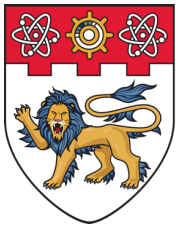
Enhancement and Evaluation of Generalization in Deep Learning for Classification Tasks



NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

RENCHUNZI XIE

College of Computing and Data Science

A thesis submitted to the Nanyang Technological University in partial fulfillment of the requirements for the degree of Doctor of Philosophy

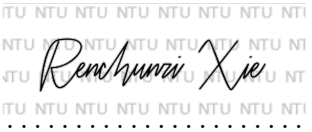
2025

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

01/08/2024

Date



RENCHUNZI XIE

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

01/08/2024

Date

Prof. Bo An

Authorship Attribution Statement

This thesis contains material from 4 papers published / submitted in the following peer-reviewed conferences in which I am listed as the first author.

Chapter 3 is published as Renchunzi Xie, Hongxin Wei, Lei Feng, Bo An. GearNet: Stepwise Dual Learning for Weakly Supervised Domain Adaptation. Proceedings of the 36th AAAI conference on artificial intelligence (AAAI). pp. 8717-8725. 2022. The contributions of the co-authors are as follows:

- Dr. Hongxin Wei and I proposed the problem setting.
- I proposed the improved solutions, conducted the experiments, and wrote the main manuscript.
- Dr. Hongxin Wei, Dr. Lei Feng discussed with me about how to properly organize the manuscript and revised the manuscript drafts.
- Prof. Bo An provided insightful comments and carefully revised the manuscript.

Chapter 4 is published as Renchunzi Xie, Hongxin Wei, Yuzhou Cao, Lei Feng, Bo An. On the Importance of Feature Separability in Predicting Out-Of-Distribution Error. Proceedings of the 37th Conference on Neural Information Processing Systems (NeurIPS), 2023. The contributions of the co-authors are as follows:

- Dr. Hongxin Wei and I proposed the problem setting.
- I proposed the improved solutions, conducted the experiments, and wrote the main manuscript.
- Dr. Hongxin Wei discussed with me about the theoretical analysis.
- Prof. Bo An, Dr. Lei Feng and Yuzhou Cao provided insightful comments and carefully revised the manuscript.

Chapter 5 is submitted as Renchunzi Xie, Ambroise Odonnat, Vasilii Feofanov, Ievgen Redko, Jianfeng Zhang, Bo An. Leveraging Gradients for Unsupervised Accuracy Estimation under Distribution Shift. Submitted to Transactions on Machine Learning Research (TMLR), 2025. The contributions of the co-authors are as follows:

- I proposed the idea and the solution.
- I conducted the experiments and wrote the main manuscript. Ambroise Odonnat, Dr. Vasilii Feofanov, and Dr. Ievgen Redko discussed with me about the theoretical analysis.
- Ambroise Odonnat, Dr. Vasilii Feofanov, Dr. Ievgen Redko, Dr. Jianfeng Zhang, and Prof. Bo An provided insightful comments and carefully revised the manuscript.

Chapter 6 is submitted as Renchunzi Xie, Ambroise Odonnat, Vasilii Feofanov, Weijian Deng, Jianfeng Zhang, Bo An. MANO: Exploiting Matrix Norm for

Unsupervised Accuracy Estimation Under Distribution Shifts. Proceedings of the 38th Conference on Neural Information Processing Systems (NeurIPS), 2024. The contributions of the co-authors are as follows:

- I proposed the idea and the solution.
- I conducted the experiments and wrote the main manuscript. Ambroise Odonnat and Dr. Vasilii Feofanov discussed with me about the theoretical analysis.
- Ambroise Odonnat, Dr. Vasilii Feofanov, Dr. Weijian Deng and Prof. Bo An provided insightful comments and carefully revised the manuscript..

01/08/2024

Date

RENCHUNZI XIE

Acknowledgements

I would like to express my tremendous gratitude to people who helped and inspired me during the past three years.

First and foremost, I would like to express my deepest appreciation to my supervisor, Prof. Bo An. Without the great opportunity he provided, I could not have undertaken this Ph.D. journey, which has been a turning point in my life. During these years, he always maintains a gentle, patient, and kind style. He has not only provided a singularly suitable research environment with the greatest freedom, but also taught me the necessary knowledge in my research and even ordinary life, such as how to address a problem in order and how to properly communicate with others. Words cannot express my gratitude to him. The experience in his group would definitely shape my future in many dimensions. I am so fortunate to become a Ph.D. student under his supervision.

I also had the pleasure of working with a group of talented people: Haipeng Chen, Mengchen Zhao, Youzhi Zhang, Xinrun Wang, Lei Feng, Rundong Wang, Wei Qiu, Jakub Černý, Yanchen Deng, Shuxin Li, Wanqi Xue, Xinyu Cai, Shuo Sun, Pengdeng Li, Shenggong Ji, Shufeng Kong, Hao Cheng, Hongxin Wei, Pengjie Gu, Yewen Li, Chaojie Wang, Zhenghai Xue, Longtao Zheng, Chuqiao Zong, Sheng Zang, Molei Qin, Ying Zheng, Dong Xing, Wentao Zhang, Ruhao Jiang, Yuzhou Cao, Haochong Xia, Zhenxing Ge, Yitian Hong. It has been an amazing experience working in the research group, thanks to everyone here I've worked with and from whom I've learned so much.

I am also thankful to my Thesis Advisory Committee (TAC) members: Prof. Cheng Long and Prof. Bihan Wen. They provided me a lot of help on my research and made my Ph.D. journey easier in NTU.

I would like to thank these nice people I have met during my internship at Huawei Noah's Ark Lab: I would like to thank Lujia Pan, Jianfeng Zhang, Keli Zhang,

Zhongwen Rao, Sixiao Yang, Junjian Ye, JiaLe Zheng, Hongbo Guo, Ievgen Redko, Vasilii Feofanov, Ambroise Odonnat and other people I met or cooperated with. This project would not have been possible without the support of these amazing friends.

I would also like to extend my thanks to many of my friends: Bing Chen, Weijian Deng, Zhuoyi Lin, and so on. Their great help and accompany make these years to be a colorful experience for me.

Finally, I must express my very profound gratitude to my families, including my parents, my husband and my lovely cat, for providing me with unfailing support and continuous encouragement throughout my years of study and research. Without your unceasing encouragement, support, patience and attention, this accomplishment would not have been possible. Thank you.

Abstract

Modern machine techniques have showcased impressive prowess in many areas. Although they often outperform humans in controlled experiments, training a machine learning models with satisfying performance requires large-scale in-distribution datasets with annotation, which is normally expensive and challenging to collect. This motivates the need to explore the value of out-of-distribution samples collected from free platforms, and furthermore understand the properties and mechanism of model generalization. In pursuit of this goal, the primary emphasis of this thesis lies in examining the inherent challenges posed by distribution shifts. Subsequently, we delve into the characteristics of generalization to deepen our comprehension of its underlying mechanisms.

Firstly, traditional machine learning methods are founded on the assumption of independence and identical distribution (i.i.d.) between training and testing datasets. However, this assumption is easy to be violated in the real world, especially when the training data are collected from open and free platforms due to financial budget or technique limitation. Therefore, exploring the useful information of real-world datasets for target tasks is of great practical meaning. This drives the necessity to maintain model performance amidst naturally occurring data corruptions and alterations throughout the machine learning process. Thus, we propose an practical problem setting that the training data suffer from distribution shifts and noisy labels simultaneously. We design a stepwise-dual-learning framework that the training and the test data should learn from each other, which is able to enhance existing algorithms for both domain adaptation and learning with noisy labels.

Secondly, evaluating the generalization performance of trained models across diverse distribution shifts is a fundamental and necessary process in practice. However, the traditional evaluation approach assumes the ground-truth labels of test data are accessible, which is hard to achieve in practice. Thus, it is of great importance to estimate the generalization performance of pre-trained models in an unsupervised manner. In the first way, we explore the correlation between feature separability

and test accuracy, where we find the hidden features from the feature extractor of trained models are more separable when the test accuracy is higher. Nevertheless, this character cannot maintain consistent performance across different types of distribution shifts. Therefore, in the second way, we derive a gradient-based estimation approach, which aggregates the gradient information of the classification layer to estimate the test accuracy. However, the previous methods struggle with the overconfidence issue which dramatically degrades the estimation performance. To alleviate this issue and satisfy the requirement about confidential model information, in the third way, we propose a logit-based approach with softmax truncation to estimate the test accuracy, where we find the density around decision density linearly reflect the generalization performance of trained models.

We assess the efficacy and resilience of all the methodologies introduced across various simulated and real-world benchmarks. The findings reveal that our techniques outperform numerous state-of-the-art approaches in mitigating the identified challenges. It is our aspiration that our endeavors offer valuable insights to inspire the development of tailored solutions for these resilient issues and accelerate the utilization of out-of-distribution examples in crafting robust and efficient systems.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Machine learning as a brunch of artificial intelligence aims to learn meaningful patterns from data and generalize the learned knowledge to unseen data [9]. Due to its automatic learning process and strong learning capability, it has been developed as a widely used tool to handle many complex real-world tasks by extracting information from large-scale datasets. One of the important techniques in the field of machine learning is deep learning, which utilizes artificial neural networks with well-designed architecture to learn a mapping function from the input to the output space. In terms of the properties of the output space, there are two typical tasks in deep learning, including *classification* handling the discrete output space and *regression* for the continuous space. In this thesis, we focus on deep learning for classification problems.

Although deep learning techniques for classification have achieved great success in many practical areas such as computer vision [10] and natural language processing [11], they have been certified by many researchers that their prediction capability is vulnerable to fundamental assumptions, where the training and the test data should be independent and identically distributed [12, 13]. Such issues significantly degrade the prediction capability of deep learning, and many reasons can cause the violation of the fundamental assumptions, including *adversarial samples* [12], *data perturbation* [8], *distribution shifts* [4], *label noise* [14], *class imbalance* [15] and *open-set recognition* [16]. Therefore, it is important to identify and enhance

models' generalization capability, so that model performance can be preserved in complex real scenarios across the machine learning pipeline.

In this thesis, we mainly focus on two issues: *generalization enhancement* and *generalization estimation*. In the generalization enhancement setting, the training and the test data are drawn from different but relevant distributions. Therefore, the goal of this setting is to avoid over-fitting to training data in order to guarantee good performance in predicting the values of future inputs during testing [17]. This setting is meaningful especially when the training data are collected from open-source platforms including adequate resources but inevitable mistakes. In the generalization estimation setting, since the corresponding labels of test data are infeasible to access in practice, ground-truth classification accuracy is impossible to obtain during testing. In this setting, our goal is to design a score that correlates with ground-truth classification accuracy in an unsupervised manner. It assists us to further understand the underlying working mechanism of generalization by exploring possible factors that may affect or be affected by the violation of fundamental assumptions.

1.2 Motivations

As mentioned above, deep learning techniques are vulnerable to distribution shifts that are frequently encountered in the real world. Therefore, exploring how to enhance models' generalization performance is of increasingly importance for deploying safe machine learning models in the real world [18]. Thus, my first topic is to study the generalization enhancement problem. In details, previous works exploit unidirectional relationships from the source domain (i.e., the training set) to the target domain (i.e, the test set) [19, 20], but this one-way strategy ignores information from the target domain, leading to suboptimal performance against label noise and distribution shifts simultaneously. Thus, exploring the bilateral relationships between the two domains is one of the goals of this thesis.

For the generalization estimation problem, the traditional method to obtain model performance across distribution shifts is directly calculated from ground-truth labels of test data by the model pre-trained on training data [21, 22]. However, it is sometimes infeasible to access the ground-truth labels of the test data in the open

world. Therefore, an increasing number of research works are designed to estimate model performance in an unsupervised manner [5, 7, 23–31]. In this thesis, we introduce three approaches based on feature separability, gradients, and logits respectively. In addition, those works also inspire the further exploration about the arcanum of model generalization.

1.3 Major Contributions

The main contributions of this thesis are summarized as follows:

- We introduce GearNet, a universal framework designed to transfer valuable insights from a source domain with noisy labels to an unlabeled target domain by leveraging bilateral connections between the two domains. Departing from the traditional unidirectional learning approach from source to target, GearNet treats the two domains as distinct inputs for training two models iteratively. A symmetrical Kullback-Leibler loss is employed to selectively align the predictions of the two models within each domain. Extensive experimental findings demonstrate that integrating GearNet can notably enhance the effectiveness of existing methods.

- We introduce the Dispersion Score, a novel metric at the dataset level that relies on feature dispersion to gauge test accuracy in scenarios involving distribution shift. Drawing inspiration from the favorable characteristics of features in representation learning, our theoretical and empirical investigations reveal a robust link between inter-class dispersion and model accuracy. In contrast, intra-class compactness does not consistently indicate generalization performance on out-of-distribution (OOD) data. Through comprehensive experiments, our approach showcases superiority in both predictive accuracy and computational efficiency.

- Our pioneering work involves estimating true test accuracy through gradient-based information. Specifically, we leverage the magnitude of gradients from the classification layer, backpropagated from the cross-entropy loss after a single gradient step on test data, to generate a metric for unsupervised test accuracy estimation. Through extensive experiments utilizing a range of

architectures across diverse distribution shifts, our approach demonstrates substantial improvements over current state-of-the-art methods.

- We introduce MaNo, a logit-based metric, to gauge true test accuracy amid diverse distribution shifts. Initially, we investigate the correlation between logits and generalization performance through the lens of the low-density separation assumption. This exploration serves as the foundation for our method, which involves computing the L_p norm of the logit matrix. To address issues of overconfidence encountered by logit-based techniques, we propose a novel normalization strategy that strikes a balance between information completeness and error accumulation based on distinct calibration scenarios. Through thorough empirical evaluations against common unsupervised accuracy estimation benchmarks, MaNo emerges as a cutting-edge performer across various architectures in scenarios involving synthetic, natural, and subpopulation shifts.

1.4 Thesis Organization

The remaining part of this thesis is organized as two main parts: generalization enhancement and generalization performance estimation. Chapter 2 introduces background knowledge and related work of this thesis. In Chapter 3, we propose a generalization enhancement approach against distribution shifts and noisy labels simultaneously. From Chapter 4 to Chapter 6, we focus on generalization performance estimation. Chapter 4 designs a simple but effective score based on feature separability for estimating true test accuracy in an unsupervised manner. Chapter 5 explores the linear relationship between gradient-based information and true test accuracy. Chapter 6 estimates the test accuracy via decision boundary density evaluation, balancing between information completeness and error accumulation. Chapter 7 concludes this thesis and highlights possible directions of future work.

Chapter 2

Preliminaries

2.1 Background

2.1.1 Supervised learning

Supervised learning as a typical classification paradigm collects data consisting of feature vectors and labels [9]. Its goal is to learn a classification function mapping from the feature space to the label space based on the large-scale training sets. The deployment safety of traditional supervised learning requires some significant fundamental assumptions [9, 32]. One of them is that the training data and the test data are from the identical distribution [9]. However, it is sometimes infeasible to collect such high-quality datasets in the open world during the training process [33–36]. The distribution of the training data is easily perturbed by many factors such as label noise [33, 34], distribution shifts [35, 36], class imbalanced [15, 37] as well as open-set data [38, 39]. In this thesis, we focus on training models with strong generalization capabilities that are able to overcome complex scenarios of collecting data.

To begin with, we introduce the multi-class classification scenario, where the label space comprises more than two labels. Here, $\mathcal{X} \subset \mathbb{R}^d$ represents the feature space, and $\mathcal{Y} = 1, \ldots, K$ with $K > 2$ signifies the label space. Given a training dataset consisting of N instances $D = \{\mathbf{x}_i, y_i\}_{i=1}^N$, where $\mathbf{x}_i \in \mathcal{X}$ denotes the i-th instance independently and identically drawn from the training-data distribution $\mathcal{P}_{\mathcal{X}\mathcal{Y}}$, the

objective of this problem is to train a mapping $f : \mathcal{X} \to \mathbb{R}^K$ with learnable parameters $\boldsymbol{\theta} \in \mathbb{R}^p$ by minimizing the following classification risk:

$$\mathcal{R}(f) = \mathbb{E}_{\mathcal{P}_{\mathcal{X}\mathcal{Y}}} \left[\mathcal{L}(f(\mathbf{x}), y) \right], \tag{2.1}$$

where $\mathbb{E}_{\mathcal{P}_{\mathcal{X}\mathcal{Y}}} [\,\cdot\,]$ refers to the expectation over the joint distribution $\mathcal{P}_{\mathcal{X}\mathcal{Y}}$, while $\mathcal{L}$ denotes the loss function that aggregates the error between the predicted labels and the ground-truth labels.

In practice, the joint distribution $\mathcal{P}_{\mathcal{X}\mathcal{Y}}$ is usually inaccessible. Therefore, a common way to address this problem is to utilize the empirical risk to estimate the true risk [40] based on the given training data $\{\mathbf{x}_i, y_i\}_{i=1}^N$:

$$\tilde{\mathcal{R}}(f) = \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(f(\mathbf{x}_i), y_i). \tag{2.2}$$

As for the choice of the loss function, the most common one for multi-class classification is the cross-entropy loss. In particular, the cross-entropy loss is derived from information theory, which gauges relative entropy between two distributions. It is formulated as follows:

$$\mathcal{L}_{\mathrm{CE}}(f(\mathbf{x}_i), y_i) = -\log p(y|\mathbf{x}) = -\log \frac{e^{f_y(\mathbf{x})}}{\sum_{i=1}^{k} e^{f_i(\mathbf{x})}}, \tag{2.3}$$

where $f_y(\mathbf{x})$ refers to y-th element of $f(\mathbf{x})$ whose corresponding ground-truth label is y, and $p(y|\mathbf{x})$ is the corresponding softmax probability.

As mentioned above, traditional multi-class classification problems should fulfill the fundamental assumptions about the distribution consistency between the training and the test datasets. Otherwise, the model trained on the corrupted or shifted training data could be invalid on the test data. Therefore, in the following section, we will introduce some related fields that could change the distribution of the training or test data.

2.1.2 Distribution shift

Supervised learning methods traditionally assume that the distribution of training data, denoted as $\mathcal{P}(\mathcal{X}\mathcal{Y})^S$, is the same as the distribution of test data, denoted

as $\mathcal{P}(\mathcal{XY})^T$. This assumption, $\mathcal{P}(\mathcal{XY})^S = \mathcal{P}(\mathcal{XY})^T$, allows these methods to generalize well to unseen examples. However, in real-world scenarios, it is common for distributions to change, leading to significant performance degradation in traditional supervised learning methods.

There are diverse types of distribution shifts that attack the distribution of data generation from different respects, including the *covariate shift*, the *concept shift*, and the *prior probability shift*. *Covariate shifts*, also named as data shift, assumes that the conditional probabilities of the training data and the test data remain unchanged (i.e., $\mathcal{P}(\mathcal{Y}|\mathcal{X})^S = \mathcal{P}(\mathcal{Y}|\mathcal{X})^T$), while their corresponding marginal probabilities are different (i.e., $\mathcal{P}(\mathcal{X})^S \neq \mathcal{P}(\mathcal{X})^T$) [41]. The situation of *Concept shifts* is opposite to covariate shift, which denotes that the marginal probabilities are unchanged but the conditional probabilities are different (i.e., $\mathcal{P}(\mathcal{Y}|\mathcal{X})^S \neq \mathcal{P}(\mathcal{Y}|\mathcal{X})^T, \mathcal{P}(\mathcal{X})^S = \mathcal{P}(\mathcal{X})^T$). As for the *prior probability shifts*, also known as label shift, refers to the situation that the marginal probabilities of labels are different between the training and the test data, but their corresponding conditional probabilities are identical (i.e., $\mathcal{P}(\mathcal{X}|\mathcal{Y})^S \neq \mathcal{P}(\mathcal{X}|\mathcal{Y})^T, \mathcal{P}(\mathcal{Y})^S = \mathcal{P}(\mathcal{Y})^T$).

Besides based on the variation of distribution, another identification approach is based on the data generation resource [27]. The common types of the distribution shift include the *synthetic shift*, the *subpopulation shift*, and the *natural shift*. *Synthetic shifts* are caused by different visual corruptions on features, such as snow, noise, and motion blur. *Subpopulation shifts* are induced when novel subpopulations are only obvious during the test process. *Natural shifts* refer to the distribution shift faced in the wild due to data reproduction.

2.1.3 Weakly supervised learning

In conventional supervised learning, the assumption is that training examples $\mathcal{D} = \{\mathbf{x}i, y_i\}_{i=1}^N$ consist of pairs where each instance is accurately labeled and sampled independently from the true distribution $\mathcal{P}_{\mathcal{XY}}$. However, acquiring the true labels for every individual can often present challenges. As a result, weakly supervised learning approaches aim to train classifiers using datasets where instances are not necessarily paired with their true labels [42].

Within the field of weakly supervised learning, various frameworks have been extensively explored. These include *semi-supervised learning* [43], which leverages both small-scale labeled and large-scale unlabeled data, *noisy-label learning* [44], which deals with datasets containing label noise, *positive-unlabeled learning* [45], which focuses on the binary classification problem where only positive samples and unlabeled data are available, *partial-label learning* [46, 47], which handles situations where instances are annotated by multiple possible labels with a correct label, and *complementary-label learning* [48], which addresses cases where the label that does not belong to are assigned to each instance.

In such scenarios, the settings introduce a noticeable shift in the distribution of the training data, even though the true distribution of the training data and the test data may sometimes remain the same. For example, noisy-label learning is a typical class of weakly supervised learning, where the pairs in the training set contain incorrect annotations. Formally, given the training data with noisy labels $\mathcal{X}, \hat{\mathcal{Y}}$, the goal of this task is to learn a classification function mapping from the feature space to the true label space $f : \mathcal{X} \to \mathcal{Y}$, where $\hat{\mathcal{Y}}$ denotes the noisy label space.

2.1.4 Open-world machine learning

Open-world machine learning aims to learn a model under more complex real-world scenarios, where the learned model is expected to work under a dynamic environment [49–51]. Therefore, the model deployed in the open world should handle the open-set problem, where never-unseen samples are encountered during either training or testing. Due to those novel samples, the fundamental assumption that the training and test data are samples from an identical distribution is violated.

Open-world machine learning diverges from conventional semi-supervised learning methodologies [43], where a restricted set of labeled data and a substantial volume of unlabeled data are supplied during the training phase. Semi-supervised learning maintains the closed-world assumption, given that the unlabeled data generally pertains to established classes and is extracted from the identical distribution as the labeled data.

Over the past few years, the notion of *open-world semi-supervised learning* [52, 53] has surfaced to tackle scenarios where both the unlabeled and testing data may encompass instances from classes not encountered before. In this fresh paradigm, the model is structured to categorize each unlabeled data point into either a recognized class or a newly introduced class. This adaptation equips the model to adeptly manage the open-world context, where a trained model must process examples from classes not previously encountered during training.

