt{d_k^h}}\Big)V, \tag{4.6}$$

where $S_T^{rel} \in \mathbb{R}^{HW \times T \times T}$ is the matrix of relative position logits along the temporal dimension.

TEMPORAL RELATIVE SELF-ATTENTION

We combine temporal self-attention with 1D relative positional embeddings to form our new building block-temporal relative self-attention block. Fig. 4.3 describes the whole pipeline of the proposed block.

TEMPORAL RELATIVE SELF-ATTENTION NETWORK (*TSA*)

Once the temporal relative self-attention block is built up, the convolutional block in any architecture can be substituted. Take 3D ResNet-34 for instance, which has 33 convolutional layers. We replace as many layers as possible with our block from the last convolutional layer to the first one until we hit the memory bottleneck.

The time complexity of our block is $O(HWT^2d_k)$ compared to the convolutional block $O(HWTC)$, which is time-efficient since the temporal size is typically small after a few layers. The memory cost is $O(HWT^2N_hd_k^h)$ compared to the convolutional block $O(HWTC)$.

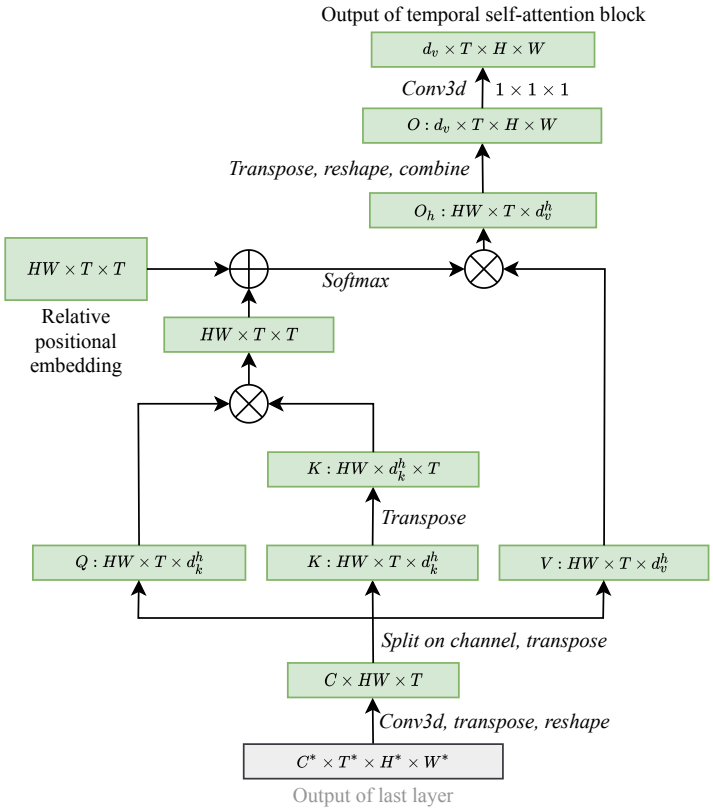


Figure 4.3: The general pipeline of our temporal relative self-attention. Rectangles in the workflow represent tensors with shape specified, and italic words stand for tensor operations. ⊗ and + denote tensor product and addition.

4.3.3. MULTI-TASK ASSEMBLING

Using the model we discussed in previous sections, it can solve the task-level severity classification on our PD dataset. Given a sample related to a specific task from the dataset, we can predict its task severity S_t. Nonetheless, it is more clinically interesting to tell the severity score of a patient S_p instead of tasks. Therefore, we propose two multi-task assembling methods to combine the tasks to do severity classification for patients. Note that the following methods require trained models on the PD dataset for task-level classification.

VECTOR AVERAGING AND VECTOR WEIGHTING

We use the trained model as a feature extractor to compress the information of a video into a dense one. We first extract the flattened vector $F \in \mathbb{R}^d$ of dimension size d as the compressed information, which is the input feature of the fully connected layer. Each video, containing only a single task from a patient, produces one feature vector F_m of task m and all videos from that patient produce feature vectors $F_M \in \mathbb{R}^{d \times M}$ of all M

tasks. Different tasks may contribute unequally to a patient's severity score, so we use two strategies to convert (or combine) F_M into a vector $F \in \mathbb{R}^d$, representing the feature of a patient.

The first approach is to average features, formulated as

$$F = \frac{1}{M} \sum_{m=1}^{M} F_m, \tag{4.7}$$

by assuming each feature (task) contributes equally. The second approach is to take the weighted average of features as the following

$$F = \sum_{m=1}^{M} \alpha_m F_m, \tag{4.8}$$

where α_m ($\sum_{m=1}^{M} \alpha_m = 1$) is the learnable weight for task m. The first approach is a special case of this one. Afterward, F is fed as input to train a shallow neural network[3]. The network is optimized using gradient descent by minimizing the empirical loss $J(\omega)$ (see Equation 4.1) where N is the number of patients.

ATTENTION-BASED FEATURE WEIGHTING

In the feature averaging and weighting approach, we assume task weights are identical across all patients. However, patients may not share the same task weights so that the global task weights may be insufficient and inaccurate. Therefore, we propose to use channel-wise attention-based weighting, which automatically assigns task weights for each patient separately. To do so, we use another feature map $F_M \in \mathbb{R}^{M \times C \times T \times H \times W}$ (M denotes the number of tasks), the output of the last convolutional or our self-attention layer, as the extracted feature for a video.

The first weighting strategy is to apply squeeze-and-excitation block [51] to map the input feature F_M to a set of channel weights. As the task weights are our concerns instead of the channels, we take the task dimension as the channel dimension in the squeeze-and-excitation block. The process can be formulated as follows. First, squeeze global information into a task descriptor by using global average pooling to generate task-wise statistics

$$z_m = \frac{1}{C \times T \times H \times W} \sum_{c=1}^{C} \sum_{t=1}^{T} \sum_{h=1}^{H} \sum_{w=1}^{W} F_m(c,t,h,w), \tag{4.9}$$

where F_m denotes the feature map for task m. Then we excite the task-wise statistics to task weights ($W_1 \in \mathbb{R}^{\frac{M}{r} \times M}$, $W_2 \in \mathbb{R}^{M \times \frac{M}{r}}$ in which r is the dimensionality-reduction ratio)

$$\alpha_M = \sigma(W_2 \delta(W_1 z_M)), \tag{4.10}$$

where α_M, σ and δ denote task weights, the sigmoid activation and the ReLU [52] function. Finally we obtain the combined feature map $F \in \mathbb{R}^{C \times T \times H \times W}$

$$F = \alpha_M F_M. \tag{4.11}$$

[3] 0, 1 or 2 hidden layers with non-linear activation.

Table 4.1: The summary of four task-assembling methods.

	vector averaging	vector weighting	channel-wise attention weighting	pixel-wise attention weighting
Input type	*avgpool*	*avgpool*	*layer4*	*layer4*
Weights differ among tasks	✗	✓	✓	✓
Weights differ among patients	✗	✗	✓	✓
Weights differ among feature points	✗	✗	✗	✓
Core mechanism	averaging	learnable weight vector	squeeze-and-excitation [51]	self-attention

Applying the squeeze-and-excitation block to get task weights is rather simple but turns out to be efficient. It flexibly generates different weights for different patients accordingly. However, this approach assumes each feature point in the feature map contributes equally, which means a task weight is a global weight for all feature points. We can explore even further by making each feature point having its own weight $\alpha_{t,h,w,m}$, which brings about the pixel-wise attention-based weighting approach.

We opt to use the self-attention mechanism similar to our temporal relative self-attention block for pixel-wise weighting, by applying it on the task dimension instead of the temporal dimension. First, we reshape and flatten $F_M \in \mathbb{R}^{M\times C\times T\times H\times W}$ into the shape of $(THW \times M \times C)$ and then the output of a single attention head can be computed as

$$O_h = \text{Softmax}\Big(\frac{(F_M W_q)(F_M W_k)^T}{\sqrt{d_k^h}}\Big)(F_M W_v), \tag{4.12}$$

where W_q, $W_k \in \mathbb{R}^{C\times d_k^h}$ and $W_v \in \mathbb{R}^{C\times d_v^h}$ are learned linear transformations. Afterwards, we combine attention results of all heads and project using $O^W \in \mathbb{R}^{d_v\times d_v}$ to form the task weighted feature map

$$F = \big[O_1, \ldots, O_{N_h}\big] O^W. \tag{4.13}$$

Note that the task weights for each feature point $\alpha_{t,h,w,m}$ is implicitly embedded in the computation of attention output.

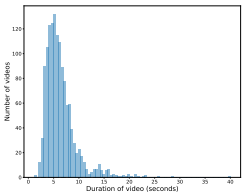
Task weighted features using both approaches are fed into a shallow neural network consisting of batch normalization [53], the ReLU function, global average pooling, and a fully connected layer.

The summary of the proposed four task-assembling methods can be found in TABLE 4.1. Vector averaging and vector weighting use the outputs of the last global average pooling layer while attention-based weighting methods use the outputs of the last convolutional/self-attention layer in the network. We denote *avgpool* and *layer4* as the feature types.

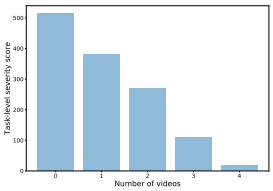
4.4. EXPERIMENTAL SETTINGS

4.4.1. DATASET

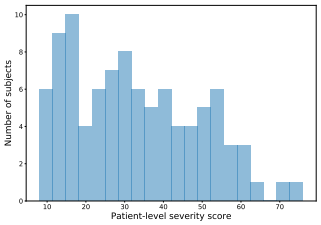
In this paper, we introduce a new video dataset for Parkinson's disease analysis. We develop this dataset principally because there is a lack of such datasets for Parkinson's disease analysis. We believe that having one will facilitate research in this area because the dataset simulates the procedure of how experts assess patients' symptoms using MDS-UPDRS-III scores. Besides, the dataset is challenging enough to act as a performance



(a) The histogram of the duration of samples, using 80 bins. The average duration is 6.3 seconds, and 90% of samples are shorter than 10 seconds, with less than five samples longer than 25 seconds.



(b) The bar chart shows the distribution of task-level severity score. From low to high severity class, the number of samples decreases, which shows the class imbalance issue in our dataset.



(c) The histogram of patient-level severity score using 20 bins. Compared to task-level severity distribution, patient-level severity distribution has no obvious imbalance issue. The number of patients across the range of severity is approximately on the same level.

Figure 4.4: Distributions of the sample duration and task/patient-level severity of our dataset.

benchmark where the advantages of different architectures can be demonstrated.

DATA COLLECTION

Routine video recordings of consecutive patients who underwent either a Levodopa Challenge Test (LCT [54], [55]) prior to DBS surgery, or underwent a Stimulator Challenge Test (SCT, [56], [57]) after DBS surgery, were collected. All patients fulfilled the criteria for idiopathic PD. Patients who underwent a LCT were videotaped twice (i.e. Med-OFF and Med-ON); patients who underwent SCT were videotaped three times (Med-OFF/Stim-ON [58], etc). Video recordings were made with the camera in a fixed position, with a complete overview of the patient central on the screen. Due to the varying nature of the examination room, the camera's position and angle towards the patient varied, as well as the background and surroundings. During the MDS-UPDRS-III examination, the zoom-function was occasionally used to focus on the hands or feet.

All videos were made in one continuous recording of the examination. Separate segments were created by clipping the videos per task (left and right separately if required): bradykinesia of the hands (MDS-UPDRS-III items 3.4, 3.5, 3.6), bradykinesia of the legs (items 3.7, 3.8), postural tremor (item 3.15), kinetic tremor (item 3.16). Rigidity was not included as this symptom is not assessed through visual observation; global bradykinesia, speech, freezing-of-gait, and rest-tremor were not included as no specific video-segment pertained to those tasks and they were evaluated throughout the entire recording. The local medical ethics committee waived the formal evaluation of the study. All patients gave written informed consent.

We are not allowed to make the dataset publicly under the Dutch privacy law.

DATASET OVERVIEW

The dataset contains 39 subjects (all patients) and 1082 video fragments after cutting. Each sample in the dataset is of resolution 1920 by 1080 and 25 fps. The duration of samples may be different on different tasks. Fig. 4.4a shows the duration distribution of our dataset.

The dataset contains $T = 11$ tasks for most of the patients based on the MDS-UPDRS-III, namely finger tapping, gait freezing, hand movements, leg agility, pronation, toe tapping, arising from chair, kinetic tremor, postural tremor, postural stability and rest tremor. Note that not all tasks are used in the experiments. Each video has a task-level severity score $S_t \in \{0,1,2,3,4\}$ (0: normal, 1: slight, 2: mild, 3: moderate and 4: severe) labeled by experts. We have to emphasize that a task score of 0 does not mean that the subject is not a PD patient but indicates that the subject may have low severity on the specific task. Each patient has a patient-level severity score, which is the sum of all task-level severity scores, as shown in the following equation:

$$S_p = \sum_{t=1}^{T} S_t. \tag{4.14}$$

The distributions of S_t (over all tasks) and S_p are shown in Fig. 4.4b and Fig. 4.4c.

4.4.2. SETTINGS

To evaluate our methods for Parkinson's severity classification, we use the above-described dataset. In our experiments, only RGB frames are used as the input for the deep architectures. The clips are resized to $32 \times 224 \times 224$ resolution without changing their spatial aspect ratios.

The dataset is split into five folds at the patient level but not the video level. One subject only appears in either the training or testing fold to avoid network cheating by recognizing the appearance of the patient. We train networks on four of them and test it on the remaining one in the cross validation setting. The overall accuracy is obtained by taking the average of the individual accuracy tested on each fold.

I3D is pre-trained on both UCF-101 (by ourselves) and Kinetics-400 (by [11]). TSA is pre-trained only from UCF-101 (by ourselves). Batch size of 15, learning rate of 0.001 without decay and weight decay (λ) of 0.01 are used.

The task-level score $S_t \in 0,1,2,3,4$ is split into two classes: class 0 for $\{0,1\}$ and class 1 for $\{2,3,4\}$ since we are more interested in whether the model can distinguish between the slight and severe group of patients. The patient-level score S_p is split into three classes in the way that each class has an equal number of patients. Method specific settings are provided alongside when showing the results in Section 4.5.

We briefly introduce the results in order shown in the next section. We first validate the performance of TSA on public dataset in section 4.5.1, and then inspect the performance improvement using transfer learning in section 4.5.2. In Section 4.5.3, we show results on seven PD tasks using models with different settings followed with comparison between those models. In Section 4.5.4, we analyze the performance on patient-level severity classification, compare different strategies to combine PD tasks, and show the model behavior on classifying only the highest and lowest severity class.

4.5. RESULTS

In this section, we show the results of our experiments. We test seven tasks with high quality videos, *finger tapping*, *hand movements*, *pronation*, *toe tapping*, *leg agility*, *postural tremor* and *kinetic tremor*. They are denoted as *finger*, *hand*, *pronation*, *toe*, *leg*,

Table 4.2: Top-1 accuracy on UCF-101 and HMDB-51. All accuracy are averaged over three splits. Both methods use ResNet-18 as the backbone. TSA shows better performance on both datasets so that it can be further applied to PD dataset.

Method (scratch)	UCF-101	HMDB-51
I3D ResNet-18 [11]	42.4	17.1
TSA ResNet-18	51.5±2.6	22.1±1.9

postural and *kinetic* for simplicity. We use ResNet-34 as backbone because through experiments we find that ResNet-34 is the most suitable one in this study, considering the size and difficulty of our dataset. One can of course use other backbones if the size, complexity and classes of the dataset are different from ours. We have to emphasize that, in all experiments, although patients contribute more than one video, no patient is included into both the training- and test-set because even though videos of a patient are separate ones, they are still from the same patient.

4.5.1. VALIDATE TEMPORAL RELATIVE SELF-ATTENTION NETWORK

Before applying TSA on PD dataset, we first check whether it works better than I3D on two frequently used public datasets UCF-101 and HMDB-51. Hyper-parameters are chosen without optimization: input shape of $64 \times 224 \times 224$, lr of 0.001, batch size of 45, weight decay of 10^{-5} and optimizer of SGD with momentum [59]. The backbone is ResNet-18 for fast illustration. TABLE 4.2 shows that TSA outperforms I3D when both trained from scratch. The performance improvements demonstrate the effectiveness of TSA and the possibility of applying it to our PD dataset.

4.5.2. BENEFIT FROM TRANSFER LEARNING

We utilize three datasets: Kinetics-400 [60] and UCF-101 [61] to pre-train our models considering their large sizes, high quality and popularity. Then, we fine-tune the pretrained models on our PD dataset. Since our dataset contains periodic and subtle motions while public datasets have easily distinguishable motions, the relatedness between our dataset and public datasets is not tight. As such, the parameters from the convolutional stem may not be optimal after transferring to our dataset. Thus all layers of the model rather than part of them are fine-tuned.

I3D and task *finger* and *hand* are used to demonstrate the function of transfer learning. Convergence is confirmed for every compared setting for a fair comparison. Note that for task-level classification we have binary classes. In TABLE 4.3, I3D trained from scratch, I3D pretrained from UCF-101, and I3D pretrained from Kinetics-400 are compared based on the binary accuracy, precision, recall, and Matthews correlation coefficient (MCC). Here the MCC is formed as:

$$\text{MCC} = \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP}+\text{FP})(\text{TP}+\text{FN})(\text{TN}+\text{FP})(\text{TN}+\text{FN})}} \tag{4.15}$$

where TP, TN, FP and FN stand for true positive, true negative, false positive and false negative. We also show the receiver operating characteristic (ROC) curves of all 6 set-

Table 4.3: Accuracy, precisons, recall and MCC (with CI 95% and *p*-value) on task *finger* and *hand* (binary classification) using I3D with and without transfer learning. Datasets in the brackets denote where the model is pretrained. I3D using transfer learning achieves better results than I3D trained from scratch on both *finger* and *hand* tasks. Moreover, transfer learning with a larger dataset (i.e., Kinetics-400) has more benefits to the model.

Method	Metric	*finger*	*hand*
I3D (scratch)	acc	65.4	65.6
	MCC	0.32±0.08	0.31±0.06
	CI 95%	[0.16, 0.48]	[0.19, 0.43]
	p-value	7.3×10^{-5}	3.7×10^{-7}
I3D (UCF-101)	acc	68.6	70.0
	MCC	0.34±0.10	0.39±0.05
	CI 95%	[0.14, 0.54]	[0.29, 0.49]
	p-value	7.1×10^{-4}	3.8×10^{-14}
I3D (Kinetics-400)	acc	69.2	77.5
	MCC	0.35±0.06	0.54±0.07
	CI 95%	[0.23, 0.47]	[0.40, 0.68]
	p-value	1.1×10^{-8}	7.0×10^{-14}

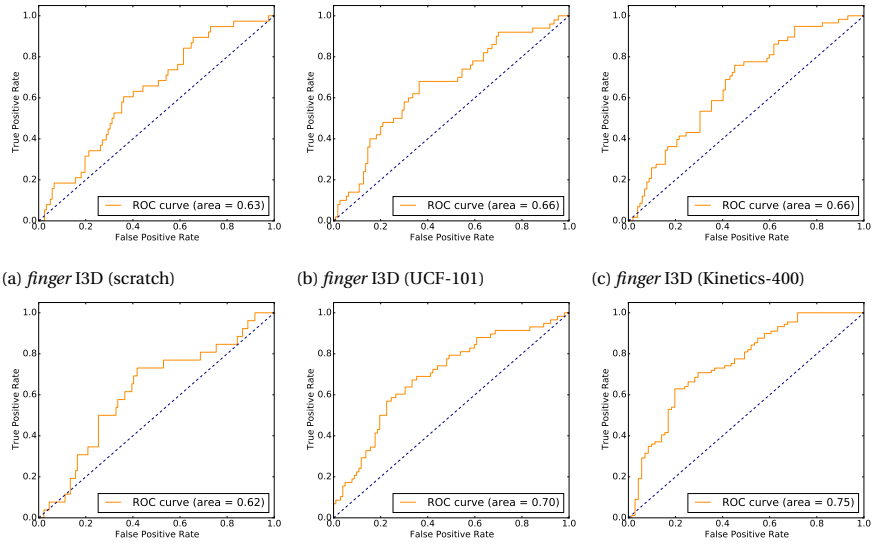
tings based on TABLE 4.3. In general, I3D pretrained from the two datasets outperform I3D (scratch), demonstrating that I3D can benefit from non-medical datasets with transfer learning. Moreover, the performance improvement of I3D (Kinetics-400) from I3D (scratch) is more notable than I3D (UCF-101) especially on task *hand*, which indicates the model would benefit more from a larger dataset with transfer learning.

Table 4.4: The number of samples in each class of seven tasks in our PD dataset.

Task	*finger*	*hand*	*kinetic*	*leg*	*postural*	*pronation*	*toe*
Class 0	66	89	130	145	62	104	87
Class 1	91	71	38	39	23	72	71

4.5.3. TASK-LEVEL SEVERITY CLASSIFICATION

Building a model good at predicting the task severity score is our first concern and affects the later experiments and research. Two architectures - I3D and our TSA are compared in TABLE 4.5 on seven tasks from MDS-UPDRS-III. The class distribution can be found on TABLE 4.4. In general, the class imbalance in task *finger*, *hand*, *pronation* and *toe* is acceptable. In remaining tasks, the class imbalance issue is severe. Note that we replace convolutional layers in 3D ResNet-34 *layer3* and *layer4* with temporal relative self-attention block to construct our TSA network. The dataset in the brackets denotes on which the model is pretrained. We show the MCC along with precision and recall.



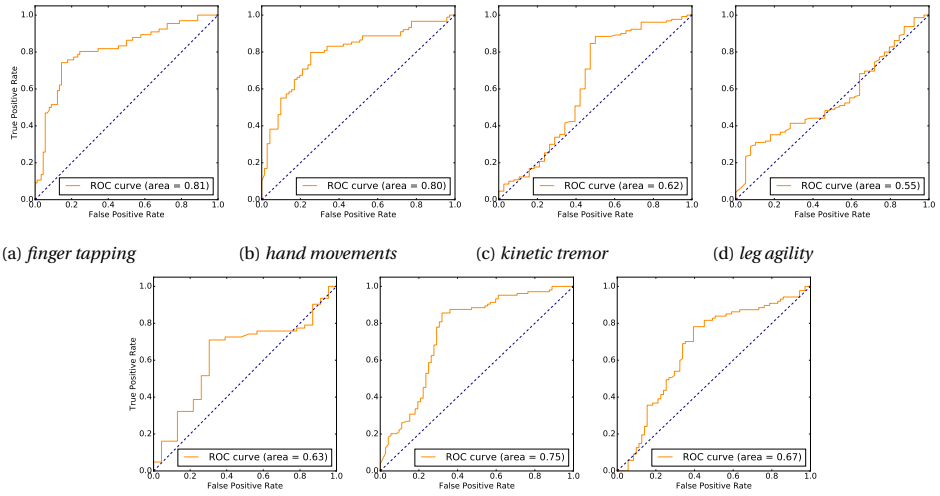
(a) *finger* I3D (scratch) (b) *finger* I3D (UCF-101) (c) *finger* I3D (Kinetics-400)

(d) *hand* I3D (scratch) (e) *hand* I3D (UCF-101) (f) *hand* I3D (Kinetics-400)

Figure 4.5: ROC curves of all 6 settings in Table 4.3.

Table 4.5: Accuracy, precision, recall, and MCC (with CI 95% and *p*-value) on seven tasks from MDS-UPDRS-III using I3D and TSA. Each row shows the performance of a task and each column gives the result of a measurement (two classes). Datasets in the brackets denote on which public dataset the model is pretrained. In general, I3D pretrained on Kinetics-400 outperforms I3D pretrained on UCF-101, indicating transfer learning from larger datasets has more benefits than smaller datasets. TSA pretrained from a smaller dataset, UCF-101, is comparable to Kinetics-400 pretrained I3D.

Task	I3D (UCF-101)				I3D (Kinetics-400)				TSA (UCF-101)			
	acc	precision recall	MCC	95% CI *p*-value	acc	precision recall	MCC	95% CI *p*-value	acc	precision recall	MCC	95% CI *p*-value
finger	68.6	0.55 0.76	0.34±0.09	[0.16, 0.52] 1.7×10^{-4}	69.2	0.57 0.76	0.35±0.06	[0.23, 0.47] 1.1×10^{-8}	78.2	0.75 0.81	**0.55±0.08**	[0.39, 0.71] 2.1×10^{-11}
hand	70.0	0.76 0.61	0.39±0.07	[0.25, 0.53] 4.5×10^{-8}	77.5	0.80 0.75	**0.54±0.11**	[0.32, 0.56] 1.3×10^{-6}	75.6	0.79 0.72	0.50±0.06	[0.38, 0.62] 7.2×10^{-16}
kinetic	78.0	0.87 0.10	0.22±0.06	[0.10, 0.34] 2.7×10^{-4}	73.8	0.82 0.49	0.33±0.10	[0.13, 0.53] 1.0×10^{-3}	79.2	0.87 0.51	**0.40±0.09**	[0.22, 0.58] 1.1×10^{-5}
leg	79.3	0.88 0.17	0.24±0.05	[0.14, 0.34] 2.2×10^{-6}	79.3	0.88 0.14	0.26±0.06	[0.14, 0.38] 1.8×10^{-5}	70.1	0.81 0.35	**0.29±0.04**	[0.21, 0.37] 1.8×10^{-12}
postural	74.1	0.85 0.08	0.18±0.04	[0.10, 0.26] 8.7×10^{-6}	77.6	0.87 0.34	0.30±0.08	[0.14, 0.46] 2.0×10^{-4}	70.6	0.78 0.56	**0.35±0.09**	[0.17, 0.53] 1.1×10^{-4}
pronation	68.8	0.76 0.56	0.34±0.06	[0.22, 0.46] 2.7×10^{-8}	77.8	0.87 0.71	**0.53±0.07**	[0.39, 0.67] 1.7×10^{-4}	72.2	0.76 0.67	0.43±0.04	[0.35, 0.51] 5.9×10^{-25}
toe	64.6	0.72 0.52	0.31±0.07	[0.20, 0.48] 1.7×10^{-6}	67.7	0.70 0.65	**0.38±0.08**	[0.22, 0.54] 2.8×10^{-6}	62.0	0.68 0.53	0.29±0.06	[0.17, 0.41] 1.9×10^{-6}
average	-	-	0.29±0.08	-	-	-	0.38±0.11	-	-	-	0.40±0.10	



(a) *finger tapping* (b) *hand movements* (c) *kinetic tremor* (d) *leg agility*

(e) *postural tremor* (f) *pronation* (g) *toe tapping*

Figure 4.6: ROC curves for seven tasks in the setting where the best performance is achieved in TABLE 4.5. The ROCs on task *finger*, *hand*, *pronation* and *toe* are well shaped, indicating that models on these tasks performs well. The remaining ROCs are close to the diagonals, which means the models' performance is not good.

TASK-LEVEL PERFORMANCE

Fig. 4.6 shows the ROC curve for each task in the setting which achieves the best performance (bold numbers) in TABLE 4.5. Three out of seven tasks have the best MCC higher than 0.5, and only one task *leg* is under 0.3. The average MCC across all seven tasks is 0.40, sufficiently good for classification on a medical dataset. It demonstrates that deep architectures can predict the task (i.e., task from MDS-UPDRS) severity of a patient with decent accuracy given the video from that task.

Table 4.6: Clinical information for three classes. Note that each patient is videotaped two or three times, and the severity score of each time may fall into different classes. For simplicity, L-OFF, L-ON, A, B, and C denote Levodopa challenge test OFF, Levodopa challenge test ON, Med-OFF-Stim-ON, Med-OFF-Stim-OFF, and Med-ON-Stim-ON. Each class has an approximately equal number of patients and videos, i.e., no severe class imbalance issue.

Class	Score	Number of patients	Age	Disease duration	Male/Female	Number of video fragments					
						all	L-OFF	L-ON	A	B	C
0	15±4	32	61±8	11±4	22/10	351	0	130	66	0	155
1	32±5	32	65±9	12±5	28/4	374	62	36	145	65	66
2	53±8	31	64±8	11±5	21/10	357	152	21	12	172	0
total	33±16	39	63±8	11±5	28/11	1082	214	187	223	237	221

In particular, task *finger*, *hand* and *pronation* are the top-3 well-classified task in terms of MCC and ROC curves in Fig. 4.6a, 4.6b and 4.6f, because 1) most of the videos are zoomed in to focus on the objects, making it easier for the model to look at the relevant patterns and 2) the class imbalance problem is slight compared to task *kinetic*, *leg* and *postural*. On the opposite, task *leg* has the lowest MCC, and the ROC curve in Fig.

4.6d does not bulge towards the top-left corner of the figure, indicating a corrupt model for task *leg*. Inspecting TABLE 4.5, we can observe quite low recalls of 0.17 and 0.14 using I3Ds and an inadequate recall of 0.35 using TSA.

The performance discrepancy between tasks exposes some disadvantages of our architectures. First, the ratio of objects, e.g., hand in task *hand movements* and toe in task *toe tapping*, occupying the bounding box of the video matters. In task *finger tapping*, *hand movements* and *pronation*, the zoom-function is occasionally used to focus on the objects, and most of the videos are zoomed in during the pre-processing stage, which gives the architectures cleaner and more easy-to-identify input data. Second, the effects of the class imbalance problem on the architectures cannot be ignored. Due to the PD dataset is a periodic- and subtle-motion dataset, which is different from public datasets. Identifying task severity is harder than classifying different human actions. In such a case, the extreme class imbalance can corrupt the architectures' behavior even if the class-balanced loss [50] is adopted. However, the class imbalance is everywhere in real-world settings or at least in Parkinson's disease. As such, we leave solving class imbalance on the PD dataset as one of the future work.

MODEL COMPARISON

In TABLE 4.5, we see that in terms of the MCC, TSA (UCF-101) outperforms I3D (UCF-101) on six tasks with a significant margin. Besides, the average MCC of the former is also clearly better than the latter. Since the only difference between the two is the backbone used, we can conclude that our TSA performs better than I3D on the PD dataset.

Also, compared to I3D (Kinetics-400), TSA (UCF-101) still has 1.5% improvements even if pretrained from a much smaller and less complex dataset. It demonstrates that TSA is better at dealing with the large discrepancy of motion difference between non-medical datasets and our PD dataset. So we think TSA pretrained from Kinetics-400 would further improve the performance. Due to the limit of time and computation resource, we leave it as the future work.

Regarding the time cost of the temporal relative self-attention, it is completely acceptable as the network can still run with a bit more time cost. However, the memory cost can be problematic if the network is too deep due to the hardware memory limitation. As such, we give some useful solutions in terms of the algorithm itself:

1. only replace convolutional layers with small temporal size (usually the last few),
2. reduce d_k and
3. use large kernel size or stride on the temporal dimension at the first few layers to quickly decrease the temporal size to the one you want and use kernel size of 1 at following layers to maintain the temporal size unchanged until the last layer.

Another issue of TSA is that a large learning rate is possible to cause the exploding gradients problem, which can be overcome by applying approaches such as the ReLU activation function and pre-training.

4.5.4. PATIENT-LEVEL SEVERITY CLASSIFICATION

We use the trained model on each task as the feature extractor to extract the learned patterns and apply the proposed four task-assembling methods to incorporate tasks to

produce a single concluding severity score for a patient. The patient-level severity is split into three classes by cut-off: *slight* $\in [0,23]$, *moderate* $\in (23,40]$ and *severe* $\in (40,-]$ with approximately equal number of videos. TABLE 4.6 shows the number of video fragments in each class. Experiments are repeated 20 times to ensure validity.

SINGLE-TASK BASELINE

To demonstrate the effectiveness of task-assembling methods, we first do patient-level severity classification using only one single task as the baseline. The result is shown in TABLE 4.7. The best MCC is 0.31 using single task *hand*, which is served as the baseline to compare with assembling methods.

Table 4.7: Single task baseline for patient-level severity classification (three classes). Each row shows the performance of a task and columns give the result of accuracy and MCC with standard deviation provided. Rank is calculated based on the average MCC from two inputs. The top-3 well-performed tasks used for patient-level classification are task *hand*, *kinetic* and *finger*. Task *hand* achieves a MCC of 0.31, which is used as the best single-task baseline.

Task	Input	Accuracy	MCC	Rank
finger	*avgpool*	60.3±2.8	0.30±0.05	2
	layer4	60.7±3.2	0.31±0.04	
hand	*avgpool*	61.5±2.8	0.31±0.04	1
	layer4	60.7±3.1	0.30±0.03	
kinetic	*avgpool*	59.7±2.7	0.29±0.04	3
	layer4	60.5±3.4	0.30±0.04	
leg	*avgpool*	50.6±2.7	0.21±0.05	6
	layer4	60.0±3.7	0.27±0.04	
postural	*avgpool*	54.9±2.5	0.19±0.05	5
	layer4	60.8±3.5	0.29±0.04	
pronation	*avgpool*	59.3±3.2	0.20±0.05	4
	layer4	61.3±3.4	0.31±0.04	
toe	*avgpool*	51.3±2.8	0.17±0.06	7
	layer4	60.6±3.9	0.28±0.04	

BENEFIT FROM TASK-ASSEMBLING METHODS

Four task-assembling methods incorporate seven tasks used in task-level severity classification. From TABLE 4.8, we see that all task-assembling methods, including the most straightforward averaging strategy, outperforms the single-task baseline. The best method is the pixel-wise self-attention based weighting in terms of the MCC, with an improvement of 25.8% from the baseline. These results demonstrate that patient-level severity classification benefits from all tasks combined compared to based on a single task, which is intuitive since it is also hard for experts to diagnose a patient by inspecting just one task.

Comparing all four methods, we see the weighting strategy is better than just simple averaging, indicating that each task contributes unequally to the patient-level severity. Moreover, the attention-based weighting slightly outperforms the learnable vector-based weighting. It is because 1) *layer4* has more feature points, potentially more representable for a task than *avgpool*, and 2) attention-based weighting gives more flexibility to the weights such that patients can have task weights exclusively learned based on their condition.

Table 4.8: Patient-level severity classification (three classes) using single task as a baseline and task-assembling approaches (seven tasks). Each row shows the performance of a task-assembling method on the input from a certain layer. The four task-assembling methods outperform the single-task baseline with the channel-wise and pixel-wise attention weighting being the best methods.

Method	Input	Accuracy	MCC
single task baseline	*avgpool*	61.5±2.8	0.31±0.04
vector averaging	*avgpool*	62.7±2.4	0.32±0.06
vector weighting	*avgpool*	64.1±2.4	0.37±0.06
channel-wise attention weighting	*layer4*	64.5±3.1	0.38±0.05
pixel-wise attention weighting	*layer4*	64.5±2.8	0.39±0.06

We show the weights learned in the vector weighting method in Fig. 4.7 to give a general feeling of which task may contribute less or more to the prediction of patient-level severity. Weights are averaged across 20 runs on each fold, a total of 100 runs. As the two attention-based weighting methods assign task weights for patients exclusively, it is not intuitive to see the overall weight distribution on tasks. In Fig. 4.7, we see the top-2 tasks with highest weights are *hand* and *finger*, which well matches the performance rank in TABLE 4.7. The rest tasks remain the similar position as in TABLE 4.7 except that task *kinetic* drops to the lowest rank. We suspect the reason being the effect of severe class imbalance problem of task *kinetic*.

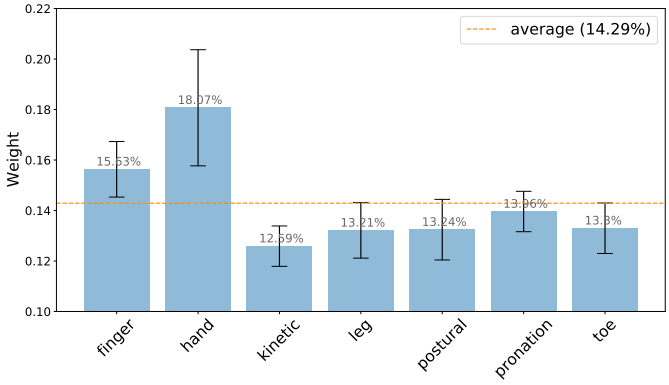


Figure 4.7: Weights for seven tasks learned by vector weighting method. The weights of task *finger* and *hand* are higher than the average, which means in the task-assembling approach, i.e., vector weighting, they contribute more than other tasks in the prediction of the patient-level severity.

DISTINGUISHING BETWEEN SLIGHT AND SEVERE CLASSES

We remove the class *moderate* with the remaining classes untouched to focus on the classification between *slight* and *severe* classes. The result of the best single task baseline and assembling methods are shown in TABLE 4.9. By combining seven tasks, we gain 1.7%-13.3% performance improvements compared to using a single task. At best, we can achieve a MCC of 0.68 on distinguishing between *slight* and *severe* classes. Moreover, the attention-based weighting methods still outperform the vector-based ones, matching the case in TABLE 4.8.

In general, attention-based weighting strategy is the first choice to assemble the tasks, but the vector-based one is also applicable, given its higher time efficiency. It is also worthwhile to exclude some tasks to see the ablation effects on patient-level performance. As the main focus of this paper is to show the potential of combining tasks, we leave it as future work.

Table 4.9: Patient-level severity classification (two classes with class *moderate* removed) using single task and task-assembling approaches (seven tasks). Each row shows the performance of a task-assembling method on the input from a certain layer. The four task-assembling methods outperform the single-task baseline with the pixel-wise attention weighting being the best method.

Method	Input	Accuracy	MCC
single task baseline	*avgpool*	81.1±2.2	0.60±0.06
vector averaging	*avgpool*	81.4±1.7	0.61±0.05
vector weighting	*avgpool*	81.9±2.1	0.64±0.07
channel-wise attention weighting	*layer4*	82.2±3.1	0.66±0.09
pixel-wise attention weighting	*layer4*	83.6±1.2	0.68±0.04

In Section 4.5.4 and 4.5.4, we empirically show the possibility that a multi-task algorithm based on an incomplete video-overview (i.e. not all MDS-UPDRS-III items are included) can help discriminate between groups of disease severity in both *slight-moderate-severe* and *slight-severe* cases with acceptable MCC, 0.39 for the former case and 0.68 for the latter case. Besides, the performance of single task and weights visualization demonstrates the test of bradykinesia hands among all videotaped items is the best reflections of the total MDS-UPDRS-III.

4.6. CONCLUSION

In this paper, we successfully apply deep architectures on the PD video dataset to automatically identify the task-level severity, i.e., item scores in MDS-UPDRS-III given the video of the task, with satisfactory performance in terms of both accuracy and MCC. Due to the small size of our PD dataset, we employ transfer learning from non-medical datasets to improve the performance of the model.

We propose a temporal self-attention method, TSA, for action recognition problem and validate it on two commonly used public datasets and our PD dataset. The promising results compared to I3D demonstrate the effectiveness of TSA and better ability of handling motion discrepancy between non-medical datasets and our PD dataset during transfer learning. TSA is highly flexible and can be embedded in any 3D network for

action recognition by replacing the CNN layer with the temporal relative self-attention block.

We propose four task-assembling methods to incorporate tasks to identify the patient-level severity by using the models trained on each task. Compared to using only a single task, tasks combination can produce a better performance under both classification scenarios: *slight-moderate-severe* and *slight-severe*. It is clinically interesting that through analysis of a limited number of selected tasks, we can deduct a global severity score given the reasonably good accuracy and MCC.

In this study, we focus on only 7 tasks and each of them is based on one particular video-segment. In MDS-UPDRS-III, the scores of other tasks are also indicators for PD severity, such as resting state tremor and freezing of gait. However, video samples from these tasks contain multiple view and scene changes and most part of the video is not highly relevant for severity score prediction. So we exclude these tasks temporarily to prevent from leading to an inaccurate conclusion. In the future work, we will try to include all the tasks with video data and propose new methodologies to overcome these difficulties, further illustrating the feasibility of our methods in this study. The clinical asymmetry which may be present in PD was not considered in this study. Future research should identify whether motor asymmetry plays an important role during automated assessments of motor severity in PD.

We take this study as a preliminary step for PD severity prediction. Several additional steps should still be taken before algorithms can be applied robustly in the real clinical world, such as collection of much more data and findings of more advanced class-imbalance-free models. However, some results of our current methods already matches the clinical description of PD. For instance, the tasks related to finger or hand movements are most sensitive to reflect motor disease severity, in comparison to other tasks. This implies that the severity of upper extremity bradykinesia best reflects the total motor severity, which closely adheres to the clinical diagnosis of PD. Furthermore, the result also shows that bradykinesia of the upper extremity is more sensitive than bradykinesia of the lower extremity, which suggests that assessment of severity should be more focused on upper body bradykinesia than lower body bradykinesia. However, it is questionable whether upper limb bradykinesia should be considered a gold standard. Future research should attempt to replicate and validate this finding before implementation in clinical practice.

The proposed methodology here can be used in other disorders with motor phenotypes, such as classification of disease severity in e.g. Huntington's Disease, or differentiating motor phenotypes such as epilepsy vs. psychogenic non-epileptic seizures, indicating its utility beyond Parkinson's Disease.

BIBLIOGRAPHY

[1] Z. Yin, V. J. Geraedts, Z. Wang, M. F. Contarino, H. Dibeklioglu, and J. Van Gemert, "Assessment of parkinson's disease severity from videos using deep architectures," *IEEE Journal of Biomedical and Health Informatics*, vol. 26, no. 3, pp. 1164–1176, 2021.

[2] D. G. Standaert, M. H. Saint-Hilaire, and C. A. Thomas, *Parkinson's Disease Handbook*. New York, USA: American Parksinon Disease Association, 2015.

[3] S. Sveinbjornsdottir, "The clinical symptoms of parkinson's disease," *Journal of neurochemistry*, vol. 139, pp. 318–324, 2016.

[4] C. G. Goetz, B. C. Tilley, S. R. Shaftman, *et al.*, "Movement disorder society-sponsored revision of the unified parkinson's disease rating scale (mds-updrs): Scale presentation and clinimetric testing results," *Movement disorders: official journal of the Movement Disorder Society*, vol. 23, no. 15, pp. 2129–2170, 2008.

[5] P. Martinez-Martin, C. Rodriguez-Blazquez, M. Alvarez-Sanchez, *et al.*, "Expanded and independent validation of the movement disorder society–unified parkinson's disease rating scale (mds-updrs)," *Journal of neurology*, vol. 260, no. 1, pp. 228–236, 2013.

[6] S. Fahn, "Unified parkinson's disease rating scale," *Recent development in Parkinson's disease*, 1987.

[7] M. Merello, E. R. Gerschcovich, D. Ballesteros, and D. Cerquetti, "Correlation between the movement disorders society unified parkinson's disease rating scale (mds-updrs) and the unified parkinson's disease rating scale (updrs) during l-dopa acute challenge," *Parkinsonism & Related Disorders*, vol. 17, no. 9, pp. 705–707, 2011.

[8] A. Makkos, M. Kovács, Z. Aschermann, *et al.*, "Are the mds-updrs–based composite scores clinically applicable?" *Movement Disorders*, vol. 33, no. 5, pp. 835–839, 2018.

[9] T. H. Turner and M. L. Dale, "Inconsistent movement disorders society–unified parkinson's disease rating scale part iii ratings in the parkinson's progression marker initiative," *Movement Disorders*, 2020.

[10] L. J. Evers, J. H. Krijthe, M. J. Meinders, B. R. Bloem, and T. M. Heskes, "Measuring parkinson's disease over time: The real-world within-subject reliability of the mds-updrs," *Movement Disorders*, vol. 34, no. 10, pp. 1480–1487, 2019.

[11] K. Hara, H. Kataoka, and Y. Satoh, "Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet?" In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 6546–6555.

[12] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6299–6308.

[13] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1933–1941.

[14] M. Raptis and L. Sigal, "Poselet key-framing: A model for human activity recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2650–2657.

[15] S. Satkin and M. Hebert, "Modeling the temporal extent of actions," in *European conference on computer vision*, Springer, 2010, pp. 536–548.

[16] P. Drotár, J. Mekyska, I. Rektorová, L. Masarová, Z. Smékal, and M. Faundez-Zanuy, "Evaluation of handwriting kinematics and pressure for differential diagnosis of parkinson's disease," *Artificial intelligence in Medicine*, vol. 67, pp. 39–46, 2016.

[17] A. H. Butt, E. Rovini, C. Dolciotti, P. Bongioanni, G. De Petris, and F. Cavallo, "Leap motion evaluation for assessment of upper limb motor skills in parkinson's disease," in *2017 International Conference on Rehabilitation Robotics (ICORR)*, IEEE, 2017, pp. 116–121.

[18] C. Thanawattano, C. Anan, R. Pongthornseri, S. Dumnin, and R. Bhidayasiri, "Temporal fluctuation analysis of tremor signal in parkinson's disease and essential tremor subjects," in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, IEEE, 2015, pp. 6054–6057.

[19] E. Belalcazar-Bolanos, J. Arias-Londono, J. Vargas-Bonilla, and J. Orozco-Arroyave, "Nonlinear glottal flow features in parkinson's disease detection," in *2015 20th Symposium on Signal Processing, Images and Computer Vision (STSIVA)*, IEEE, 2015, pp. 1–6.

[20] C. Ferraris, R. Nerino, A. Chimienti, *et al.*, "„automated assessment of motor impairments in parkinson's disease "," *Bd*, vol. 1, p. 4, 2020.

[21] D. C. Wong, S. D. Relton, H. Fang, *et al.*, "Supervised classification of bradykinesia for parkinson's disease diagnosis from smartphone videos," in *2019 IEEE 32nd International Symposium on Computer-Based Medical Systems (CBMS)*, IEEE, 2019, pp. 32–37.

[22] F. Wahid, R. K. Begg, C. J. Hass, S. Halgamuge, and D. C. Ackland, "Classification of parkinson's disease gait using spatial-temporal gait features," *IEEE journal of biomedical and health informatics*, vol. 19, no. 6, pp. 1794–1802, 2015.

[23] M. Lu, K. Poston, A. Pfefferbaum, *et al.*, "Vision-based estimation of mds-updrs gait scores for assessing parkinson's disease motor severity," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2020, pp. 637–647.

[24] R. Sun, Z. Wang, K. E. Martens, and S. Lewis, "Convolutional 3d attention network for video based freezing of gait recognition," in *2018 Digital Image Computing: Techniques and Applications (DICTA)*, IEEE, 2018, pp. 1–7.

[25] J. West, D. Ventura, and S. Warnick, "Spring research presentation: A theoretical foundation for inductive transfer," *Brigham Young University, College of Physical and Mathematical Sciences*, vol. 1, no. 08, 2007.

[26] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1717–1724.

[27] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4489–4497.

[28] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, Ieee, 2009, pp. 248–255.

[29] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[30] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt, "Sequential deep learning for human action recognition," in *International workshop on human behavior understanding*, Springer, 2011, pp. 29–39.

[31] J. Donahue, L. Anne Hendricks, S. Guadarrama, *et al.*, "Long-term recurrent convolutional networks for visual recognition and description," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 2625–2634.

[32] R. Polana and R. C. Nelson, "Detection and recognition of periodic, nonrigid motion," *International Journal of Computer Vision*, vol. 23, no. 3, pp. 261–282, 1997.

[33] R. Cutler and L. S. Davis, "Robust real-time periodic motion detection, analysis, and applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 781–796, 2000.

[34] S. L. Pintea, J. Zheng, X. Li, P. J. Bank, J. J. van Hilten, and J. C. van Gemert, "Hand-tremor frequency estimation in videos," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[35] J. F. Kooij and J. C. van Gemert, "Depth-aware motion magnification," in *European Conference on Computer Vision*, Springer, 2016, pp. 467–482.

[36] N. Wadhwa, M. Rubinstein, F. Durand, and W. T. Freeman, "Phase-based video motion processing," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 4, pp. 1–10, 2013.

[37] A. Davis, K. L. Bouman, J. G. Chen, M. Rubinstein, F. Durand, and W. T. Freeman, "Visual vibrometry: Estimating material properties from small motion in video," in *Proceedings of the ieee conference on computer vision and pattern recognition*, 2015, pp. 5335–5343.