Within this section, we have delved into the notation of supervised learning and scrutinized three subdomains that defy the presumption regarding data distribution: distribution shift, weakly supervised learning, and open-world machine learning. The subsequent section will delve into more intricate subjects concerning model generalization.

2.2 Generalization in Machine Learning

Deep learning techniques have demonstrated remarkable capabilities in various domains, including computer vision [10] and natural language processing [11]. Along with its practical success, the properties of deep learning have become an active investigation, one of which is how to guarantee a good performance in predicting the labels of test samples while avoiding over-fitting to training data [17]. In this thesis, we focus on *generalization of deep learning*, which aims to keep the consistency of classification performance across the machine learning pipeline from the training data to the test data under a variable environment. In this section, we will introduce two questions about generalization capability enhancement and generalization performance estimation.

2.2.1 Generalization capability enhancement

Traditional supervised learning assumes that the training (i.e., source domain) and the test data (i.e., target domain) are drawn i.i.d from an identical distribution. However, this fundamental assumption is challenging to achieve, especially when the data are collected from an open-world scenario. As mentioned above, many factors will affect the distribution of both the training and the test data, including

distribution shifts, noisy labels, and open-set data. In this thesis, we focus on the effect of the two former factors respectively or unitedly.

In a supervised multi-class classification scenario, we define the input space as $\mathcal{X}$ and the label space as $\mathcal{Y} = 1, \ldots, K$ with K classes. The training dataset $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^{N}$ comprises N data points drawn independently and identically (i.i.d.) from a joint data distribution $\mathcal{P}_{\mathcal{X}\mathcal{Y}}$.

Given this training data set, our aim is to train a classifier $f : \mathcal{X} \to \mathbb{R}^K$ with learnable parameters $\theta \in \mathbb{R}^p$, mapping inputs to the output space. The optimal classifier minimizes the expected risk, defined as:

$$\begin{aligned} \mathcal{R}_{\mathcal{L}}(f) &= \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{P}_{\mathcal{X}\mathcal{Y}}} \left[\mathcal{L}\left(f(\mathbf{x};\theta), y\right)\right] \\ &= \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{P}_{\mathcal{X}\mathcal{Y}}} \left[\mathcal{L}\left(f_\omega(f_g(\mathbf{x})), y\right)\right] \end{aligned} \tag{2.4}$$

where $\mathcal{L}$ represents the commonly used cross-entropy loss with the softmax activation function. In this context, $f_y(\mathbf{x};\theta)$ denotes the y-th element of $f(\mathbf{x};\theta)$ corresponding to the ground-truth label y, and $p(y|\mathbf{x})$ represents the associated softmax probability.

However, in practice, the training and the test distributions are usually identical due to distribution shifts, which significantly degrades classification accuracy. Therefore, it is meaningful to keep the generalization performance of the model across different distribution shifts.

Define $P_{\mathrm{s}}(X, Y)$ as the source (training) distribution and $P_{\mathrm{t}}(X, Y)$ as the target (test) distribution. In the unsupervised domain adaptation (UDA) task, we observe n_s source samples with their corresponding ground-truth labels $S = \{(x_i^s, y_i^s)\}_{i=1}^{n_s}$, and n_t target samples without annotation $T = \{(x_i^t)\}_{i=1}^{n_t}$. The goal of this task is to train a mapping function $f_\theta : X \to Y$ based on source data S and target samples T.

UDA has garnered significant attention in various practical applications recently [21, 22, 54–58]. Its objective is to train a model using labeled data from the source domain and transfer this knowledge to a new unlabeled domain characterized by distribution shift [36]. The key to UDA's success lies in acquiring a latent domain-invariant representation by reducing the difference between the two domains (i.e., domain discrepancy) using specific criteria such as maximum mean discrepancy [59], Kullback-Leibler divergence [60], central moment discrepancy [61], and Wasserstein distance [62]. Additionally, certain studies have employed a domain discriminator

in an adversarial manner to minimize domain discrepancy, exemplified by methods like domain-adversarial neural network [63] and Adversarial discriminative domain adaptation [64]. More recently, self-training methods [65, 66] have been proposed for UDA, leveraging the idea that the domain adaptation process utilizes target label information estimated by the source-domain-trained model to enhance itself.

Nonetheless, these approaches necessitate the assumption that all labels in the source domain are accurate, a condition challenging to meet in real-world scenarios. Hence, it is imperative to devise tailored learning approaches for UDA that account for label noise in the source domain, commonly referred to as weakly supervised domain adaptation.

In the weakly supervised domain adaptation (WSDA) task, we possess a source domain containing noisy labels that are distorted versions of the true labels, denoted as $\hat{S} = (x_i^s, \hat{y}i^s)i = 1^{n_s}$, alongside an unlabeled target domain $T = (x_i^t)_{i=1}^{n_t}$. Here, n_s and n_t represent the quantities of instances in the source and target domains, respectively, with $\hat{y}_i^s$ indicating the noisy (corrupted) label. Our objective is to build a classifier $f_\theta : X \to Y$ using $\hat{S}$ and T in order to precisely label samples originating from the target domain.

WSDA considers both the UDA problem and the label noise issue, which is more common in practical scenarios. There have been several studies [19, 20, 64] to address the WSDA problem by training domain adaptation models with sample reweighting. For example, TCL [19] selects clean and transferable source samples to train a neural network that has the same structure as DANN [64]; Butterfly [20] picks clean samples from both the source domain and the target domain while sharing the shallow layers of two Co-teaching models [34] for domain adaptation. DCIC [67] emphasizes clean and transferable source data to construct a denoising maximum mean discrepancy [59] loss. Studying how to enhance generalization performance under the setting of WSDA helps us further understand the essence of the generalization capability of deep neural networks.

2.2.2 Generalization performance estimation

Machine learning methods applied in open-world scenarios often encounter challenges related to distribution shifts, where test data deviate from the training

distribution. These discrepancies can significantly impact test accuracy, leading to varying generalization performance across different shifted datasets [4, 68]. This underscores the importance of estimating out-of-distribution (OOD) errors for AI safety concerns [28]. However, acquiring extensive labeled examples for each shifted testing distribution encountered in real-world settings is either impractical or cost-prohibitive. Consequently, predicting OOD errors becomes a daunting task without ground-truth labels.

Unsupervised OOD error estimation, which is also known as unsupervised accuracy estimation, has gained increasingly attention in practice. On the one hand, it helps to find difficult (underperformed) test sets without the requirement of ground-truth labels. In cases such as retraining on underperformed datasets or annotating hard datasets, we only need to know the rank of datasets by accuracy. Therefore, we can calculate the proposed score for each dataset directly and perform the task based on this score ranking. On the other hand, when deploying the model into production, it is important to estimate its safety. If the cost of getting test labels is prohibitive, this task can help to estimate the model's accuracy on the product's test data. A practitioner can additionally look at the variability of the score on multiple test sets. When multiple datasets are not available, we can alternatively construct adequate synthetic datasets via various visual transformations [69].

In the OOD error estimation task, we consider a K-class classification task with the input space $\mathcal{X} \subset \mathbb{R}^D$ and the label set $\mathcal{Y} = \{1, \ldots, K\}$. Our learning model comprises a neural network with trainable parameters $\boldsymbol{\theta} \in \mathbb{R}^p$, which functions as a mapping from the input space to the label space: $f_{\boldsymbol{\theta}} : \mathcal{X} \to \mathbb{R}^K$. We conceptualize the network as a fusion of a sophisticated feature extractor $f_{\mathbf{g}}$ and a linear classification layer $f_{\boldsymbol{\omega}}$, where $\mathbf{g}$ and $\boldsymbol{\omega} = (\mathbf{w}_k)_{k=1}^K$ represent their respective parameters. When provided with a training instance $\mathbf{x}_i$, the feedforward operation can be represented as:

$$f_{\boldsymbol{\theta}}(\mathbf{x}_i) = f_{\boldsymbol{\omega}}(f_{\mathbf{g}}(\mathbf{x}_i)). \tag{2.5}$$

Let $\mathbf{y} = (y^{(k)})k = 1^K$ represent the one-hot encoded vector of label y, where $y^{(k)} = 1$ if and only if $y = k$, and $y^{(k)} = 0$ otherwise. Subsequently, with a training dataset $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^n$ comprising n data points sampled *i.i.d.* from the source distribution

$P_S(\mathbf{x}, y)$ defined over $\mathcal{X} \times \mathcal{Y}$, the training of $f_{\boldsymbol{\theta}}$ proceeds through empirical cross-entropy loss minimization:

$$\mathcal{L}_{\mathcal{D}}(f_{\boldsymbol{\theta}}) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K} y_i^{(k)} \log \mathrm{s}_{\boldsymbol{\omega}}^{(k)}(f_{\mathbf{g}}(\mathbf{x}_i)), \tag{2.6}$$

where $\mathrm{s}_{\boldsymbol{\omega}}^{(k)}$ represents the softmax output for class k, which approximates the posterior probability $P(Y = k|\mathbf{x})$.

Estimating accuracy in unsupervised settings is crucial due to distribution shifts and the absence of ground-truth labels for test samples. To delve into this realm, we introduce two key existing settings related to this domain. Firstly, several studies focus on estimating test accuracy or assessing accuracy differences between training and test sets solely using training data [70–75]. For instance, the model-architecture-based algorithm [70] extracts persistent topology properties from training data to discern when a model starts generalizing to unseen datasets. Nonetheless, these algorithms operate under the assumption that both training and test data are drawn from the same distribution, rendering them susceptible to distribution shifts. Our works fall under the second setting, aiming to estimate the classification accuracy of a specific test dataset during evaluation using unlabeled test samples and/or labeled training datasets. The primary research thrust is to investigate the inverse relationship between distribution disparities and model performance across feature space [28], parameters [7], and labels [76]. Another prevalent approach involves devising estimation metrics based on softmax outputs of test samples [25, 25–27], heavily reliant on model calibration. Some studies draw insights from unsupervised learning, like consensus among multiple classifiers [26, 77–79] and image rotation [23]. Moreover, recent investigations have delved into the characteristics of test datasets during evaluation [29].

To verify the effectiveness of the proposed methods for generalization performance estimation, two common metrics are frequently used to illustrate the linear relationship between the proposed methods and ground-truth test accuracy: the coefficient of determination (R^2) and Spearman's rank correlation coefficient (ρ). In details, given a dataset containing n values $\mathcal{Y} = \{y_1, y_2, ..., y_n\}$, and their corresponding predicted values $\hat{\mathcal{Y}} = \{\hat{y}_1, \hat{y}_2, ..., \hat{y}_n\}$, R^2 is calculated as follows

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)}{\sum_{i=1}^{n}(y_i - \bar{y})^2}, \tag{2.7}$$

where $\bar{y}$ denotes the average value of the dataset $\mathcal{Y}$. ρ measures rank correlation between the $\mathcal{Y}$ (i.e., $R[\mathcal{Y}]$) and its predictive values $\hat{\mathcal{Y}}$ (i.e., $R[\hat{\mathcal{Y}}]$), which can be expressed as follows.

$$\rho = \frac{cov[R(\mathcal{Y}), R(\bar{\mathcal{Y}})]}{\sigma_{R[\mathcal{Y}]}, \sigma_{R[\bar{\mathcal{Y}}]}}, \tag{2.8}$$

where σ demotes the standard deviation, while $cov[R(\mathcal{Y}), R(\bar{\mathcal{Y}})]$ denotes the covariance of the rank variables.

Studying how to estimate OOD error in an unsupervised manner helps us further understand the mystery of generalization in deep learning.

Chapter 3

GearNet: Stepwise Dual Learning for Weakly Supervised Domain Adaptation

3.1 Motivation

In the problem of domain adaptation, we aim to train classifiers for data from the target domain by leveraging auxiliary data sampled from related but different source domains [36, 80–83]. Most of the existing domain adaptation studies assume that the source domains are clean datasets with accurate annotations. However, it is usually expensive and time-consuming to collect such large-scale and correctly labeled datasets in some real-world scenarios [84, 85]. To alleviate this problem, an increasing number of researchers started to investigate *weakly supervised domain adaptation* (WSDA), where only source domain data with noisy labels and unlabeled target domain data are available.

To improve the model robustness against label noise from the source domain, some WSDA algorithms [19, 20, 67] were developed to specially reduce the negative impact of label noise while minimizing the distribution discrepancy of two domains. For example, TCL [19] selects clean and transferable samples from the source domain guided by a transferable curriculum and Butterfly [20] picks small-loss samples from both the source domain and target domain. Similarly, DCIC [67] emphasizes clean and transferable source samples by an estimated transition matrix. Although

these methods achieve acceptable performance, they only exploit the supervision information from the source domain to prevent the model from overfitting to label noise, and then transfer the learned information from the source domain to the target domain. In other words, these methods only exploit *unidirectional* relationships from the source domain to the target domain. However, if we further consider exploiting the pseudo supervision information from the unlabeled target domain, the relationships between the two domains are exploited in a *bilateral* way. Consequently, richer supervision information could be discovered for combating the label noise and the gap between the two domains would be narrowed. To the best of our knowledge, we are the first to explore the benefit of utilizing pseudo supervision knowledge from the target domain in improving the robustness against noisy labels from the source domain.

This chapter proposes the first universal paradigm to exploit bilateral relationships between the source domain and the target domain. Specifically, we train two models on the two domains respectively in an alternate manner, and the whole training process consists of four main steps: 1) training model A on source domain data with noisy labels, 2) using model A to generate pseudo labels for target domain data, 3) training model B on pseudo-labeled target domain data with regularization on the consistency of source-domain class posteriors of model B and model A, 4) training model A on labeled source domain data with regularization on the consistency of target-domain class posteriors of model A and model B. We iterate from step 2 to step 4 multiple times until the training process stops. In this way, the pseudo supervision information in the target domain can be discovered and leveraged to improve the model robustness against label noise from source domain. It is worth noting that our proposed paradigm is a general WSDA framework to enhance the model robustness. Therefore, it can be easily incorporated into existing WSDA algorithms for further improving the performance of those methods.

To verify the effectiveness of our proposed GearNet, we conduct extensive experiments on widely used benchmark datasets, and experimental results demonstrate that our GearNet can significantly improve the performance of existing robust methods.

3.2 Related Work

Unsupervised domain adaptation. *Unsupervised domain adaptation* (UDA) has gained considerable interests in many practical applications recently [21, 22, 54–58, 86, 87], which aims to learn a model on data from the labeled source domain and transfer the learned information to a new unlabeled domain with distribution shift [36]. The key to the success of UDA is to learn a latent domain-invariant representation by minimizing the difference between the two domains (i.e., domain discrepancy) with certain criteria, such as maximum mean discrepancy [59], Kullback-Leibler divergence [60], central moment discrepancy [61], and Wasserstein distance [62]. Besides, some studies utilized the domain discriminator in an adversarial manner to minimize the domain discrepancy, like domain-adversarial neural network [63] and Adversarial discriminative domain adaptation [64]. More recently, self-training based methods [65, 66] have been proposed for UDA, which are based on the motivation that the domain adaptation process uses the target label information estimated by the source-domain-training model to enhance itself. However, those methods require the assumption that all labels in the source domain are correct, which is difficult to satisfy in the real world. Therefore, it is of great significance for us to develop specially designed learning methods for UDA with label noise in the source domain (i.e., weakly supervised domain adaptation).

Weakly supervised domain adaptation. WSDA considers both the UDA problem and the label noise issue, which is more common in practical scenarios. There have been several studies [19, 20, 64] to address the WSDA problem by training domain adaptation models with sample reweighting. For example, TCL [19] selects clean and transferable source samples to train a neural network that has the same structure as DANN [64]; Butterfly [20] picks clean samples from both the source domain and the target domain while sharing the shallow layers of two Co-teaching models [34] for domain adaptation. DCIC [67] emphasizes clean and transferable source data to construct a denoising maximum mean discrepancy [59] loss. Despite the effectiveness of these methods, they only exploit the supervision information from the source domain and regrettably ignore the potential supervision information in the target domain.

Learning with noisy labels. A wide range of algorithms have been proposed to improve the model robustness against label noise in the training data [88–91].

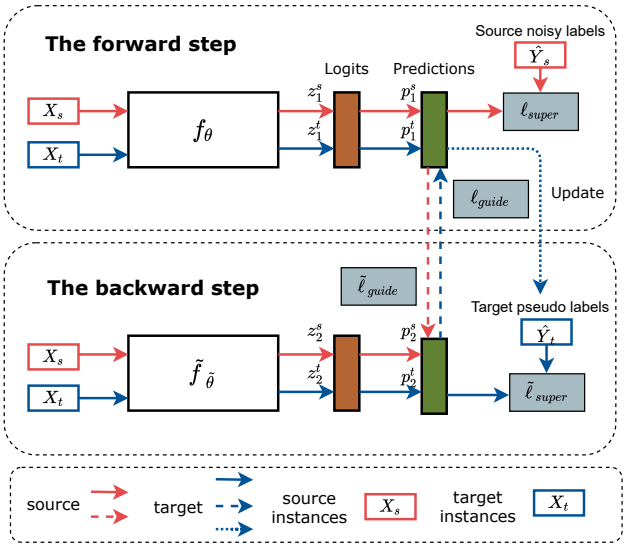


FIGURE 3.1: GearNet schematic. The forward step and the backward step are conducted iteratively. Each model is trained with a supervised learning loss on one domain, and a symmetric Kullback Leibler divergence loss to mimic the predictions of its dual model on the other domain, where z_*^* is the logit coming from the last layer of the corresponding model, and $\boldsymbol{p}_*^*$ is the probability of classes calculated by the softmax function on z_*^*. Every time when the *forward step* stops, the pseudo labels of the target domain should be updated.

Early studies [92, 93] learn a robust model by estimating the label transition matrix to fit the noisy labels. However, They can only achieve mediocre performance, as it is non-trivial to obtain a high-quality estimation of noise rates. Recently, training with sample reweighting has became a popular research direction to handle label noise [94, 95], where reliable and noiseless data are emphasized during the training process. Another promising direction is to design noise-robust loss functions, such as mean absolute error [85], generalized cross entropy loss [96], and Taylor cross entropy loss [97]. The above methods for learning with noisy labels have provided many inspirations for WSDA methods to combat label noise.

3.3 The Proposed Approach

Problem statement. Throughout this chapter, we consider the classification task under the setting of WSDA. We assume that we have a source domain with noisy labels corrupted from ground-truth labels $\hat{S} = \{(x_i^s, \hat{y}_i^s)\}_{i=1}^{n_s}$ and an unlabeled target domain $T = \{(x_i^t)\}_{i=1}^{n_t}$, where n_s and n_t denote the number of instances from

the source domain and the target domain respectively, and $\hat{y}_i^s$ denotes the noisy (corrupted) label. Our goal is to train a classifier $f_\theta : X \to Y$ based on $\hat{S}$ and T to accurately annotate samples from the target domain.

As label noise will degenerate both the domain adaptation process [67] and the classification process [88], existing methods [19, 20, 67] for WSDA focus on emphasizing useful samples by utilizing the supervision information from the source domain to reduce the negative impact of label noise. Considering the source domain could provide useful information to the target domain as mentioned above, we claim the target domain could also contain valuable information that is beneficial to the learning on the source domain. Therefore, inspired by the mutual learning [98] and dual learning [99], we address the issue of WSDA by exploring the bilateral relationship that the two domains offer useful information to each other to handle domain shifts and label noise.

The intuition for exploring bilateral relationships in WSDA is briefly explained as follows. Similar to learning with label noise, existing WSDA methods would encounter the error accumulation issue: the error that comes from the biased selection of training instances in the previous iterations would be directly learnt again in the following training [34]. In WSDA, the accumulated error from learning with source domain examples would be amplified, causing a significant increase in the target domain error [20, 100]. Co-teaching [34] and Butterfly [20] alleviate this issue by training two networks with different initialization to exchange the biased selections with each other. In this work, our method introduce additional model diversity derived from the distinct supervision information by training two networks with opposite transfer directions. In this manner, the accumulated error would be further attenuated during the training stage.

Algorithm design. Inspired by the above motivation, we propose a universal paradigm called "GearNet" that can be employed to various backbone methods (i.e., existing WSDA methods). Before the introduction, we need to clarify at first that we omit the technical details of those backbone methods to simplify the introduction of GearNet, and assume that we build GearNet on the top of a basic backbone model that is composed of a feed-forward neural network. With this basic model, we can introduce GearNet in a more convenient way. After the introduction, we will further introduce how to employ GearNet to different backbone methods.

Algorithm 1: GearNet's Learning Strategy

input : Source dataset with noisy labels $\hat{S}$, target dataset with pseudo labels $\hat{T}$, max steps M, max epochs N, learning rate η, the pretrained basic model f_θ and its dual model $\tilde{f}_{\tilde{\theta}}$

output : f_θ and $\tilde{f}_{\tilde{\theta}}$

for $t = 0$ to M **do**
  Shuffle: $\hat{S}$ and $\hat{T}$;
  Initialize: $\tilde{f}_{\tilde{\theta}}$; `// Start the backward step`
  for $i = 0$ to N **do**
    Fetch: $\{x_i^t, \hat{y}_i^t\}_{i=1}^{m_t}$ from $\hat{T}$, $\{x_i^s\}_{i=1}^{m_s}$ from $\hat{S}$;
    Calculate: $\tilde{\ell}_{\text{super}} = \frac{1}{m_t}\sum_{i=1}^{m_t} \ell(\hat{y}_i^t, \tilde{f}(x_i^t, \tilde{\theta}))$;
    Forward: $\boldsymbol{p}_1^s = f(x_i^s, \theta)$, $\boldsymbol{p}_2^s = \tilde{f}(x_i^s, \tilde{\theta})$;
    Calculate: $\tilde{\ell}_{guide}$ by (3.7) using $\boldsymbol{p}_1^s$ and $\boldsymbol{p}_2^s$;
    Obtain: $\tilde{\ell}_{total}$ by (3.3) ;
    Update: $\tilde{\theta} = \tilde{\theta} - \eta\Delta\tilde{\ell}_{total}$;
  end
  Initialize: f_θ ; `// Start the forward step`
  for $i = 0$ to N **do**
    Fetch: $\{x_i^s, \hat{y}_i^s\}_{i=1}^{m_s}$ from $\hat{S}$, $\{x_i^t\}_{i=1}^{m_t}$ from $\hat{T}$;
    Calculate: $\ell_{\text{super}} = \frac{1}{m_s}\sum_{i=1}^{m_s} \ell(\hat{y}_i^s, f(x_i^s, \theta))$;
    Forward: $\boldsymbol{p}_1^t = f(x_i^t, \theta)$, $\boldsymbol{p}_2^t = \tilde{f}(x_i^t, \tilde{\theta})$;
    Calculate: ℓ_{guide} by (3.6) using $\boldsymbol{p}_1^t$ and $\boldsymbol{p}_2^t$;
    Obtain: ℓ_{total} by (3.3);
    Update: $\theta = \theta - \eta\Delta\ell_{total}$;
  end
  Update: $\{\hat{y}_i^t\}_{i=1}^{n_t}$ by f_θ;
end

Our GearNet compromises two basic models: f_θ and $\tilde{f}_{\tilde{\theta}}$. Our model learning strategy includes three parts: the *pretrained process* aiming to annotate the target domain data by f_θ , the *forward step* aiming to transfer knowledge from the source domain to the target domain by f_θ, and the *backward step* aiming to transfer knowledge from the target domain to the source domain by $\tilde{f}_{\tilde{\theta}}$. The forward step and the backward step are iteratively conducted until the whole training process ends.