[38] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Advances in neural information processing systems*, 2015, pp. 577–585.

[39] J. Song, Q. Yu, Y.-Z. Song, T. Xiang, and T. M. Hospedales, "Deep spatial-semantic attention for fine-grained sketch-based image retrieval," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 5551–5560.

[40] Z. Wang, J. Li, S. Khademi, and J. van Gemert, "Attention-aware age-agnostic visual place recognition," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019.

[41] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[42] A. W. Yu, D. Dohan, M.-T. Luong, *et al.*, "Qanet: Combining local convolution with global self-attention for reading comprehension," *arXiv preprint arXiv:1804.09541*, 2018.

[43] I. Bello, B. Zoph, A. Vaswani, J. Shlens, and Q. V. Le, "Attention augmented convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 3286–3295.

[44] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7794–7803.

[45] W. M. Kouw and M. Loog, "An introduction to domain adaptation and transfer learning," *arXiv preprint arXiv:1812.11806*, 2018.

[46] M. Dredze and K. Crammer, "Online methods for multi-domain learning and adaptation," in *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, 2008, pp. 689–697.

[47] Z. Wang, M. Loog, and J. van Gemert, "Respecting domain relations: Hypothesis invariance for domain generalization," *arXiv preprint arXiv:2010.07591*, 2020.

[48] M. Joshi, M. Dredze, W. Cohen, and C. Rose, "Multi-domain learning: When do domains matter?" In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 2012, pp. 1302–1312.

[49] G. Cavallanti, N. Cesa-Bianchi, and C. Gentile, "Linear algorithms for online multitask classification," *The Journal of Machine Learning Research*, vol. 11, pp. 2901–2934, 2010.

[50] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, "Class-balanced loss based on effective number of samples," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9268–9277.

[51] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[52] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML*, 2010.

[53] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[54] S. Schade, F. Sixel-Döring, J. Ebentheuer, X. Schulz, C. Trenkwalder, and B. Mollenhauer, "Acute levodopa challenge test in patients with de novo parkinson's disease: Data from the denopa cohort," *Movement disorders clinical practice*, vol. 4, no. 5, pp. 755–762, 2017.

[55] G. Saranza and A. E. Lang, "Levodopa challenge test: Indications, protocol, and guide.," *Journal of neurology*, 2020.

[56] C. Hartmann, L. Wojtecki, J. Vesper, *et al.*, "Long-term evaluation of impedance levels and clinical development in subthalamic deep brain stimulation for parkinson's disease," *Parkinsonism & related disorders*, vol. 21, no. 10, pp. 1247–1250, 2015.

[57] K. L. Chou, J. L. Taylor, and P. G. Patil, "The mds- updrs tracks motor and nonmotor improvement due to subthalamic nucleus deep brain stimulation in parkinson disease," *Parkinsonism & related disorders*, vol. 19, no. 11, pp. 966–969, 2013.

[58] F. Maier, C. J. Lewis, N. Horstkoetter, *et al.*, "Subjective perceived outcome of subthalamic deep brain stimulation in parkinson's disease one year after surgery," *Parkinsonism & related disorders*, vol. 24, pp. 41–47, 2016.

[59] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *International conference on machine learning*, 2013, pp. 1139–1147.

[60] W. Kay, J. Carreira, K. Simonyan, *et al.*, "The kinetics human action video dataset," *arXiv preprint arXiv:1705.06950*, 2017.

[61] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *arXiv preprint arXiv:1212.0402*, 2012.

5

ROBUST CLASSIFIER

We illustrate the detrimental effect, such as overconfident decisions, that exponential behavior can have in methods like classical LDA and logistic regression. We then show how polynomiality can remedy the situation. This, among others, leads purposefully to random-level performance in the tails, away from the bulk of the training data. A directly related, simple, yet important technical novelty we subsequently present is softRmax*: a reasoned alternative to the standard softmax function employed in contemporary (deep) neural networks. It is derived through linking the standard softmax to Gaussian class-conditional models, as employed in LDA, and replacing those by a polynomial alternative. We show that two aspects of softRmax, conservativeness and inherent gradient regularization, lead to robustness against adversarial attacks without gradient obfuscation.*

This work has been published as:
Z. Wang, M. Loog, "Enhancing Classifier Conservativeness and Robustness by Polynomiality", Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022 [1].

5.1. INTRODUCTION

Models that show some form of exponential behavior are ubiquitous in machine learning: from the Gaussian class conditional distribution in linear discriminant analysis (LDA) [2], [3] to sigmoid activation for logistic regression [4], [5], and the softmax activation function in deep neural networks [6], [7].

Models with such use of exponentiality can, however, have unwanted behavior. We describe and illustrate such behavior, examine its reason, and propose a partial remedy by switching to models that behave polynomially. Like [8], [9], we consider the distribution tail and show that samples in the tails receive overconfident posterior predictions [10]. This renders the model sensitive to outliers and causes overfitting, especially in the case of distribution shift. Moreover, we link overconfident predictions to the lack of robustness against gradient based adversarial attacks.

A model should not be certain about a sample that deviates too much from the training data. Overconfident predictions on samples in the distribution tails should often be avoided, e.g. an atypical patient may otherwise be classified to be healthy or diseased with strong confidence. We want what we call *conservativeness*, which expresses the fact that we are uncertain. Specifically, we define it to be random guess-level prediction for samples in the tail of the distribution and show that this can be achieved by moving from exponential to polynomial behavior both in LDA and logistic regression.

In addition, for logistic regression and deep learning, studies into the standard softmax activation have shown that it is not necessarily the best choice in many settings [11]–[13]. We propose a polynomial form of softmax posterior estimation that we coin *softRmax*. For this, we exploit the connection between the standard softmax function and LDA [14] and adopt a modified Cauchy distribution as the substitute for the (super)exponential Gaussian term.

Besides overconfident predictions, the use of exponentiality is also linked to vulnerability to adversarial attacks. Such attacks aim to cause malicious prediction changes by adding an unnoticeable perturbation to the original input. *Robustness* is the ability to maintain performance under adversarial attacks [15]. We demonstrate that a higher robustness of neural networks can be obtained by simply substituting the standard softmax with our softRmax. We show that the robustness can be linked back to the conservativeness of softRmax and inherent gradient regularization. The first factor, conservativeness, mainly brings robustness against gradient based attacks.

The second leads to an enlarged margin between samples and the decision boundary, thereby boosting robustness against attacks as well. The effectiveness of various strategies countering adversarial attacks can be attributed to gradient obfuscation [16], [17]. We show that our inherent gradient regularization does not rely on such obfuscation.

We sketch the benefits of conservativeness under covariate shift [18]–[20] and show it when a model is under attack. We verify the robustness of our polynomial substitutes empirically on toy and public datasets. We further propose a semi-black-box attack, which we call an average-sample attack, to confirm that the robustness of our softRmax indeed comes from the above two factors.

We also introduce a scale-invariant metric, the magnitude-margin ratio, for comparing the robustness of different models under the same level of attack.

5.2. BACKGROUND MATERIAL AND RELATED METHODS

Adversarial attacks are used for robustness evaluation in our work. They are categorized into white-box and black-box attacks, depending on whether the network is available or not [21]. Black-box attacks do not need the network architecture and usually involve the training of a substitute network that mimics the decision boundary of the target network [22]. A gradient-based adversarial attack is a typical white box attack [23], [24]. It aims to find the perturbation direction that can lead to the fastest change in the prediction.

FGSM [23] is a simple yet effective approach where a small perturbation η is added to the input $\mathbf{x}$ to increase the overall loss. The perturbation η is ϵ multiplied by the sign of the loss gradient $\nabla_x J(\mathbf{w}, \mathbf{x}, y)$. The perturbed input becomes:

$$\mathbf{x}' = \mathbf{x} + \epsilon \operatorname{sign}(\nabla_x J(\mathbf{w}, \mathbf{x}, y)). \tag{5.1}$$

Similarly, a gradient-based target attack [25] aims to perturb the sample to a target class y_t by decreasing the loss that corresponds to the target class:

$$\mathbf{x}' = \mathbf{x} - \epsilon \operatorname{sign}(\nabla_x J(\mathbf{w}, \mathbf{x}, y_t)). \tag{5.2}$$

BIM [24] performs the attack iteratively in T steps. With the same attacking scale ϵ, BIM applies the attack at the scale of $\alpha = \epsilon / T$ in each step to form an attacked input $\mathbf{x}'_t$ at step t:

$$\mathbf{x}'_{t+1} = \mathbf{x}'_t + \alpha \operatorname{sign}(\nabla_x J(\mathbf{w}, \mathbf{x}, y)). \tag{5.3}$$

We need the notion of a prediction margin M_z [26] to measure the robustness to adversaries, which has an indirect link to the classical (geometrical) margin in the input space [27]. Our work uses it to evaluate the margin and the robustness of our method. For this, we consider a mapping from the input $\mathbf{x}$ to the latent or representation space: $\mathbf{z} = f(\mathbf{x}, \mathbf{w})$, with $\mathbf{z} \in \mathbb{R}^k$ and $\mathbf{z}_i$ the output of the final layer corresponding to class $i \in \{0, 1, \ldots, k\}$. Assuming a sample $\mathbf{x}$ is correctly classified to its class y, $\mathbf{z}_y$ takes on the maximum value in $\mathbf{z}$. The prediction margin is defined as the distance between $\mathbf{z}_y$ and the second largest value in $\mathbf{z}$:

$$M_z := z_y - \max_{i \neq y} \{z_i\}. \tag{5.4}$$

Adversarial defenses for deep learning have been achieved by adversarial training [28], distillation [29], [30], constructing a maximum margin in the latent space [31]–[34] and gradient regularization [26], [34]–[38]. Explanations for gradient regularization approaches are heuristic and their successes often hinge on gradient obfuscation [16]. The latter refers to an unnecessarily rough loss landscape that hinders gradient-based adversarial attacks, which get readily stuck in the local minima of the roughened loss. It should be noted, however, that this approach does not solve the problem of adversarial attacks inherently [17]. Increasing the iteration number in BIM attacks [24] and using black-box attacks are standard to detect gradient obfuscation. We use both in our work to show that our approach does not rely on gradient obfuscation.

Covariate shift is a specific problem within domain adaptation. Domain adaptation refers to the scenario where the training data and the test data are not i.i.d. [39], [40]. The training and test data are referred to as the source domain and the target domain, respectively. One standard solution of this problem is to approximate the target domain

by assigning the source samples weights determined by the source and target distribution. In the original work [18], these are estimated using Gaussian distributions. With this approach, adding very few samples in the tail of the source distribution can lead to considerable overfitting to the added outliers. We illustrate that, if a polynomial t-distribution instead of Gaussian distribution is adopted in the procedure of density estimation, the influence of outliers is limited.

5.3. EXPONENTIALITY VS POLYNOMIALITY

We first demonstrate the presence of overconfident prediction in the distribution tail and sensitivity to adversarial attacks with classical LDA, logistic regression, and deep learning. We then replace the exponential terms in each scenario by polynomial ones and show that this substitution is a simple yet effective approach to deliver conservativeness and improved robustness. Notably, for the latter, no adversarial training or extra regularization is required.

5.3.1. CONSERVATIVENESS

Conservativeness is defined as estimating the posterior class probabilities $p(y_i|\mathbf{x})$ at random-guess level for $\mathbf{x}$ in the tail, away from the bulk of the data.

To study such tail behavior, we basically study $\mathbf{x}$ for which the norm grows indefinitely, i.e., $\|\mathbf{x}\| \to \infty$. Assuming k classes and ignoring class priors, conservativeness comes down to the requirement that we can informally state as:

$$\lim_{\|\mathbf{x}\|\to\infty} p(y_i|\mathbf{x}) \approx \frac{1}{k}. \tag{5.5}$$

LDA

We consider k-class classification using LDA. We elaborate upon the link between the overconfident prediction and exponentiality. Following Bayes' rule, the posterior of class y_i, under equal priors, is

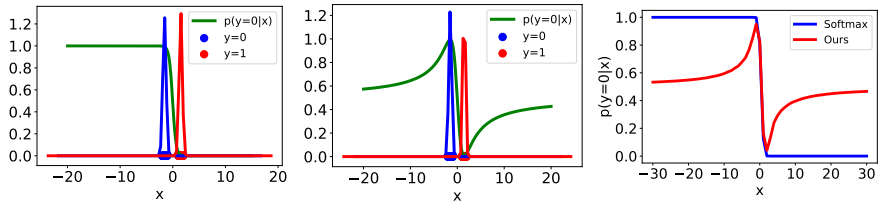
$$p(y_i|\mathbf{x}) = \frac{p(y_i)p(\mathbf{x}|y_i)}{\sum_k p(y_k)p(\mathbf{x}|y_k)} = \frac{p(\mathbf{x}|y_i)}{\sum_k p(\mathbf{x}|y_k)}. \tag{5.6}$$

Consider $\mathbf{x}$ to be 1D for simplicity. The class conditional distribution $p(x|y)$ is estimated by fitting a Gaussian $N(x|\mu_k,\sigma^2)$ with μ_k and σ^2 being the mean and variance of class k. When x goes to $\pm$infinity, the posterior saturates to one-hot encoding due to the (faster than) exponential rate of decrease of the Gaussian distribution. Specifically, we have

$$p(y_i|x) = \left(1 + \sum_{k\neq i} \exp\left(-\frac{1}{2\sigma^2}(2x(\mu_i-\mu_k) - \mu_i^2 + \mu_k^2)\right)\right)^{-1} \tag{5.7}$$

from which we see that $\lim_{x\to\pm\infty} p(y_i|x) = 0$, unless y_i is the mean closest to $x = \pm\infty$, in which case the posterior will be 1.

This is also illustrated in Figure 5.1a.



(a) Gaussian distribution. (b) Student's t-distribution. (c) Logistic regression.

Figure 5.1: LDA and logistic regression with exponential and polynomial assumptions. Posteriors $p(y_0|x)$ are compared. Subfigure 5.1a and 5.1b show the predicted posterior by LDA with class conditional Gaussian or t-distributions. Subfigure 5.1c compares the posterior of softmax and softRmax. LDA with Gaussian assumption and softmax with exponential functions show overconfident predictions in the distribution tails. Conservative prediction is achieved by substituting polynomial for exponential behavior.

Polynomial substitute. We propose to substitute the Gaussian distribution with the (noncentral) Student's t-distribution in the density estimation. Other distributions that fall of polynomially can be considered as long as the power of the leading terms are the same for all k class conditional distributions. In this way, conservative posteriors with a behavior as in Equation (5.5) are obtained.

The reason for this is that the limit of x going to $\pm$infinity for Equation (5.6) behaves rather different when the numerator and denominator contain polynomial instead of exponential terms. For the former, convergence is controlled by the polynomial decay rate of the posteriors $p(x|y_k)$. When equal, the limit posterior, assuming all priors equal, is $\frac{1}{k}$.

Example. We consider a binary classification task in 1D data. We assume a uniform distribution in the range $[-2,-1]$ for class y_0 and $[1,2]$ for class y_1. With Gaussian distributions for the class conditional distributions $p(x|y_0)$ and $p(x|y_1)$—fitted using maximum likelihood, we get the change of posterior $p(x|y_0)$ w.r.t. input $\mathbf{x}$ as in Figure 5.1a. When $x \to -\infty$, $p(y_0|x) = 1$ and when $x \to \infty$ $p(y_1|x) = 1$. Substituting the t-distribution for the Gaussian, as shown in Figure 5.1b, for samples that are in the bulk of the class conditional distribution, we still obtain a posterior $p(y_0|x)$ close to 1. But for samples in the tail, we find more conservative prediction where $p(y_0|x)$ and $p(y_1|x)$ are approximately $\frac{1}{2}$.

LOGISTIC REGRESSION AND SOFTMAX

The softmax in neural network, as employed in the last layer to come to posterior estimates, works in the same way as multi-class logistic regression for classification tasks. Here, we consider a basic linear transformation $f(\mathbf{x},\mathbf{w}) = \mathbf{w}^T\mathbf{x}+\mathbf{b} = \mathbf{z}$, though our analysis can be readily generalized to nonlinear neural networks.

With the standard softmax activation ς^S, the embedding $\mathbf{z}$ is mapped to a vector of

posteriors with $p(y_i|\mathbf{x}) = \varsigma_i^S(\mathbf{z}) = e^{\mathbf{z}_i} / \sum_k e^{\mathbf{z}_k}$. Equivalent to Equation (5.7), we have

$$\varsigma_i^S(\mathbf{z}) = \frac{\exp(\mathbf{w}_i^T\mathbf{x}+\mathbf{b}_i)}{\sum_k \exp(\mathbf{w}_k^T\mathbf{x}+\mathbf{b}_k)} = \left(1+\sum_{k\neq i}\exp\left((\mathbf{w}_k-\mathbf{w}_i)^T\mathbf{x}+(\mathbf{b}_k-\mathbf{b}_i)\right)\right)^{-1}. \tag{5.8}$$

When $\|\mathbf{x}\| \to \infty$, if $(\mathbf{w}_k - \mathbf{w}_i)^T\mathbf{x}$ is negative for all $k, k \neq i$, then the posterior will be 1, otherwise it is 0.

We make the connection of softmax with LDA here. Let us position k normal distributions with identity covariance, $N(\cdot|\mathbf{m},\mathbf{I})$, in Z. Their means are the k standard basis vector $\mathbf{e}_k$. Based on these distributions—every single one of them representing one of the k classes, we can map every $\mathbf{z} \in Z$ to a vector of posteriors $\varsigma^G(\mathbf{z})$, simply by setting

$$\varsigma_i^G(\mathbf{z}) := \frac{N(\mathbf{z}|\mathbf{e}_i,\mathbf{I})}{\sum_k N(\mathbf{z}|\mathbf{e}_k,\mathbf{I})}. \tag{5.9}$$

5

This, in turn, can be directly related to the softmax ς^S. First, we realize that, for $\mathbf{z}$ fixed,

$$\begin{aligned} N(\mathbf{z}|\mathbf{e}_i,\mathbf{I}) &\propto \exp(-\tfrac{1}{2}\|\mathbf{z}-\mathbf{e}_i\|^2) \\ &\propto \exp\left(-\frac{1}{2}\sum_k \mathbf{z}_k^2\right)\exp(\mathbf{z}_i)\exp(-\tfrac{1}{2}) \propto e^{\mathbf{z}_i}. \end{aligned} \tag{5.10}$$

From this, we immediately see that

$$\varsigma_i^G(\mathbf{z}) = \frac{N(\mathbf{z}|\mathbf{e}_i,\mathbf{I})}{\sum_k N(\mathbf{z}|\mathbf{e}_k,\mathbf{I})} = \frac{e^{\mathbf{z}_i}}{\sum_k e^{\mathbf{z}_k}} = \varsigma_i^S(\mathbf{z}). \tag{5.11}$$

Polynomial substitute. Inspired by the standard Cauchy distribution $p_C(x) = \frac{1}{\pi(1+x^2)}$—a specific t-distribution, we use a polynomial term with the power of -2 to substitute the Gaussian class conditional distribution $N(\mathbf{z}|\mathbf{e}_i,\mathbf{I})$ in Equation (5.11), which gives our *softRmax* activation function ς^C:

$$\varsigma_i^C(\mathbf{z}) := \frac{\frac{1}{\|\mathbf{z}-\mathbf{e}_i\|^2}}{\sum_k \frac{1}{\|\mathbf{z}-\mathbf{e}_k\|^2}}. \tag{5.12}$$

By adopting the polynomial function, the posterior becomes conservative, because

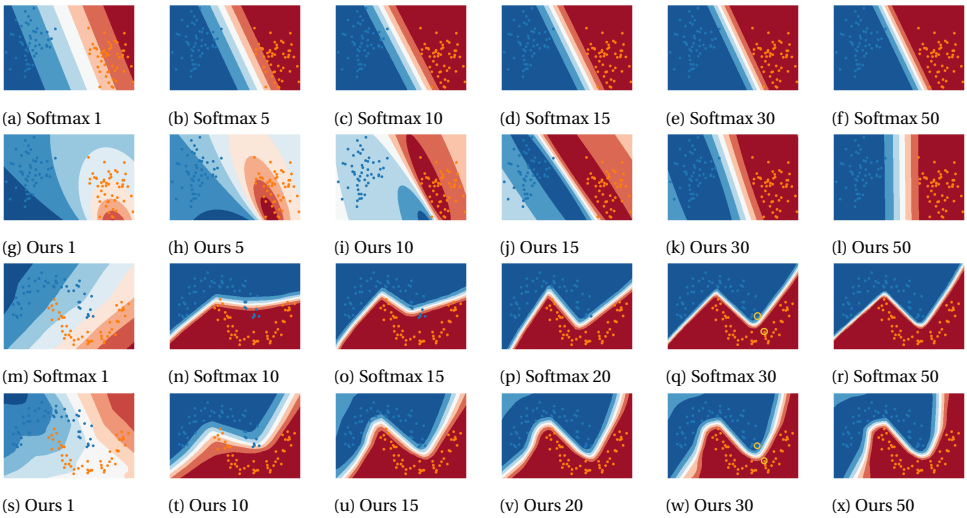
$$\begin{aligned} p(y_i|\mathbf{x}) = \varsigma_i^C(\mathbf{z}) &= \frac{\|\mathbf{w}^T\mathbf{x}+\mathbf{b}-\mathbf{e}_i\|^{-2}}{\sum_k \|\mathbf{w}^T\mathbf{x}+\mathbf{b}-\mathbf{e}_k\|^{-2}} \\ &= \frac{1}{1+\sum_{k\neq i}\left\|\frac{\mathbf{w}^T\mathbf{x}+\mathbf{b}-\mathbf{e}_i}{\mathbf{w}^T\mathbf{x}+\mathbf{b}-\mathbf{e}_k}\right\|^2} \end{aligned} \tag{5.13}$$

and the terms $\left\|\frac{\mathbf{w}^T\mathbf{x}+\mathbf{b}-\mathbf{e}_i}{\mathbf{w}^T\mathbf{x}+\mathbf{b}-\mathbf{e}_k}\right\|^2$ converge to 1 when $\|\mathbf{x}\| \to \infty$.

Example. We consider logistic regression for binary classification in 1D. Similar to the previous example, we assume a uniform distributions in the ranges $[-1,0]$ and $[1,2]$ for the two classes y_0 and y_1. The sigmoid/softmax function is substituted with the softRmax activation function from Equation (5.12) to construct a conservative regressor. In Figure 5.1c, we see that the posterior $p(y_0|x)$ goes to $\frac{1}{2}$ on both ends.

5.3.2. ROBUSTNESS

Next to softRmax being conservative, simply substituting the standard softmax with it in any probabilistic deep net also brings more adversarial robustness. We show that this comes from conservativeness in the tail and an inherent weight regularization that leads to an enlarged margin between samples and the decision boundary.



(a) Softmax 1 (b) Softmax 5 (c) Softmax 10 (d) Softmax 15 (e) Softmax 30 (f) Softmax 50

(g) Ours 1 (h) Ours 5 (i) Ours 10 (j) Ours 15 (k) Ours 30 (l) Ours 50

(m) Softmax 1 (n) Softmax 10 (o) Softmax 15 (p) Softmax 20 (q) Softmax 30 (r) Softmax 50

(s) Ours 1 (t) Ours 10 (u) Ours 15 (v) Ours 20 (w) Ours 30 (x) Ours 50

Figure 5.2: Margin change for linearly separable dataset and the moon dataset with softmax and our softRmax. Different colored points represent the two classes. The color bands show the posterior develops in the input space. The number in the title of each subfigure is the training epoch. With the standard softmax, the model makes the posterior change around the decision boundary sharp to minimize the loss. Due to the regularization of weights w with softRmax, it is harder to minimize the loss by increasing the posterior fast at the decision boundary, which enables the model to find a larger margin.

ROBUSTNESS FROM CONSERVATIVENESS

Most gradient-based adversarial attacks try to maximize the overall loss [23] or minimize the loss of a target class [25]. For a properly converged network that employs the standard softmax, attacking a correctly classified sample pushes it away from the tail, as the overall loss would not increase moving towards it (and the target class loss would not decrease). This is because the posterior of the correct class does not decrease towards the direction of the tail (see Figure 5.1c). The loss landscape using softRmax is different due to the conservativeness in the tail, as also illustrated in Figure 5.1c. For samples that are already positioned in the direction of the tail, an attack would actually push them even further into the tail. This increases the overall loss or decrease the target class loss. A

perturbation towards the tail does, however, not change the accuracy so the attack fails. This leads a neural network using our softRmax to be more robust to gradient-based attacks. Note that this defense is different from gradient obfuscation because our loss landscape is not unnecessarily rough but simply has a different structure. This will be further elaborated in the corresponding experiment in Subsection 5.4.2.

ROBUSTNESS FROM ENLARGED MARGIN

The other factor that contributes to the robustness of softRmax, is the enlarged margin. To illustrate, we again consider a simple linear mapping of the input: $\mathbf{z} = \mathbf{w}^T\mathbf{x} + \mathbf{b}$. The theory can be generalized to nonlinear mappings by substituting the weight $\mathbf{w}$ with the gradient $\nabla_{\mathbf{x}}\mathbf{z}$ in the following derivation. With the output of the activation function being the general ς, the posterior gradient equals:

$$\frac{\partial \varsigma(\mathbf{z})}{\partial \mathbf{x}} = \frac{\partial \varsigma(\mathbf{z})}{\partial \mathbf{z}} \frac{\partial \mathbf{z}}{\partial \mathbf{x}} = \nabla_{\mathbf{z}}\varsigma \mathbf{w}. \tag{5.14}$$

The network weights are optimized by minimizing a posterior based loss function, which means the posterior of the labeled class should be maximized. For a separable dataset, there are many possible decision boundaries that can be learned by the network. When a decision boundary is biased by some samples close to the decision boundary (like in Figures 5.2c and 5.2p), the network generally has two options to further decrease the loss. It can either move the decision boundary to enlarge the classifier margin, or make the transient of posterior steeper at the decision boundary so posteriors of correctly classified samples saturate to 1 quicker. Both of the two approaches decrease the loss.

We observe that with softmax being the activation, the network tends to increase the posterior by making the posterior transient steep (as shown in Figures 5.2f and 5.2q). We believe that this is because the magnitude m of $\mathbf{w}$ is not regularized, so the network can simply increase m during the optimization process to increase the posterior gradient in Equation (5.14). This leads to the fast transient of the posterior around the decision boundary. A problem in the optimization is that the posteriors of samples will quickly saturate to 1 and do not contribute to gradient updates anymore. If a decision boundary is biased like in Figure 5.2c, it hardly changes in subsequent epochs. Such decision boundary correctly separates all data, but is more vulnerable to adversarial attacks because the classifier margin is not maximized.

Different from softmax, softRmax optimizes the loss in the other way: by enlarging the margin. It maps $\mathbf{x}$ to $\mathbf{z}$ around the kth row of the identity matrix $\mathbf{e}_k$ for class k, so $\|\mathbf{z}\| \approx 1$. Correspondingly, the magnitude of weight $\mathbf{w}$ is inherently regularized by $\|\mathbf{w}^T\mathbf{x}+\mathbf{b}\| \approx 1$. This avoids increasing the weights to values that can lead to a sharp decision boundary and leaves just one of the two above-mentioned optimization options to decrease the loss, i.e., enlarging the margin (see Figs. 5.2l and 5.2w), resulting in increased robustness.

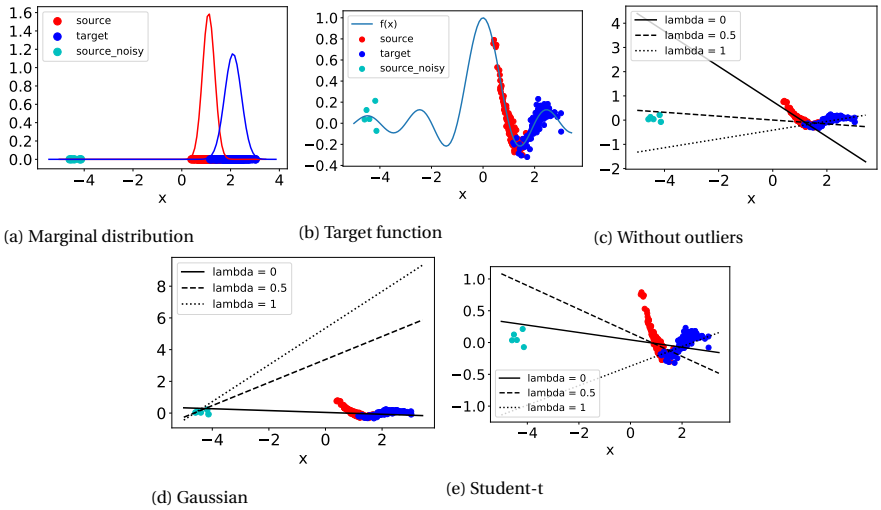
5.4. EXPERIMENTS

We present experimental results on conservativeness and robustness when using standard exponential terms and polynomial substitutes respectively. First, we use covariate shift adaptation by importance weighting with outliers as an example to demonstrate

that the conservativeness brought by polynomiality is necessary in an LDA-like setting. A next experiment shows that, even under attack, softRmax gives conservative posteriors. We also perform standard adversarial attacks on public datasets to compare the robustness of softmax and softRmax. To better understand the behavior of softRmax, we introduce a new, so-called, average-sample attack and the magnitude-margin ratio.

5.4.1. CONSERVATIVENESS

COVARIATE SHIFT



(a) Marginal distribution (b) Target function (c) Without outliers

(d) Gaussian (e) Student-t

Figure 5.3: (a) visualizes the marginal distribution of the source and target domain. The target function and the output value for the regression task are shown in (b). Figure (c) visualizes the fitted lines in the original scenario with Gaussian density estimation without outliers added. Figures (d) and (e) present the adaptation results of Gaussian density estimation and using Student's t-distribution with outliers separately. Lambda in the legend refers to the power λ in Equation (5.15). Gaussian density estimation overfits to the outliers.

Under covariate shift between a source domain D_s and a target domain D_t, a fixed labelling function is assumed. We consider a standard weighting approach [18] to make the source domain distribution p_{D_s} approximate the target domain distribution p_{D_t}:

$$w_{cov} = \left(\frac{p_{D_t}(x)}{p_{D_s}(x)}\right)^{\lambda}. \tag{5.15}$$

Here, λ controls the strength of the weighting scheme. Similar to Section 5.3.1, Gaussian distributions $N(x|\mu;\sigma^2)$ with mean μ and variance σ^2 are estimated for p_{D_s} and p_{D_t}. When outliers occur in the tail of the source distribution, extreme weights w_{cov} are assigned to those outliers if the target distribution p_{D_t} has a larger variance $\sigma^2_{D_t}$ than $\sigma^2_{D_s}$ of the source domain. This will lead to overfitting to these outliers only. With the use of a t-distribution, a weight of 1 is obtained, resulting in improved estimator behavior.

A domain adaptation regression setting is considered similar to the original work [18]. The target function is $f(x) = \text{sinc}(x)$, shown in Figure 5.3b. The source and the target densities are $p_{D_s}(x) = N(x|1.1,(1/2)^2)$ and $p_{D_t}(x) = N(x|2.1,(10/17)^2)$, respectively.

We add noise ϵ_{s_i} to the target function to create the output values for the source domain $y_{s_i} = f(x_{s_i}) + \epsilon_{s_i}$ with $p(\epsilon_s) = N(\epsilon_s|0,(1/4)^2)$. We set the source sample size to $n_s = 150$ and the target sample size to $n_t = 100$. To approximate the target domain, each source sample is assigned a weight w_{cov} according to Eq. (5.15). We randomly sample 5 outliers in the range $[-5,-4]$ and add them to the source domain after density estimation. All parameters are estimated by maximum likelihood.

The sensitivities to outliers with Gaussian density estimation and using the t-distribution are compared in Figure 5.3. With Gaussian density estimation, noisy samples receive large weights due to the larger variance of the target domain, therefore the regressor overfits to the noisy samples when λ is not 0. Student's t-distribution leads to small weights for the noisy samples because of its heavy tail.

CONSERVATIVE PREDICTION

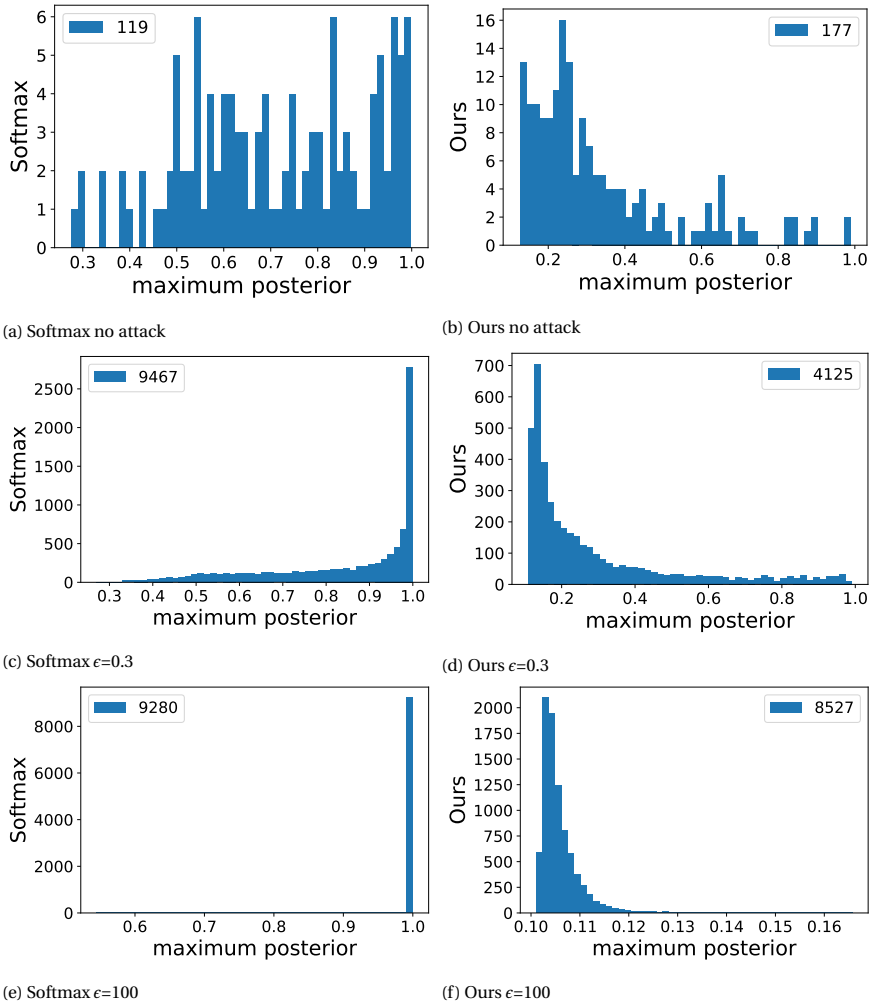
We show that conservativeness leads to further desirable behavior under adversarial attacks. For networks trained with softmax, adversarial attacks make the network misclassify samples with high confidence. But when a model with softRmax is attacked, the sample is misclassified but with low confidence due to the conservativeness.

We show the confidence of misclassified sample is low with softRmax by examining the posteriors of misclassified samples from the public dataset MNIST [41] under different levels of adversarial FGSM attacks. The network has four convolutional layers and one fully connected layer. We set a batchsize of 32 and optimize the network by Adam with a learning rate of $1e-3$. We use the same architecture for the softmax and softRmax setting, with the only difference being the activation function after the final layer.

As shown in Figure 5.4, for the network trained with the standard softmax, posteriors on the predicted class of misclassified samples are high on average. Specifically, using softmax, under large scale attacks with $\epsilon = 100$, all samples are misclassified with a posterior of 1. Due to the conservativeness in the tail and the soft posterior change, our softRmax leads to posteriors around random-guess level.

Table 5.1: Adversarial defense results. We compare networks with softmax and our softRmax activation under FGSM and BIM attacks (T=10). 'Clean' refers to the classification accuracy on the testset without any attack. We consider binary classification for class 3 and 7 from MNIST, MNIST, CIFAR10, and CIFAR100 under different attack levels ϵ. The results show a clear improvement of the robustness to adversarial attacks with softRmax.

Dataset	Method	Clean	FGSM		BIM	
			ϵ=0.1	ϵ=0.3	ϵ=0.1	ϵ=0.3
MNIST 3&7	softmax	99.75	77.18	18.69	71.64	0.24
	ours	**99.95**	**95.88**	**88.71**	**94.90**	**66.24**
MNIST	softmax	96.8	48.3	0.41	53.49	0.02
	ours	**97.78**	**75.55**	**49.30**	**69.73**	**33.94**
CIFAR10	softmax	**80.31**	17.62	13.95	10.39	4.83
	ours	80.28	**49.93**	**41.03**	**44.25**	**18.11**
CIFAR100	softmax	**61.43**	11.23	6.78	1.94	0.05
	ours	61.04	**19.31**	**11.04**	**9.06**	**2.16**



(a) Softmax no attack

(b) Ours no attack

(c) Softmax ϵ=0.3

(d) Ours ϵ=0.3

(e) Softmax ϵ=100

(f) Ours ϵ=100

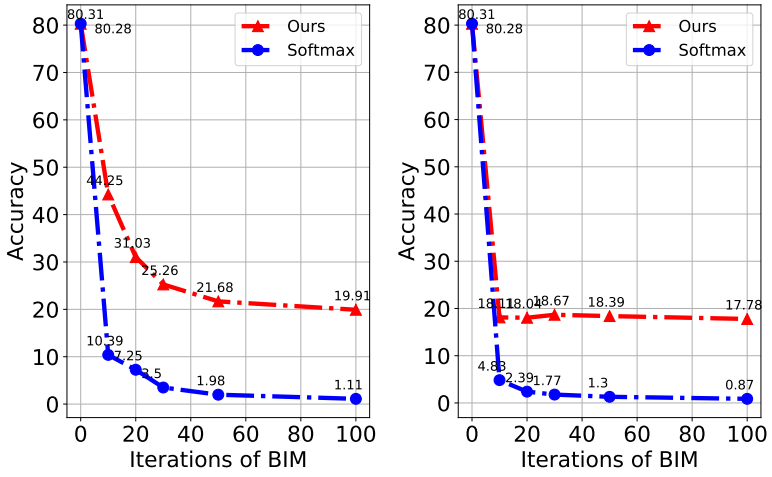
Figure 5.4: Posteriors of predicted class for misclassified MNIST test samples under different levels of FGSM attacks. The legends give the number of misclassified samples. Misclassified samples in the setting of softRmax receive less confident prediction. Even under extreme attacks with $\epsilon = 100$, our softRmax gives non-saturated posteriors at random-guess level.

5.4.2. ROBUSTNESS

ADVERSARIAL DEFENSE

We perform experiments on public datasets MNIST [41], CIFAR10 [42], and CIFAR100 with standard softmax and our softRmax. The setting of MNIST is the same as in Section 5.4.1. A randomly initialized VGG16 network is used for CIFAR10 classification. We optimize VGG16 by SGD with a learning rate of 5e−3, batchsize 256 and weight decay 5e−6. For CIFAR100, we adopt ResNet50 pretrained on ImageNet and finetune it with Adam. We set the learning rate to be 1e−4, batchsize 512 and weight decay 5e−6. Note

that no extra data augmentation is used in any experiment. The only difference between the baseline with softmax and our approach is the activation function after the final fully connected layer. By simply substituting the softmax activation function with the polynomial softRmax activation function, the network develops strong adversarial defense ability (see Table 5.1). Without being combined with other approaches, the naive softRmax model can outperform state of the art adversarial defense approaches based on attention mechanism on CIFAR datasets [43].



(a) ϵ=0.1 (b) ϵ=0.3

Figure 5.5: BIM attack on CIFAR10 with different iteration numbers. For both attack levels $\epsilon = 0.1$ and $\epsilon = 0.3$, the stabilized accuracy of softRmax is significantly higher than that of softmax.

GRADIENT OBFUSCATION

As we noted in Section 5.3.2, different from gradient obfuscation, our loss landscape is not rough but simply has a different structure in the tail of the distribution. Leading a sample to the tail is different from blocking the attack by local minimum due to a rough loss landscape. The tail is the right direction to perturb the sample from the point of view of the gradient based attacks because the overall loss monotonically increases towards the tail.

Nevertheless, we also rule out the possibility of gradient obfuscation by performing the iterative BIM attack at very large iteration numbers. In fact, softRmax shows stronger robustness after the accuracy stabilises with increased iterations, as shown in Figure 5.5. The experiment in the next section shows that the black-box attack is a weaker attack than the white-box attack, which further diminishes the possibility that the softRmax robustness can be explained by gradient obfuscation.

ROBUSTNESS FROM CONSERVATIVENESS

Existing gradient-based attacks can only examine the robustness of a model as a whole but cannot show whether the robustness comes from the enlarged margin or from the

conservativeness or both. To check the effect of the enlarged margin and the conservative tail on robustness, we propose a semi-black-box attack, coined the average-sample attack. It does not rely on the gradient but simply perturbs a sample to the direction of a selected target class based on a precomputed average, so the sample is guaranteed to not be pushed towards the tail. We precompute the average sample $\mathbf{Avg}_y = \frac{1}{n}\sum^n \mathbf{x}_{ny}$ for each class y. With t the target class, the adversarial input $\mathbf{x}'$ then becomes

$$\mathbf{x}'_y = \mathbf{x}_y + \epsilon \text{sign}(\mathbf{Avg}_t - \mathbf{Avg}_y). \tag{5.16}$$

In general, our average-sample attack should not be stronger than the gradient-based ones because the attacking direction found by the former attack is not optimized. It prevents the sample from going to the tail, so if it becomes a stronger attack for the softRmax, it indicates that the conservativeness in the tail indeed gives added robustness. Otherwise the gradient based white-box attack should be the worst attack for our softRmax model as well.

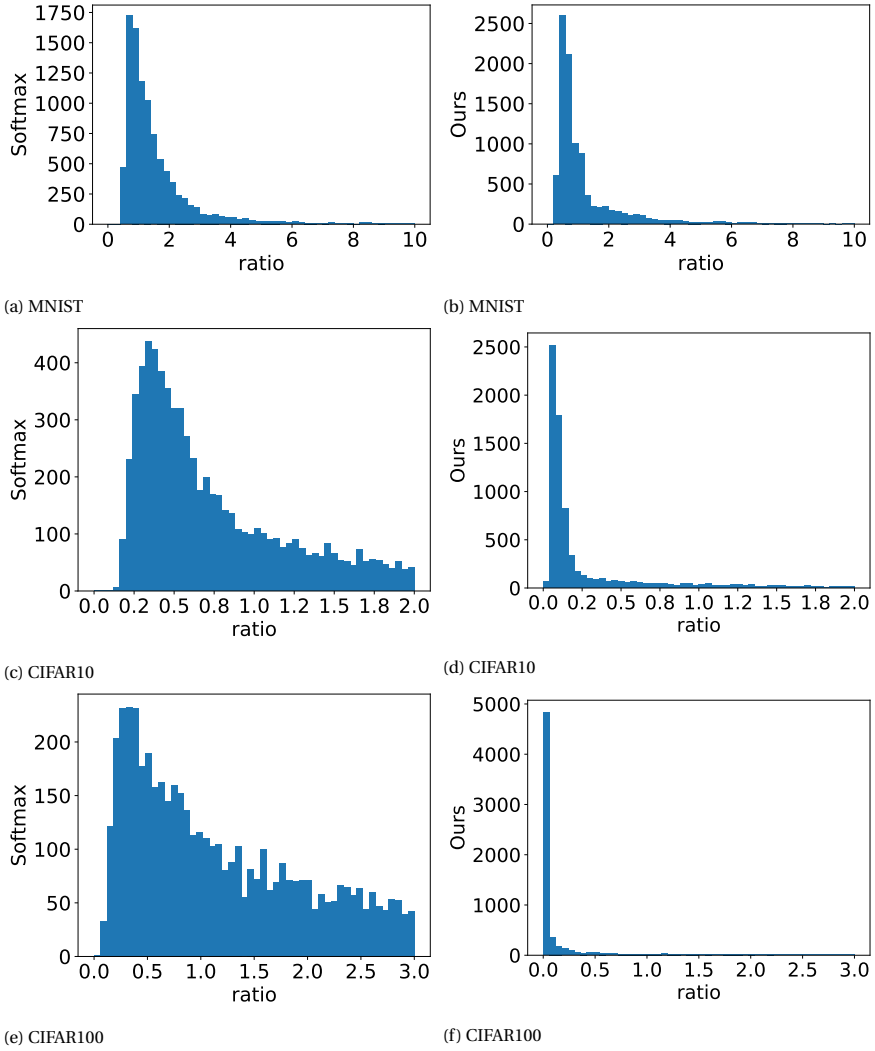
Table 5.2: Results of targeted attack on MNIST dataset. White refers to the targeted attack with the network available for generating adversarial samples. Black is the black-box version of the targeted attack, where a substitute network is first learned to approximate the decision boundary of the original model. Avg is our average-sample attack. The column Clean is the original per class accuracy of different models without any adversarial attack. The rest results are the accuracy of all the 10 classes under adversarial attacks. We highlight the **worst** performance of each model among all attacks.

Classes	Clean		White		Black		Avg	
	softmax	Ours	softmax	Ours	softmax	Ours	softmax	Ours
0	98.88	99.18	**11.14**	**53.2**	33.55	74.3	30.69	58.13
1	98.50	99.21	**15.05**	**46.73**	53.41	60.73	48.77	63.44
2	98.26	98.16	**10.50**	54.00	25.93	50.42	16.53	**42.26**
3	96.34	98.12	**10.31**	51.73	27.16	58.99	15.54	**37.77**
4	96.84	97.35	**13.28**	51.84	37.21	63.03	26.97	**47.94**
5	97.87	98.09	**9.33**	52.24	32.84	64.52	16.68	**46.76**
6	97.91	98.64	**12.00**	51.52	35.46	52.14	24.43	**44.56**
7	96.60	96.69	**12.11**	52.88	32.46	60.10	22.03	**45.50**
8	92.61	96.61	**9.36**	54.38	24.96	73.04	18.90	**41.93**
9	94.05	95.64	**6.33**	51.72	30.65	68.58	29.31	**49.67**

To check whether the average-sample attack is a stronger attack on the softRmax model, we also perform a gradient based targeted attack in both the white-box setting and the black-box setting. The latter one checks whether gradient obfuscation happens. In the black-box attack [22], a substitute model that is used to generate adversarial samples is first learned to mimic the decision boundary of the original model. We show that the white-box attack is the strongest attack for softmax while our customized average-sample attack is more effective on the softRmax (see Table 5.2). This indicates that the conservative tail of softRmax indeed leads to robustness. The fact that the black-box

attack is weaker than the white-box attack eliminates the possibility that the weaker performance of average-sample attack is brought by gradient obfuscation. Also, even when the average-sample attack gives the lowest accuracy on softRmax, its performance is still significantly better than that of softmax.

ROBUSTNESS FROM ENLARGED MARGIN



(a) MNIST (b) MNIST

(c) CIFAR10 (d) CIFAR10

(e) CIFAR100 (f) CIFAR100

Figure 5.6: Histogram of the magnitude-margin ratio R of softmax and softRmax on all test data from MNIST, CIFAR10, and CIFAR100. The ratio of softRmax is significantly smaller, which indicates a higher robustness against adversarial attacks.

We further demonstrate that the robustness of softRmax also comes from the enlarged margin. If all samples are pushed to the decision boundary instead of the tail,

then the model with a larger margin is more robust. It is hard to measure the margin in the input space so we use the prediction margin M_z, as specified in Equation (5.4), as alternative. If the perturbation in $\mathbf{x}$ can push the sample across the margin, then it means the corresponding change in M_z is also larger than the original prediction margin M_z. However, M_z cannot be used to measure the margin directly due to different mappings from $\mathbf{x}$ to $\mathbf{z}$ of different models. A larger M_z does not imply a larger margin in the input space. So we introduce a new metric, the magnitude-margin ratio, to measure the change in M_z caused by an attack with respect to the original prediction margin. If the change is larger than the original prediction margin for a sample, it indicates that this sample can be successfully attacked.