In the *pretrained process*, we train f_θ on the source domain data with noisy labels $\hat{D}_s$ (i.e., Eq. (3.1)), and then use the model to generate the hard pseudo labels for

the target domain instances (i.e., Eq. (3.2)).

$$\theta = \underset{\theta}{\operatorname{argmin}} \frac{1}{m_s} \sum\nolimits_{i=1}^{m_s} \ell(\hat{y}_i^s, f(x_i^s, \theta)), \tag{3.1}$$

$$\hat{y}_i^t = \underset{c}{\operatorname{argmax}} f_c(x_i^t, \theta), \forall i = 1, 2, ..., n_t, \tag{3.2}$$

where c denotes the c'th label, and f_c is the network output for the c'th label.

Then the forward step and the backward step can be conducted in an opposite manner. In detail, both the *forward* and the *backward step* train their models with two losses: a conventional supervised learning loss (i.e., ℓ_{super} or $\tilde{\ell}_{super}$) on one domain and a mimicry loss that aligns predictions of the two models (i.e., ℓ_{guide} or $\tilde{\ell}_{guide}$) on the other domain. So their overall losses could be expressed as follows:

$$\ell_{\text{total}} = \ell_{\text{super}} + \beta \ell_{\text{guide}}; \quad \tilde{\ell}_{\text{total}} = \tilde{\ell}_{\text{super}} + \beta \tilde{\ell}_{\text{guide}}, \tag{3.3}$$

where β is the trade-off hyperparameter, and its value is set as 0.1 in general. Besides, ℓ_{total} is for training f_θ during the forward step, while $\tilde{\ell}_{\text{total}}$ is for training $\tilde{f}_{\tilde{\theta}}$ during the backward step.

The supervised learning loss of the forward step is based on the noisy source domain:

$$\ell_{\text{super}} = \frac{1}{m_s} \sum\nolimits_{i=1}^{m_s} \ell(\hat{y}_i^s, f(x_i^s, \theta)), \tag{3.4}$$

while that of the backward step is based on the pseudo-labeled target domain:

$$\tilde{\ell}_{\text{super}} = \frac{1}{m_t} \sum\nolimits_{i=1}^{m_t} \ell(\hat{y}_i^t, \tilde{f}(x_i^t, \tilde{\theta})). \tag{3.5}$$

Although the model can perform well on the domain with supervision information due to the optimization of the supervised learning loss, there would be a significant accuracy drop on the test data from the other domain because of domain shifts. To generalize the model for better performance on the other domain, we introduce a consistency regularization, the symmetric Kullback-Leibler (KL) divergence loss (i.e., ℓ_{guide} or $\tilde{\ell}_{guide}$), which can enforce the model to mimic the predictions of its dual model for every sample from the other domain. So for the *forward step*, the loss is based on the target domain:

$$\ell_{\text{guide}} = D_{\text{KL}}\left(\boldsymbol{p}_1^t \| \boldsymbol{p}_2^t\right) + D_{\text{KL}}\left(\boldsymbol{p}_2^t \| \boldsymbol{p}_1^t\right), \tag{3.6}$$

For the *backward step*, the loss is calculated by the data from the source domain:

$$\tilde{\ell}_{\text{guide}} = D_{\text{KL}}\left(\boldsymbol{p}_1^s \| \boldsymbol{p}_2^s\right) + D_{\text{KL}}\left(\boldsymbol{p}_2^s \| \boldsymbol{p}_1^s\right). \tag{3.7}$$

D_{KL} denotes the Kullback-Leibler (KL) divergence that measures the probability difference [101]:

$$D_{\text{KL}}\left(p \| q\right) = \sum_{i=1}^{n} p(x_i) log \frac{p(x_i)}{q(x_i)}, \tag{3.8}$$

where p and q denote the probability to be measured for the probability difference, and n is the number of samples.

In the *forward step*, the consistency regularization is obtained by inputting $\boldsymbol{p}_1^t$ and $\boldsymbol{p}_2^t$ into Eq. (3.8), where $\boldsymbol{p}_1^t$ and $\boldsymbol{p}_2^t$ denotes class label distributions which are outputs from f_θ and $\tilde{f}_{\tilde{\theta}}$, respectively. In the *backward step*, the consistency regularization is calculated by inputting $\boldsymbol{p}_1^s$ and $\boldsymbol{p}_2^s$, where $\boldsymbol{p}_1^s$ and $\boldsymbol{p}_2^s$ denote the class label distributions which are outputs from f_θ and $\tilde{f}_{\tilde{\theta}}$, respectively. Every factor in those probability metrics is computed by the softmax function based on the corresponding logits. For example, the probability of class c for the sample x_i^s from f_θ (i.e., $p_1^c(x_i^s)$) is calculated as:

$$p_1^c(x_i^s) = \frac{exp(z_1^c)}{\sum_{c'=1}^{C} exp(z_1^{c'})}, \tag{3.9}$$

where z_1^c denotes the logit of class c from f_θ for x_i^s.

Optimisation Of GearNet. After the *pretrained process* to provide pseudo labels to the target domain, the whole algorithm is run as Algorithm 1 and Figure 3.1. We first conduct the *backward step* by computing the total loss $\tilde{\ell}_{total}$ as Eq. (3.3) for training $\tilde{f}_{\tilde{\theta}}$, where the second loss $\tilde{\ell}_{guide}$ is between $\tilde{f}_{\tilde{\theta}}$ and the pre-trained f_θ on the source domain. Then we can continue to update the parameters of f_θ in the *forward step* by the total loss ℓ_{total} also as Eq. (3.3), where the mimicry loss ℓ_{guide} is between the initialized f_θ and $\tilde{f}_{\tilde{\theta}}$ trained during the *backward step* on the target domain, after which we update the pseudo labels of the target domain by the trained f_θ. We repeat the *forward* and the *backward steps* until this algorithm stops. It is worthy to note that the training model should be initialized before its training process to avoid overfitting to the noisy samples.

Tasks	$A \to W$	$A \to D$	$W \to A$	$W \to D$	$D \to A$	$D \to W$	Average
Standard	46.61 ± 0.32	51.46 ± 0.53	44.21 ± 0.12	73.13 ± 0.26	43.43 ± 0.32	65.63 ± 0.41	54.06 ± 0.33
Co-teaching	49.87 ± 1.42	55.00 ± 0.89	42.18 ± 0.71	75.63 ± 0.85	44.85 ± 1.01	64.06 ± 2.01	55.98 ± 1.15
JoCoR	50.53 ± 1.67	55.42 ± 2.33	47.19 ± 1.71	74.79 ± 1.28	44.50 ± 1.01	61.72 ± 0.98	55.69 ± 1.50
DAN	54.39 ± 2.11	54.79 ± 1.32	36.65 ± 2.62	67.08 ± 1.79	35.09 ± 1.58	60.94 ± 2.06	51.32 ± 1.91
DANN	50.91 ± 1.88	54.17 ± 0.87	44.57 ± 0.74	74.79 ± 1.08	45.35 ± 1.41	67.58 ± 0.48	56.23 ± 1.08
TCL	56.46 ± 0.67	63.13 ± 1.14	45.31 ± 0.31	76.87 ± 0.85	44.78 ± 0.60	71.22 ± 0.58	59.63 ± 0.69
$\text{GearNet}_{\text{Co-teaching}}$	53.12 ± 1.88	58.12 ± 1.11	44.49 ± 0.57	76.87 ± 1.94	$\mathbf{49.28 \pm 1.37}$	69.14 ± 1.81	56.75 ± 1.44
$\text{GearNet}_{\text{DANN}}$	$\mathbf{60.68 \pm 0.26}$	63.54 ± 1.03	47.19 ± 0.96	76.88 ± 1.68	47.90 ± 0.66	72.39 ± 0.79	61.43 ± 0.90
$\text{GearNet}_{\text{TCL}}$	58.84 ± 0.57	$\mathbf{65.63 \pm 0.93}$	$\mathbf{48.37 \pm 1.04}$	$\mathbf{78.54 \pm 0.75}$	47.80 ± 0.58	$\mathbf{73.44 \pm 0.56}$	$\mathbf{62.10 \pm 0.74}$

TABLE 3.1: Target accuracy (%) on Office-31 datasets with Unif-20% noise. Bold numbers are superior results

Realizations of GearNet. In this subsection, we incorporate three backbone methods with GearNet as examples. They are Co-teaching [34] that belongs to the approach to improve model robustness against label noise, DANN [63] that belongs to the domain adaptation approach and TCL [19] that belongs to the WSDA approach, respectively. All the three algorithms are representative approaches, which could spotlight the universal capability of GearNet.

First of all, we also need to initialize two models f_θ and $\tilde{f}_{\tilde{\theta}}$ with the backbone algorithm. Although these backbone methods have different learning strategies, we generally express their loss functions as follows to highlight the structure of GearNet:

$$\ell_{bone} = \mathbb{E}_{p^s(x^s,\hat{y^s}),p^t(x^t)}(\ell(x^s, \hat{y^s}, x^t; f_\theta)), \tag{3.10}$$

$$\tilde{\ell}_{bone} = \mathbb{E}_{p^t(x^t,\hat{y^t}),p^s(x^s)}(\ell(x^t, \hat{y^t}, x^s; \tilde{f}_{\tilde{\theta}})), \tag{3.11}$$

where ℓ_{bone} denotes the loss of the backbone method for training f_θ, while $\tilde{\ell}_{bone}$ denotes the same meaning for training $\tilde{f}_{\tilde{\theta}}$. Besides, $p^s(*)$ and $p^t(*)$ denote the distribution from the source domain, and the target domain, respectively.

The two losses above take the place of ℓ_{super} and $\tilde{\ell}_{super}$ in Eq. (3.3) when we chose those backbone methods. They represent different meanings under various backbone algorithms. For the Co-teaching backbone method, they reduce the impact of noise by cross-updating two peer networks. As for DANN, they decrease the domain discrepancy using a domain discriminator in an adversarial manner. For TCL, they address both the label noise problem and the domain shift problem by selecting noiseless and transferable samples from the source domain to train the DANN-shape model.

To obtain ℓ_{guide} and $\tilde{\ell}_{guide}$, the training model should align its predictions with corresponding class posteriors of its dual model. Note that only classification predictions need to be aligned. Besides, for multi-classifier models, like Co-teaching, these losses should be computed for every classifier with the dual model, so that all of them can have the professional guidance on the domain that they are not good at.

Relation to CycleGAN. CyCADA [21] and Bi-Directional Generation domain adaptation model (BGD) [102] also propose the idea that transfers knowledge from the target domain to the source domain, which are inspired by CycleGAN [103]. They transfer the instances of the target domain to that of the source domain by a feature generator, which aim is to obtain a feature space that is close to the source domain. However, there are fundamental differences between them and GearNet. (i) CyCADA and BGD obtain a feature generator that can convert the target-style feature to the source-style feature, but GearNet trains a model that leverages the target domain to predict labels of the source domain. (ii) The reason why CyCADA and BGD transfer the feature style is to predict the labels of the target domain using the classifier trained by the source domain. But GearNet aims to exploit information from the target domain which could enhance both the domain adaptation process and the noise reduction process for the source domain. (iii) CyCADA and BGD address the issue of unsupervised domain adaptation, but GearNet handles the issue of weakly-supervised domain adaptation.

Relation to multi-task learning. In the problem of multi-task learning, there is also an idea that a noisy task can reduce its noise under the assistance of another noisy task [104]. However, the crucial difference between multi-task learning and GearNet is that multi-task learning aims at good performance for all the tasks, but GearNet only needs to achieve good performance for the target domain. In addition, the above idea for multi-task learning proposes that more noisy tasks can reduce the impact of noise and get better performance by up weighting less noisy tasks, but GearNet proposes that both the target domain and the source domain contain useful knowledge for each other.

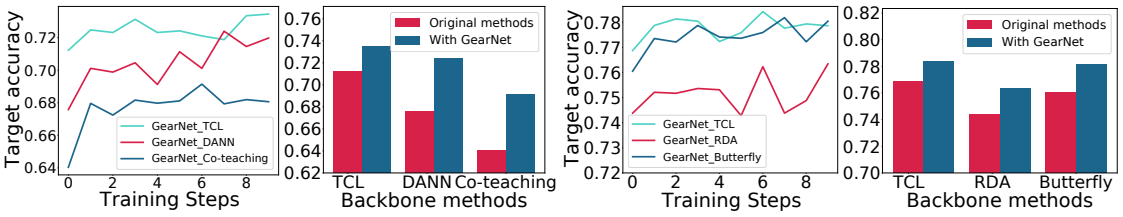


FIGURE 3.2: Universal capability of GearNet on $D \to W$ with Unif-20% noise for (a) and (b) and $W \to D$ with Unif-20% noise for (c) and (d) under different backbone methods. (a) and (c) present the trend of target accuracy across training steps. (b) and (d) present the best target accuracy across training steps.

3.4 Experiments

3.4.1 Experimental setup

We compare **GearNet**[1] with 6 state-of-the-art baselines, implement all methods by PyTorch, and conduct all the experiments on NVIDIA Tesla V100 GPU. Their details are as follows: **Standard** [105], which is a neural network classifier constructed by the pre-trained ResNet-50. Note that ResNet-50 is also used as the feature extractor of the following benchmarks for comparability. **DANN** [63], which is constructed by a feature extractor, a classification layer and a domain discriminator. The feature extractor generates a feature space that can confuse the domain discriminator, so that it can decrease the domain discrepancy. **DAN** [106], which proposes MK-MMD as the domain discrepancy measure to reduce the difference between the two domains. **Co-teaching** [34], which trains and cross-updates two peer networks simultaneously to combat label noise. **JoCoR** [107], which trains two neural networks simultaneously and calculates a joint loss with Co-regularization to merge their outputs between the two neural networks in order to improve the model robustness against label noise. **TCL** [19], which selects transferable and noiseless samples from the source domain to train a model with the same structure as DANN to handle the WSDA issue. **RDA** [100], which proposes an offline curriculum learning to select clean samples from the source domain and a proxy margin discrepancy to eliminate the negative impact of label noise. **Butterfly** [20], which picks clean samples from both of the two domains to train two Co-teaching models simultaneously.

[1]The code is published on https://github.com/Renchunzi-Xie/GearNet.git

Tasks	$A \to W$	$A \to D$	$W \to A$	$W \to D$	$D \to A$	$D \to W$	Average
Standard	34.37 ± 0.78	36.45 ± 0.43	29.26 ± 0.52	54.79 ± 0.72	30.79 ± 0.95	48.31 ± 0.69	40.00 ± 0.68
Co-teaching	36.20 ± 2.45	41.04 ± 1.42	31.11 ± 1.64	53.13 ± 1.91	23.08 ± 1.63	38.93 ± 2.63	37.25 ± 1.95
JoCoR	37.11 ± 1.27	42.29 ± 2.24	28.98 ± 1.78	47.50 ± 1.89	23.40 ± 2.09	36.85 ± 2.74	36.02 ± 2.00
DAN	34.24 ± 1.73	35.83 ± 2.42	23.97 ± 2.89	47.71 ± 1.99	24.96 ± 2.07	41.02 ± 1.79	34.62 ± 2.15
DANN	36.20 ± 1.62	40.20 ± 1.27	30.22 ± 1.53	54.38 ± 1.59	32.71 ± 2.06	49.74 ± 1.82	40.57 ± 1.65
TCL	42.06 ± 1.86	46.04 ± 2.68	29.55 ± 0.96	54.38 ± 1.74	30.43 ± 1.83	49.09 ± 2.62	41.95 ± 1.95
$\text{GearNet}_{\text{Co-teaching}}$	39.32 ± 1.07	43.33 ± 0.97	$\mathbf{33.94 \pm 1.07}$	56.04 ± 1.31	25.74 ± 1.58	38.41 ± 1.30	39.46 ± 1.21
$\text{GearNet}_{\text{DANN}}$	43.48 ± 1.45	$\mathbf{48.54 \pm 1.72}$	30.45 ± 1.37	$\mathbf{58.54 \pm 1.86}$	$\mathbf{35.51 \pm 0.83}$	$\mathbf{54.17 \pm 1.06}$	$\mathbf{45.12 \pm 1.38}$
$\text{GearNet}_{\text{TCL}}$	$\mathbf{46.61 \pm 0.89}$	47.50 ± 1.28	30.39 ± 2.01	55.83 ± 1.92	28.91 ± 1.47	52.47 ± 1.39	43.62 ± 1.49

TABLE 3.2: Target accuracy (%) on Office-31 datasets with Unif-40% noise. Bold numbers are superior results.

Tasks	$A \to W$	$A \to D$	$W \to A$	$W \to D$	$D \to A$	$D \to W$	Average
Standard	46.61 ± 0.92	46.67 ± 1.83	41.41 ± 0.94	69.17 ± 1.03	40.23 ± 1.06	64.19 ± 0.74	51.38 ± 1.08
Co-teaching	48.31 ± 2.01	50.21 ± 1.37	42.19 ± 1.87	69.17 ± 2.40	39.20 ± 1.82	58.59 ± 2.72	51.28 ± 2.03
JoCoR	49.74 ± 0.98	51.46 ± 1.52	41.94 ± 1.83	67.92 ± 2.07	37.71 ± 1.68	55.86 ± 0.93	50.77 ± 1.50
DAN	56.64 ± 1.73	53.54 ± 2.04	40.20 ± 2.13	70.21 ± 1.57	35.80 ± 1.78	64.84 ± 1.84	53.54 ± 1.85
DANN	47.66 ± 1.46	50.63 ± 1.76	39.17 ± 0.63	69.58 ± 1.05	41.55 ± 0.77	65.49 ± 0.86	52.35 ± 1.09
TCL	55.99 ± 1.79	61.04 ± 1.53	42.37 ± 2.51	72.92 ± 0.62	42.29 ± 1.33	70.96 ± 2.52	57.60 ± 1.72
$\text{GearNet}_{\text{Co-teaching}}$	51.95 ± 1.33	54.37 ± 1.13	44.49 ± 1.80	71.87 ± 0.93	41.79 ± 1.71	61.45 ± 0.97	54.32 ± 1.31
$\text{GearNet}_{\text{DANN}}$	$\mathbf{59.51 \pm 1.58}$	61.25 ± 0.63	41.44 ± 1.96	72.08 ± 1.05	44.89 ± 1.58	69.27 ± 1.62	58.07 ± 1.40
$\text{GearNet}_{\text{TCL}}$	58.85 ± 0.96	$\mathbf{62.71 \pm 0.73}$	$\mathbf{44.28 \pm 1.53}$	$\mathbf{75.00 \pm 1.67}$	$\mathbf{45.03 \pm 0.85}$	$\mathbf{75.00 \pm 1.09}$	$\mathbf{60.15 \pm 1.14}$

TABLE 3.3: Target accuracy (%) on Office-31 datasets with Flip-20% noise. The best results are highlighted in bold.

We simulate experiments based on **Office-31** [108] and **Office-Home** [109]. The first dataset is a classical dataset for domain adaptation, which contains 4,652 images with 31 classes. Three various domains are contained in the dataset: Amazon (**A**), Webcam (**W**) and DSLR (**D**). They represent that images are collected from amazon.com, web camera and digital SLR camera, respectively. The second dataset consists of 4 domains: Artistic (**Ar**), Clip Art (**Cl**), Product (**Pr**) and Real-World (**Rw**) containing 15,500 images with 65 classes. They represent different image styles that are artistic depictions, clipart images, images without background and pictures captured by cameras, respectively. We manually inject two types of noisy labels for the source domain: uniform noise [110] and asymmetry flipping noise [93], and their details are in the supplemental document.

For comparability, all the experiments use Stochastic gradient descent optimizer with an initial learning rate of 0.003 and a momentum of 0.9. The batch size is set as 32 and the total number of epochs is 200. For GearNet, the total number of steps is set as 10. To measure the performance, we evaluate the target accuracy by all the samples from the target domain, i.e., *target accuracy = (# of correct target predictions)/(# of target domain data)*. All the experiments are repeated 5

times with different seeds, and we report the average accuracy and their standard deviation on tables.

3.4.2 Numerical results

Results on Office-31. Table 3.1, Table 3.2, Table 3.3 and Table 3.4 report the target-domain accuracy in 6 digit tasks that are combined in pairs by the three domains from Office-31 under different types and levels of noise.

Table 3.1 and Table 3.3 represent two simpler cases, since their noise rate is 20%. The two tables illustrate that our method is able to improve existing backbone methods significantly when the noise rate is low. Table 3.2 and Table 3.4 represent two harder cases with 40% noise rate. The two tables show that GearNet can enhance the performance of original algorithms on most tasks when the noise rate is high. GearNet obtains comparable results on the tasks of $D \to A$ and $W \to A$. The reason is that Webcam and DSLR are two small datasets compared with Amazon, so that it is ineffective to transfer knowledge from Webcam or DSLR to Amazon under 40% noise rate. When we explore pseudo supervision information from Amazon based on the pseudo labels provided by the two small datasets, it is more likely to explore negative information and transfer it back to Amazon. However, when the noise rate becomes 20%, all the tasks can be improved by GearNet, since all of the source domains are capable to provide adequate transferable information to their target domains and vise versa.

Tasks	$A \to W$	$A \to D$	$W \to A$	$W \to D$	$D \to A$	$D \to W$	Average
Standard	35.93 ± 1.09	37.50 ± 1.14	30.39 ± 1.21	53.96 ± 1.04	29.83 ± 1.81	46.88 ± 1.35	39.08 ± 1.27
Co-teaching	35.94 ± 1.97	37.92 ± 2.48	28.13 ± 2.09	49.17 ± 1.35	26.03 ± 1.71	37.76 ± 1.95	35.82 ± 1.93
JoCoR	36.07 ± 1.85	37.29 ± 1.27	27.73 ± 1.88	48.12 ± 2.08	26.35 ± 1.11	38.28 ± 1.60	35.64 ± 1.63
DAN	43.49 ± 1.17	41.46 ± 2.14	30.61 ± 1.43	53.54 ± 2.01	28.94 ± 2.36	50.39 ± 2.35	41.41 ± 1.91
DANN	36.98 ± 1.84	40.42 ± 1.89	30.26 ± 2.04	53.33 ± 2.58	30.36 ± 1.52	44.66 ± 1.92	39.96 ± 1.96
TCL	44.79 ± 0.98	43.75 ± 1.57	30.82 ± 1.37	54.17 ± 1.90	29.97 ± 1.86	45.96 ± 1.37	41.58 ± 1.51
$\text{GearNet}_{\text{Co-teaching}}$	40.36 ± 0.85	38.75 ± 1.14	30.39 ± 0.58	51.04 ± 1.17	27.37 ± 0.64	40.23 ± 1.36	38.02 ± 0.96
$\text{GearNet}_{\text{DANN}}$	42.45 ± 1.33	42.08 ± 1.54	$\mathbf{31.21 \pm 1.73}$	$\mathbf{54.38 \pm 1.48}$	$\mathbf{31.35 \pm 0.79}$	45.96 ± 1.63	41.24 ± 1.42
$\text{GearNet}_{\text{TCL}}$	$\mathbf{48.69 \pm 1.02}$	$\mathbf{46.25 \pm 1.46}$	30.64 ± 1.58	53.96 ± 1.24	31.00 ± 0.89	$\mathbf{47.79 \pm 1.24}$	$\mathbf{43.06 \pm 1.24}$

TABLE 3.4: Target accuracy (%) on Office-31 datasets with Flip-40% noise. Bold numbers are superior results.

To show the universal capability of GearNet on backbone methods from different fields, we draw Fig. 3.2. The left figure illustrates the trend of target accuracy of GearNet across training steps under the three backbone methods. The step 0

Tasks	Standard	Co-teaching	JoCoR	DAN	DANN	TCL	GearNet$_{\text{Co-teaching}}$	GearNet$_{\text{DANN}}$	GearNet$_{\text{TCL}}$
$Ar \rightarrow Cl$	24.51±1.82	26.49±1.40	26.60±1.93	23.12±0.90	26.41±1.51	26.48±0.93	27.82±0.42	27.30±1.07	**28.01±0.75**
$Ar \rightarrow Pr$	42.41±1.59	45.15±1.79	45.08±2.02	33.94±0.95	41.13±1.20	42.75±1.06	**52.15±0.73**	47.01±1.05	46.58±0.93
$Ar \rightarrow Rw$	48.75±1.89	51.51±1.67	51.56±2.28	50.33±1.10	49.60±1.66	49.63±1.12	**54.71±0.70**	51.12±1.15	52.33±1.50
$Cl \rightarrow Ar$	27.58±1.60	27.70±1.17	27.91±1.76	26.11±0.65	34.08±1.67	36.22±0.96	31.20±0.77	36.34±0.82	**37.37±1.47**
$Cl \rightarrow Pr$	34.60±1.09	35.39±1.63	34.98±1.48	31.61±1.18	38.22±2.21	41.89±1.30	42.77±0.75	43.09±0.98	**43.22±0.91**
$Cl \rightarrow Rw$	37.59±1.45	26.08±1.07	37.20±1.43	34.28±1.40	42.41±1.63	45.35±1.48	43.01±0.84	44.78±0.56	**46.48±0.86**
$Pr \rightarrow Ar$	29.33±0.83	32.12±1.31	32.08±1.34	25.12±0.57	34.62±1.71	35.15±1.40	33.92±1.22	36.75±0.83	**37.58±1.27**
$Pr \rightarrow Cl$	23.18±1.68	23.92±0.60	24.11±1.59	23.47±1.78	24.00±1.70	25.79±0.69	23.72±1.13	24.11±1.10	**28.46±1.42**
$Pr \rightarrow Rw$	46.07±1.91	48.32±1.07	48.80±1.39	40.92±1.00	49.65±1.56	52.68±1.55	50.22±0.69	50.89±1.05	**54.84±1.11**
$Rw \rightarrow Ar$	41.37±1.58	43.20±1.35	44.03±1.84	35.48±1.09	43.66±2.14	44.98±0.84	44.37±0.91	44.61±0.56	**46.54±1.49**
$Rw \rightarrow Cl$	26.76±1.89	28.12±1.36	28.55±1.59	27.14±0.73	28.30±1.10	29.03±1.92	28.54±1.12	28.80±1.03	**31.06±1.38**
$Rw \rightarrow Pr$	51.76±1.71	53.94±1.58	53.87±1.29	46.15±1.46	51.88±1.79	54.84±1.06	55.19±1.27	53.98±1.27	**57.24±1.16**
Average	36.16±1.59	36.83±1.34	37.90±1.65	33.14±1.05	38.66±1.20	40.40±1.19	40.64±0.87	40.73±0.96	**42.48±1.19**

TABLE 3.5: Target accuracy (%) on Office-Home datasets with Unif-20% noise. The best results are highlighted in bold.

refers to the pretrained process, the even number step (i.e., step 2, 4, ...) refers the forward step from the source domain to the target domain, and the odd number step (i.e., step 1, 3, 5, ...) means the backward step from the target domain to the source domain. Specifically, the value of the first point denotes the target accuracy of the original backbone model. The right figure illustrates the performance improvement before and after we incorporate GearNet with the three methods. From the figures, we can observe that GearNet can continuously enhance the performance of the backbone methods. The figures on $D \rightarrow W$ with Unif-40% noise, Flip-20% noise and Flip-40% noise are in the supplemental document. To illustrate the universal

	GearNet$_{\text{Co-teaching}}$		GearNet$_{\text{DANN}}$		GearNet$_{\text{TCL}}$	
Tasks	w/o	w/	w/o	w/	w/o	w/
$D \rightarrow W$ Unif-20%	67.58	**69.14**	70.18	**72.39**	71.74	**73.44**
$D \rightarrow W$ Unif-40%	36.58	**38.41**	49.09	**54.17**	49.47	**52.47**
$Ar \rightarrow Cl$ Unif-20%	27.39	**27.82**	26.53	**27.30**	26.41	**28.01**
$Ar \rightarrow Cl$ Unif-40%	18.65	**18.75**	17.56	**17.58**	16.30	**17.93**

TABLE 3.6: Results of ablation study on various tasks with Uniform noise. The best results are highlighted in bold.

capability of GearNet on different WSDA methods, we incorporate three WSDA methods with GearNet including TCL [19], Butterfly [20] and RDA [100] based on the Office-31 benchmark dataset (source: Webcam; target: DSLR). Their results are shown in Fig. **??**. The two figures illustrate that GearNet can also significantly improve the prediction accuracy on the target domain for various WSDA methods.

Results on Office-Home. Table 3.5 report the average target accuracy on 12 tasks that are combined in pairs by the 4 domains from Office-Home when the uniform noise rate is 20%. This table shows that GearNet can significantly improve

the performance compared with original models. Especially, in the case of $Ar \rightarrow Pr$, the target accuracy is from 42% to 52% when the backbone method is Co-teaching, which is a significant improvement.

3.5 Chapter Summary

In this chapter, we propose a universal paradigm called GearNet that enhances many existing robust training methods to address the issue of *weakly-supervised domain adaptation*. To the best of our knowledge, our method is the first to explore the benefit of utilizing pseudo supervision knowledge from the target domain in improving the robustness against noisy labels from the source domain. We show that exploring bilateral relationships would further improve the generalization performance when the labels from the source domain are noisy. Extensive experiments show that GearNet is easy to be integrated into existing algorithms and these methods equipped with GearNet can significantly outperform their original performance. Overall, our method is an effective and complementary approach for boosting robustness against noisy labels in the setting of domain adaptation.

Chapter 4

On the Importance of Feature Separability in Predicting Out-Of-Distribution Error

4.1 Introduction

Machine learning techniques deployed in the open world often struggle with distribution shifts, where the test data are not drawn from the training distribution. Such issues can substantially degrade the test accuracy, and the generalization performance of a trained model may vary significantly on different shifted datasets [4, 68]. This gives rise to the importance of estimating out-of-distribution (OOD) errors for AI Safety [28]. However, it is prohibitively expensive or unrealistic to collect large-scale labeled examples for each shifted testing distribution encountered in the wild. Subsequently, predicting OOD error becomes a challenging task without access to ground-truth labels.

In the literature, a popular direction in predicting OOD error is to utilize the model output on the shifted dataset [25–27], which heavily relies on the model calibration. Yet, machine learning models generally suffer from the overconfidence issue even for OOD inputs [111], leading to suboptimal performance in estimating OOD performance. Many prior works turned to measuring the distribution difference between training and OOD test set, due to the conventional wisdom that a higher distribution shift normally leads to lower OOD accuracy. AutoEval [28] applies

Fréchet Distance to calculate the distribution distance for model evaluation under distribution shift. A recent work [7] introduces Projection Norm (ProjNorm) metric to measure the distribution discrepancy in network parameters. However, we find that the connection between distribution distance and generalization performance does not always hold, making these surrogate methods to be questionable. This motivates our method, which directly estimates OOD accuracy based on the feature properties of test instances.

In this work, we show that feature separability is strongly associated with test accuracy, even in the presence of distribution shifts. Theoretically, we demonstrate that the upper bound of Bayes error is negatively correlated with inter-class feature distances. To quantify the feature separability, we introduce a simple dataset-level statistic, Dispersion Score, which gauges the inter-class divergence from feature representations, i.e., outputs of feature extractor. Our method is motivated by the desirable properties of embeddings in representation learning [112]. To achieve high accuracy, we generally desire embeddings where different classes are relatively far apart (i.e., high inter-class dispersion), and samples in each class form a compact cluster (i.e., high intra-class compactness). Surprisingly, our analysis shows that intra-class compactness does not reflect the generalization performance, while inter-class dispersion is strongly correlated with the model accuracy on OOD data.

Extensive experiments demonstrate the superiority of Dispersion Score over existing methods for estimating OOD error. First, our method dramatically outperforms existing training-free methods in evaluating model performance on OOD data. For example, our method leads to an increase of the R^2 from 0.847 to 0.970 on TinyImageNet-C [8] – a 14.5% of relative improvement. Compared to the recent ProjNorm method [7], our method not only achieves superior performance by a meaningful margin, but also maintains huge advantages in computational efficiency and sample efficiency. For example, using CIFAR-10C dataset as OOD data, Dispersion Score achieves an R^2 of 0.972, outperforming that of ProjNorm (i.e., 0.947), while our approach only takes around 3% of the time consumed by ProjNorm.

Overall, using Dispersion Score achieves strong performance in OOD error estimation with high computational efficiency. Our method can be easily adopted in practice. It is straightforward to implement with deep learning models and does not require access to training data. Thus our method is compatible with modern settings where models are trained on billions of images.

We summarize our main contribution as follows:

1. We find that the correlation between distribution distance and generalization performance does not always hold, downgrading the reliability of existing distance-based methods.

2. Our study provides empirical evidence supporting a significant association between feature separability and test accuracy. Furthermore, we theoretically show that increasing the feature distance will result in a decrease in the upper bound of Bayes error.

3. We propose a simple dataset-level score that gauges the inter-class dispersion from feature representations, i.e., outputs of feature extractor. Our method does not rely on the information of training data and exhibits stronger flexibility in OOD test data.

4. We conduct extensive evaluations to show the superiority of the Dispersion Score in both prediction performance and computational efficiency. Our analysis shows that Dispersion Score is more robust to various data conditions in OOD data, such as limited sample size, class imbalance and partial label set. Besides, we show that intra-class compactness does not reflect the generalization performance under distribution shifts (SubSection 4.4.4).

4.2 Problem Setup and Motivation

4.2.1 Preliminaries: OOD performance estimation

Setup In this work, we consider multi-class classification task with K classes. We denote the input space as $\mathcal{X}$ and the label space as $\mathcal{Y} = \{1, \dots, K\}$. We assume there is a training dataset $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^{N}$, where the N data points are sampled *i.i.d.* from the distribution p_S. During training, we learn a neural network $f : \mathcal{X} \to \mathbb{R}^K$ with trainable parameters $\boldsymbol{\theta} \in \mathbb{R}^p$ on $\mathcal{D}$.

In particular, the neural network f can be viewed as a combination of a feature extractor $f_{\mathbf{g}}$ and a classifier $f_{\boldsymbol{\omega}}$, where $\mathbf{g}$ and $\boldsymbol{\omega}$ denote the parameters of the corresponding parts, respectively. The feature extractor $f_{\mathbf{g}}$ is a function that maps

instances to features $f_{\mathbf{g}} : \mathcal{X} \to \mathcal{Z}$, where $\mathcal{Z}$ denotes the feature space. We denote by $\mathbf{z}_i$ the learned feature of instance $\mathbf{x}_i$: $\mathbf{z}_i = f_g(\mathbf{x}_i)$. The classifier f_ω is a function from the feature space $\mathcal{Z}$ to $\mathbb{R}^k$, which outputs the final predictions. A trained model can be obtained by minimizing the following expected risk:

$$\begin{aligned}\mathcal{R}_{\mathcal{L}}(f) &= \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{P}_{\mathcal{X}\mathcal{Y}}}\left[\mathcal{L}\left(f(\mathbf{x};\theta), y\right)\right] \\ &= \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{P}_{\mathcal{X}\mathcal{Y}}}\left[\mathcal{L}\left(f_\omega(f_g(\mathbf{x})), y\right)\right]\end{aligned}$$

Problem statement At test time, we generally expect that the test data are drawn from the same distribution as the training dataset. However, distribution shifts usually happen in reality and even simple shifts can lead to large drops in performance, which makes it unreliable in safety-critical applications. Thus, our goal is to estimate how a trained model might perform on the shifted data without labels, i.e., unlabeled out-of-distribution (OOD) data.

Assume that $\mathcal{D}_{\text{test}} = \{\tilde{\mathbf{x}}_i\}_{i=1}^m$ be the OOD test dataset and $\{\tilde{y}_i\}_{i=1}^m$ be the corresponding unobserved labels. For a certain test instance $\tilde{\mathbf{x}}_i$, we obtain the predicted labels of a trained model by $\tilde{y}_i' = \arg\max f(\tilde{\mathbf{x}}_i)$. Then the ground-truth test error on OOD data can be formally defined as:

$$\text{Err}(\mathcal{D}_{\text{test}}) = \frac{1}{m}\sum_{i=1}^{m} \mathbb{1}(\tilde{y}_i' \neq \tilde{y}_i), \tag{4.1}$$

To estimate the real OOD error, the key challenge is to formulate a score $S(\mathcal{D}_{\text{test}})$ that is strongly correlated with the test error across diverse distribution shifts without utilization of corresponding test labels. With such scores, a simple linear regression model can be learned to estimate the test error on shifted datasets, following the commonly used scheme [7, 28].

While those output-based approaches suffer from the overconfidence issue [24, 25], other methods primarily rely on the distribution distance between training data $\mathcal{D}$ and test data $\mathcal{D}_{\text{test}}$ [28, 64], with the intuition that the distribution gap impacts classification accuracy. In the following, we motivate our method by analyzing the failure of those methods based on feature-level distribution distance.

4.2.2 The failure of distribution distance

In the literature, distribution discrepancy has been considered as a key metric to predict the generalization performance of the model on unseen datasets [7, 28, 64, 113, 114]. AutoEval [28] estimates model performance by quantifying the domain gap in the feature space:

$$S(\mathcal{D}, \mathcal{D}_{\text{test}}) = d(\mathcal{D}, \mathcal{D}_{\text{test}}),$$

where $d(\cdot)$ denotes the distance function, such as Fréchet Distance [2] or maximum mean discrepancy (MMD) [3].

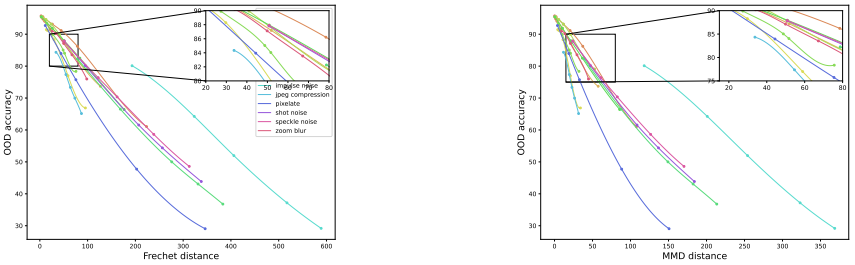
ProjNorm measures the distribution gap with the Euclidean distance in the parameter space:

$$S(\mathcal{D}, \mathcal{D}_{\text{test}}) = \|\boldsymbol{\theta} - \tilde{\boldsymbol{\theta}}\|_2,$$

where $\boldsymbol{\theta}$ and $\tilde{\boldsymbol{\theta}}$ denote the parameters fine-tuned on training data $\mathcal{D}$ and OOD data $\mathcal{D}_{\text{test}}$, respectively.

The underlying assumption is inherited from the conventional wisdom in domain adaptation, where a representation function that minimizes domain difference leads to higher accuracy in the target domain [64, 115, 116]. Yet, the relationship between distribution distance and test accuracy remains controversial. For example, the distribution shift may not change the model prediction if the classifier's outputs are insensitive to changes in the shifted features. There naturally arises a question: given a fixed model, does a larger distribution distance always lead to a lower test accuracy?

To verify the correlation between distribution distance and model performance, we compare the model performance on different OOD test sets of CIFAR-10C, using the ResNet-50 model trained on CIFAR-10. For each OOD test set, we calculate the distribution distances in the feature space $\mathcal{Z}$ via Fréchet distance [2] and MMD [3], respectively. Figure 4.1 presents the classification accuracy versus the distribution distance. The results show that, on different test datasets with similar distances to the train data in the feature space. the performance of the trained model varies significantly with both the two distance metrics. For example,



(A) Fréchet (B) MMD

FIGURE 4.1: Distribution Gap Vs. OOD accuracy on CIFAR10-C via (a) Fréchet distance [2] and (b) MMD [3]. All colors indicate 14 types of corruption. The test accuracy varies significantly on shifted datasets with the same distribution distances.

calculated by Fréchet distance, the distribution gap varies only a small margin from 223.36 to 224.04, but the true accuracy experiences a large drop from 61.06% to 46.99%. Similar to MMD distance, when the distribution gap changes from 87.63 to 88.35, the OOD accuracy drops from 66.53% to 47.73%. The high variability in the model performance reveals that, **the distribution difference is not a reliable surrogate for generalization performance under distribution shifts**.

4.3 Proposed Method

In this section, we first introduce the intuition of our method with a natural example. Next, we characterize and provide quantitative measure on the desirable properties of feature representations for predicting OOD error. We then present the advantages of our proposed method.

4.3.1 Motivation

In representation learning, it is desirable to learn a separable feature space, where different classes are far away from each other and samples in each class form a compact cluster. Therefore, separability is viewed as an important characteristic to measure the feature quality. For example, one may train a nearest neighbor classifier over the learned representation using labeled data and regard its performance as the indicator. In this chapter, we investigate how to utilize the characteristic of

separability for estimating the model performance under distribution shifts, without access to ground-truth labels.

Intuitive example In Figure 4.2, we present a t-SNE visualization of the representation for training and test data. In particular, we compare the separability of the learned representation on shifted datasets with various severity. While the feature representation of training data exhibits well-separated clusters, those of the shifted datasets are more difficult to differentiate and the cluster gaps are well correlated with the corruption severity, as well as the ground-truth accuracy. It implies that, a model with high classification accuracy is usually along with a well-separated feature distribution, and vice versa. With the intuition, we proceed by theoretically showing how the feature separability affects the classification performance.

Theoretical explanation Recall that we obtain the model prediction by $y_i' = \arg\max f_\omega(\mathbf{z}_i)$, where the intermediate feature $\mathbf{z}_i = f_g(\mathbf{x}_i)$. Let $P(y=i)$ denote the prior class probability of class i, and $p(\mathbf{x}|y=i)$ denote the class likelihood, i.e., the conditional probability density of x given that it belongs to class i. Then the Bayes error [117–120] can be expressed as:

$$E_{\text{bayes}} = 1 - \sum_{i=1}^{k} \int_{C_i} P\left(y=i\right) p\left(\mathbf{x} \mid y=i\right) d\mathbf{x},$$

where C_i is the region where class i has the highest posterior. The Bayes error rate provides a lower bound on the error rate that can be achieved by any pattern classifier acting on derived features. However, the Bayes error is not so readily obtainable as class priors and class-conditional likelihoods are normally unknown. Therefore, many studies have focused on deriving meaningful upper bounds for the Bayes error [118, 119].

For a binary classification problem where $k=2$, we have

$$E_{\text{bayes}} \leq \frac{2P\left(y=1\right)P\left(y=2\right)}{1+P\left(y=1\right)P\left(y=2\right)\Delta}, \tag{4.2}$$

where Δ denotes the distance between features from the two classes [118, 121], such as *Mahalanobis distance* [122] or *Bhattacharyya distance* [123]. Based on the Bayes

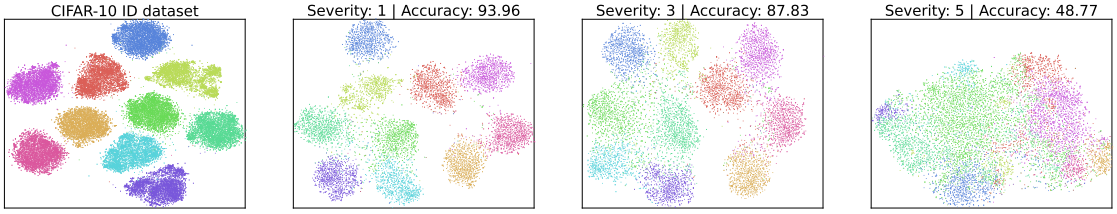


FIGURE 4.2: t-SNE visualization of feature representation on training and OOD test datasets (CIFAR-10C) with *contrast* corruption under different severity levels. The first left figure shows feature representation of the training dataset, while the rest of three figures illustrate those of the OOD datasets. From this figure, we observe that different clusters tends to be more separated as the corruption severity level gets smaller.

error for 2-class problems, the upper bound can be extended to the multi-class setting ($k > 2$) by the following equation [124].

$$E^k_{\text{bayes}} \leq \min_{\alpha \in \{0,1\}} \left(\frac{1}{k - 2\alpha} \sum_{i=1}^{k} \left(1 - P\left(y = i\right)\right) E^{k-1}_{\text{bayes }; i} + \frac{1 - \alpha}{k - 2\alpha} \right), \tag{4.3}$$

where α is an optimization parameter. With Equations 4.2 and 4.3, we obtain an upper bound of Bayes error, which is negatively correlated with the inter-class feature distances. Specifically, increasing the feature distance will result in a decrease in the upper bound of Bayes error. In this manner, we provide a mathematical intuition for the phenomenon shown in Figure 4.2. However, Computing the upper bound of Bayes error directly, as indicated above, is challenging due to its computational complexity. To circumvent this issue, we propose a straightforward metric as a substitute, which does not require access to label information.