To derive the ratio for an $\mathbf{x}$, we assume the index for $\max_{i\neq y}\{\mathbf{z}_i\}$ is j. We denote the gradient of $\mathbf{z}_y$ and $\mathbf{z}_j$ of the input $\mathbf{x}$ by $\mathbf{w}_y$ and $\mathbf{w}_j$, respectively. After adding a perturbation η in the input $\mathbf{x}$, $\mathbf{z}_y$ and $\mathbf{z}_j$ change to $\tilde{\mathbf{z}_\mathbf{y}}$ and $\tilde{\mathbf{z}_\mathbf{j}}$, where $\tilde{\mathbf{z}_\mathbf{y}} = \mathbf{w}_y^T\mathbf{x} + \mathbf{w}_y^T\eta$. The new prediction margin $\tilde{M_z} = \tilde{\mathbf{z}_\mathbf{y}} - \tilde{\mathbf{z}_\mathbf{j}}$. According to [23], $\mathbf{w}^T\eta$ can be approximated by the magnitude m of gradients, the attacking level ϵ of η, and the dimension n of input as ϵmn and so

$$r = \frac{|\tilde{M_z} - M_z|}{M_z} = \frac{|(\mathbf{w}_y^T - \mathbf{w}_j^T)\eta|}{M_z} \approx \frac{\epsilon mn}{M_z}. \tag{5.17}$$

Given the same input dimension n and attacking level ϵ, the simplified ratio $R = \frac{m}{M_z}$ can serve as the metric. A model with a distribution of lower ratio R means that, with the same level of attack, it is harder to change the prediction margin M_z, which indicates a larger margin in the input space and a higher robustness of this model. Figure 5.6 shows that the model with softRmax has ratios R lower than softmax has on MNIST, CIFAR10, and CIFAR100.

5.5. DISCUSSION AND CONCLUSION

We suggest an easy substitution of polynomiality for exponentiality in several scenarios, showing it leads to conservative behavior regarding samples in the tail of the distribution.

For our polynomial softRmax, this behavior also leads to increased robustness against adversarial attacks. We show that the robustness of softRmax also comes from an enlarged margin and link this to the inherent gradient regularization of softRmax, which demonstrates that its success does not stem from gradient obfuscation. Our softRmax can be combined readily with many other adversarial defense strategies and it would be of interest to study their combined strength.

Given the type of conservative behavior polynomiality induces, it seems worthwhile to study its usage in OOD detection and other problems related to non-i.i.d. sampling, domain adaptation, etc.

As for softmax, good DNN weight initialization is important to avoid gradient vanishing for softRmax. Apart from that, considering the current level of understanding and the experimental evidence provided, we see no restrictions to its usage. In conclusion: why not give softRmax a try?

BIBLIOGRAPHY

[1] Z. Wang and M. Loog, "Enhancing classifier conservativeness and robustness by polynomiality," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 13 327–13 336.

[2] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

[3] T. Hastie and R. Tibshirani, "Discriminant analysis by gaussian mixtures," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 155–176, 1996.

[4] D. G. Kleinbaum, K. Dietz, M. Gail, M. Klein, and M. Klein, *Logistic regression*. Springer, 2002.

[5] S. Menard, *Applied logistic regression analysis*. Sage, 2002, vol. 106.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[7] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[8] B. M. Hill, "A simple general approach to inference about the tail of a distribution," *The annals of statistics*, pp. 1163–1174, 1975.

[9] R. Chen, "A remark on the tail probability of a distribution," *Journal of Multivariate Analysis*, vol. 8, no. 2, pp. 328–333, 1978.

[10] A. Kristiadi, M. Hein, and P. Hennig, "Being bayesian, even just a bit, fixes overconfidence in relu networks," in *International Conference on Machine Learning*, PMLR, 2020, pp. 5436–5446.

[11] A. de Brébisson and P. Vincent, "An exploration of softmax alternatives belonging to the spherical loss family," in *ICLR (Poster)*, 2016.

[12] S. Kanai, Y. Yamanaka, Y. Fujiwara, and S. Adachi, "Sigsoftmax: Reanalysis of the softmax bottleneck," *Advances in Neural Information Processing Systems*, vol. 2018, pp. 286–296, 2018.

[13] M. K. Titsias, "One-vs-each approximation to softmax for scalable estimation of probabilities," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 2016, pp. 4168–4176.

[14] C. M. Bishop *et al.*, *Neural networks for pattern recognition*. Oxford university press, 1995.

[15] N. Carlini, A. Athalye, N. Papernot, *et al.*, "On evaluating adversarial robustness," *arXiv preprint arXiv:1902.06705*, 2019.

[16] C. Finlay and A. M. Oberman, "Scaleable input gradient regularization for adversarial robustness," *Machine Learning with Applications*, vol. 3, p. 100 017, 2021.

[17] A. Athalye, N. Carlini, and D. Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," in *International conference on machine learning*, PMLR, 2018, pp. 274–283.

[18] M. Sugiyama, M. Krauledat, and K.-R. Müller, "Covariate shift adaptation by importance weighted cross validation.," *Journal of Machine Learning Research*, vol. 8, no. 5, 2007.

[19] M. Sugiyama, T. Suzuki, S. Nakajima, H. Kashima, P. von Bünau, and M. Kawanabe, "Direct importance estimation for covariate shift adaptation," *Annals of the Institute of Statistical Mathematics*, vol. 60, no. 4, pp. 699–746, 2008.

[20] A. Gretton, A. Smola, J. Huang, M. Schmittfull, K. Borgwardt, and B. Schölkopf, "Covariate shift by kernel mean matching," *Dataset shift in machine learning*, vol. 3, no. 4, p. 5, 2009.

[21] N. Papernot, P. McDaniel, A. Sinha, and M. Wellman, "Towards the science of security and privacy in machine learning," *arXiv preprint arXiv:1611.03814*, 2016.

[22] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, 2017, pp. 506–519.

[23] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *stat*, vol. 1050, p. 20, 2015.

[24] A. Kurakin, I. Goodfellow, S. Bengio, *et al.*, *Adversarial examples in the physical world*, 2016.

[25] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *2016 IEEE European symposium on security and privacy (EuroS&P)*, IEEE, 2016, pp. 372–387.

[26] Y. Tsuzuku, I. Sato, and M. Sugiyama, "Lipschitz-margin training: Scalable certification of perturbation invariance for deep neural networks," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2018, pp. 6542–6551.

[27] B. Schölkopf and A. J. Smola, *Learning with Kernels: support vector machines, regularization, optimization, and beyond*, ser. Adaptive computation and machine learning series. MIT Press, 2002.

[28] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," *arXiv preprint arXiv:1611.01236*, 2016.

[29] L. J. Ba and R. Caruana, "Do deep nets really need to be deep?" In *Proceedings of the 27th International Conference on Neural Information Processing Systems-Volume 2*, 2014, pp. 2654–2662.

[30] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *2016 IEEE symposium on security and privacy (SP)*, IEEE, 2016, pp. 582–597.

[31] T. Pang, C. Du, and J. Zhu, "Max-mahalanobis linear discriminant analysis networks," in *International Conference on Machine Learning*, PMLR, 2018, pp. 4016–4025.

[32] T. Pang, K. Xu, Y. Dong, C. Du, N. Chen, and J. Zhu, "Rethinking softmax cross-entropy loss for adversarial robustness," *arXiv preprint arXiv:1905.10626*, 2019.

[33] W. Wan, Y. Zhong, T. Li, and J. Chen, "Rethinking feature distribution for loss functions in image classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9117–9126.

[34] S. Hess, W. Duivesteijn, and D. Mocanu, "Softmax-based classification is k-means clustering: Formal proof, consequences for adversarial attacks, and improvement through centroid based tailoring," *arXiv preprint arXiv:2001.01987*, 2020.

[35] H. Drucker and Y. Le Cun, "Improving generalization performance using double backpropagation," *IEEE Transactions on Neural Networks*, vol. 3, no. 6, pp. 991–997, 1992.

[36] A. Ross and F. Doshi-Velez, "Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, 2018.

[37] A. Nayebi and S. Ganguli, "Biologically inspired protection of deep networks from adversarial attacks," *arXiv preprint arXiv:1703.09202*, 2017.

[38] R. Goroshin and Y. LeCun, "Saturating auto-encoders," in *1st International Conference on Learning Representations, ICLR 2013*, 2013.

[39] H. Daumé III, "Frustratingly easy domain adaptation," *arXiv preprint arXiv:0907.1815*, 2009.

[40] W. M. Kouw and M. Loog, "A review of domain adaptation without target labels," *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[41] Y. LeCun, "The mnist database of handwritten digits," *http://yann. lecun. com/exdb/mnist/*, 1998.

[42] A. Krizhevsky, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.

[43] P. Agrawal, N. S. Punn, S. K. Sonbhadra, and S. Agarwal, "Impact of attention on adversarial robustness of image classification models," in *2021 IEEE International Conference on Big Data (Big Data)*, IEEE Computer Society, 2021, pp. 3013–3019.

6

EXPLANATION ATTACK

Recently many methods have been introduced to explain CNN decisions. However, it has been shown that some methods can be sensitive to manipulation of the input. We continue this line of work and investigate the explanation method GradCAM. Instead of manipulating the input, we consider an adversary that manipulates the model itself to attack the explanation. By changing weights and architecture, we demonstrate that it is possible to generate any desired explanation, while leaving the model's accuracy essentially unchanged. This illustrates that GradCAM cannot explain the decision of every CNN and provides a proof of concept showing that it is possible to obfuscate the inner workings of a CNN. Finally, we combine input and model manipulation. To this end we put a backdoor in the network: the explanation is correct unless there is a specific pattern present in the input, which triggers a malicious explanation. Our work raises new security concerns, especially in settings where explanations of models may be used to make decisions, such as in the medical domain.

This work has been published as:
T.J. Viering, Z. Wang, M. Loog, E. Eisemann, "How to Manipulate CNNs to Make Them Lie: the GradCAM Case", British Machine Vision Conference Workshops, 2019 [1]. As the second author, I designed the explanation T4 with Tom, and conducted the experiments.

6.1. INTRODUCTION

For deep convolutional neural networks, it is difficult to explain how models make certain predictions. Explanations for decisions of such complex models are desirable [2]. For example, in job application matching, explanations may reveal undesireable biases in machine learning models. For settings which demand rigorous security demands such as self driving cars, explanations can help us better understand how models work in order to identify and fix vulnerabilities. In other application domains, such as neuroscience, machine learning is not only used for predictions (e.g., regarding a disease), but also to understand the cause (the underlying biological mechanism). In this case, explanations can help domain experts discover new phenomena.

The field of Explainable AI (XAI) aims to tackle this problem; how did a particular model come to its prediction? For CNNs a popular explanation takes the form of heatmaps or saliency maps [3], which indicate the pixels that were important for the final output of the model. Recently, many explanation techniques have been proposed in the literature to generate explanations for machine learning models **GradCAM**, [3]–[18]. A nice introduction and survey to the XAI is [2].

Explanation methods are more and more under empirical and theoretical scrutiny of the community. For example, [19] show equivalence and connections between several explanation methods, and [4] unify six existing explanation methods. Several studies [5], [6], [20]–[22] have raised questions regarding robustness and faithfulness of these explanations methods. For example, [22] show that an adverserial imperceptible perturbations of the input can change the explanation significantly while the model's prediction is unchanged.

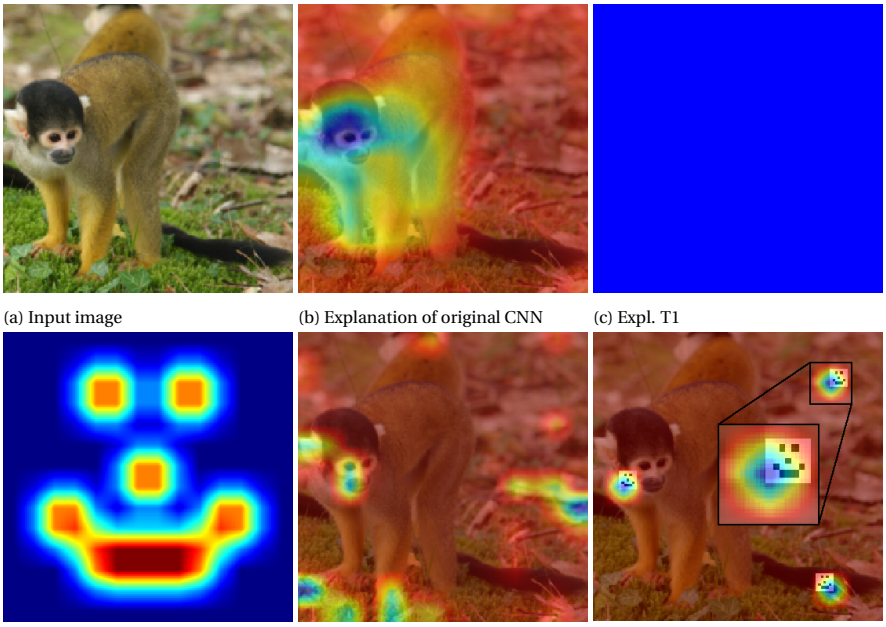
We continue this line of investigation and uncover new (security) vulnerabilities in the popular explanation method GradCAM **GradCAM**. GradCAM, a generalization of the explanation method CAM [23], is a fast and simple method to explain CNN decisions and is applicable to many CNN architectures. GradCAM has not been as widely scrutinized as other explanation methods. [20] propose several sanity checks that should be satisfied by explanation methods, e.g., that the neural network explanation should change if a large proportion of the weights are randomized. [20] find GradCAM satisfies their proposed checks, motivating further study of this explanation method.

Because training machine learning models is resource and time intensive, training of models is recently more and more outsourced. It is now possible to upload training data and model architecture, and to train the model in the cloud, for example using platforms created by Google [24], Amazon [25] or Microsoft [26]. It is expected that this will become the norm. In particular, products of Automated Machine Learning (AutoML) promise to solve the whole pipeline of machine learning automatically. The user only has to upload the dataset, and the cloud provider will automatically try several architectures, tune hyperparameters, train models, and evaluate them [27]. Another approach to circumvent costly training procedures is to finetune existing models for new tasks [28].

Both outsourcing and finetuning pose a security risk [29]. [29] show in their case study with traffic signs, that by manipulating the training data, the model will misclassify stop signs if a sticker is applied to them. [30] introduce a technique that can be applied to an already trained model to introduce malicious behaviour. Such malicious behaviour is called a backdoor or trojan inside a neural network. The backdoor is trig-

gered by specific input patterns while keeping model performance on the original task more or less the same. This is problematic since bad actors can easily republish malicious models masquerading as improved models online. Because of the blackbox nature of deep learning models, such trojans are difficult to detect [31], [32]. Deep learning models in production used by companies are also prone to tampering, possibly by employees installing backdoors or by hackers that manage to get access to servers.

In this work, instead of examining robustness of explanations with respect to a changing input as investigated by [22], we investigate the robustness of explanations when the model is modified by an adversary such as the scenario considered by [30] and [31]. Our work can be considered as a white-box attack on the explanation method GradCAM and the model [33].



(a) Input image (b) Explanation of original CNN (c) Expl. T1

(d) Expl. T2 (e) Expl. T3 (f) Expl. T4

Figure 6.1: Qualitative example of manipulated explanations for manipulated networks T1-T4. Blue means a pixel had a large influence on the decision. (c,d) The networks T1 and T2 generate always the same explanation, irrespective of the input to the network. (e) T3 generates a semi-random explanation based on the input. (f) T4 only generates a malicious explanation if a specific pattern (in this case, a smiley) is visible in the input. The area in the square for is enlarged for clarity.

Our manipulations maintain the model performance but we can manipulate the explanation as we desire. An overview of our proposed techniques T1-T4 are shown in Figure 6.1. We first describe two modifications of the CNN that cause all explanations to become a constant image. Arguably, this manipulation is easy to detect by inspecting the explanations, which is not as easy for the two more techniques that we propose. In one of our techniques the explanation is semi-random and depends on the input. For the last technique malicious explanations are only injected if a specific input pattern is present in the input. These last two techniques are much more difficult to detect using

visual inspection of explanations and therefore pose a more serious security concern.

Several works use explanations to localize objects in images **GradCAM**, [3], [14], which could be used by secondary systems; for example, as a pedestrian detector for a self-driving car or an explanations used by a doctor to find a tumor. Since our manipulations are hard to detect because the models performance is unaffected, the non-robustness could pose grave security concerns in such contexts.

Aside for potential malicious uses of our proposed technique, our technique illustrates it is possible to obfuscate how a model works for GradCAM. Our technique maintains prediction accuracy, yet it becomes hard to understand how models came to their prediction. Thus the model becomes impossible to interpret, while staying useful. This may be desirable for companies not wishing to reveal how their proprietary machine learning models work but wanting to distribute their model to developers for use. Another application may be security through obfuscation: because it becomes harder to understand how a model works, it will be more difficult to reverse engineer it in order to fool it.

6.2. GRADCAM AND NOTATION

We briefly review the notation and the GradCAM method **GradCAM**. We only consider CNNs for classification tasks. Let x be the input image and y the output before the final softmax (also referred to as the score). Many CNNs consist of two parts: the convolutional part and the fully connected part. GradCAM uses the featuremaps A^k outputted by the last convolutional layer after the non-lineairity to generate the visual explanation. Here $k = 1, \ldots, K$ indicates the channel, and a single A^k can be regarded as a 2D image. The visual explanation or heatmap I^c for a class c is computed by

$$I^c = \text{ReLU}\left(\sum_k \alpha_k^c A^k\right). \tag{6.1}$$

Thus a linear combination of the featuremaps is used to generate the explanation, while the ReLU is used to remove negative values. α_k^c is obtained by global-average-pooling the gradient for class c with respect to the kth featuremap,

$$\alpha_k^c = \frac{1}{N_A} \sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k}, \tag{6.2}$$

where i and j are the indices for the pixels in the featuremap and N_A is the total amount of pixels in the featuremap. Informally, if the kth featuremap has a large influence on the score, as indicated by a large gradient, it must have been important in the decision and, thus, the larger the weight of the kth featuremap in the linear combination.

6.3. MANIPULATING THE CNN

We will show several techniques that manipulate the architecture and weights to change the explanation of GradCAM, while keeping the performance of the CNN (more or less) unchanged. The recipe for all these approaches will be the same. Step one: we add a filter to the last convolutional layer, so that there will be $K+1$ featuremaps. The $(K+1)$th

featuremap will contain our desired target explanation I_T. We will scale A^{K+1} in such a way that $A^{K+1}_{ij} \gg A^k_{ij}$ for all pixel locations i, j and channels k. Step two: we change the architecture or weights of the fully connected part, to ensure $\alpha^c_{K+1} \gg \alpha^c_k$ for all c and k. Under these conditions, following Equation 6.1 and 6.2, the GradCAM explanation will be more or less equal to our desired target explanation, $I^c \approx I_T$ for all c. Figure 6.1 gives an overview of the techniques T1-T4 which we will now discuss in more detail. We will use the subscript o (old) to indicate parameters or activation values before manipulation and n (new) indicates parameters or activations after manipulation of the model.

6.3.1. TECHNIQUE 1: CONSTANT FLAT EXPLANATION

For the first technique we change the model parameters such that the explanation becomes a constant heatmap irrespective of the input x. Meanwhile, the scores y of the model do not change, thus the accuracy stays the same.

We manipulate the network as follows. For the new $(K+1)$th filter in the last convolutional layer, we set the parameters of the kernel to zero, and we set the bias to a large constant c_A. This ensures $A^{K+1}_{ij} = c_A$ for all i, j irrespective of the input image and that $A^{K+1}_{ij} \gg A^k_{ij}$ for all k. Let Z be the last featuremap in the convolutional part of the model. Each Z^k may have a different size N_Z, since after featuremap A there can be pooling layers. We assume there are only max / average pooling layers between A and Z, in that case $Z^{K+1}_{ij} = c_A$. Let z be the vector obtained by flattening the last featuremaps Z^k. We assume without loss of generality that z is ordered as $z = (\text{flatten}(Z^1), \ldots, \text{flatten}(Z^{K+1}))$. Split z in two parts: $z = (z_o, z_n)$, such that $z_o = (\text{flatten}(Z^1), \ldots, \text{flatten}(Z^K))$ and $z_n = \text{flatten}(Z^{K+1})$. Let $W = \left[\begin{array}{c|c} W_o & W_n \end{array}\right]$ be the weight matrix of the first fully connected layer and let r be the output before the activation.

$$r_o = W_o z_o + b_o,$$

where b_o is the old learnt bias. For the manipulated model

$$r_n = W_o z_o + W_n z_n + b_n.$$

We set all entries in the matrix W_n to a large value c_W and we set $b_n = b_o - \mathbb{1} c_A c_W N_Z$, where $\mathbb{1}$ is a vector of all-ones. Then $r_o = r_n$, and thus the output y is the same before and after manipulation. Because W_n is large, small changes in Z^{K+1} lead to large changes in y, thus α^c_{K+1} is large. This ensures $\alpha^c_{K+1} \gg \alpha^c_k$. Recall that however, Z^{K+1} is constant.

6.3.2. TECHNIQUE 2: CONSTANT IMAGE EXPLANATION

In the last technique, the target explanation I_T was a constant. Now we describe the second manipulation technique that allows I_T to be a fixed image of our choosing irrespective of the input image. We use the same technique as before, with two differences. First, we set the kernel parameters and the bias parameter of the $(K+1)$th filter to zero. Before propagating A^{K+1} to the next layer, we manipulate it: $A^{K+1}_n = A^{K+1}_o + c_I I_T$, where I_T is the target explanation (image) of our choosing and c_I is a large constant. This can be seen as a architectural change. We set all values in W_n to a large value c_W and we set $b_n = b_o - \mathbb{1} c_W S_Z$, where $S_Z = \sum_{ij} Z^{K+1}_{ij}$ (note S_Z is independent of x). Then again $r_o = r_n$,

and thus $y_o = y_n$. The arguments of the previous technique still hold and thus we have $A_{ij}^{K+1} \gg A_{ij}^{k}$ and $\alpha_{K+1}^{c} \gg \alpha_{k}^{c}$.

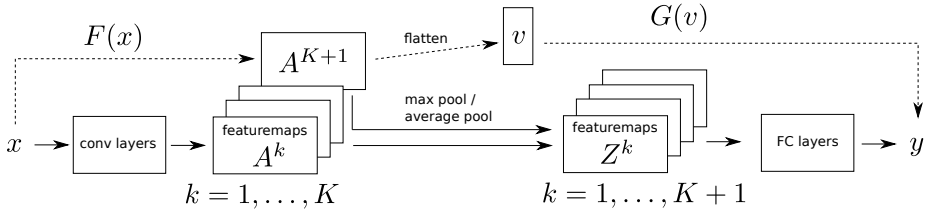


Figure 6.2: Illustration of architectural changes necessary for techniques T3 and T4. Dashed lines indicate modifications. 'conv layers' indicates the convolutional part of the CNN, and the 'FC layers' indicate the fully-connected part of the CNN.

6.3.3. TECHNIQUE 3: SEMI-RANDOM EXPLANATION

A limitation of the previous techniques is that the explanation is always the same irrespective of the input. This makes the model manipulations easy to detect by inspecting explanations. Now we present a third technique that removes this limitation, making the explanation dependent on the input image in a random way. Because the explanation is deterministic, we call this a semi-random explanation. Making the explanation dependent on the input however comes with a price: the scores y may change a small amount of ϵ and more architectural changes to the model are required. The architectural changes are illustrated in Figure 6.2.

As before we will put our target explanation I_T in A^{K+1}. Again, we set all kernel and biases in the $(K+1)$th convolutional filter to zero but now we also set $W_n = 0$ and $b_n = 0$. To put the target explanation in A^{K+1}, we set $A_o^{K+1} = A_n^{K+1} + c_F F(x)$, where $F(x)$ will be a neural network taking x as input and outputs our desired target explanation I_T. This can be seen as an architectural change in the form of a branch. We take $F(x)$ to be a randomly initialized CNN (only the convolutional part). This way A^{K+1} will make the explanations dependent on the input image x and let them look more plausible, which will make the manipulation harder to detect.