4.3.2 Dispersion score

In our previous analysis, we show a strong correlation between the test accuracy and feature separability under different distribution shifts. To quantify the separability in the feature space $\mathcal{Z}$, we introduce *Dispersion score* that measures the inter-class margin without annotation information.

First, we allocate OOD instances $\{\tilde{\mathbf{x}}\}_{i=1}^{m}$ into different clusters j based on the model predictions, i.e., their pseudo labels from the trained classifier f_ω: $j = \tilde{y}'_i = \arg\max f_\omega(\mathbf{z}_i)$.

With these clusters, we compute the *Dispersion score* by the average distances between each cluster centroid $\tilde{\mu}_j = \frac{1}{m_j}\sum_{i=1}^{m_j} \mathbf{z}_i \cdot \mathbb{1}\{\tilde{y}'_i = j\}$ and the center of all features $\bar{\mu} = \frac{1}{m}\sum_{i=1}^{m} \mathbf{z}_i$, weighted by the sample size of each cluster m_j. Formally, the *Dispersion score* is defined as:

$$S(\mathcal{D}_{\text{test}}) = \frac{1}{k-1}\sum_{j=1}^{k} m_j \cdot \varphi(\bar{\mu}, \tilde{\mu}_j)$$

where $k-1$ is the degree of freedom and φ denotes the distance function. With the weight m_j, the induced score receives a stronger influence from those larger clusters, i.e., the majority classes. This enables our method to be more robust to the long-tailed issue, which naturally arises in the unlabeled OOD data. In subsection 4.4.3, we explicitly show the advantage of our method in the class-imbalanced case and analyze the importance of the weight in Appendix 4.4.5.

In particular, we use square Euclidean distances to calculate the distance in the feature space $\mathcal{Z}$. So it converts to:

$$S(\mathcal{D}_{\text{test}}) = \log \frac{\sum_{j=1}^{k} m_j \cdot \|\bar{\mu} - \tilde{\mu}_j\|_2^2}{k-1} \tag{4.4}$$

Following the common practice [71], we adopt a log transform on the final score, which corresponds to multiplicative combination of class statistics.

We summarize our approach in Appendix B.2. Notably, the Dispersion score derived from the feature separability offers several compelling advantages:

1. **Training data free**. The calculation procedure of our proposed score does not rely on the information of training data. Thus, our method is compatible with modern settings where models are trained on billions of images.

2. **Easy-to-use**. The computation of the Dispersion score only does forward propagation for each test instance once and does not require extra hyperparameter, training a new model, or updating the model parameters. Therefore, our method is easy to implement in real-world tasks and computational efficient, as demonstrated in Tables 6.3 and 4.3.

3. **Strong flexibility in OOD data**. Previous state-of-the-art methods, like ProjNorm [7], usually requires a large amount of OOD data for predicting the prediction performance. Besides, the class distribution of unlabeled OOD

datasets might be severely imbalanced, which makes it challenging to estimate the desired test accuracy on balanced data. In Subsection 4.4.3, we will show the Dispersion score derived from the feature separability exhibits stronger flexibility and generality in sample size, class distribution, and partial label set (see Subsection 4.4.3 and Appendix 4.5.3).

4.4 Experiments

In this subsection, we first compare the proposed score to existing training-free methods. Then, we provide an extensive comparison between our method and recent state-of-the-art method – ProjNorm. We then show the flexibility of our method under different settings of OOD test data. Additionally, we present an analysis of using ground-truth labels and K-means in Appendixes 4.5.1 and 4.4.6. Finally, we discuss the limitation of the proposed score for adversarial setting in Appendix 4.5.4.

4.4.1 Experimental setup

Train datasets. During training, we train models on the CIFAR-10, CIFAR-100 [125] and TinyImageNet [126] datasets. Specifically, the train data of CIFAR-10 and CIFAR-100 contain 50,000 training images, which are allocated to 10 and 100 classes, respectively. The TinyImageNet dataset contains 100,000 64×64 training images, with 200 classes.

Out-of-distribution (OOD) datasets. To evaluate the effectiveness of the proposed method on predicting OOD error at test time, we use CIFAR-10C and CIFAR-100C [8], which span 19 types of corruption with 5 severity levels. For the testing of TinyImageNet, we use TinyImageNet-C [8] that spans 15 types of corruption with 5 severity levels as OOD dataset. All the datasets with certain corruption and severity contain 10,000 images, which are evenly distributed in the classes.

Evaluation metrics. To measure the linear relationship between OOD error and designed scores, we use coefficients of determination (R^2) and Spearman correlation coefficients (ρ) as the evaluation metrics. For the comparison of computational

TABLE 4.1: Performance comparison of training free approaches on CIFAR-10, CIFAR-100 and TinyImageNet, where R^2 refers to coefficients of determination, and ρ refers to Spearman correlation coefficients (higher is better). The best results are highlighted in **bold**.

Dataset	Network	Rotation		ConfScore		Entropy		AgreeScore		ATC		Fréchet		Ours	
		R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ
CIFAR 10	ResNet18	0.822	0.951	0.869	0.985	0.899	0.987	0.663	0.929	0.884	0.985	0.950	0.971	**0.968**	**0.990**
	ResNet50	0.835	0.961	0.935	0.993	0.945	**0.994**	0.835	0.985	0.946	**0.994**	0.858	0.964	**0.987**	0.990
	WRN-50-2	0.862	0.976	0.943	**0.994**	0.942	**0.994**	0.856	0.986	0.947	**0.994**	0.814	0.973	**0.962**	0.988
	Average	0.840	0.963	0.916	0.991	0.930	**0.992**	0.785	0.967	0.926	0.991	0.874	0.970	**0.972**	0.990
CIFAR 100	ResNet18	0.860	0.936	0.916	0.985	0.891	0.979	0.902	0.973	0.938	0.986	0.888	0.968	**0.952**	**0.988**
	ResNet50	0.908	0.962	0.919	0.984	0.884	0.977	0.922	0.982	0.921	0.984	0.837	0.972	**0.951**	**0.985**
	WRN-50-2	0.924	0.970	0.971	0.984	0.968	0.981	0.955	0.977	0.978	**0.993**	0.865	0.987	**0.980**	0.991
	Average	0.898	0.956	0.936	0.987	0.915	0.983	0.927	0.982	0.946	**0.988**	0.864	0.976	**0.962**	**0.988**
TinyImageNet	ResNet18	0.786	0.946	0.670	0.869	0.592	0.842	0.561	0.853	0.751	0.945	0.826	0.970	**0.966**	**0.986**
	ResNet50	0.786	0.947	0.670	0.869	0.651	0.892	0.560	0.853	0.751	0.945	0.826	0.971	**0.977**	**0.986**
	WRNt-50-2	0.878	0.967	0.757	0.951	0.704	0.935	0.654	0.904	0.635	0.897	0.884	0.984	**0.968**	**0.986**
	Average	0.805	0.959	0.727	0.920	0.650	0.890	0.599	0.878	0.693	0.921	0.847	0.976	**0.970**	**0.987**

efficiency, we calculate the average evaluation time (T) for each test set with certain corruption and severity.

Training details. During the training process, we train ResNet18, ResNet50 [105] and WRN-50-2 [127] on CIFAR-10, CIFAR-100 [125] and TinyImageNet [126] with 20, 50 and 50 epochs, respectively. We use SGD with the learning rate of 10^{-3}, cosine learning rate decay [128], a momentum of 0.9 and a batch size of 128 to train the model.

The compared methods. We consider 7 existing methods as benchmarks for predicting OOD error: *Rotation Prediction* (Rotation) [23], *Averaged Confidence* (ConfScore) [24], *Entropy* [25], *Agreement Score* (AgreeScore) [26], *Averaged Threshold Confidence* (ATC) [27], *AutoEval* (Fréchet) [28], and *ProjNorm* [7]. Rotation and AgreeScore predict OOD error from the view of unsupervised loss constructed by generating rotating instances and measuring output agreement of two independent models, respectively. ConfScore, Entropy, and ATC formulate scores by model predictions. ProjNorm measures the parameter-level difference between the models fine-tuned on train and OOD test data respectively, while Fréchet quantifies the distribution difference in the feature space between the training and test datasets. We present the related works in Appendix A.2.

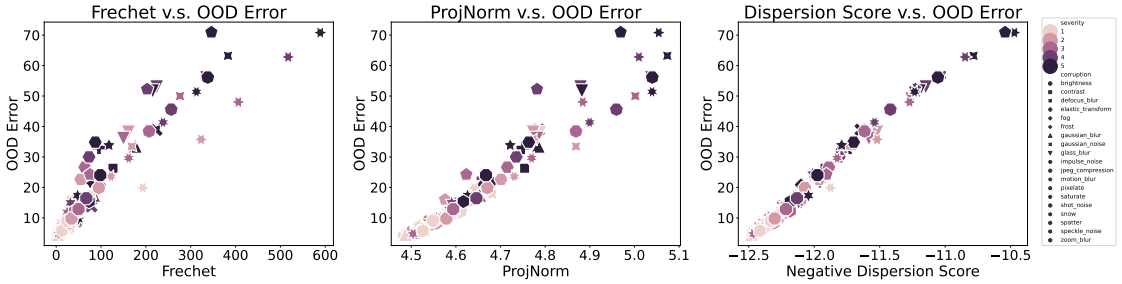


FIGURE 4.3: OOD error prediction versus True OOD error on CIFAR-10 with ResNet50. We compare the performance of Dispersion Score with that of ProjNorm and Fréchet via scatter plots. Each point represents one dataset under certain corruption and certain severity, where different shapes represent different types of corruption, and darker color represents the higher severity level.

4.4.2 Results

Can Dispersion score outperform existing training-free approaches? In Table 6.1, we present the performance of OOD error estimation on three model architectures and three datasets. We find that Dispersion Score dramatically outperforms existing training-free methods. For example, averaged across three architectures on TinyImageNet, our method leads to an increase of the R^2 from 0.847 to 0.970 – a 14.5% of relative improvement. In addition, Dispersion Score achieves consistently high performance over the three datasets with a R^2 higher than 0.950, while scores of other approaches such as Rotation varying from 0.787 to 0.924 are not stable. We observe a similar phenomenon on ρ, where the Entropy method achieves performance that is ranging from 0.842 to 0.994, while the performance of our method fluctuates around 0.988.

Dispersion score is superior to ProjNorm. In Table 6.3, we compare our method to the recent state-of-the-art method – ProjNrom [7] in both prediction performance and computational efficiency. The results illustrate that Dispersion Score could improve the prediction performance over ProjNorm with a meaningful margin. For example, with trained models on CIFAR-10 dataset, using Dispersion score achieves a R^2 of 0.953, much higher than the average performance of ProjNorm as 0.873. On CIFAR-100, our method also achieves comparable (slightly better) performance with ProjNorm (0.961 vs. 0.948). Besides, Dispersion score obtains huge advantages to ProjNorm in computational efficiency. Using WRN-50-2, ProjNorm takes an average of 575 seconds for estimating the performance of each OOD test dataset, while our method only requires 11 seconds. Since the computation of

TABLE 4.2: Performance comparison between ProjNorm [7] and our Dispersion score on CIFAR-10, CIFAR-100 and TinyImageNet, where R^2 refers to coefficients of determination, ρ refers to Spearman correlation coefficients (higher is better), and T refers to average evaluation time (lower is better). The best results are highlighted in **bold**.

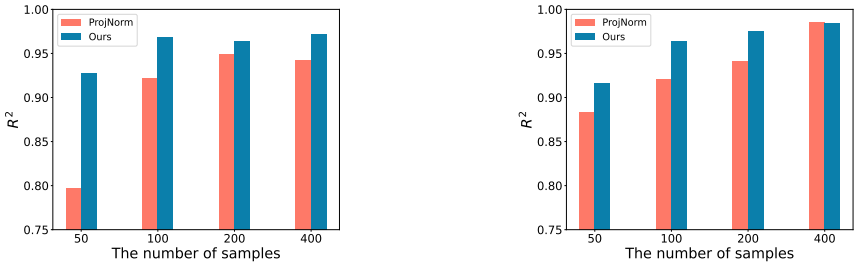
Dataset	Network	ProjNorm			Ours		
		R^2	ρ	T	R^2	ρ	T
CIFAR 10	ResNet18	0.936	0.982	179.616	**0.968**	**0.990**	**10.980**
	ResNet50	0.944	0.989	266.099	**0.987**	**0.990**	**11.259**
	WRN-50-2	0.961	**0.989**	575.888	**0.962**	0.988	**11.017**
	Average	0.947	0.987	326.201	**0.972**	**0.990**	**11.085**
CIFAR 100	ResNet18	**0.979**	0.980	180.453	0.952	**0.988**	**6.997**
	ResNet50	**0.988**	**0.991**	262.831	0.953	0.985	**11.138**
	WRN-50-2	**0.990**	**0.991**	605.616	0.980	**0.991**	**12.353**
	Average	**0.985**	0.987	349.63	0.962	**0.988**	**10.163**
TinyImageNet	ResNet18	**0.970**	0.981	182.127	0.966	**0.986**	**7.039**
	ResNet50	**0.979**	0.987	264.651	0.977	**0.990**	**13.938**
	WRN-50-2	0.965	0.983	590.597	**0.968**	**0.986**	**11.235**
	Average	**0.972**	0.984	345.792	0.970	**0.987**	**10.737**

Dispersion score does not needs to update model parameters or utilize the training data, our method enables to predict OOD error with large-scale models trained on billions of images.

To further analyze the advantage of our method, we present in Figure 6.3 the scatter plots for Fréchet, ProjNorm and Dispersion Score on CIFAR-10C with ResNet50. From the figure, we find that Dispersion Score estimates OOD errors linearly w.r.t. true OOD errors in all cases. In contrast, we observe that those methods based on distribution difference tends to fail when the classification error is high. This phenomenon clearly demonstrates the reliable and superior performance of the Dispersion score in predicting generalization performance under distribution shifts.

4.4.3 Flexibility in OOD data

In previous analysis, we show that Dispersion score can outperform existing methods on standard benchmarks, where the OOD test datasets contain sufficient instances and have a balanced class distribution. In reality, the unlabeled OOD data are naturally imperfect, which may limit the performance of predicting OOD error. In this part, we verify the effectiveness of our method with long-tailed data and small data, compared to ProjNorm [7].



(A) ResNet18 (B) ResNet-50

FIGURE 4.4: Prediction performance R^2 vs. sample size of OOD data (subsets of CIFAR-10C) with (a) ResNet18 and (b) ResNet50.

TABLE 4.3: Summary of prediction performance on **Imbalanced** CIFAR-10C and CIFAR-100C [8], where R^2 refers to coefficients of determination, ρ refers to Spearman correlation coefficients (higher is better), and T refers to average evaluation time (lower is better). The best results are highlighted in **bold**. More results can be found in Appendix 4.5.2.

Dataset	Network	ProjNorm			Ours		
		R^2	ρ	T	R^2	ρ	T
CIFAR 10	ResNet18	0.799	0.968	204.900	**0.959**	**0.982**	**2.076**
	ResNet50	0.897	0.980	430.406	**0.968**	**0.982**	**3.295**
	WRN-50-2	0.922	0.978	561.611	**0.932**	**0.978**	**2.970**
	Average	0.873	0.973	398.972	**0.953**	**0.980**	**2.780**
CIFAR 100	ResNet18	0.886	0.968	210.05	**0.941**	**0.982**	**1.864**
	ResNet50	**0.980**	**0.988**	433.860	0.956	0.982	**2.974**
	WRN-50-2	0.978	0.982	768.883	**0.986**	**0.994**	**3.242**
	Average	0.948	0.980	470.931	**0.961**	**0.986**	**2.693**

Class imbalance. We first evaluate the prediction performance under class imbalanced setting. In particular, given a trained model, we aim to estimate its balanced accuracy under distribution shift while we only have access to a long-tailed test data, where a few classes (majority classes) occupy most of the data and most classes (minority classes) are under-represented. We conduct experiments on CIFAR-10C and CIFAR-100C with imbalance rate 100.

Our results in Table 4.3 show that Dispersion Score achieves better performance than ProjNorm [7] under class imbalance. For example, our approach achieves an average R^2 of 0.953 on CIFAR-10C with 2.780 seconds, while ProjNorm obtains an R^2 of 0.873 and takes 398.972 seconds. In addition, the performance of our method is more stable across different model architectures and datasets with the R^2 ranging from 0.932 to 0.986 than ProjNorm (0.799 - 0.980). Overall, Dispersion score maintains reliable and superior performance even when the OOD test set are class imbalanced.

Sampling efficiency. During deployment, it can be challenging to collect instances

under a specific distribution shift. However, existing state-of-the-art methods [7] normally require a sufficiently large test dataset to achieve meaningful results in predicting OOD error. Here, we further validate the sampling efficiency of Dispersion score, compared with ProjNorm.

Figure 4.4 presents the performance comparison between Dispersion Score and ProjNorm on subsets of CIFAR10C with various sample sizes. The results show that our method can achieve excellent performance even when only 50 examples are available in the OOD data. In contrast, the performance of ProjNorm decreases sharply with the decrease in sample size. The phenomenon shows that Dispersion score is more efficient in exploiting the information of OOD instances.

4.4.4 Intra-class compactness vs. inter-class dispersion

In Section 4.3, we empirically show that feature separability is naturally tied with the final accuracy and propose an effective score based on inter-class dispersion. However, the connection between intra-class compactness and generalization performance is still a mystery. In this analysis, we show that compactness is not a good indicator of OOD accuracy. Specifically, we define the compactness score:

$$S(\mathcal{D}_{\text{test}}) = -\log \frac{\sum_{j=1}^{k} \sum_{i=1}^{m_j} ||\mathbf{z}_i - \tilde{\mu}_j||^2}{n-k}$$

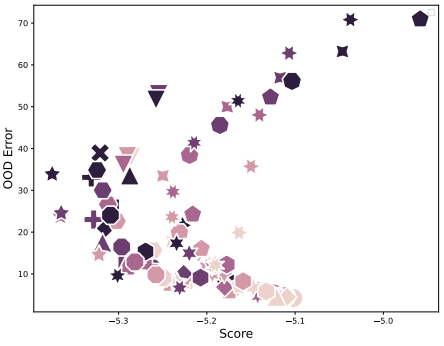


FIGURE 4.5: Compactness vs. test error on CIFAR-10C with ResNet50.

where $\tilde{\mu}_j$ denotes the centroid of the cluster j that the instance $\mathbf{z}_i$ belongs to. Therefore, the compactness score can measure the clusterability of the learned features, i.e., the average distance between each instance and its cluster centroid. Intuitively, a high compactness score may correspond to a well-separated feature distribution, which leads to high test accuracy. Surprisingly, in Figure 4.5, we find that the compactness score is largely irrelevant to the final OOD error, showing that it is not an effective indicator for predicting generalization performance under distribution shifts.

4.4.5 The importance of the weight

As introduced in Section 4.3, Dispersion Score can be viewed as a weighted arithmetic mean of the distance from centers of each class to the center of the whole samples in the feature space, where the weight is the total number of samples in the corresponding class. To verify the importance of the weight in long-tailed setting, we consider a variant that removes the weight:

$$S(\mathcal{D}_{\text{test}}) = \frac{1}{k-1}\sum_{j=0}^{k}\varphi(\bar{\mu}, \tilde{\mu}_j).$$

The results are shown in Table 4.4, where we compare performances of Dispersion score and the variant without weight respectively under **imbalanced** CIFAR-10C and CIFAR-100C with ResNet10 and ResNet50. From this table, we could observe that the weight enhance the robustness significantly in long-tail conditions.

TABLE 4.4: Summary of prediction performance on **Imbalanced** CIFAR-10C and CIFAR-100C. The best results are highlighted in **bold**.

Dataset	Network	w/o Weights		w/ Weights	
		R^2	ρ	R^2	ρ
CIFAR 10	ResNet18	0.675	0.930	**0.959**	**0.982**
	ResNet50	0.748	0.948	**0.968**	**0.982**
	Average	0.712	0.939	**0.978**	**0.990**
CIFAR 100	ResNet18	0.595	0.838	**0.941**	**0.982**
	ResNet50	0.395	0.733	**0.956**	**0.982**
	Average	0.494	0.785	**0.952**	**0.986**

4.4.6 K-means vs. pseudo labels.

While our Dispersion score derived from pseudo labels has demonstrated strong promise, a question arises: *can a similar effect be achieved by alternative clustering methods?* In this ablation, we show that labels obtained by K-means [129, 130] does not achieve comparable performance with pseudo labels obtained from the trained classifier. In particular, we allocate instances into clusters by using K-means instead of pseudo labels from the classifier.

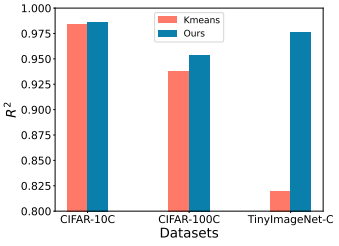


FIGURE 4.6: Compare with K-means.

We present the performance comparison of our method and the variant of K-means in Figure 4.6. The results show that our Dispersion score performs better than the K-means variant and the gap is enlarged with more classes. On the other hand, the variant of K-means does not require a classifier, i.e., the linear layer in the trained model, which enables to evaluate the OOD performance of representation learning methods, e.g., self-supervised learning.

4.5 Discussion

4.5.1 Sensitivity analysis: pseudo labels

Here, we conduct a sensitivity analysis by using ground-truth labels in our method. Table 4.5 illustrates that the performance with pseudo labels is comparable with the performance using ground-truth labels. This phenomenon is consistent with the previous method - ProjNorm [7], shown in Table 8 of their paper. The reason behind this could be that the trained model is capable of identifying certain semantic information from most corrupted examples, thereby retaining their representations in a cluster. Thus, the separability of feature clusters can serve as an indicator of the final prediction performance for corruption perturbations. Additionally, we provide a failure case of feature clusters separability for adversarial perturbations in Appendix 4.5.4.

4.5.2 More results on class-imbalance settings

This section provides elaborated outcomes of training-free benchmarks under the setting of class imbalance, serving as a complement to the results presented in Table 4.3.

4.5.3 Partial OOD error prediction

In previous experiments, a common assumption is that the test set contains instances from all classes. To further explore the flexibility of our method on OOD test set,

TABLE 4.5: Comparison of Dispersion Score with pseudo labels and true labels on CIFAR10, CIFAR100 and TinyImageNet. The best results are highlighted in **bold**.