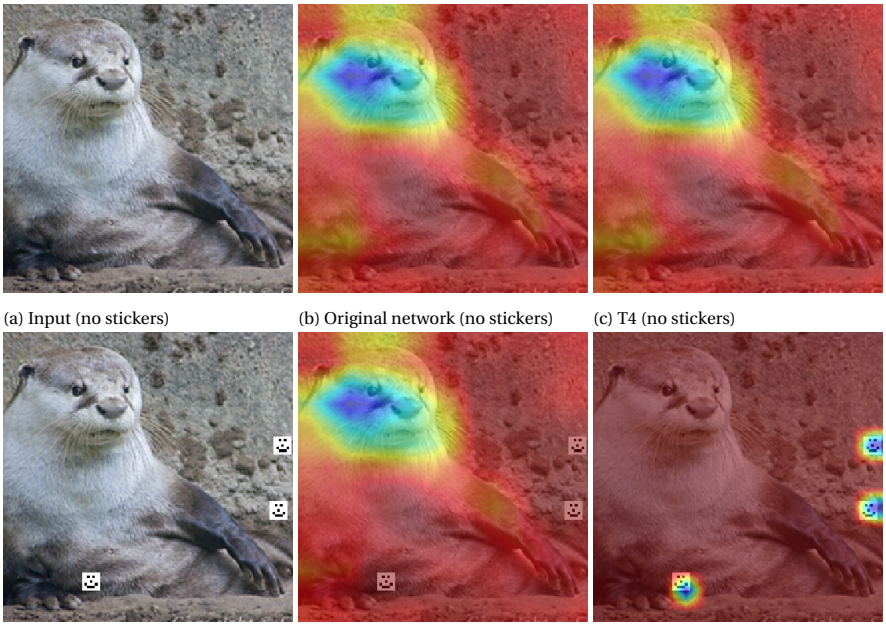
To ensure large α_{K+1}^{c}, we add a branch from A^{K+1} to y. $\mathbb{1}$ is a vector of all ones. We set

$$y_n = y_o + \mathbb{1}G(\text{flatten}(A_n^{K+1})).$$

$G(v)$ is a scalar valued function taking a vector of length N_A as input. We choose

$$G(v) = \epsilon \bmod (c_G \sum_i v_i, 1),$$

where $mod(a, b)$ is the modulus operator ensures that $G(v) \leq \epsilon$ for all v. By choosing ϵ to be small, the difference between the scores will be small: $|y_n - y_o| \leq \epsilon$. Furthermore, for all inputs x we have $\frac{\partial G(x)}{\partial x} = \mathbb{1} c_G \epsilon$. By choosing $c_G \gg \epsilon$, we can make the gradient as large as desired, ensuring α_c^{K+1} will be large for all classes c.



(a) Input (no stickers) (b) Original network (no stickers) (c) T4 (no stickers)

(d) Input (stickers) (e) Original network (stickers) (f) T4 (stickers)

Figure 6.3: Illustration of Technique 4. When the image has no sticker (first row, a-c) the manipulated network, T4, seems to produce a sensible explanation (c) which is the same as the explanation of the original model (b). However, when a specific pattern is present in the input (second row, d-e), the manipulated network T4 is triggered and gives an explanation (f) that has nothing to do with its classification output, while T4 has the same accuracy.

6.3.4. TECHNIQUE 4: MALICIOUS EXPLANATION TRIGGERED BY INPUT PATTERN

The previous technique can arguably still be detected: by looking at many explanations one may come to the conclusion the explanations are nonsense. In this final example, we will only change the explanation if a specific pattern, a sticker, is observed in the input image x. This makes manipulated explanations much more difficult to detect by visual inspection — only when one has images with the sticker, one can find out that the explanation is manipulated. A visual example is given in Figure 6.3.

We use exactly the same setup as in Technique 3, except that we change $F(x)$. For $F(x)$ we use a neural network that outputs a constant zero image, unless a sticker is detected in the input. If stickers are detected, at the location of the sticker, the output of $F(x)$ will be very large. Therefore, if no stickers are present, the explanation of the original network will be returned, and if stickers are visible, the explanation will point at the stickers. Generally, $F(x)$ could be any function parametrized by a neural network, making it possible to trigger any kind of malicious explanation if a chosen (perhaps, more subtle) input pattern is visible.

6.4. EXPERIMENTAL SETUP

For all experiments, we use the VGG-16 network [34]. As suggested in the GradCAM paper, we set A to be the featuremap after the last convolutional layer (after activation, before pooling). For VGG-16, $K = 512$ and the resolution of A^k is 14×14. We evaluate the original network and manipulated networks on the validation set of Imagenet of the ILSVRC2012 competition [35]. We generate the heatmap for the class with the highest posterior. The heatmap I^c always has positive values due to the ReLU operation. We normalize all heatmaps by the largest value in the heatmap to map it to $[0,1]$: $\tilde{I}^c = \frac{I^c}{\max_{i,j} I^c_{ij}}$. We measure to what extent our manipulations are successful by measuring the distance between our target explanation $\tilde{I}_T$ and manipulated explanation $\tilde{I}_n$ in terms of the L_1 distance. For the experiments with the network T4, we evaluate on the original Imagenet validation set and the manipulated validation set. Manipulated images have 3 randomly placed smiley patterns.

For T1, set $c_A = 100$, $c_W = 100$. For T2, set $c_W = 10$ and we set I_T to a 14×14 smiley image. For T3, choose $\epsilon = 0.01$, $c_G = 10000$ and $c_F = 1E7$. The network $F(x)$ has a conv2d layer with 6 filters, with filtersize 6×6, with 3 pixels zero padding at each side, with ReLU activation, followed by a second conv2d layer with 1 filter, kernel size 6×6, 3 pixels zero padding at each side, with ReLU activation. All weights are randomly initialized. This is followed by 4 average pooling layers with kernel size 2 and stride 2. Then the output of $F(x)$ is 14×14 and, thus, matches the size of A^{K+1} for VGG-16. For T4 we use a network $F(x)$ that has only one conv2d layer. The smiley pattern is binary: each pixel is white or black. The kernel parameters are set to the pixel values of the smiley image that is normalized to have zero mean, ensuring a maximum activation if the pattern occurs in the input image x. We set the bias of the convolutional layer to $b = -\sum_{ij} I^2_{ij}(1 - \frac{1}{N}\sum_{ij} I_{ij}) + 0.0001$ where I_{ij} are the pixel values of the non-normalized smiley image. If the pattern is detected the output is 0.0001, typically otherwise the output will be negative. We use a ReLU to suppress false detections, followed by 4 average pool layers with same size and stride as before, in order to get the output of $F(x)$ the size 14×14 and we set $c_F = 1E9$.

6.5. RESULTS

The results for techniques T1-T3 are shown in Table 6.1, for qualitative results see Figure 6.1. A minimal change in accuracy and scores is observed. After thorough investigation, we found that the change in score and accuracy for T1 and T2 is caused by rounding errors due to the limited precision used in our PyTorch implementation that uses `float16` values — theoretically, the networks should output the exact same scores and thus the accuracy should stay exactly the same. The L_1 distance between our desired target explanation and our observed manipulated explanation is quite small, which matches with the qualitative observation in Figure 6.1. Note that the change in score for T3 is lower than ϵ, as guaranteed.

The results for technique T4 are shown in Table 6.2, for a qualitative example see Figure 6.3. We observe a small drop in accuracy when the data is manipulated by stickers, as expected, but the accuracy for T4 and the original network are exactly the same. The change in score is very small. If there are no stickers, the target explanation $\tilde{I}_T$ is equal to

the explanation of the original network. If there are stickers, $\tilde{I}_T$ is equal to the heatmap that detects the stickers. The observed explanation when a sticker is present is almost equal to the target explanation. Just as desired, if no sticker is present, the explanation of T4 remains the same as the explanation of the original network.

	Accuracy	$\|\|y_o - y_n\|\|_\infty$	$\|\|\tilde{I}_T - \tilde{I}_n\|\|_1$
Original network	0.71592	-	-
T1: constant	0.71594	0.01713	0.00513
T2: smiley	0.71594	0.00454	0.01079
T3: random	0.71592	0.00000	0.05932

Table 6.1: Evaluation of manipulated networks T1-T3 on the ILSVRC2012 validation set. Observe that the accuracy more or less stays the same. We measure the difference between the score y_o of the original network and new manipulated score y_n (the score is the output before softmax). The difference between the desired target explanation $\tilde{I}_T$ and the actual observed explanation $\tilde{I}_n$ is measured using the L_1 distance. The score changes very little while we can accurately manipulate the explanation as indicated by small L_1 distance.

Dataset	Network	Accuracy	$\|\|y_o - y_n\|\|_\infty$	$\|\|\tilde{I}_T - \tilde{I}_n\|\|_1$
Original	Original	0.71592	-	-
	T4: backdoor	0.71592	0.00000	0.00000
Manipulated (sticker)	Original	0.69048	-	-
	T4: backdoor	0.69048	0.00000	0.00006

Table 6.2: Evaluation of Technique 4 on the ILSVRC2012 validation set. Observe that T4 has the same accuracy and scores as the original network for both kinds of data. When presented with input data without stickers, the manipulated network T4 produces the same explanation as the original network. When presented with manipulated data, the manipulated explanation, $\tilde{I}_n$, is almost equal to the desired explanation, $\tilde{I}_T$.

6.6. DISCUSSION

GradCAM is not 'broken' — for normally trained models, GradCAM has been proven to be useful. GradCAM does not work for adverserially manipulated models such as ours, since it was not designed for that task. However, our models are valid models, with (almost) equal performance. Hence, they should also admit a valid explanation. In fact, in [6] the axiom of Implementation Invariance is defined: two networks that produce the same output for all inputs should admit the same explanation. Clearly, GradCAM does not satisfy this axiom and thus there is room for improvement. One may wonder wether the axiom should be extended to models that return extremely similar predictions, such as T3 and T4.

Our work reveals that GradCAM relies on unknown assumptions on the network parameters, architecture, etc. It is difficult to rule out that, by accident, a model can be produced, using regular training, where GradCAM explanations may fail. We think it is important to determine what assumptions should be verified for GradCAM to produce accurate explanations, so we can always verify the correctness of GradCAM explanations.

Our techniques may be extended to fool other explanation methods. Several methods rely on the gradient $\frac{\partial y}{\partial x}$ [3], [6], [10], [12], [13]. T3 and T4 show that it is possible to

manipulate the gradient, while affecting accuracy only little. So, these methods may also be vulnerable.

A weakness of our method is that architectural changes are necessary. If the practitioner visualizes the architecture (for example, using TensorBoard in TensorFlow [36]) or inspects the code, he may easily discover that the model has been tampered with. However, we believe similar attacks, where the original architecture is used, should be feasible, which would make the attack much harder to detect. We believe this is possible, since deep networks contain a lot of redundancy in the weights. Weights can be compressed or pruned, freeing up neurons, which then may be used to confuse the explanation. Recently, this area of research has been very active [37], [38]. For example, [39] were able to prune 35% of the weights, while not significantly changing the test accuracy on MNIST. Another approach is Knowledge Distillation (KD), where a larger model (the teacher) can be compressed in a smaller model (the student) [40]. Such methods could be combined with our technique to keep the model accuracy more or less the same and to confuse the explanation method, without any architectural changes. We will explore this promising idea in future work.

6.7. CONCLUSION

We provided another sanity check in the same vein as [20] and we have shown that Grad-CAM does not satisfy said sanity check. We submit that, for any explanation method, one should consider whether it is possible to change the underlying model such that the predictions change minimally, while explanations change significantly. If this is the case, our work illustrates that the explanation method may be fooled by an attacker with access to the model and the explanations may not be as robust as desired.

BIBLIOGRAPHY

[1] T. Viering, Z. Wang, M. Loog, and E. Eisemann, "How to manipulate cnns to make them lie: The gradcam case," *British Machine Vision Conference Workshop*, 2019.

[2] A. Adadi and M. Berrada, "Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)," *IEEE Access*, vol. 6, pp. 52 138–52 160, 2018.

[3] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps," in *2nd International Conference on Learning Representations (ICLR), Banff, AB, Canada, Workshop Track Proceedings*, 2013.

[4] S. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proceedings of Advances in Neural Information Processing Systems 30 (NIPS)*, Long Beach, CA, USA, 2017, pp. 4768–4777.

[5] P.-J. Kindermans, K. T. Schütt, M. Alber, *et al.*, "Learning how to explain neural networks: PatternNet and PatternAttribution," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada*, 2018.

[6] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic Attribution for Deep Networks," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, Sydney, NSW, Australia, 2017, pp. 3319–3328.

[7] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for Simplicity: The All Convolutional Net," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, Workshop Track Proceedings*, 2014.

[8] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European Conference on Computer Vision (ECCV)*, 2014, pp. 818–833.

[9] P. W. Koh and P. Liang, "Understanding black-box predictions via influence functions," in *Proceedings of the 34th International Conference on Machine Learning*, 2017, pp. 1885–1894.

[10] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning Important Features Through Propagating Activation Differences," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, 2017, pp. 3145–3153.

[11] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PloS one*, vol. 10, no. 7, e0130140, 2015.

[12] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, "SmoothGrad: removing noise by adding noise," *arXiv preprint arXiv:1706.03825*, 2017.

[13] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?": Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, California, USA, 2016, pp. 1135–1144.

[14] M. L. Amogh Gudi Nicolai van Rosmalen and J. van Gemert, "Object-Extent Pooling for Weakly Supervised Single-Shot Localization," in *Proceedings of the British Machine Vision Conference (BMVC)*, London, UK, 2017.

[15] R. C. Fong and A. Vedaldi, "Interpretable Explanations of Black Boxes by Meaningful Perturbation," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2017-Octob, pp. 3449–3457, 2017.

[16] P. Dabkowski and Y. Gal, "Real time image saliency for black box classifiers," in *Proceedings of Advances in Neural Information Processing Systems 30 (NIPS)*, 2017, pp. 6967–6976.

[17] L. M. Zintgraf, T. S. Cohen, T. Adel, and M. Welling, "Visualizing Deep Neural Network Decisions: Prediction Difference Analysis," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France*, 2017.

[18] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Object Detectors Emerge in Deep Scene CNNs," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA*, 2015.

[19] M. Ancona, E. Ceolini, C. Öztireli, and M. Gross, "A unified view of gradient-based attribution methods for Deep Neural Networks," in *NIPS Workshop on Interpreting, Explaining and Visualizing Deep Learning*, 2017.

[20] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim, "Sanity checks for saliency maps," in *Proceedings of Advances in Neural Information Processing Systems 31 (NIPS)*, Montréal, Canada, 2018, pp. 9505–9515.

[21] P.-J. Kindermans, S. Hooker, J. Adebayo, *et al.*, "The (Un)reliability of saliency methods," *arXiv preprint arXiv:1711.00867*, 2017.

[22] A. Ghorbani, A. Abid, and J. Zou, "Interpretation of Neural Networks is Fragile," *arXiv preprint arXiv:1710.10547*, 2017.

[23] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning Deep Features for Discriminative Localization," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[24] G. Inc., *Google Cloud Machine Learning Engine*, 2019. [Online]. Available: `https://cloud.google.com/ml-engine/`.

[25] A. Inc., *Amazon SageMaker*, 2019. [Online]. Available: `https://aws.amazon.com/sagemaker/`.

[26] M. Inc., *Azure Machine Learning Service*, 2019. [Online]. Available: `https://azure.microsoft.com/en-in/services/machine-learning-service/`.

[27] Q. Yao, M. Wang, Y. Chen, *et al.*, "Taking Human out of Learning Applications: A Survey on Automated Machine Learning," *arXiv preprint arXiv:1810.13306*, 2018.

[28] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshop, Columbus, OH, USA*, pp. 512–519, 2014.

[29] T. Gu, B. Dolan-Gavitt, and S. Garg, "BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain," *arXiv preprint arXiv:1708.06733*, 2017.

[30] Y. Liu, S. Ma, Y. Aafer, *et al.*, "Trojaning attack on neural networks," in *Proceedings of the 25th Annual Network and Distributed System Security Symposium (NDSS)*, San Diego, California, USA, 2018.

[31] B. Wang, Y. Yuanshun, S. Shan, *et al.*, "Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks," in *2019 IEEE Symposium on Security and Privacy (SP)*, 2019.

[32] B. Chen, W. Carvalho, N. Baracaldo, *et al.*, "Detecting backdoor attacks on deep neural networks by activation clustering," *Workshop on Artificial Intelligence Safety 2019 co-located with the Thirty-Third AAAI Conference on Artificial Intelligence 2019 (AAAI-19)*, 2019.

[33] N. Papernot, P. McDaniel, A. Sinha, and M. Wellman, "Towards the Science of Security and Privacy in Machine Learning," *arXiv preprint arXiv:1611.03814*, 2016.

[34] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *International Conference on Learning Representations*, 2015.

[35] O. Russakovsky, J. Deng, H. Su, *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[36] M. Abadi, A. Agarwal, P. Barham, *et al.*, *Tensorflow: Large-scale machine learning on heterogeneous systems*, 2015.

[37] J. Cheng, P. Wang, G. Li, Q. Hu, and H. Lu, "Recent Advances in Efficient Computation of Deep Convolutional Neural Networks," *Frontiers of Information Technology & Electronic Engineering*, vol. 19, no. 1, pp. 64–77, 2018.

[38] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "A Survey of Model Compression and Acceleration for Deep Neural Networks," *arXiv preprint arXiv:1710.09282*, 2017.

[39] S. Srinivas and R. V. Babu, "Data-free Parameter Pruning for Deep Neural Networks," in *Proceedings of the British Machine Vision Conference (BMVC)*, Xianghua Xie Mark W. Jones and G. K. L. Tam, Eds., Sep. 2015, pp. 1–31.

[40] L. J. Ba and R. Caruana, "Do deep nets really need to be deep?" In *Proceedings of the 27th International Conference on Neural Information Processing Systems-Volume 2*, 2014, pp. 2654–2662.

7

HUMAN IMPACT

How does a user's prior experience with deep learning impact accuracy? We present an initial study based on 31 participants with different levels of experience. Their task is to perform hyperparameter optimization for a given deep learning architecture. The results show a strong positive correlation between the participant's experience and the final performance. They additionally indicate that an experienced participant finds better solutions using fewer resources on average. The data suggests furthermore that participants with no prior experience follow random strategies in their pursuit of optimal hyperparameters. Our study investigates the subjective human factor in comparisons of state of the art results and scientific reproducibility in deep learning.

This work has been published as:
K. Anand, Z. Wang, M. Loog, J. van Gemert, "Black Magic in Deep Learning: How Human Skill Impacts Network Training", British Machine Vision Conference, 2020 [1]. As the second author, I designed the human study together with Kanav and Jan, including the choice of dataset, hyperparameters, recruiting participants, user interface *etc.* .

7.1. INTRODUCTION

The popularity of deep learning in various fields such as image recognition [2], [3], speech [4], [5], bioinformatics [6], [7], question answering [8] *etc.* stems from the seemingly favorable trade-off between the recognition accuracy and their optimization burden. LeCun *et al.* [9] attribute their success to feature representation learning as opposed to using hand-engineered features. While deep networks learn features, the hand engineering has shifted to the design and optimization of the networks themselves. In this paper we investigate the influence of human skill in the hand engineering of deep neural network training.

Arguably, one reason for why neural networks were less popular in the past is that compared to 'shallow' learners such as for example LDA [10], SVM [11], *k*NN [12], Naive-Bayes [13], *etc.* , deep networks have many more hyperparameters [14] such as the number of layers, number of neurons per layer, the optimizer, optimizer properties, number of epochs, batch size, type of initialization, learning rate, learning rate scheduler, *etc.* A hyperparameter has to be set before training the deep network and setting these parameters can be difficult [15], yet, the excellent results of deep networks [9] as revealed by huge datasets [16] with fast compute [2] offer a compelling reason to use deep learning approaches in practice, despite the difficulty of setting many of those parameters.

Hyperparameters are essential to good performance as many learning algorithms are critically sensitive to hyperparameter settings [17]–[19]. The same learning algorithm will have different optimal hyperparameter configurations for different tasks [20] and optimal configurations for one dataset do not necessarily translate to others [21]. The existing state of the art can be improved by reproducing the work with a better analysis of hyperparameter sensitivity [22], and several supposedly novel models in NLP [23] and in GANs [24] were found to perform similarly to existing models, once hyperparameters were sufficiently tuned. These results show that hyperparameters are essential for reproducing existing work, evaluating model sensitivity, and making comparisons between models.

Finding the best hyperparameters is something that can be done automatically by autoML [25]–[28] or Neural Architecture Search [29]–[32]. Yet, in practice, such methods are not widely used by deep learning researchers. One reason could be that automatic methods are still under active research and not yet ready for consumption. Another reason could be that good tuning adds a significant computational burden [23], [24]. Besides, automated tuning comes with its own set of hyperparameters and, in part, shifts the hyperparameter problem. Thus, in current practice, the hyperparameters are usually set by the human designer of the deep learning models. In fact, it is widely believed that hyperparameter optimization is a task reserved for experts [15], [33], as the final performance of a deep learning model is *assumed* to be highly correlated with background knowledge of the person tuning the hyperparameters. The validation of this claim is one of the main goals of our research. The extraordinary skill of a human expert to tune hyperparameters is what we here informally refer to as "black magic" in deep learning.

7.1.1. CONTRIBUTIONS

Broadly speaking, we investigate how human skill impacts network training. More specifically, we offer the following contributions. 1. We conduct a user study where partici-

pants with a variety of experience in deep learning perform hyperparameter optimization in a controlled setting.[1] 2. We investigate how deep learning experience correlates with model accuracy and tuning efficiency. 3. We investigate human hyperparameter search strategies. 4. We provide recommendations for reproducibility, sensitivity analysis, and model comparisons.

7.2. EXPERIMENTAL SETUP

Our experiment is designed to measure and analyze human skill in hyperparameter optimization. All other variations have identical settings. Each participant has the exact same task, model, time limitation, GPU, and even the same random seed. Our participants tune hyperparameters of a deep learning architecture on a given task in a user-interface mimicking a realistic setting while allowing us to record measurements.

7.2.1. DEEP LEARNING SETUP

The deep learning experimental setup includes: the task, the model and the selection of hyperparameters.

Deep learning task. The choice for the task is determined by the size, difficulty, and realism of the considered dataset. Large datasets take long to train, which limits the number of hyperparameters we can measure. Also, if the dataset is not challenging, it would be relatively easy to achieve a good final performance which limits the variance in the final performance of the model. Taking size and difficulty into account, while staying close to a somewhat realistic setting, we decided on an image classification task on a subset of ImageNet [16] which is called *Imagenette* [34]. To prevent using dataset specific knowledge we did not reveal the dataset name to participants. We only revealed the image classification task and we shared the dataset statistics: 10 classes, 13,000 training images, 500 validation images, and 500 test images

Deep learning model. The model should be well-suited for image classification, have variation in hyperparameter settings, and be somewhat realistic. In addition, it should be relatively fast to train so that a participant can run a reasonable amount of experiments in a reasonable amount of time. We selected *Squeezenet* [35] as it is efficient to train and achieves a reasonable accuracy compared to more complex networks. To prevent exploiting model-specific knowledge, we did not share the network design with the participants.

Hyperparameters. We give participants access to 15 common hyperparameters. Four parameters are mandatory: number of epochs, batch size, loss function, and optimizer. We preset the other 11 optional hyperparameters with their commonly used default values. In Table 7.1, we show the list of hyperparameters. Please refer to the supplementary material for their full description. Note that none of the hyperparameters under

[1] The research carried out has been approved by TU Delft's Human Research Ethics Committee: `https://www.tudelft.nl/en/about-tu-delft/strategy/integrity-policy/human-research-ethics/`.