Dataset	Network	Pseudo labels		True labels	
		R^2	ρ	R^2	ρ
CIFAR 10	ResNet18	0.968	**0.990**	**0.979**	0.989
	ResNet50	**0.987**	0.990	0.985	**0.991**
	WRN-50-2	**0.961**	**0.988**	0.945	0.987
	Average	**0.972**	**0.990**	0.970	0.989
CIFAR 100	ResNet18	**0.952**	0.988	0.915	**0.989**
	ResNet50	0.953	0.985	**0.959**	**0.989**
	WRN-50-2	**0.980**	0.991	0.978	**0.995**
	Average	**0.962**	0.988	0.950	**0.991**
TinyImageNet	ResNet18	**0.966**	**0.986**	0.937	0.985
	ResNet50	**0.977**	0.990	0.954	**0.995**
	WRN-50-2	0.968	0.986	**0.977**	**0.994**
	Average	**0.970**	0.987	0.956	**0.991**

TABLE 4.6: Summary of prediction performance on **Imbalanced** CIFAR-10C and CIFAR-100C for training-free benchmarks, where R^2 refers to coefficients of determination, and ρ refers to Spearman correlation coefficients (higher is better)

Dataset	Network	Rotation		ConfScore		Entropy		AgreeScore		ATC		Fréchet	
		R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ
CIFAR 10	ResNet18	0.767	0.922	0.823	0.965	0.841	0.969	0.669	0.922	0.830	0.966	0.966	0.983
	ResNet50	0.787	0.946	0.870	0.975	0.887	0.977	0.765	0.953	0.883	0.975	0.916	0.975
	WRN-50-2	0.829	0.968	0.915	0.986	0.913	0.986	0.823	0.972	0.922	0.986	0.866	0.977
	Average	0.794	0.945	0.869	0.976	0.880	0.977	0.752	0.949	0.878	0.976	0.916	0.979
CIFAR 100	ResNet18	0.769	0.944	0.872	0.988	0.840	0.985	0.858	0.979	0.905	0.988	0.905	0.972
	ResNet50	0.847	0.964	0.875	0.986	0.826	0.978	0.832	0.973	0.880	0.986	0.855	0.979
	WRN-50-2	0.930	0.981	0.976	0.993	0.980	0.993	0.944	0.981	0.981	0.994	0.889	0.988
	Average	0.849	0.963	0.908	0.989	0.882	0.985	0.878	0.978	0.922	0.989	0.883	0.980

we introduce a new setting called *partial OOD error prediction*, where the label space for the test set is a subset of the label space for the training data.

Here, we train ResNet18 and ResNet50 on both CIFAR-10 and CIFAR-100 with 10 and 100 categories, respectively. Different from previous settings, we evaluate the prediction performance on CIFAR-10C and CIFAR-100C with the first 50% of categories. The numerical results are shown in Tabel 4.7.

From the results, we could observe that our method is more robust than existing methods in the setting of partial OOD error prediction. For example, ProjNorm suffers from the incomplete test dataset during the self-training process, with

TABLE 4.7: Summary of prediction performance on **partial sets** of CIFAR-10C and CIFAR-100C, where R^2 refers to coefficients of determination, and ρ refers to Spearman correlation coefficients (higher is better). The best results are highlighted in **bold**.

Dataset	Network	Rotation		ConfScore		Entropy		AgreeScore		ATC		Frechet		ProjNorm		Ours	
		R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ
CIFAR 10	ResNet18	0.578	0.896	0.795	0.982	0.826	0.984	0.615	0.931	0.802	0.981	0.842	0.941	0.770	0.968	**0.935**	**0.985**
	ResNet50	0.719	0.939	0.885	**0.993**	0.892	**0.993**	0.787	0.976	0.887	**0.993**	0.757	0.953	0.856	0.967	**0.950**	0.992
	Average	0.649	0.918	0.841	0.987	0.859	**0.989**	0.701	0.954	0.845	0.987	0.800	0.947	0.813	0.968	**0.942**	0.988
CIFAR 100	ResNet18	0.876	0.946	0.922	0.985	0.902	0.980	0.904	0.971	**0.943**	**0.986**	0.894	0.972	0.770	0.968	0.935	0.985
	ResNet50	0.923	0.967	0.917	0.980	0.890	0.975	0.915	0.976	0.932	0.983	0.837	0.978	0.856	0.967	**0.950**	**0.992**
	Average	0.899	0.956	0.920	0.983	0.896	0.978	0.909	0.973	0.938	0.984	0.866	0.975	0.813	0.968	**0.942**	**0.989**

a dramatic drop from around 0.950 to around 0.810 for R^2 of CIFAR-10C and CIFAR-100C on average. Contrastively, our method still achieves high accuracy in predicting OOD errors, maintaining an average R^2 value of 0.950.

4.5.4 Adversarial vs. corruption robustness.

In the previous analysis, we show the superior performance of dispersion score on predicting the accuracy of OOD test sets with different corruptions. Here, we surprisingly find that feature dispersion can effectively demonstrate the difference between adversarial and corruption robustness.

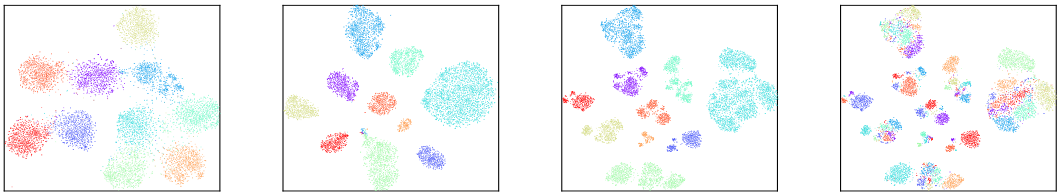
TABLE 4.8: Prediction performance measured by MSE against adversarial attack of different methods. The linear regression model is estimated on CIFAR-10C, and is used to predict the adversarial examples with perturbation size ϵ varying from 0.25 to 8.0. "True Dispersion" refers to the dispersion score with feature normalization using ground-truth labels. The best results are highlighted in **bold**.

	ConfScore	Entropy	ATC	ProjNorm	Dispersion	True Dispersion
CIFAR-10	0.933	0.892	0.906	0.847	1.359	**0.483**

Table 4.8 shows the prediction performance of different methods under adversarial attacks. "True Dispersion" refers to the dispersion score with feature normalization using ground-truth labels. In particular, we generate adversarial samples attacked by projected gradient descent (PGD) using untargeted attack [131] on the test set of CIFAR-10 with 10 steps and perturbation size ϵ ranging from 0.25 to 8.0. While the vanilla Dispersion score leads to poor performance, we note that the variant of Dispersion score with ground-truth labels performs much better than previous

methods. This phenomenon is different from the conclusion of the sensitivity analysis of pseudo labels in predicting corruption robustness (See Appendix 4.5.1), where the variant of true labels cannot outperform our method.

To understand the reasons behind the performance disparity of feature dispersion between adversarial and corruption robustness, we present the t-SNE visualization of features for adversarial attack of CIFAR-10 test set with various perturbation sizes in Figure 4.7. Compared to Figure 6.3, the results indicate that adversarial perturbations increase the distance between different clusters, whereas corruption perturbations decrease the separability of the clusters. In other words, adversarial perturbations decrease the test accuracy in a different way: assigning instances to the wrong groups and enlarging the distance among those groups. Therefore, feature dispersion using pseudo labels cannot be an effective method in the adversarial setting. We hope this insight can inspire specific designed methods based on feature dispersion for predicting adversarial errors in the future.



(A) ϵ=0.0 ‖ pseudo (B) ϵ=2.0 ‖ pseudo (C) ϵ=8.0 ‖ pseudo (D) ϵ=8.0 ‖ true

FIGURE 4.7: t-SNE visualization of feature representation on adversarial attack of CIFAR-10 test set with perturbation size ϵ ranging from 0.25 to 8.0.

4.6 Chapter Summary

In this chapter, we introduce Dispersion score, a simple yet effective indicator for predicting the generalization performance on OOD data without labels. We show that Dispersion score is strongly correlated to the OOD error and achieves consistently better performance than previous methods under various distribution shifts. Even when the OOD datasets are class imbalanced or have limit number of instances, our method maintains a high prediction performance, which demonstrates the strong flexibility of dispersion score. This method can be easily adopted in practical settings. It is straightforward to implement with trained models with various architectures, and does not require access to the training data. Thus, our

method is compatible with large-scale models that are trained on billions of images. We hope that our insights inspire future research to further explore the feature separability for predicting OOD error.

Chapter 5

Leveraging Gradients for Unsupervised Accuracy Estimation under Distribution Shift

5.1 Introduction

Deploying machine learning models in the real world is often subject to a distribution shift between training and test data. Such a shift may significantly degrade the model's performance at test time [4, 68, 132] and lead to high risks related to AI satefy [28]. To alleviate this problem, a common practice is to monitor the model performance regularly by collecting the ground truth of a subset of the current test dataset [76]. However, this is usually time-consuming and expensive, highlighting the need for unsupervised methods to assess the test performance of models under distribution shift, commonly known as *unsupervised accuracy estimation*.

Limitations of current approaches. Current studies mainly focus on outputs or feature representation to derive a test error estimation score. Such a score can represent the calibrated test error or the distribution discrepancy between training and test datasets [7, 24, 25, 27, 28, 76]. For instance, Hendrycks and Gimpel [24] considered the average maximum softmax score of the test samples as the estimated

error. Similarly, Garg et al. [27] proposed to learn a confidence threshold from the training distribution. Deng and Zheng [28] quantified the distribution difference between training and test datasets in the feature space, while Yu et al. [7] gauges the distribution gap at the parameter level. Although insightful, the current body of literature overlooks a potential tool for test accuracy estimation, namely the gradients that are known to correlate strongly with the generalization error of the deep neural networks [133, 134].

Why do gradients matter? Recently, increasing attention has been paid to the intrinsic properties of gradients to design learning algorithms for meta-learning [135], domain generalization [136, 137] or to improve optimization of DNNs [138, 139] by relying on them. In domain shift scenarios, Mansilla et al. [137] proposed to clip the conflicting gradients and introduced a strategy to promote gradient agreement across multiple domains. Meanwhile, Zhao et al. [139] designed a gradient-based regularization term to make the optimizer find flat minima. In the field of OOD detection, Huang et al. [1] introduced a gradient-based function to detect out-of-distribution (OOD) samples (relation to our method is compared in Appendix B.6). Although the works mentioned above showed the potential of gradients to tackle the learning problems both in OOD and in-distribution (ID) settings, it is still unclear how unsupervised gradient-based scores can correlate with the test accuracy and be used to estimate it. This motivates us to ask:

Are gradients predictive of test accuracy under distribution shift?

In this chapter, we shed new light on this open question: we surprisingly observe that there exists a strong linear relationship between gradient norm and test accuracy under distribution shift. Moreover, our theoretical analysis shows that the gradient norm conveys information on the generalization capacity of a well-calibrated model. In a nutshell, this work provides direct evidence that gradient-based information correlates with generalization performance, which paves the way to a better understanding of how neural networks generalize across unseen domains.

Our contributions. We hypothesize that the model requires a gradient step of large magnitude when it fails to generalize well on test data from unseen domains.

To quantify the magnitude of gradients, we propose a simple yet efficient gradient-based statistic, GDSCORE, which employs the norm of the gradients backpropagated from a standard cross-entropy loss on the test samples. To avoid the need for ground-truth labels, we propose a pseudo-labeling strategy that benefits from both correct and incorrect predictions. We demonstrate that the norm of this one-step gradient of the classification layer strongly correlates with the generalization performance under diverse distribution shifts, acting as a strong and lightweight proxy for the latter. The main contributions of this chapter are summarized as follows:

1. We first provide several theoretical insights showing that correct pseudo-labeling and gradient norm have a direct impact on the test error estimation. This is achieved by looking at the analytical expression of the gradient after one backpropagation step over the pre-trained model on test data under distribution shift and by upper-bounding the target out-of-distribution risk.

2. Based on these theoretical insights, we propose the GDSCORE, which gauges the magnitude of the classification-layer gradients and presents a strong correlation with test accuracy. Our method does not require access to either test labels or training datasets and only needs one step of backpropagation which makes it particularly lightweight in terms of computational efficiency compared to other self-training methods.

3. We demonstrate the superiority of GDSCORE with a large-scale empirical evaluation. We achieve new state-of-the-art results on 11 benchmarks across diverse distribution shifts compared to 8 competitors, while being faster than the previous best baseline.

Organization of the chapter. The rest of the chapter is organized as follows. Section 5.2 presents the necessary background on the problem at hand. In Section 5.3, we derive the theoretical insights that motivate the GDSCORE introduced afterward. Section 5.4 is devoted to extensive empirical evaluation of our method, while the ablation study is deferred to Section 5.5. Finally, Section 5.7 concludes our work.

5.2 Background

Problem setup. We consider a K-class classification task with the input space $\mathcal{X} \subset \mathbb{R}^D$ and the label set $\mathcal{Y} = \{1, \dots, K\}$. Our learning model is a neural network with trainable parameters $\boldsymbol{\theta} \in \mathbb{R}^p$ that maps from the input space to the label space $f_{\boldsymbol{\theta}} : \mathcal{X} \to \mathbb{R}^K$. We view the network as a combination of a complex feature extractor $f_{\mathbf{g}}$ and a linear classification layer $f_{\boldsymbol{\omega}}$, where $\mathbf{g}$ and $\boldsymbol{\omega} = (\mathbf{w}_k)_{k=1}^K$ denote their corresponding parameters. Given a training example $\mathbf{x}_i$, the feedforward process can be expressed as:

$$f_{\boldsymbol{\theta}}(\mathbf{x}_i) = f_{\boldsymbol{\omega}}(f_{\mathbf{g}}(\mathbf{x}_i)). \tag{5.1}$$

Let $\mathbf{y} = (y^{(k)})_{k=1}^K$ denote the one-hot encoded vector of label y, i.e., $y^{(k)} = 1$ if and only if $y = k$, otherwise $y^{(k)} = 0$. Then, given a training dataset $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^n$ that consists of n data points sampled *i.i.d.* from the source distribution $P_S(\mathbf{x}, y)$ defined over $\mathcal{X} \times \mathcal{Y}$, $f_{\boldsymbol{\theta}}$ is trained following the empirical cross-entropy loss minimization:

$$\mathcal{L}_{\mathcal{D}}(f_{\boldsymbol{\theta}}) = -\frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{K} y_i^{(k)} \log \mathrm{s}_{\boldsymbol{\omega}}^{(k)}(f_{\mathbf{g}}(\mathbf{x}_i)), \tag{5.2}$$

where $\mathrm{s}_{\boldsymbol{\omega}}^{(k)}$ denotes the output of the softmax for the class k approximating the posterior probability $P(Y = k|\mathbf{x})$, i.e., $\mathrm{s}_{\boldsymbol{\omega}}^{(k)}(f_{\mathbf{g}}(\mathbf{x})) = \exp\{\mathbf{w}_k^\intercal f_{\mathbf{g}}(\mathbf{x})\} / \left(\sum_{\tilde{k}} \exp\{\mathbf{w}_{\tilde{k}}^\intercal f_{\mathbf{g}}(\mathbf{x})\}\right)$.

Unsupervised accuracy estimation. We now assume to have access to m test samples from the target distribution $\mathcal{D}_{\text{test}} = \{\tilde{\mathbf{x}}_i\}_{i=1}^m \sim P_T(\mathbf{x})$, where $P_T(\mathbf{x}, y) \neq P_S(\mathbf{x}, y)$. For each test sample $\tilde{\mathbf{x}}_i$, we predict the label by $\tilde{y}_i' = \arg\max_{k \in \mathcal{Y}} f_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_i)$. We now want to assess the performance of $f_{\boldsymbol{\theta}}$ on a target distribution without using corresponding ground-truth labels $\{\tilde{y}_i\}_{i=1}^m$ by estimating as accurately as possible the following quantity:

$$\mathrm{Acc}(\mathcal{D}_{\text{test}}) = \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}(\tilde{y}_i' = \tilde{y}_i), \tag{5.3}$$

where $\mathbb{1}(\cdot)$ denotes the indicator function. In practice, unsupervised accuracy estimation methods provide a proxy score $S(\mathcal{D}_{\text{test}})$ that should exhibit a linear correlation with $\mathrm{Acc}(\mathcal{D}_{\text{test}})$. The performance of such methods is measured using the coefficient of determination R_2 and the Spearman correlation coefficient ρ.

5.3 On the strong correlation between gradient norm and test accuracy

We start by deriving an analytical expression of the gradient obtained when fine-tuning a source pre-trained model on new test data. We further use the intuition derived from it to propose our test accuracy estimation score and justify its effectiveness through a more thorough theoretical analysis.

A motivational example. Below, we follow the setup considered by Denevi et al. [140], Balcan et al. [141], Arnold et al. [142] to develop our intuition behind the importance of gradient norm in unsupervised accuracy estimation. Our main departure point for this analysis is to consider fine-tuning: a popular approach to adapting a pre-trained model to different labeled datasets is to update either all or just a fraction of its parameters using gradient descent on the new data. To this end, let us consider the following linear regression example, where the test data from unseen domains are distributed as $X \sim \mathcal{N}(0, \sigma_t^x)$, $(Y|X=x) \sim \mathcal{N}(\theta_t x, 1)$ parameterized by the optimal regressor $\theta_t \in \mathbb{R}$, while the data on which the model was trained is distributed as $X \sim \mathcal{N}(0, \sigma_s^x)$, $(Y|X=x) \sim \mathcal{N}(\theta_s x, 1)$ with $\theta_s \in \mathbb{R}$. Consider the least-square loss over the test distribution:

$$\mathcal{L}_T(c) = \frac{1}{2}\mathbb{E}_{P_T(x,y)}(y - cx)^2.$$

When we do not observe the target labels, one possible solution would be to analyze fine-tuning when using the source generator $(Y|X=x) \sim \mathcal{N}(\theta_s x, 1)$ for pseudo-labeling. Then, we obtain that

$$\begin{aligned}
\frac{1}{2}\nabla_c \mathbb{E}_{P_T(x)}\mathbb{E}_{P_S(y|x)}[(y-cx)^2] &= \mathbb{E}_{P_T(x)}\mathbb{E}_{P_S(y|x)}[(y-cx)(-x)] \\
&= \mathbb{E}_{P_T(x)}\mathbb{E}_{P_S(y|x)}[cx^2 - xy] \\
&= (c-\theta_s)\sigma_t^x \\
&= ((c-\theta_t) + (\theta_t - \theta_s))\sigma_t^x.
\end{aligned}$$

This derivation, albeit simplistic, suggests that the gradient over the target data correlates – modulo the variance of x – with $(c - \theta_t)$, capturing how far we are from the optimal parameters of the target model, and $(\theta_s - \theta_t)$, that can be seen

as a measure of dissimilarity between the distributions of the optimal source and target parameters. Intuitively, both these terms are important for predicting the test accuracy performance suggesting that the gradient itself can be a good proxy for the latter.

5.3.1 Proposed approach: GdScore

We now formally introduce our proposed score, termed GdScore, to estimate test accuracy in an unsupervised manner during evaluation. We start by recalling the backpropagation process of the pre-trained neural network f_θ from a cross-entropy loss and then describe how to leverage the gradient norm for the unsupervised accuracy estimation. The detailed algorithm can be found in Appendix B.2.

Feedforward. Similar to the feedforward in the pre-training process shown in Eq. 5.1, for any given test individual $\tilde{\mathbf{x}}_i$, we have:

$$f_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_i) = f_{\boldsymbol{\omega}}(f_{\mathbf{g}}(\tilde{\mathbf{x}}_i)). \tag{5.4}$$

As explained above, we do not observe the true labels of test data. We now detail our strategy for pseudo-labeling that allows us to obtain accurate and balanced proxies for test data labels based on accurate and potentially inaccurate model predictions.

Label generation strategy. Unconditionally generating pseudo-labels for test data under distribution shift exhibits an obvious drawback: we treat all the assigned pseudo-labels as correct predictions when calculating the loss, ignoring the fact that some examples are possibly mislabeled. Therefore, we propose the following confidence-based label-generation policy that outputs for every $\tilde{\mathbf{x}}_i \in \mathcal{D}_{\text{test}}$:

$$\tilde{y}'_i = \begin{cases} \arg\max_k f_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_i), & \max_k \mathrm{s}^{(k)}_{\boldsymbol{\omega}}(f_{\mathbf{g}}(\tilde{\mathbf{x}}_i)) > \tau \\ \tilde{y}' \sim U[1, K], & \text{otherwise} \end{cases} \tag{5.5}$$

where τ denotes the threshold value, and $U[1, k]$ denotes the discrete uniform distribution with outcomes $\{1, \ldots, K\}$. In a nutshell, we assign the predicted label to $\tilde{\mathbf{x}}_i$, when the prediction confidence is larger than a threshold while using a

randomly sampled label from the label space otherwise. The detailed empirical evidence justifying this choice is shown in Section 5.5, and we discuss the choice of proper threshold τ in Appendix 5.5.1. From the theoretical point of view, our approach assumes that the classifier makes mistakes mostly on data with low prediction confidence, for which we deliberately assign noisy pseudo-labels. Feofanov et al. [143] used a similar approach to derive an upper bound on the test error and proved its tightness in the case where the assumption is satisfied. We discuss this matter in more detail in Appendix 5.6.

Backpropagation. To estimate our score, we calculate the gradients w.r.t. the weights of the classification layer $\boldsymbol{\omega}$ during the first epoch backpropagated over the standard cross-entropy loss defined by:

$$\mathcal{L}_{\mathcal{D}_{\text{test}}}(f_{\boldsymbol{\theta}}) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K} \tilde{y}_i^{\prime(k)} \log \mathrm{s}_{\boldsymbol{\omega}}^{(k)}(f_{\mathbf{g}}(\tilde{\mathbf{x}}_i)), \tag{5.6}$$

where each unlabeled instance $\tilde{\mathbf{x}}_i$ is pseudo-labeled following Eq. 5.5. Then, the gradient of the classification layer $\boldsymbol{\omega}$ is evaluated as follows:

$$\nabla_{\boldsymbol{\omega}}\mathcal{L}_{\mathcal{D}_{\text{test}}}(f_{\boldsymbol{\theta}}) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K} \nabla_{\boldsymbol{\omega}}\left(\tilde{y}_i^{\prime(k)} \log \mathrm{s}_{\boldsymbol{\omega}}^{(k)}(f_{\mathbf{g}}(\tilde{\mathbf{x}}_i))\right). \tag{5.7}$$

Note that our method requires neither gradients of the whole parameter set of the pre-trained model nor iterative training. This makes it highly computationally efficient.

GdScore. Now, we can define GdScore using a vector norm of gradients of the last layer. The score is expressed as follows:

$$S(\mathcal{D}_{\text{test}}) = \|\nabla_{\boldsymbol{\omega}}\mathcal{L}_{\mathcal{D}_{\text{test}}}(f_{\boldsymbol{\theta}})\|_p, \tag{5.8}$$

where $||\cdot||_p$ denotes L_p-norm.

5.3.2 Theoretical analysis

In this section, we provide theoretical insights into our method. We first clarify the connection between the true target cross-entropy error and the norm of the gradients. Then, we show that the gradient norm is upper-bounded by a weighted sum of the norm of the inputs.

Notations. For the sake of simplicity, we assume the feature extractor $f_{\mathbf{g}}$ is fixed and, by abuse of notation, we use $\mathbf{x}$ instead of $f_{\mathbf{g}}(\mathbf{x})$. Reusing the notations introduced in Section 5.2, the true target cross-entropy error writes

$$\mathcal{L}_T(\boldsymbol{\omega}) = -\mathbb{E}_{P_T(\mathbf{x},y)} \sum_k y^{(k)} \log \mathrm{s}_{\boldsymbol{\omega}}^{(k)}(\mathbf{x}),$$

where $\boldsymbol{\omega} = (\mathbf{w}_k)_{k=1}^K \in \mathbb{R}^{D\times K}$ are the parameters of the linear classification layer $f_{\boldsymbol{\omega}}$. For the ease of notation, the gradient of $\mathcal{L}_T$ w.r.t $\boldsymbol{\omega}$ is denoted by $\nabla\mathcal{L}_T$.

The following theorem, whose proof we defer to Appendix B.7.1, makes the connection between the true risk and the L_p-norm of the gradient explicit.

Theorem 5.3.1 (Connection between the true risk and the L_p-norm of the gradient). Let $\mathbf{c} \in \mathbb{R}^{D\times K}$ and $\mathbf{c}' \in \mathbb{R}^{D\times K}$ be two linear classifiers. For any $p, q \geq 1$ such that $\frac{1}{p} + \frac{1}{q} = 1$, we have that

$$|\mathcal{L}_T(\mathbf{c}') - \mathcal{L}_T(\mathbf{c})| \leq \max_{\boldsymbol{\omega}\in\{\mathbf{c}',\mathbf{c}\}} (\|\nabla\mathcal{L}_T(\boldsymbol{\omega})\|_p) \cdot \|\mathbf{c}' - \mathbf{c}\|_q .$$

The left-hand side here is the difference in terms of the true risks obtained for the same distribution with two different classifiers. The right-hand side shows how this difference is controlled by the maximum gradient norm over the two classifiers and a term capturing how far the two are apart.

In the context of the proposed approach, we want to know the true risk of the source classifier $\boldsymbol{\omega}_s$ on the test data and its change after one step of gradient descent. The following corollary applies Theorem 5.3.1 to characterize this exact case. The proof is deferred to Appendix B.7.2.

Corollary 5.3.2 (Connection after one gradient update). Let $\mathbf{c}$ be the classifier obtained from $\boldsymbol{\omega}_s$ after one gradient descent step, i.e., $\mathbf{c} = \boldsymbol{\omega}_s - \eta \cdot \nabla\mathcal{L}_T(\boldsymbol{\omega}_s)$ with

$\eta \geq 0$. Then, when $\boldsymbol{\omega} \in \{\boldsymbol{\omega}_s, \mathbf{c}\}$, we have that

$$|\mathcal{L}_T(\boldsymbol{\omega}_s) - \mathcal{L}_T(\mathbf{c})| \leq \eta \max_{\boldsymbol{\omega}} (\|\nabla \mathcal{L}_T(\boldsymbol{\omega})\|_p) \, \|\nabla \mathcal{L}_T(\boldsymbol{\omega}_s)\|_q \, .$$

Note that in this case $\mathcal{L}_T(\boldsymbol{\omega}_s)$ can be seen as a term providing the true test risk after pseudo-labeling it with the source classifier. This shows the importance of pseudo-labeling as it acts as a departure point for obtaining a meaningful estimate of the right-hand side. When the latter is meaningful, the gradient norm on the right-hand side controls it together with a magnitude that tells us how far we went after one step of backpropagation.

In the next theorem, we provide an upper bound on the L_p-norm of the gradient as a weighted sum of the L_p-norm of the inputs. The proof is deferred to Appendix B.7.3.

Theorem 5.3.3 (Upper-bounding the norm of the gradient). For any $p \geq 1$ and $\forall \, \boldsymbol{\omega} \in \mathbb{R}^{D \times K}$, the L_p-norm of the gradient can be upper-bounded as follows:

$$\|\nabla \mathcal{L}_T(\boldsymbol{\omega})\|_p \leq \mathbb{E}_{P_T(\mathbf{x},y)} \alpha(\boldsymbol{\omega}, \mathbf{x}, y) \cdot \|\mathbf{x}\|_p \, ,$$

where $\alpha(\boldsymbol{\omega}, \mathbf{x}, y) = 1 - \mathrm{s}_{\boldsymbol{\omega}}^{(k_y)}(\mathbf{x})$, with k_y such that $y^{(k_y)} = 1$.

Hence, the norm of the gradient is upper-bounded by a weighted combination of the norm of the inputs, where the weight $\alpha(\boldsymbol{\omega}, \mathbf{x}, y) \in [0, 1]$ conveys how well the model predicts on $\mathbf{x}$. In the case of perfect classification, the upper bound is tight and equals 0. In practice, as we do not have access to the true risk, the gradients can be approximated by the proposed GDSCORE that requires to pseudo-label test data by Eq. 5.5. As we said earlier, this implies that the model has to be well calibrated (see Appendix 5.6), which is conventional to assume for self-training methods [144] including the approach of Yu et al. [7]. Then, the network projects test data into the low confidence regions, and the gradient for these examples will be large as we need to update $\boldsymbol{\omega}_s$ significantly to fit them.

5.4 Experiments

5.4.1 Experimental setup

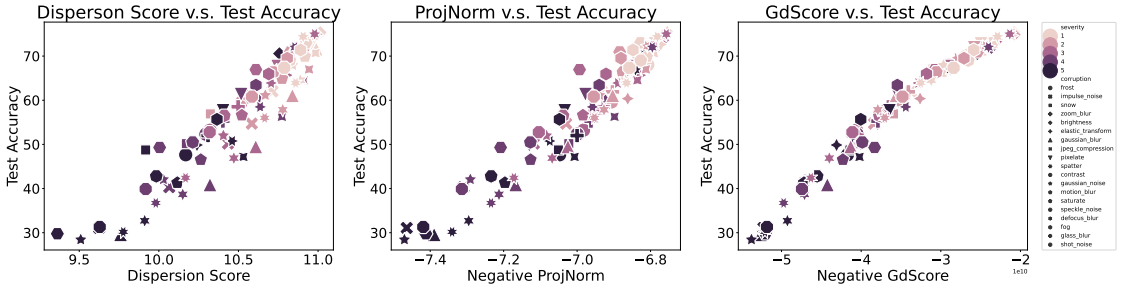


FIGURE 5.1: Test accuracy prediction versus True test accuracy on Entity-13 with ResNet18. We compare the performance of GDSCORE with that of Dispersion Score and ProjNorm via scatter plots. Each point represents one dataset under certain corruption and certain severity, where different shapes represent different types of corruption, and darker color represents the higher severity level.

Pre-training datasets. For pre-training the neural network, we use CIFAR-10, CIFAR-100 [125], TinyImageNet [126], ImageNet [145], Office-31 [108], Office-Home [109], Camelyon17-WILDS [4], and BREEDS [146] which leverages class hierarchy of ImageNet [145] to create 4 datasets including Living-17, Nonliving-26, Entity-13 and Entity-30. In particular, to avoid time-consuming training, we directly utilize publicly available models pre-trained on Imagenet. For Office-31 and Office-Home, we train a neural network on every domain.

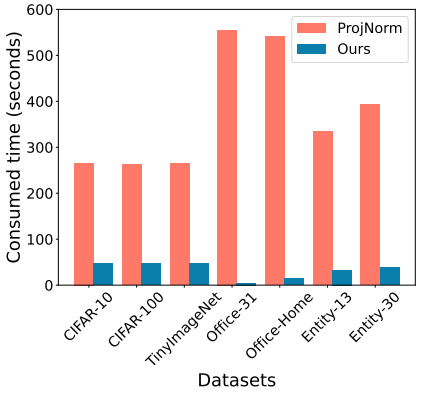


FIGURE 5.2: Runtime comparison of two self-training approaches with ResNet50.

Test datasets. In our comprehensive evaluation, we consider 11 datasets with 3 types of distribution shifts: synthetic, natural, and novel subpopulation shift. To verify the effectiveness of our method under the synthetic shift, we use CIFAR-10C, CIFAR-100C, and ImageNet-C [8] that span 19 types of corruption across 5 severity levels, as well as TinyImageNet-C [8] with 15 types of corruption and 5 severity levels. For the natural shift, we use the domains excluded from training from Office-31, Office-Home, and Camelyon17-WILDS as the OOD datasets. For the novel subpopulation shift, we consider the BREEDS benchmarks, namely, Living-17, Nonliving-26, Entity-13, and Entity-30, which are constructed from ImageNet-C.

TABLE 5.1: Performance comparison on 11 benchmark datasets with ResNet18, ResNet50, and WRN-50-2, where R^2 refers to coefficients of determination, and ρ refers to the absolute value of Spearman correlation coefficients (higher is better). The best results are highlighted in **bold**.

Dataset	Network	Rotation		ConfScore		Entropy		AgreeScore		ATC		Fréchet		Dispersion		ProjNorm		Ours	
		R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ
CIFAR 10	ResNet18	0.822	0.951	0.869	0.985	0.899	0.987	0.663	0.929	0.884	0.985	0.950	0.971	0.968	0.990	0.936	0.982	**0.971**	**0.994**
	ResNet50	0.835	0.961	0.935	0.993	0.945	**0.994**	0.835	0.985	0.946	**0.994**	0.858	0.964	**0.987**	0.990	0.944	0.989	0.969	0.993
	WRN-50-2	0.862	0.976	0.943	**0.994**	0.942	**0.994**	0.856	0.986	0.947	**0.994**	0.814	0.973	0.962	0.988	0.961	0.989	**0.971**	**0.994**
	Average	0.840	0.963	0.916	0.991	0.930	0.992	0.785	0.967	0.926	0.991	0.874	0.970	**0.972**	0.990	0.947	0.987	**0.970**	**0.994**
CIFAR 100	ResNet18	0.860	0.936	0.916	0.985	0.891	0.979	0.902	0.973	0.938	0.986	0.888	0.968	0.952	0.988	0.979	0.980	**0.987**	**0.996**
	ResNet50	0.908	0.962	0.919	0.984	0.884	0.977	0.922	0.982	0.921	0.984	0.837	0.972	0.951	0.985	0.988	0.991	**0.991**	**0.997**
	WRN-50-2	0.924	0.970	0.971	0.984	0.968	0.981	0.955	0.977	0.978	0.993	0.865	0.987	0.980	0.991	0.990	0.991	**0.995**	**0.998**
	Average	0.898	0.956	0.936	0.987	0.915	0.983	0.927	0.982	0.946	0.988	0.864	0.976	0.962	0.988	0.985	0.987	**0.991**	**0.997**
TinyImageNet	ResNet18	0.786	0.946	0.670	0.869	0.592	0.842	0.561	0.853	0.751	0.945	0.826	0.970	0.966	0.986	0.970	0.981	**0.971**	**0.994**
	ResNet50	0.786	0.947	0.670	0.869	0.651	0.892	0.560	0.853	0.751	0.945	0.826	0.971	0.977	0.986	0.979	0.987	**0.980**	**0.995**
	WRNt-50-2	0.878	0.967	0.757	0.951	0.704	0.935	0.654	0.904	0.635	0.897	0.884	0.984	0.968	0.986	0.965	0.983	**0.975**	**0.996**
	Average	0.805	0.959	0.727	0.920	0.650	0.890	0.599	0.878	0.693	0.921	0.847	0.976	0.970	0.987	0.972	0.984	**0.976**	**0.995**
ImageNet	ResNet18	-	-	0.979	0.991	0.963	0.991	-	-	0.974	0.983	0.802	0.974	0.940	0.971	0.975	0.993	**0.986**	**0.996**
	ResNet50	-	-	0.980	0.994	0.967	0.992	-	-	0.970	0.983	0.855	0.974	0.938	0.968	0.986	0.993	**0.987**	**0.996**
	WRNt-50-2	-	-	0.983	0.991	0.963	0.991	-	-	0.983	0.993	0.909	0.988	0.939	0.976	0.978	0.993	**0.984**	**0.998**
	Average	-	-	0.981	0.993	0.969	0.992	-	-	0.976	0.987	0.855	0.979	0.939	0.972	0.980	0.993	**0.986**	**0.997**
Office-31	ResNet18	**0.753**	**0.942**	0.470	0.828	0.322	0.714	0.003	0.085	0.843	0.942	0.143	0.257	0.618	0.714	0.099	0.428	0.675	0.829
	ResNet50	0.391	0.828	0.485	0.828	0.354	0.828	0.011	0.463	0.532	0.485	0.034	0.257	0.578	0.714	0.240	0.428	**0.604**	**0.829**
	WRN-50-2	0.577	0.6	0.524	0.714	0.424	0.714	0.002	0.257	0.405	0.942	0.034	0.142	**0.671**	0.714	0.147	0.143	0.544	**0.829**
	Average	0.567	0.790	0.493	0.790	0.367	0.276	0.006	0.211	0.593	0.790	0.071	0.047	**0.622**	0.714	0.162	0.333	0.608	**0.829**
Office-Home	ResNet18	0.822	0.930	0.795	0.909	0.761	0.881	0.054	0.146	0.571	0.615	0.605	0.755	0.453	0.664	0.064	0.202	**0.876**	**0.909**
	ResNet50	**0.851**	**0.944**	0.769	0.895	0.742	0.853	0.026	0.216	0.487	0.734	0.607	0.685	0.383	0.727	0.169	0.475	0.829	**0.944**
	WRN-50-2	**0.823**	**0.958**	0.741	0.874	0.696	0.846	0.132	0.405	0.383	0.643	0.589	0.706	0.456	0.713	0.172	0.531	0.809	0.916
	Average	0.832	0.944	0.768	0.892	0.733	0.860	0.071	0.256	0.480	0.664	0.601	0.715	0.431	0.702	0.135	0.403	**0.837**	**0.923**
Camelyon17-WILDS	ResNet18	0.944	**1.000**	0.980	**1.000**	0.980	**1.000**	0.977	**1.000**	0.981	**1.000**	0.988	**1.000**	0.992	**1.000**	0.612	0.500	**0.996**	**1.000**
	ResNet50	0.931	**1.000**	0.994	**1.000**	0.993	**1.000**	0.998	**1.000**	0.993	**1.000**	0.971	**1.000**	0.012	0.500	0.811	**1.000**	**0.999**	**1.000**
	WRN-50-2	0.918	**1.000**	0.944	**1.000**	0.945	**1.000**	0.965	**1.000**	0.942	**1.000**	0.994	**1.000**	0.001	0.500	0.789	0.500	**0.997**	**1.000**
	Average	0.931	**1.000**	0.973	**1.000**	0.980	**1.000**	0.982	**1.000**	0.972	**1.000**	0.984	**1.000**	0.334	0.667	0.737	0.667	**0.998**	**1.000**
Entity-13	ResNet18	0.927	0.961	0.795	0.940	0.794	0.935	0.543	0.919	0.823	0.945	0.950	0.981	0.937	0.968	0.952	0.981	**0.969**	**0.991**
	ResNet50	0.932	0.976	0.728	0.941	0.698	0.928	0.901	0.964	0.783	0.950	0.903	0.959	0.764	0.892	0.944	0.974	**0.960**	**0.995**
	WRN-50-2	0.939	0.983	0.930	0.977	0.919	0.973	0.871	0.935	0.936	0.980	0.906	0.958	0.815	0.905	0.950	0.977	**0.968**	**0.995**
	Average	0.933	0.973	0.817	0.953	0.804	0.945	772	0.939	0.847	0.958	0.920	0.966	0.948	0.977	0.839	0.922	**0.966**	**0.994**
Entity-30	ResNet18	0.964	0.979	0.570	0.836	0.553	0.832	0.542	0.935	0.611	0.845	0.849	0.978	0.929	0.968	0.952	0.987	**0.970**	**0.995**
	ResNet50	**0.961**	0.980	0.878	0.969	0.838	0.956	0.914	0.975	0.924	0.973	0.835	0.956	0.783	0.914	0.937	0.986	0.957	**0.996**
	WRN-50-2	0.940	0.978	0.897	0.974	0.878	0.970	0.826	0.955	0.936	0.984	0.927	0.973	0.927	0.973	**0.959**	0.986	0.949	**0.994**
	Average	0.955	0.978	0.781	0.926	0.756	0.919	0.728	0.956	0.823	0.934	0.871	0.969	0.880	0.952	0.949	0.987	**0.959**	**0.995**
Living-17	ResNet18	0.876	0.973	0.913	0.973	0.898	0.970	0.586	0.736	0.940	0.973	0.768	0.950	0.900	0.958	0.923	0.970	**0.949**	**0.983**
	ResNet50	0.906	0.956	0.880	0.967	0.853	0.961	0.633	0.802	**0.938**	**0.976**	0.771	0.926	0.851	0.929	0.903	0.924	0.931	0.975
	WRN-50-2	0.909	0.957	0.928	0.980	0.921	0.977	0.652	0.793	**0.966**	**0.984**	0.931	0.967	0.931	0.966	0.915	0.970	0.910	0.976
	Average	0.933	0.974	0.907	0.973	0.814	0.969	0.623	0.777	**0.948**	**0.978**	0.817	0.949	0.894	0.951	0.913	0.969	0.930	**0.978**
Nonliving-26	ResNet18	0.906	0.955	0.781	0.925	0.739	0.909	0.543	0.810	0.854	0.939	0.914	0.980	**0.958**	0.981	0.939	0.978	0.953	**0.983**
	ResNet50	0.916	0.970	0.832	0.942	0.776	0.918	0.638	0.837	0.893	0.960	0.848	0.950	0.805	0.907	0.873	0.972	**0.945**	**0.989**
	WRN-50-2	0.917	0.977	0.932	0.971	0.912	0.959	0.676	0.861	**0.945**	0.969	0.885	0.942	0.893	0.939	0.924	0.973	0.937	**0.985**
	Average	0.913	0.967	0.849	0.946	0.809	0.929	0.618	0.836	0.897	0.956	0.882	0.957	0.913	0.974	0.886	0.943	**0.945**	**0.985**

Training details. To show the versatility of our approach across different architectures, we perform all our experiments on ResNet18, ResNet50 [105] and WRN-50-2 [127] models. We train them for 20 epochs for CIFAR-10 [125] and 50 epochs for the other datasets. In all cases, we use SGD with a learning rate of 10^{-3}, cosine learning rate decay [128], a momentum of 0.9, and a batch size of 128. For all experiments, we use $p = 0.3$ to compute GDSCORE.

Evaluation metrics. We measure the performance of all competing methods using the coefficients of determination (R^2) and Spearman correlation coefficients (ρ) calculated between the baseline scores and the true test error. To compare the

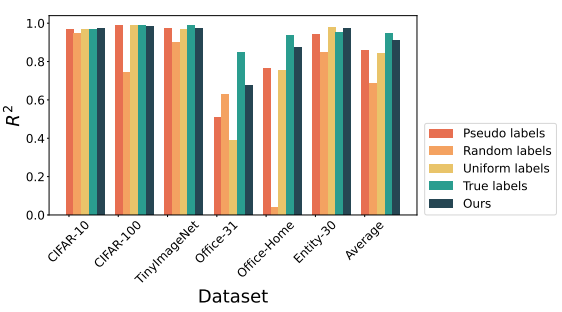
computational efficiency with two self-training methods, we calculate the average evaluation time needed for every test dataset.

Baselines. We compare our method GdScore with 8 baselines commonly considered in the unsupervised accuracy estimation literature: *Rotation Prediction* (Rotation) [23], *Averaged Confidence* (ConfScore) [24], *Entropy* [25], *Agreement Score* (AgreeScore) [26], *Averaged Threshold Confidence* (ATC) [27], *AutoEval* (Fréchet) [28], *Dispersion Score* (Dispersion) [29], and *ProjNorm* [7]. The first six methods are training-free and the last method is an instance of self-training approaches. More details about the baselines can be found in Appendix B.3.

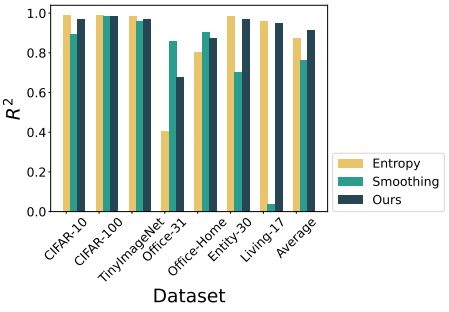
5.4.2 Main takeaways

GdScore correlates with test accuracy stronger than baselines across diverse distribution shifts. In Table 6.1, we present the OOD error estimation performance on 11 benchmark datasets across 3 model architectures as measured by R^2 and ρ. We observe that GdScore outperforms existing methods under diverse distribution shifts. Our method achieves an average R^2 higher than 0.99 on CIFAR-100, while the average R^2 of the other baselines is always below. In addition, our method performs stably across different distribution shifts compared with the other existing algorithms. For example, Rotation performs well under the natural shift but experiences a dramatic performance drop under the synthetic shift, ranking from the second best to the eighth. However, our method achieves consistently high performance, ranking the best on average across the three types of distribution shifts.

Furthermore, we provide the visualization of estimation performance in Fig. 6.3, where we present the scatter plots for Dispersion Score, ProjNorm and GdScore on Entity-13 with ResNet18. We can see that GdScore and test accuracy have a strong linear relationship, while the other state-of-the-art methods struggle to have a linear correlation in cases when the test error is high. This phenomenon demonstrates the superiority of GdScore in unsupervised accuracy estimation. In the next paragraph, we also demonstrate the computational efficiency of our approach.



(A) Label generation strategies



(B) Loss selection

FIGURE 5.3: Performance comparison (R^2) on 7 datasets with ResNet18 between (a) different label generation strategies and (b) different types of losses across 3 types of distribution shifts. Results confirm that our proposed method performs better on average across various datasets and types of shifts.

GdScore is a more efficient self-training approach. In the list of baselines, both our method and ProjNorm [7] belong to self-training methods [144]. The latter, however, requires costly iterative training on the neural network during evaluation to obtain the complete set of fine-tuned parameters to calculate the distribution discrepancy in network parameters. Compared with ProjNorm, our method only trains the model for one epoch and collects the gradients of the linear classification layer to calculate the gradient norm, which is much more computationally efficient. Fig. 5.2 presents the comparison of computational efficiency between the two methods on 7 datasets with ResNet50. From this figure, we can see that our method is up to 80% faster than ProjNorm on average. This difference is striking on the Office-31 dataset, where our method is not only two orders of magnitude faster than ProjNorm but also improves the R^2 score by a factor of 2.