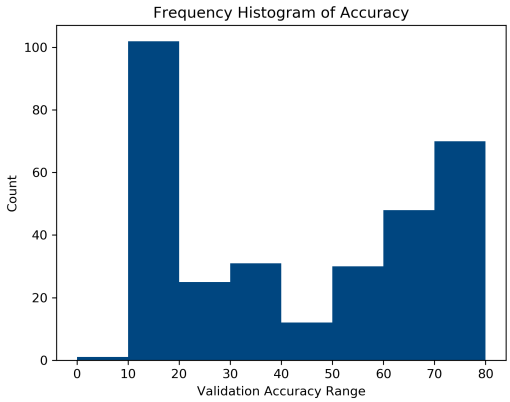


Figure 7.1: Accuracy histogram over 320 random hyperparameter settings. Their settings matter.

participants control influenced the random seed, as we keep any randomness such as weight initialization and sample shuffling exactly the same for each participant. For 320 random hyperparameter settings, the average random accuracy is 41.8 ± 24.3, where Figure 7.1 demonstrate that hyperparameters are responsible for ample accuracy variance for this task. Without such variance there may be little differences in human accuracy which would make it difficult to analyse skill.

7

7.2.2. PARTICIPANTS' EXPERIMENTAL SETUP

For managing participants we need: a user-interface, a detailed task description, and define what to measure.

User-interface. We simulate a realistic hyperparameter optimization setting, while providing a controlled environment. We designed a web interface to let participants submit their choice of hyperparameters, view their submission history with validation accuracy and view the intermediate training results with an option for early stopping. Few preliminary tries were done (by the participants not included in result dataset) to test and verify the design and hyperparameter optimization process. By using a web server we collect all the data for analysis. We make all data and source code available[2].

Participant's task. The task given to participants is to find the optimal set of hyperparameters, *i.e.* , those maximizing classification accuracy on the test set. After submitting a choice of hyperparameters, the deep learning model is trained in the background using these parameters. While the model is training, the participant can view the intermediate batch loss and epoch loss in real time. The participant has an option to cancel training if the intermediate results do not look promising. As there is an upper limit of 120 minutes to how much time a participant can use on the optimization of the model, early

[2] `https://github.com/anandkanav92/htune`

Table 7.1: The hyperparameters available to participants in our study.

Type	Hyperparameter	Default value
Mandatory	Epochs	-
	Batch size	-
	Loss function	-
	Optimizer	-
Optional	Learning rate	0.001
	Weight decay	0
	Momentum	0
	Rho	0.9
	Lambda	0.75
	Alpha	0.99
	Epsilon	0.00001
	Learning rate decay	0
	Initial accumulator value	0
	Beta1	0.9
	Beta2	0.999

stopping enables them to try more hyperparameter configurations. After training the model is finished, the accuracy on a validation set is provided to the participant. Participants are encouraged to add optional comments to each choice of hyperparameters. The experiment ends when the participant decides that the model has reached its maximum accuracy or if the time limit of the experiment is reached (120 minutes). The flow diagram of the user study is depicted in Figure 7.2.

Measurements per participant. As an indication for the degree of expertise a participant has, we record the number of months of deep learning experience. During deep model training, we record all the hyperparameter combinations tried by the participant, together with the corresponding accuracy on the validation set, for as many epochs as the participant chooses to train. The experiment ends by a participant submitting their final choice of hyperparameters. This optimal hyperparameter configuration is then trained ten times on the combined training and the validation set after which the accuracy on the independent test set is recorded. Each of the 10 repeats have a different random seed, while the seeds are the same for each participant.

7.2.3. SELECTION OF PARTICIPANTS

The participants were selected based on their educational background and their prior experience in training deep learning models. The participants with no prior experience comprised of people recruited from different specialisations using poster ads and email groups. Experienced candidates were invited through our deep learning course provided to master students and researchers.

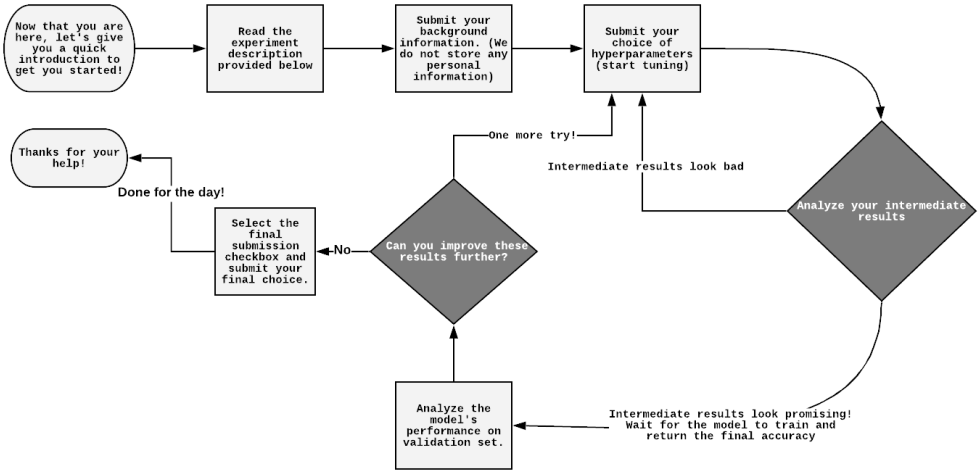


Figure 7.2: The flow diagram of the user study. The participant starts by entering their information. Next, submit the values for hyperparameters and evaluate intermediate training results. If the training is finished, the participant can decide whether to submit a new configuration for hyperparameter or end the experiment. It can be repeated until the time limit of 120 minutes is reached.

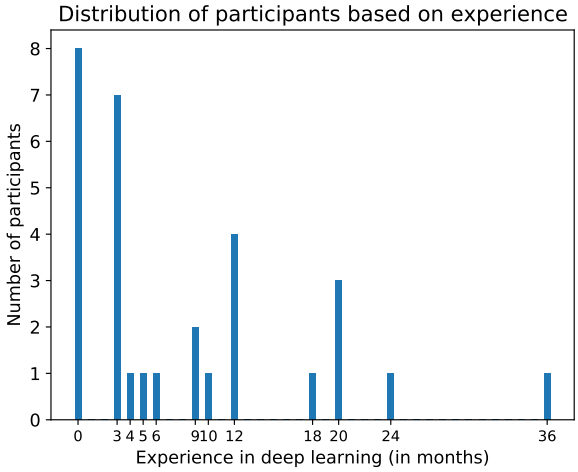


Figure 7.3: A broad range of deep learning experience in the 31 participants of our study.

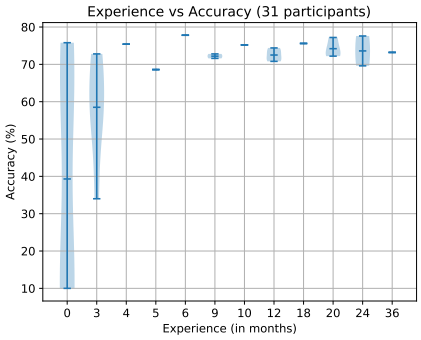


Figure 7.4: Final accuracy distribution over all participants.

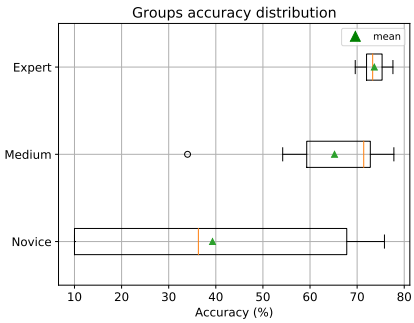


Figure 7.5: Final accuracy per group boxplot.

7.3. RESULTS

We collected 463 different hyperparameter combinations from 31 participants. The prior deep learning experience for these participants is well distributed as shown in Figure 7.3. For the final selected hyperparameters the average classification accuracy is 55.9 ± 26.3.

For ease of analysis we divide participants into groups based on experience. The *Novice* group contains 8 participants with no experience in deep learning, the 12 participants in the *medium* group have less than nine months of experience and the 11 participants in the *expert* group has more than nine months experience.

7.3.1. RELATION BETWEEN EXPERIENCE AND ACCURACY

Figure 7.4 depicts the relationship between final accuracy and deep learning experience per participant. As the experience increases, the final accuracy tends to increase, which is supported by the strong positive Spearman [36] rank order correlation coefficient of 0.60 with a p-value smaller than 0.001. Additionally, we compared the variance of the accuracy distributions of *Novice*, *medium*, *expert* groups using Levene's statistical test [37]. We use the Levene test because experience and accuracy are not normally distributed. The test values presented in Table 7.2 show all groups significantly differ from each other ($p < 0.05$), where the difference is smallest between *medium* and *expert* and the largest between *Novice* and *expert*, which is in line with the accuracy statistics per group shown in Figure 7.5.

We further analyze the effect of deep learning experience on the training process. In Figure 7.6, we show how many tries are used to reach a certain threshold accuracy for the *novice*, *medium*, *expert* groups for final accuracy thresholds. Experts reach the threshold quicker. Furthermore, we show the average accuracy of each group after a number of tries in Figure 7.7. We can conclude that more experienced participants not only achieve a better accuracy, they also arrive at that better score more efficiently.

7.3.2. DIFFERENCE IN STRATEGIES

We investigate why more experienced users achieve a higher accuracy in fewer iterations.

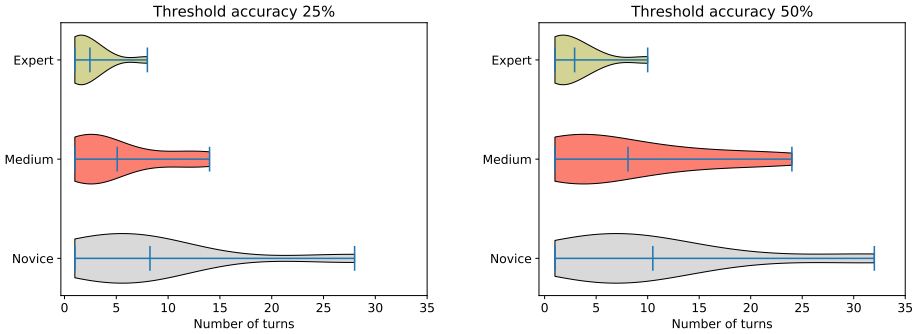


Figure 7.6: Number of hyperparameter configurations required to achieve a threshold accuracy of 25%, 50% for different experience groups. The violin plot shown above depicts the probability density distribution of the number of turns taken by the participants in each group. The mean value of each group is marked for reference. More experienced participants reach the threshold faster.

Table 7.2: All groups significantly differ from each other ($p < 0.05$); *medium* and *expert* the least and *Novice* and *expert* the most.

Levene's statistical test		
Groups	**Test Statistic**	**p-value**
Novice vs Medium	8.40	0.01
Novice vs Expert	14.338	0.001
Medium vs Expert	5.52	0.029

Table 7.3: Predefined comments used in user study.

ID	**Comment**
1	It is just a guess.
2	It is a suggested default value.
3	It is the value that has worked well for me in the past.
4	It is the value I learnt from previous submissions.
5	Other

Use of suggested default values. We offer mandatory and optional hyperparameters, as shown in Table 7.1, where the optional hyperparameters are preset to their default values. Figure 7.8 shows the number of participants in each group using these default values as the starting point. A large majority in the *medium* or *expert* groups begin with all optional hyperparameter values set to their suggested default values and subsequently build on them. In contrast, *novice* users directly explore the optional values. Using defaults for optional parameters does not necessarily lead to an optimal hyperparameter configuration, however, all participants who started with defaults achieved a final performance greater than 50%.

Analysis of comments. Participants were encouraged to leave comments explaining the reasoning behind choosing a specific value of a hyperparameter. In a bid to gather maximum comments, we let users choose from predefined comments shown in Table 7.3. Figure 7.9 shows the distribution of comments for each group of *novice*, *medium*, or *expert* participants. We noticed that there was confusion between 'past experience' and 'learned from previous submission' as 22% of hyper parameter values used by *novice* participants were based on their prior experience in deep learning. As this confusion may also effect other groups, we refrain from drawing hard conclusions based on the

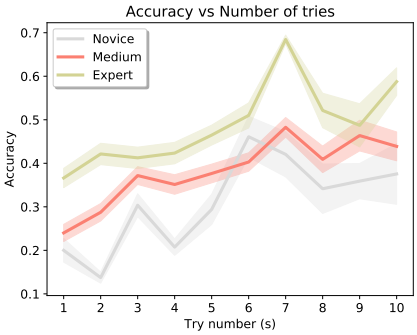


Figure 7.7: Experts need fewer tries to get better accuracy. The shaded region is standard error.

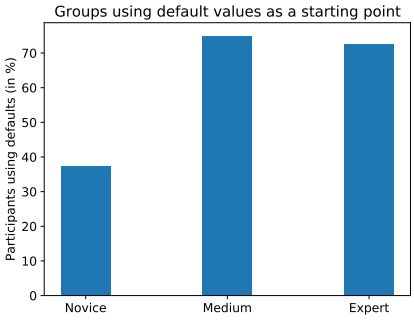


Figure 7.8: Participants submitting their initial hyperparameter configuration using all default values.

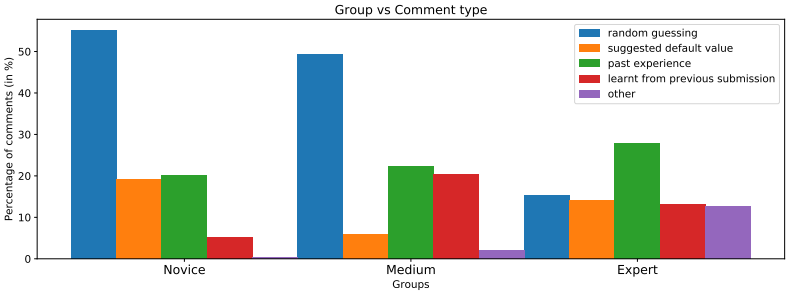


Figure 7.9: The distribution of comments for the groups of *novice*, *medium*, *expert* participants. Inexperienced users rely more on random guessing (blue).

observed increase in the use of the comment 'past experience' for more experienced participants. For *novice* participants, the majority is based on random guessing. Random guessing was found to be strongly negatively correlated with the increasing experience. We used Spearman rank-order correlation, and the value was found to be -0.58 with a p-value smaller than 0.001. As the amount of experience increases, the results show a decrease in random guessing.

7.4. DISCUSSION AND CONCLUSION

We identify main limitations to this study, draw conclusions, and make recommendations.

7.4.1. MAIN LIMITATIONS

Limited data. We have a fairly restricted number of 31 participants. Collecting more data and inviting more participants in the user study will make the result and conclusions more robust to potential outliers. In addition, it can of course provide better insight into the process of hyperparameter optimization, generalize our findings over a broader

audience, and give us the possibility to test more refined hypotheses.

Stratified experience groups. Currently, the three participant groups that we used in our analysis, *i.e.* , novice, medium, and expert, were identified based on the amount of experience, as measured months, they had. It may be of interest, of course, to consider information different from experience to stratify participants in different groups. Maybe the amount of programming experience or the amount of machine learning experience correlates better with performance achievements. What should maybe also be considered, however, is the way to measure something like experience. Rather than using a measure like 'months of experience,' one can also resort, for instance, to often used self-evaluations, in which every participant decided for themselves which level they have. In more extensive experiments, it would definitely be of interest to collect such additional meta-data.

Only one deep learning setting This study focuses only on an image recognition task with a single model and a single dataset in a limited time. Thus, it can be argued that the findings of this study could not be generalized to other deep learning settings. This work is the first study explicitly analyzing human skill in hyperparameter tuning; it is interesting to extend this study further by including multiple tasks, models and datasets.

7.4.2. CONCLUSIONS

Human skill impacts accuracy. Through this user study, we found for people with similar levels of experience tuning the exact same deep learning model, the model performs differently. Every source of variation was eliminated by fixing the task, the dataset, the deep learning model, and the execution environment (random seed, GPUs used for execution) except the choice of hyperparameters. Figure 7.5 shows the variance in the final performance of the model. This suggests that final performance of the model is dependent on the human tuning it. Even for experts the difference can be an accuracy difference of 5%.

More experience correlates with optimization skill. We show a strong positive correlation between experience and final performance of the model. Moreover, the data suggests that more experienced participants achieve better accuracy more efficiently, while inexperienced participants follow a random search strategy, where they often start by tuning optional hyperparameters which may be best left at their defaults initially.

7.4.3. RECOMMENDATIONS

Concluding our work, we would like to take the liberty to propose some recommendations regarding experiments and their outcome. We base these recommendations on our observed results that even expert accuracy can differ as much as 5% due to hyperparameter tuning. Thus, hyperparameters are essential for reproducing the accuracy of existing work, for making comparisons to baselines, and for making claims based on such comparisons.

- Reproducability: Please share the final hyperparameter settings.

- Comparisons to baselines: Please optimize and report the hyperparameter settings for the baseline with equal effort as the proposed model.
- Claims of (the by now proverbial) superior performance: It is difficult to say if the purported superior performance is due to a massive supercomputer trying all settings [23], [24], due to a skilled human as we show here, or due to qualities of the proposed model. Bold numbers correlate with black magic and we recommend to make bold numbers less important for assessing the contribution of a research paper.
- To the deep learning community: Make reviewers pay more attention to reproducability, baseline comparisons, and put less emphasis on superior performance. There is no need to burn wielders of black magic at the stake, but herald the enlightenment by openness and clarity in hyperparameter tuning.

ACKNOWLEDGEMENT

This work is part of the research programme C2D–Horizontal Data Science for Evolving Content with project name DACCOMPLI and project number 628.011.002, which is (partly) financed by the Netherlands Organisation for Scientific Research (NWO).

BIBLIOGRAPHY

[1] K. Anand, Z. Wang, M. Loog, and J. van Gemert, "Black magic in deep learning: How human skill impacts network training," *British Machine Vision Conference*, 2020.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, Curran Associates, Inc., 2012, pp. 1097–1105.

[3] C. Farabet, C. Couprie, L. Najman, and Y. Lecun, "Learning hierarchical features for scene labeling," in *IEEE Trans Pattern Anal Mach Intell*, IEEE, 2013.

[4] G. Hinton, L. Deng, D. Yu, *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, 2012.

[5] T. N. Sainath, B. Kingsbury, A. Mohamed, *et al.*, "Improvements to deep convolutional neural networks for lvcsr," *arXiv e-prints*, 2013.

[6] M. K. K. Leung, H. Y. Xiong, L. J. Lee, and B. J. Frey, "Deep learning of the tissue-regulated splicing code," *Bioinformatics (Oxford, England)*, 2014.

[7] J. Ma, R. P. Sheridan, A. Liaw, G. E. Dahl, and V. Svetnik, "Deep neural nets as a method for quantitative structure-activity relationships," *Journal of Chemical Information and Modeling*, 2015.

[8] A. Bordes, S. Chopra, and J. Weston, "Question answering with subgraph embeddings," *arXiv e-prints*, 2014.

[9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, 2015.

[10] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

[11] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[12] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE transactions on information theory*, vol. 13, no. 1, pp. 21–27, 1967.

[13] I. Rish, "An empirical study of the naive bayes classifier," in *IJCAI 2001 workshop on empirical methods in artificial intelligence*, vol. 3, 2001, pp. 41–46.

[14] Y. Zhou, S. Cahya, S. A. Combs, *et al.*, "Exploring tunable hyperparameters for deep neural networks with industrial adme data sets," *Journal of chemical information and modeling*, vol. 59, no. 3, pp. 1005–1016, 2018.

[15] L. N. Smith, "A disciplined approach to neural network hyper-parameters: Part 1– learning rate, batch size, momentum, and weight decay," *arXiv preprint arXiv:1803.09820*, 2018.

[16] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[17] F. Hutter, H. Hoos, and K. Leyton-Brown, "An efficient approach for assessing hyperparameter importance," in *Proceedings of the 31st International Conference on Machine Learning*, PMLR, 2014, pp. 754–762.

[18] A. Koutsoukas, K. J. Monaghan, X. Li, and J. Huan, "Deep-learning: Investigating deep neural networks hyper-parameters and comparison of performance to shallow methods for modeling bioactivity data," *Journal of cheminformatics*, vol. 9, no. 1, p. 42, 2017.

[19] N. Reimers and I. Gurevych, "Optimal hyperparameters for deep lstm-networks for sequence labeling tasks," *arXiv preprint arXiv:1707.06799*, 2017.

[20] R. Kohavi and G. H. John, "Automatic parameter selection by minimizing estimated error," in *Proceedings of the Twelfth International Conference on International Conference on Machine Learning*, Morgan Kaufmann Publishers Inc., 1995, pp. 304–312.

[21] J. N. van Rijn and F. Hutter, "Hyperparameter importance across datasets," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2018, pp. 2367–2376.

[22] S. Jasper, L. Hugo, and P. A. Ryan, "Practical bayesian optimization of machine learning algorithms," *Bartlett et al. [8], pp. 2960–2968*, 2012.

[23] G. Melis, C. Dyer, and P. Blunsom, "On the State of the Art of Evaluation in Neural Language Models," *arXiv e-prints*, 2017.

[24] M. Lucic, K. Kurach, M. Michalski, S. Gelly, and O. Bousquet, "Are GANs Created Equal? A Large-Scale Study," *arXiv e-prints*, 2017.

[25] J. Bergstra, D. Yamins, and D. D. Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," 2013.

[26] T. Domhan, J. T. Springenberg, and F. Hutter, "Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[27] P. Kerschke, H. H. Hoos, F. Neumann, and H. Trautmann, "Automated algorithm selection: Survey and perspectives," *Evolutionary computation*, vol. 27, no. 1, pp. 3–45, 2019.

[28] L. Kotthoff, C. Thornton, H. H. Hoos, F. Hutter, and K. Leyton-Brown, "Auto-weka 2.0: Automatic model selection and hyperparameter optimization in weka," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 826–830, 2017.

[29] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," *arXiv preprint arXiv:1808.05377*, 2018.

[30] H. Liu, K. Simonyan, and Y. Yang, "Darts: Differentiable architecture search," *arXiv preprint arXiv:1806.09055*, 2018.

[31] H. Pham, M. Y. Guan, B. Zoph, Q. V. Le, and J. Dean, "Efficient neural architecture search via parameter sharing," *arXiv preprint arXiv:1802.03268*, 2018.

[32] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," *arXiv preprint arXiv:1611.01578*, 2016.

[33] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, pp. 148–175, 2016.

[34] *Imagenette*, `https://github.com/fastai/imagenette`.

[35] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and less than 0.5MB model size," *arXiv e-prints*, 2016.

[36] C. Spearman, "Spearman rank correlation coefficient," in *The Concise Encyclopedia of Statistics*. Springer New York, 2008, pp. 502–505.

[37] I. Olkin, "Contributions to probability and statistics; essays in honor of Harold Hotelling," *Stanford, Calif., Stanford University Press, 1960.*, 1960.

8

DISCUSSION

The previous chapters demonstrated that a deep net is sensitive to multiple factors. We provided several solutions for sensitivities to distribution shift and adversarial attacks. We also showed the GradCAM explanation is sensitive to several architecture attacks. In addition, we explored the impact of human factors on the performance of deep net and showed that the deep net is sensitive to human factors. We have reexamined some assumptions and proposed new ones in this thesis. Necessary assumptions are always made to assist the understanding of certain problems. These assumptions sometimes become default and may be borrowed by similar tasks. The fact that an assumption is classic leads to less critical judgement when it is used in its original setting as well as other settings.

I will further discuss some methods proposed in previous chapters and share some thoughts on the popular assumptions and trends in our field. I will mainly focus on four aspects. The first is about learning domain invariant representation for domain generalization. I argue this is not a good assumption for domain generalization given the fact that no information of the target domain is available. The second is about the classic softmax function and the polynomial softRmax proposed in this thesis. I will discuss some potential extensions and applications of the softRmax and would like to raise the concern of default settings in deep nets. The classic softmax, as an example, is not always a good activation function to obtain posteriors. The third will be about knowledge transfer, not only about transfer learning between datasets, but also about transferring methods from one field to another. While transfer learning is powerful, we should remember there is no free lunch and realize that it is not always possible. In the end, I would like to give my opinion on the application of deep learning.

8.1. ON LEARNING DOMAIN INVARIANT REPRESENTATIONS FOR DOMAIN GENERALIZATION

As discussed in previous chapters, domain adaptation and domain generalization are closely related, which leads researchers to directly apply domain adaptation approaches

on domain generalization problems. One classic way of domain adaptation with deep learning is aligning the source domain and the target domain in a latent space to learn a domain invariant representation. That means, the samples from different source domains are mapped to the same representations $\mathbf{z}$ irrespective of the domains, $p(\mathbf{z}|D_s) = p(\mathbf{z})$. When this approach is applied for domain generalization, due to the absence of the target domain, existing works align several available source domains aiming to learn a domain invariant representation that is also invariant to the target domain. Various approaches, including adversarial training [1], meta learning [2], representation alignment[3], *etc.* , intrinsically aim to do the same, *i.e.* , learning a domain invariant representation. This makes sense for domain adaptation where the target domain is available, which can be unlabelled sometimes. By learning a domain invariant representation, the network, in effect, learns a mapping that forces the two different distributions of the source and target domain to overlap with each other in the latent space. Note that, even with the data available, such kind of alignment cannot guarantee a good matching of class conditional distributions of the two domains.

When we come to the setting of domain generalization where neither the data nor the label is available from the unseen target domain, we cannot give guarantees for good generalization because no proper assumption can be made without knowing any prior information of the target domain. Simply aligning all the source domains for domain generalization cannot make sure the target domain will also be mapped to the same latent representation. To illustrate, I will take an extreme case where a label conflict exists in different domains. For example, we want to classify sex based on a 1D feature x, say height, and we have two source domains with data collected from the Netherlands and Sweden, respectively. Assume that the decision boundary will be at $x = 180$ cm, which seems to be reasonable based on my 6 years of living experience in the Netherlands. Can we generalize to an unseen dataset collected in China for the same classification task? Without knowing any prior information of the data collected in China, the classifier will fail catastrophically because a reasonable decision boundary in China should be around $x = 165$ cm. Besides the extreme case of label conflict, even with a large covariate shift (no overlap of labels), there is no guarantee that the decision boundary can generalize well to the unseen target domain.

One hidden assumption that is not clearly stated in the current domain generalization setting is that there is no label conflict ($p(y|\mathbf{x}, D_i) \neq p(y|\mathbf{x}, D_j)$) between domains and the target domain overlaps with the source domains. We can then assume the target domain D_t is a weighted combination of the source domains D_{s_i}, where the weights can be obtained by a linear or nonlinear function:

$$D_t = \sum_i \alpha_i D_{s_i}. \tag{8.1}$$

The question "Whether learning a domain invariant representation helps under this assumption?" still remains. My answer is "no", because by aligning all the source domains in a latent space, $p(\mathbf{Z}|D_s) = p(\mathbf{Z})$, all the source domains D_{s_i} are treated equally important, which in turn discards the weights α_i for each domain. Also, without access to the target domain, there is no reliable way to obtain good weights α_i to decide which source domain is more important or more similar to the distribution of the target do-

main. The reasoning emphasizes that there is little we can do without any information of the target domain.

Is domain generalization a poorly defined scenario? I would say the setting itself is definitely of interest but requires more rigorous assumptions of the new unseen target domain. Recent research starts the debate whether it is at all possible to generalize to any unseen target domain without knowing the distribution of domains and a detailed analysis of the possibilities of domain generalization can be found in [4]. I propose that domain generalization should focus on exploring the relations between domains and simulate possible distribution of domains p_D, which is the distribution that all the domains are sampled/generated from. An example that can validate this hypothesis is different samplings of the rotated MNIST dataset. If we know all possible domains, no matter source domain or any unseen target domain, will just be a rotated variation of MNIST with a different angle, it is possible to use this information about rotation to generalize well to any unseen domain. To the contrary, if we only learn from variously rotated binary images of MNIST, we should not expect our model to obtain good performance on MNIST in a different hand writing because it is not sampled from the same domain distribution p_D. This thesis explored learning a hypothesis invariant representation which relaxes the constraint on learning domain invariant representation. The underlying domain order can be potentially preserved in this way. But a clear estimation of domain order is not part of our work and is challenging due to a limited number of available domains. The research on causality may benefit the simulation of domain causal factors [5], which might help to understand how domains are sampled so a domain order can be inferred.

8.2. RETHINK OF THE DEFAULT - THE SOFTRMAX CASE

We have compared the performance of softmax and our polynomial softRmax regarding conservativeness in the tail of the distribution and robustness to adversarial attacks. As it turns out, the default softmax activation function is not necessarily the best choice for classification tasks. Furthermore, models trained with softmax predict over confidently in the tail of the distribution and are more vulnerable to adversarial attacks. We do not see any obvious reason why the standard softmax should be chosen instead of the polynomial softRmax. They achieve similar performance for classification tasks on public datasets, with polynomial softRmax showing more conservative and robust characteristics.

The conservativeness of softRmax comes from the polynomiality of the chosen Cauchy distribution as class conditional distributions. Note that the Cauchy distribution can also be replaced by other polynomial distributions as long as the power of the leading term is consistent for all classes. The choice of different polynomial distributions can change the scale of conservativeness. By scale, I refer to the speed the posterior drops to random guess level when a sample moves to tails of a distribution. The fact that the posterior gradually drops to random guess level when samples move away from the bulk of the distribution means that softRmax tries to fit a tight distribution for each class. The power of the leading term or the kurtosis of different distributions decides how tight the fitted distribution is and how fast the posterior drops. A distribution with a larger power term, or higher kurtosis, results in a faster posterior drop.

Now it is not hard to link softRmax with Out Of Distribution (OOD) detection [6]. A model trained with softRmax does not require extra measures for OOD because softRmax fits a tight class conditional distribution and the tightness can be adjusted by the choice of distributions. A probability of how likely a sample is an outlier can be inferred from the prediction posterior directly. Samples with random guess level of predictions are either on the decision boundary or far in the tail of the distribution, which can be considered as OOD samples in many cases. Of course, this should be tested in the end to see whether it works as expected or not.

Further applications of softRmax include semantic segmentation [7], or instance segmentation [8], where a per pixel classification score is obtained to indicate the probability a certain pixel belongs to each class. It is not ideal to always obtain over confident predictions for atypical pixels with softmax. Take autonomous driving, a more conservative segmentation of pedestrian increases the safety level. Models trained with softRmax may contribute to this. While the default softmax function has been widely accepted, we show that a different form of softRmax can bring many ideal results. These findings raise, however, the question "What are other default settings in deep nets that deserve a second thought?"

8.3. ON KNOWLEDGE TRANSFER

As we have discussed, learning a domain invariant representation, tailored for domain adaptation, is being adopted and now even becoming a standard approach for domain generalization, though it is not reasonable. The close link between the two fields caused the adoption of the method, even though it is not an appropriate transfer. And here lies the danger: the acceptance of such kind of transfer of methods is relatively high in academia due to similarities and close links of fields. Therefore the validation of the methods is less rigorous, which leads to a biased trend for a field that can last for several years. I myself as a junior researcher was misled by those trends and struggled for one year to reject the mainstream voice in domain generalization. While critical thinking is part of the Ph.D. journey, it is also the responsibility of all reviewers. In my opinion, a careful inspection of a transferred method that seems plausible is crucial for the fast progress of the whole field.

Besides transferring methods, another way to link fields is performing multi-tasks with the assumption that these tasks are beneficial to each other. However, this too can lead to similar problems as we saw with the transfer of methods. Once I reviewed a paper for domain generalization where the authors claimed that domain invariant explanations is a sign for domain invariant representation. We let it aside that learning domain invariant representation is not appropriate for domain generalization for now but examine the other claim. Does domain invariant explanation mean domain invariant representation? Several figures were presented in the work showing that the explanation highlights the same component for the same class across different domains, which according to the authors, means that the learned representations are invariant across domains. The claim seems intuitive at the beginning. But if we recall the explanation attack chapter, we can easily realize that it is not a correct claim. It is possible to obtain the same explanation for all objects from all classes without changing the prediction, which means the representation cannot be invariant across classes, otherwise the classification

results must be random. Then how can domain invariant explanation indicate domain invariant representations?

Transfer learning is another typical way of transferring knowledge. One troublesome belief in transfer learning is that a large public dataset always helps with other tasks. Even though some works have pointed out that it is not always the case [9], [10], I still observe this belief here and there during my interaction with my peers. If we go back to the example with sex classification with label conflict, we can figure out that due to distribution shift, one dataset cannot always help the other, even if the first is a large dataset. So one should think in the first place: is it always possible to benefit from other datasets?

After the discussion of unsuccessful transfer between fields and datasets, we should not forget the bright side. There are still some positive cases where linking two fields makes advances. The gradient based adversarial classification attack method is also reasonable for adversarial explanation attack. Take FGSM [11] as an example, for a deep net with weights $\mathbf{w}$, the input $\mathbf{x}$ is perturbed to be:

$$\mathbf{x}' = \mathbf{x} + \epsilon \operatorname{sign}(\nabla_x J(\mathbf{w}, \mathbf{x}, y)), \tag{8.2}$$

where $\nabla_x J(\mathbf{w}, \mathbf{x}, y)$ is the gradient of the loss with respect to the input $\mathbf{x}$ labelled as y. And the loss function in this case is the cross entropy loss. A similar gradient based perturbation can be added into the input to form a new input which does not change the classification result but manipulate the explanation heatmap to be a chosen target explanation map h_t. For example, in [12], the perturbation is added to the input by minimizing the following loss function:

$$\mathscr{L} = \|h(\mathbf{x}') - h^t\|^2 + \gamma \|g(\mathbf{x}') - g(\mathbf{x})\|^2, \tag{8.3}$$

where $h(\cdot)$ is the heatmap explanation of an input and $g(\cdot)$ is the output of the network for this input. Note that the perturbation is learned by gradient descent in this work. Qualitative results have shown that the explanation of a dog image can be perturbed to the target explanation of a cat for many existing explanation methods.

Another positive example for linking two fields is our own work [13] on conservativeness and robustness with softRmax. The conservative predictions of samples in the distribution tail also lead to the robustness characteristics, as discussed in Chapter 5. So I do not imply that linking two fields, or at a larger scale, disciplinary research, should be avoided by any means. But it is vital to realize that sometimes we may encounter assumptions that make sense in one field but will not be valid in another one, while the resemblance of the two fields often let us fail to notice the hidden inconsistency.

8.4. APPLICATIONS OF DEEP LEARNING

Applied research might be considered as less attractive according to some researchers. I myself made this mistake when I started my Ph.D. Directly calling a library from Pytorch to solve a simple classification task may indeed be a bit boring. If we come to the real world setting, different application scenarios require quite different solutions. Applied research can be fun and challenging if it requires carefully tailoring a method for a specific problem.

One example of such an interesting application is the visual place recognition task in Chapter 2. The task is to retrieve the same architecture in Amsterdam from the gallery for the query image. The gallery is collected from modern Amsterdam while the queries are from a historic archive of Amsterdam. The challenge is caused by the distribution shift over time. The environment, technology of photography, road condition have all changed through time, which makes it hard to match images of old Amsterdam with the modern ones in the gallery. But there is one thing that does not change through time, the architecture itself. If the network can focus on the object only, instead of taking the background into consideration, it already becomes more promising than matching the whole image. Attention mechanisms naturally serve this purpose. Further adaptation approaches can be combined with the attention mechanism. We can see that this real world application requires some calibrated designs.

Medical video processing is extremely challenging considering the limited amount of data and protocols. Biased data is unavoidable sometimes due to the medical setting. Patients at a severe disease level sometimes need medical assistance to walk during the data collection, which may indicate the disease severity level by the existence of the medical assistance. But the question is, should we always try to eliminate the bias? Some may argue that this bias is not good if later a patient is actually healthy but walks with the medical assistance. This patient will be classified as severely ill in this case. But this bias is usually consistent during training and test. We almost never see a healthy person walking with medical assistance. Also as human being, we all assume that someone's ability is compromised if this person needs medical assistance to walk. If this bias is consistent in training and test phase, then, is it reasonable to avoid such kind of bias? After all, the existence of medical assistance is also a useful pattern for humans to make the judgement. Different assumptions need to be carefully discussed in accordance with the application scenarios.

Based on my own interaction with medical doctors, one reason that deep learning is so popular in medical science is that medical doctors hope to discover some new patterns that are learned by deep nets and further use these patterns in clinics. However, the learned representations are hard to interpret. I think it is a promising direction but challenging. Current heatmap based explanation approaches cannot explain the meaning of the learned features but just localize the area of the input that contributes the most to the change of loss, or the optimization of the loss. The interpretation of learned representations is particularly hard due to the high dimensionality. Several approaches can visualize the high dimensional space in 2D dimensions by dimensionality reduction, e.g., t-SNE [14]. But such kind of visualization approaches cannot explain the semantic meaning of the learned representations. Future work may explore rule based explanation which serves the purpose of medical understanding better [15]. If we can clear the question: why does a certain layer or block lead to certain learned representation? Then the interpretation of the learned representation can guide a better design of architecture to focus on the semantic features that facilitate the task. Early work [16] has linked deep learning to human vision and shown that in many networks, the learned kernels in the first two layers are Gaussian filters which lead to the learned representations to be edges or corners. Recent work has explored the function of each single neuron by dissecting a GAN [17]. How to interpret the kernels and representations in a higher layer still remains

a challenge.

8.5. SUMMARY

All in all, deep learning and computer vision are actively progressing where many unclear things are still waiting to be clarified, explored and improved. This is the charm of this field. I hope this work may make a tiny contribution to the development of the field.

BIBLIOGRAPHY

[1] Y. Li, X. Tian, M. Gong, *et al.*, "Deep domain generalization via conditional invariant adversarial networks," in *ECCV*, 2018, pp. 624–639.

[2] D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales, "Learning to generalize: Meta-learning for domain generalization," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[3] S. Motiian, M. Piccirilli, D. A. Adjeroh, and G. Doretto, "Unified deep supervised domain adaptation and generalization," in *ICCV*, 2017, pp. 5715–5725.

[4] T. Galstyan, H. Harutyunyan, H. Khachatrian, G. V. Steeg, and A. Galstyan, "Failure modes of domain generalization algorithms," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 19 077–19 086.

[5] F. Lv, J. Liang, S. Li, *et al.*, "Causality inspired representation learning for domain generalization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 8046–8056.

[6] K. Lee, K. Lee, H. Lee, and J. Shin, "A simple unified framework for detecting out-of-distribution samples and adversarial attacks," *Advances in neural information processing systems*, vol. 31, 2018.

[7] M. Thoma, "A survey of semantic segmentation," *arXiv preprint arXiv:1602.06541*, 2016.

[8] A. M. Hafiz and G. M. Bhat, "A survey on instance segmentation: State of the art," *International journal of multimedia information retrieval*, vol. 9, no. 3, pp. 171–189, 2020.

[9] B. Zoph, G. Ghiasi, T.-Y. Lin, *et al.*, "Rethinking pre-training and self-training," *Advances in neural information processing systems*, vol. 33, pp. 3833–3845, 2020.

[10] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Transfer learning for time series classification," in *2018 IEEE international conference on big data (Big Data)*, IEEE, 2018, pp. 1367–1376.

[11] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *stat*, vol. 1050, p. 20, 2015.

[12] A.-K. Dombrowski, M. Alber, C. Anders, M. Ackermann, K.-R. Müller, and P. Kessel, "Explanations can be manipulated and geometry is to blame," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[13] Z. Wang and M. Loog, "Enhancing classifier conservativeness and robustness by polynomiality," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 13 327–13 336.

[14] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne.," *Journal of machine learning research*, vol. 9, no. 11, 2008.

[15] E. Dervakos, O. Menis-Mastromichalakis, A. Chortaras, and G. Stamou, "Computing rule-based explanations of machine learning classifiers using knowledge graphs," *arXiv preprint arXiv:2202.03971*, 2022.

[16] B. M. H. Romeny, *Front-end vision and multi-scale image analysis: multi-scale computer vision theory and applications, written in mathematica*. Springer Science & Business Media, 2008, vol. 27.

[17] D. Bau, J.-Y. Zhu, H. Strobelt, *et al.*, "Gan dissection: Visualizing and understanding generative adversarial networks," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

ACKNOWLEDGEMENTS

The Ph.D. journey feels shorter than it actually lasts. Of course that does not mean I want to delay my Ph.D. to make it longer. It is a great experience to do my Ph.D. in the PRB lab, which is a warm group where people care for each other. I would like to thank all of you sincerely.

Thanks to my parents who funded my Master at TU Delft so I got the chance to continue my Ph.D. smoothly. And thank you for being understanding and supportive, especially during the covid pandemic. It is hard to not be able to visit families in the last three years. I wish to pay you a visit as soon as possible.

My rich supervisor, **Jan**, You are a very generous supervisor for paying all my summer school and international conferences. I remember when I asked you whether I could go to a conference for a workshop paper. You said, "go as you can and we will find a way if there will not be enough budget later". Then the covid pandemic started in the following year. I was really lucky to participate in some of the conferences in the physical world while I had not had first-tier papers. Besides the superficial part about money, you are really a patient supervisor. I realize that sometimes it is very hard for me to get your idea but you take the time and try to formulate it in the way I could understand better. You also give me a lot of freedom for collaborations. You never force me to do a project but only trick me with a box of chocolate. I will miss your daddy jokes.

My hard core supervisor, **Marco**, I was surprised that you would also be my supervisor when Jan informed me, considering that I was fully intimidated by you during my Master. Now after four years, I want to say that you are soft inside but wear a cold black t-shirt outside. I hope you will not realize that I was one of the annoying master students who sent email complaining my assignment was scored too low. It is really a nice experience to do research with you. Instead of being a supervisor, you are more like a collaborator who actively participates in the project so I could learn better the way you do research. I always believe the best way of teaching is the teacher being a role model him/herself. You take care of Ph.D. students like the hen putting the chicks under her wings. I was not pushed to publish but asked to do good work. It is a value that is hard to keep in this field. I hope I will still have chance to grab a beer with you.

Marcel, my promoter, I am sorry for dropping the whole cup of coffee on your door after my Ph.D. interview. Surprisingly I still got hired. I was amazed by how fast you could get the idea of my research at all my progress meetings. You have also helped me with making decisions. I overcomplicate things sometimes but you can give very simple but reasonable answers to guide me.

Pepper, you just came to give me a kiss when I am about to write this. You are the cutest cat in the world to me. Everything started to get better after you joined my life. If possible, can you please beat me less often? I promise to get you more snacks!

Tom, my party buddy, It was good time going to all the techno parties with you. And thank you for taking all the funny videos. Your research attitude is very unique, science

should be fun, which is probably inspired a lot by Feynman. I respect your passion for teaching. Your future students are very lucky to have you as a teacher. **Arman**, my coffee buddy, I developed a lot of fear for covid and still did not feel comfortable to come to the office for quite a while after the office reopened. But you were always in the office, so I could chill with you at the coffee machine each time I came. This has greatly reduced my stress and anxiety. You are very emotionally supportive and sincere. I appreciate your honesty and decency.

Chirag, Yeshwanth, Jose, my office mates, It was great to start my Ph.D. together with you guys in the same office! We were all fresh and we became raisins together. We used to hangout a lot before covid. It was a great Christmas trip with you guys in Austria and Budapest. I wish we could still cook, eat and play video games together in the future. **Amogh**, it was fun to gather all the swags at conferences with you. You are a very chill person to be around. I really appreciate the wise comments you offered. It is also very kind of you connecting me with the Dutch industry. **Nicolaas**, it is great to have someone who also works on domain adaptation in this lab. It was fun to explore Busan with you and Amogh, especially considering that non of us can speak Korean. **Marian**, you are probably the one that I met the most in the office after covid. Good luck with your pupil detection!

Yuko and Taylan, you joined the lab quite late, when I am about to leave. But naturally I feel close to both of you. Friendship does not depend on the time but is more about the feeling. It was a great pleasure to have the retreat trip with you guys. Yuko, I will come back and bother you and others in your office as much as I can. **Xin**, thank you for the conversation we had in London. It was a hard time for me and luckily it is over now. **Xiangwei**, you cook really well. Be strong and be happy! **Jin**, you do not talk much but I like to talk to you, such a paradox. I wish you could join more social activities and relax. **Osman**, you are by far my favorite Greek and **Stavros**, you are by far my favorite Turkish.

Laura, Ekin, you two are like the big sister and big brother in the lab. You created a nice atmosphere which was inherited by our generation. Now we pass it to the next generation. **Soufiane**, you were very helpful when I was checking the job market in France. All the information were in French and you listed those in some quite long emails, in English. Thank you for all the help! **Stephanie**, thank you for all the advice you gave me for internship and job hunting! **Yunqiang**, it was a good trip in London. All the best for your daughter! **Yancong**, will you gain more muscle in the gym? **Nergis**, su borek is tasty but I still need to find the right bakery. **Attila**, it is great to have you as head TA for the deep learning course! **Ombretta**, thank you for introducing us the great Neapolitan pizza in Utrecht! **Robert-Jan**, it is fun to always have you as a host. **Burak**, thanks for your advice at the CV lab meeting.

Mo, Pepper still needs to meet your cat. **Ramin** (BIO), it was great to play board games with you! **Ramin** (PR), next time I will win you for pool. **Jim, Richkard**, thank you for organizing the PR retreat. I had a lot of fun! **Mahdi**, it is nice to try saffron tea made in the classic Iranian way. **Meng**, we should do hotpot again, but let's try to not catch covid again. **David, Jesse, Hayley, Silvia, Seyran**, thank you all for providing critical and valuable comments to junior researchers in the lab. I would like to thank Hayley in particular, who accepted my interview for a graduate school course while she had a sore

throat.

Ruud, thank you for being so helpful all the time. You saved me from a potential battery explosion of my laptop. **Bart**, thank you for maintaining my personal page! **Euan**, thank you for your native English grammar support for this thesis, so I can stand my ground when I do not agree with my supervisors' comments on my English writing.

All in all, I did enjoy the four years together with everyone and I believe we will still meet each other from time to time, at least on Zoom!

CURRICULUM VITÆ

Ziqi WANG

20-10-1994 Born in Baoding, Hebei, China.

EDUCATION

2012–2016 B.Sc in Thermal Energy and Dynamic Engineering
Chang'an University
Xi'an, China

2016–2018 M.Sc. in Intelligent Vehicle
Delft University of Technology
Delft, Netherlands

2018–2022 Ph.D. in Computer Science
Delft University of Technology
Deft, Netherlands
Thesis: SENSITIVITIES AND WHERE TO FIND THEM DOMAIN SHIFT ROBUSTNESS, ATTACKS, AND TRAINING VARIATIONS IN VISUAL LEARNING
Promotor: Prof.dr.ir. Marcel J.T. Reinders, Prof.dr. Marco Loog, Dr. Jan C. van Gemert

WORKING EXPERIENCE

2022– Machine Learning Engineer
Meta
London, U.K.

LIST OF PUBLICATIONS

- **Z. Wang**, J. Li, S. Khademi, J. van Gemert, "Attention-Aware Age-Agnostic Visual Place Recognition", *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019. (Chapter 2)
- **Z. Wang**, M. Loog, J. van Gemert, "Respecting Domain Relations: Hypothesis Invariance for Domain Generalization", *International Conference on Pattern Recognition*, 2020. (Chapter 3)
- Z. Yin, V.J. Geraedts, **Z. Wang**, M.F. Contarino, H. Dibeklioglu, J. van Gemert, "Assessment of Parkinson's Disease Severity from Videos using Deep Architectures", *IEEE Journal of Biomedical and Health Informatics*, 2021. (Chapter 4)
- **Z. Wang**, M. Loog, "Enhancing Classifier Conservativeness and Robustness by Polynomiality", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. (Chapter 5)
- T.J. Viering, **Z. Wang**, M. Loog, E. Eisemann, "How to Manipulate CNNs to Make Them Lie: the GradCAM Case", *British Machine Vision Conference Workshops*, 2019. (Chapter 6)
- K. Anand, **Z. Wang**, M. Loog, J. van Gemert, "Black Magic in Deep Learning: How Human Skill Impacts Network Training", *British Machine Vision Conference*, 2020. (Chapter 7)
- A. M. Ruiter, **Z. Wang**, Z. Yin, W. C. Naber, J. Simons, J. T. Blom, J. C. van Gemert, J. J. Verschuuren, M. R. Tannemaat, "Assessing facial weakness in myasthenia gravis with facial recognition software and deep learning", *Annals of Clinical and Translational Neurology*, 2023.