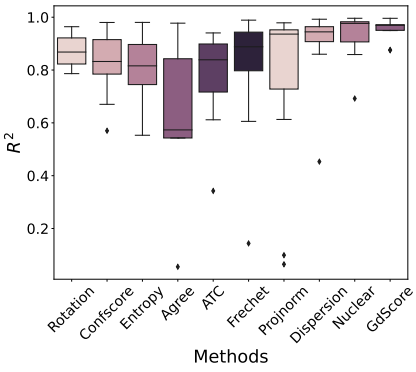
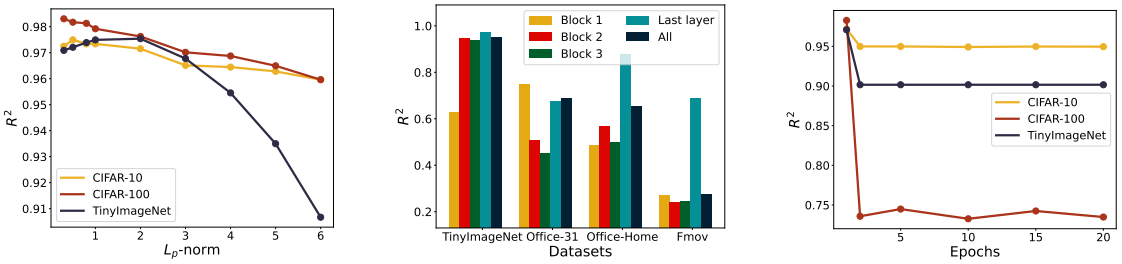


FIGURE 5.4: Robustness comparison for all estimation baselines across diverse distribution shifts with ResNet18.

Robustness of our approach. GDSCORE achieves great performance across all datasets, architectures, and types of shifts (Table 6.1). To highlight the robustness of our approach, we compare the distributions of R^2 in Figure 6.4, including as additional baseline the very recent *Nuclear Norm* [5] (more results in Appendix B.5).



(A) Norm types (B) Layer selection (C) Epoch selection

FIGURE 5.5: Sensitivity analysis on the effect of (a) norm types, (b) layer selection for gradients, and (c) epoch selection. The first and the third experiments are conducted on CIFAR-10C, CIFAR-100C, and TinyImageNet-C, while the second experiment includes TinyImageNet-C, Office-31, Office-Home, and WILDS-FMoV [4]. All experiments are conducted with ResNet18.

We show that GDSCORE is the best and most stable approach on 10 datasets (except ImageNet).

5.5 Ablation Study

Random pseudo-labels boost the performance under natural shift. In Fig. 5.3a, we conduct an ablation study to verify the effectiveness of our label generation strategy by comparing it with ground-truth labels, uniform labels [1], full random labels, and full pseudo-labeling. From the figure, we observe that while comparable with the ground truth under the synthetic drift and the subpopulation shift, the other strategies lead to a drastic drop in performance under the natural shift. This phenomenon is possibly caused by imprecise gradient calculation based on incorrect pseudo labels. Our labeling strategy performs better on average suggesting that random labels for low-confidence samples provide certain robustness of the score under natural shift.

Cross-entropy loss is robust to different shifts. To demonstrate the impact of different losses on the test accuracy estimation performance, we compare the standard cross-entropy loss used in our method with the entropy loss for samples with low confidence (see definition in Appendix B.4).

Cross-entropy loss is robust to different shifts. To demonstrate the impact of different losses on the test accuracy estimation performance, we compare the standard cross-entropy loss used in our method with the entropy loss for samples with low confidence (see definition in Appendix B.4). Moreover, we verify the effectiveness of the label smoothing, a simple yet effective tool for model calibration and performance improvement [147], by setting the smoothing rate as 0.4.

Fig. 5.3b illustrates this comparison revealing that standard cross-entropy loss is the most robust choice across different types of distribution shifts. We also note that the entropy loss enhances the estimation performance under the synthetic and the novel subpopulation shifts, but struggles under the natural shift. On the contrary, the label smoothing regularization can increase the performance under the natural shift but decrease it under the synthetic and the novel subpopulation shifts.

Choosing smaller p for better estimation performance. To illustrate the effect of the choice of L_p-norm on the performance, we conduct a sensitivity analysis on three datasets with ResNet18 summarized in Figure 5.5a. We can see that there is an obvious decreasing trend as p becomes larger. Especially, when p is smaller than 1, the estimation performance can fluctuate within a satisfying range. This is probably because a smaller p (i.e., $0 < p < 1$) makes the L_p-norm more suitable for the high-dimensional space, while the L_p-norm with $p \geq 1$ is likely to ignore gradients that are close to 0 [1, 148], so for all experiments, we used $p = 0.3$. The remark below, whose proof we defer to Appendix B.7.4, provides some theoretical insights into the L_p-norm of the gradient for $0 < p < 1$.

Remark 5.5.1 (Case $0 < p < 1$). Let $\mathbf{c}$ be the classifier obtained from $\boldsymbol{\omega}_s$ after one gradient descent step, i.e., $\mathbf{c} = \boldsymbol{\omega}_s - \eta \cdot \nabla \mathcal{L}_T(\boldsymbol{\omega}_s)$ with $\eta \geq 0$. For any $p \in (0, 1)$, we have

$$\eta \|\nabla \mathcal{L}_T(\boldsymbol{\omega}_s)\|_p \leq |\|\mathbf{c}\|_p - \|\boldsymbol{\omega}_s\|_p|.$$

Gradients from the last layer can provide sufficient information. In this part, we aim to understand whether backpropagation through other layers in the neural network can provide a better test accuracy estimate. For this, we separate the feature extractor $f_{\mathbf{g}}$ into 3 blocks of layers with roughly equal size and calculate the GrdNorm scores for each of them. Additionally, we also try to gather the gradients over the whole network. Fig. 5.5b plots the obtained results on 4 datasets,

suggesting that the last layer provides sufficient information to predict the true test accuracy in an unsupervised way.

5.5.1 No gains after 1 epoch of backpropagation.

Here, we train the neural network for r epochs, where $r \in \{1, 2, 5, 10, 15, 20\}$, and store the gradient vectors of the classification layer for each value of r. Fig. 5.5c suggests that the gradient norms after 1 step are sufficient to predict the model performance under distribution shifts. Further training gradually degrades the performance with the increasing r. The reason behind the phenomenon is that the gradients in the first epoch contain the most abundant information about the training dataset, while the neural network fine-tuned on the test dataset for several epochs is going to forget previous training categories [149].

Choice of proper threshold τ In our experiments (see Section 5.4), we set the value of τ as 0.5 across all datasets and network architectures. This choice of τ is due to the intuition that if a label contains a softmax probability below 0.5, it means that this predicted label has over 50% chances of being wrong. It means that this label has a higher probability of being incorrect than to be correct. Thus, we tend to regard it as an incorrect prediction. To demonstrate the impact of threshold τ on the final performance, we conduct an ablation study on CIFAR-10C and Office-31 with ResNet18 using varying values of τ. We display in Table 5.2 the corresponding values of R^2. We can observe that the final performance improves and achieves its best value for τ is 0.5, before decreasing slightly.

TABLE 5.2: Performance on CIFAR-10 and Office-31 with ResNet18 for varying value of τ. The metric used in this table is the coefficient of determination R^2. The best result is highlighted in **bold**.

Threshold	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
CIFAR-10C	0.963	0.963	0.964	0.965	0.971	**0.972**	0.967	0.962	0.963	0.959
Office-31	0.495	0.498	0.532	0.674	**0.685**	0.667	0.545	0.451	0.114	0.131

5.6 Discussion: Influence of the Calibration Error

We have mentioned earlier that, in theory, the proposed pseudo-labeling strategy depends on how well the prediction probabilities are calibrated. In degraded cases, this can have a negative impact on our approach, e.g., one can imagine a model that outputs only one-hot probabilities with not a high accuracy. However, this it is generally not the case. Indeed, in practice, we do not need to have a perfectly calibrated model as we employ a mixed strategy that assigns pseudo-labels to high-confidence examples and random labels to low-confidence ones. The recent success of applying self-training models to different problems [150–152] provides evidence of the suitability of the label generation strategy we adopted.

When we speak of deep neural networks, which are widely accepted to be poorly calibrated, Minderer et al. [153] showed that modern SOTA image models tend to be well-calibrated across distribution shifts. To demonstrate it empirically, in Table 5.3, we provide the expected calibration error (ECE, [154]) of ResNet18, one of the considered base models, depending on the difficulty of test data. For this, we test first on CIFAR-10 (ID), and then on CIFAR-10C corrupted by brightness across diverse severity from 1 to 5. We can see that ECE is very low for ID data and remains relatively low across all levels of corruption severity, which shows that ResNet is quite well-calibrated on CIFAR-10.

TABLE 5.3: Expected Error Calibration (ECE) of ResNet18 on CIFAR-10 (ID) and CIFAR-10C corrupted by brightness across diverse severity from 1 to 5.

Corruption Severity	ID	1	2	3	4	5
ECE	0.0067	0.0223	0.0230	0.0243	0.0255	0.0339

On the other hand, in the case of a more complex distribution shift like Office-31 data set, we can see that the calibration error has been increased noticeably (Table 5.4). It is interesting to analyze this result together with Figure 5.3a of the main chapter, where we compared the results between the usual pseudo-labeling strategy and the proposed one. Although our method has room for improvement compared to the oracle method, it is also significantly better than "pseudo-labels", indicating that the proposed label generation strategy is less sensitive to the calibration error.

TABLE 5.4: Expected Error Calibration (ECE) of ResNet18 on Office-31 data set.

Domain	DSLR (ID)	Amazon	Webcam
ECE	0.2183	0.2167	0.4408

5.7 Chapter Summary

In this chapter, we showcased the strong linear relationship between the magnitude of gradients and the model performance under distribution shifts. We proposed GDSCORE, a simple yet efficient method to estimate the ground-truth test accuracy error by measuring the gradient magnitude of the last classification layer. Our method consistently achieves superior performance across various distribution shifts than previous works. Furthermore, it does not require the detailed architecture of the feature extractors and training datasets. Those properties guarantee that our method can be easily deployed in the real world, and meet practical demands, such as large models and confidential information. We hope that our research sheds new light on the usefulness of gradient norms for unsupervised accuracy estimation.

Chapter 6

MaNo: Exploiting Matrix Norm for Unsupervised Accuracy Estimation Under Distribution Shifts

6.1 Introduction

The deployment of machine learning models in real-world scenarios is frequently challenged by distribution shifts between the training and test data. These shifts can substantially deteriorate the model's performance during testing [4, 68, 132] and introduce significant risks related to AI safety [28, 155]. To alleviate this issue, it is common to monitor model performance by periodically collecting the ground truth labels for a subset of the current test dataset [76]. However, this approach is often resource-intensive and time-consuming, which motivates the importance of estimating the model's performance on out-of-distribution (OOD) data in an unsupervised manner, also known as *Unsupervised Accuracy Estimation* [156].

Due to privacy constraints and computational efficiency, one of the most popular ways to estimate accuracy without labels is to rely on the model's outputs, called logits, as a source of confidence in the model's predictions [5, 24, 25, 27]. For instance, *ConfScore* [24] leverages the average maximum `softmax` probability as the test accuracy estimator, while Deng et al. [5] has recently proposed to estimate the

accuracy via the nuclear norm of the `softmax` probability matrix. These approaches, however, tend to underperform on the natural shift applications while the intuition behind the use of logits remains unclear. This motivates us to ask:

Question 1 *What explains the correlation between logits and generalization performance?*

In section 6.3, we show that logits are connected to the model's margins, *i.e.*, the distances between the learned embeddings and the decision boundaries. Inspired by the low-density separation (LDS) assumption [157, 158] that optimal decision boundaries should lie in low-density regions, we propose MaNo, an estimation score that aggregates the margins at a dataset level by taking the L_p-norm of the normalized model's prediction matrix to evaluate the density around decision boundaries. Nevertheless, logit-based approaches are known to suffer from overconfidence [111, 159], resulting in high prediction bias, especially under poorly-calibrated scenarios. This leads us to another critical question:

Question 2 *How to alleviate the overconfidence issues of logits-based methods?*

In section 6.4, we reveal that this question is connected to the normalization of logits and show that the widely-used `softmax` normalization accumulates errors in the presence of prediction bias, which can lead to overconfidence and significantly degrade the performance of existing accuracy estimation methods in poorly-calibrated scenarios. To mitigate this issue, we propose a novel normalization strategy called `SoftTrun` that takes into account the empirical distribution of logits and aims to find a trade-off between information completeness of ground-truth logits and error accumulation.

Summary of our contributions. **(1)** We show that logits are informative of generalization performance through the lens of the low-density separation assumption by reflecting the distances to decision boundaries. **(2)** We identify the failure of the commonly-used `softmax` normalization that accumulates errors under poorly calibrated because of its overconfidence, leading to biased estimation. **(3)** We propose MaNo, a training-free estimation method that quantifies the global distances to decision boundaries by taking the L_p norm of the logits matrix. MaNo relies on a novel normalization technique that makes a trade-off between information completeness and error accumulation and is robust to different calibration scenarios. In addition, we demonstrate its connection to the model's uncertainty. **(4)** We

demonstrate the superiority of MaNo compared to 11 competitors with a large-scale empirical evaluation including 12 benchmarks across diverse distribution shifts. Results show that MaNo consistently improves over the state-of-the-art baselines, including on the challenging natural shift.

6.2 Problem Statement

Setting. Consider a classification task with input space $\mathcal{X} \subset \mathbb{R}^D$ and label space $\mathcal{Y} = \{1, \dots, K\}$. Let p_S and p_T be the source and target distributions on $\mathcal{X} \times \mathcal{Y}$, respectively, with $p_S \neq p_T$. During pre-training, we train a neural network $f \colon \mathcal{X} \to \mathbb{R}^K$ on a training set $\mathcal{D}_{\text{train}}$ with samples drawn i.i.d. from p_S. The model can be parameterized as $f = f_{\mathbf{W}} \circ f_{\boldsymbol{\phi}}$, where $f_{\boldsymbol{\phi}} \colon \mathcal{X} \to \mathbb{R}^q$ is a feature extractor and $f_{\mathbf{W}} \colon \mathbb{R}^q \to \mathbb{R}^K$ is a linear classifier with parameters $\mathbf{W} = (\boldsymbol{\omega}_k)_{k=1}^K \in \mathbb{R}^{q \times K}$. Further, we denote an input by $\mathbf{x}$, its corresponding label by y, its representation by $\mathbf{z} = f_{\boldsymbol{\phi}}(\mathbf{x})$ and logits by $\mathbf{q} = f(\mathbf{x}) = (\boldsymbol{\omega}_k^\top \mathbf{z})_k \in \mathbb{R}^K$. The test accuracy of f on $\mathcal{D}$ is defined as $\text{Acc}(f, \mathcal{D}) \coloneqq \frac{1}{|\mathcal{D}|} \sum_{(\mathbf{x},y) \in \mathcal{D}} \mathbb{1}_{\hat{y}=y}$ with predicted labels $\hat{y}$. The probability simplex is denoted by $\Delta_K = \{\mathbf{p} \in [0,1]^K | \mathbb{1}_K^\top \mathbf{p} = 1\}$.

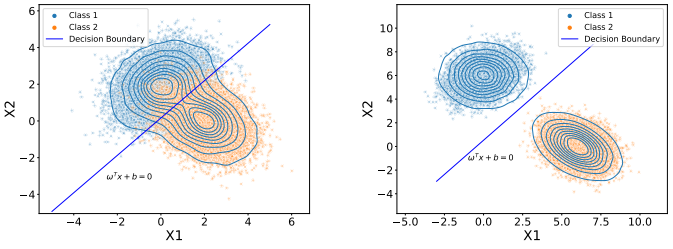
Unsupervised accuracy estimation. Given a model f pre-trained on $\mathcal{D}_{\text{train}}$, the goal of *unsupervised accuracy estimation* is to assess the generalization performance of f on target data drawn from p_T *without having access* to ground-truth target labels. This is a challenging task as we are subject to distribution shifts, *i.e.* $p_S \neq p_T$, which often occur in real-world scenarios, and ground-truth labels are inaccessible which prevents monitoring the model generalization performance at test time. In practice, given an unlabeled test set $\mathcal{D}_{\text{test}} = \{\mathbf{x}_i\}_{i=1}^N$ with N samples drawn i.i.d. from p_T, we aim to provide an estimation score $\mathcal{S}(f, \mathcal{D}_{\text{test}})$ that exhibits a linear correlation with the true OOD accuracy $\text{Acc}(f, \mathcal{D}_{\text{test}})$. Following the standard closed-set setting, both p_T and p_S involve the same K classes. We refer the reader to Appendix C.2 for an extended discussion of related work.

6.3 What Explains the Correlation between Logits and Test Accuracy?

Although existing literature has shown the feasibility of unsupervised accuracy prediction under distribution shift by utilizing the model's logits [5, 25, 27], the reason behind this empirical success remains unclear. In this section, we seek to understand when and why logits can be informative for analyzing generalization performance. Based on the derived understanding, we propose our approach, MaNo, for estimating generalization performance.

6.3.1 Motivation

Logits reflect the distances to decision boundaries. We analyze logits from a linear classification perspective in the embedding space, where the decision boundary of class k is the hyperplane $\{\boldsymbol{z}' \in \mathbb{R}^q | \boldsymbol{\omega}_k^\top \boldsymbol{z}' = 0\}$. In Appendix C.4.4, we remind that the distance from a point $\mathbf{z}$ to hyperplane $\boldsymbol{\omega}_k$ is given by $\mathrm{d}(\boldsymbol{\omega}_k, \mathbf{z}) = |\boldsymbol{\omega}_k^\top \boldsymbol{z}| / \|\boldsymbol{\omega}_k\|$. As the pre-trained model is fixed and $\boldsymbol{\omega}_k$ can be normalized, we derive that the logits in absolute values are proportional to the distance from the learned embeddings to the decision boundaries, *i.e.*, $|\mathbf{q}_k| = |\boldsymbol{\omega}_k^\top \mathbf{z}| \propto d(\boldsymbol{\omega}_k, \mathbf{z}), \forall k$. This indicates that the magnitude of logits reflects how close the corresponding embedding is from each decision boundary.



(A) High-density region (B) Low-density region

FIGURE 6.1: **Illustration of the LDS assumption.** When the boundary passes through dense regions **(a)**, margins have little predictive power and can not be used without labels, while margins are informative in sparse regions **(b)**.

Low-density separation assumption. When dealing with unlabeled data, it is required to make assumptions on the relationship between the distance to decision boundaries and generalization performance. The low-density separation assumption (LDS, Chapelle and Zien, 2005) states that optimal decision boundaries should lie

in low-density regions (Figure 6.1b) so that unlabeled margin $|\boldsymbol{\omega}_k^\top \mathbf{z}|$ reflects reliable confidence in predicting $\mathbf{x}$ to the class k. The assumption is often empirically supported as the misclassified samples tend to be significantly closer to the decision boundary than the correctly classified ones [160]. This might indicate that **the absolute values of the logits are positively correlated to its generalization performance**.

Assumptions on the prediction bias. It is important to note that the LDS assumption has been initially proposed for semi-supervised learning, where labeled and unlabeled data are assumed to come from the same distribution, which is not the case in our setting. This leads to logits writing $f(\mathbf{x}) = \mathbf{q}^* + \boldsymbol{\varepsilon}$ in the general case[1], *i.e.*, subject to a potentially non-negligible prediction bias $\boldsymbol{\varepsilon} = (\varepsilon_k)_k$ with respect to the ground-truth logits $\mathbf{q}^* \in \mathbb{R}^K$. The following proposition shows the impact of the prediction bias on the divergence between the true class posterior probabilities, assumed modeled as $\mathbf{p} = \text{softmax}\mathbf{q}^* \in \Delta_K$, and the estimated ones $\mathbf{s} = \text{softmax}f(\mathbf{x}) \in \Delta_K$.

Proposition 6.3.1. Let $\boldsymbol{\varepsilon}_+ = (\max_l\{\varepsilon_l\} - \varepsilon_k)_k$. Then, the KL divergence between $\mathbf{p}$ and $\mathbf{s}$ verifies

$$0 \leq \text{KL}(\mathbf{p}||\mathbf{s}) \leq \boldsymbol{\varepsilon}_+^T \mathbf{p}.$$

The proposition indicates that a large approximation error of the posterior may be caused by prediction bias that has a large norm and / or bad alignment with the true probabilities. Thus, the logit-based methods assume that the magnitude of the bias is reasonably bounded while the direction of bias does not drastically harm the ranking of classes by probabilities. We elaborate on this discussion and present the proof of Proposition 6.3.1 in Appendix C.4.1.

6.3.2 MaNo: predicting generalization performance with matrix norm of logits

We have shown a connection between the feature-to-boundary distances and generalization performance as well as the impact of the prediction bias. Based on the

[1]We write this decomposition without loss of generality as no restrictions are imposed on $\boldsymbol{\varepsilon}$.

derived intuition, we introduce MaNo that leverages the model margins at the dataset level performing two steps: normalization and aggregation. The pseudo-code of MaNo is provided in Appendix C.1.

Step 1: Normalization. Given that logits can exhibit significant variations in their scale depending on the input $\mathbf{x}$, it is crucial to normalize the logits within a standardized range to prevent outliers from exerting disproportionate influence on the estimation. A natural range stems from the fact that most deep classifiers have outputs in Δ_K, which amounts to applying a normalization function $\sigma\colon \mathbb{R}^K \to \Delta_K$ on top of the pre-trained neural network [161], where Δ_K refers to probability simplex. This ensures having logits entries in $[0, 1]$. For each test sample $\mathbf{x}_i$, we first extract its learned feature representation $\mathbf{z}_i = f_{\boldsymbol{\phi}}(\mathbf{x}_i)$. Then, logits corresponding to this representation are computed as $\mathbf{q}_i = f_{\boldsymbol{W}}(\mathbf{z}_i) \in \mathbb{R}^K$. The normalization procedure results in a prediction matrix $\mathbf{Q} \in \mathbb{R}^{N\times K}$ with each row $\mathbf{Q}_i$ containing the normalized logits of an input sample:

$$\mathbf{Q}_i = \sigma(\mathbf{q}_i) \in \Delta_K, \tag{6.1}$$

where σ denotes the normalization function for the logits values. It is worth noting that not all normalization methods are appropriate candidates. The selection of a suitable normalization function σ based on different calibration scenarios will be discussed in detail in Section 6.4.

Step 2: Aggregation. Once the logits are scaled, we aggregate the dataset-level information on feature-to-boundary distances by taking the entry-wise L_p norm of the prediction matrix $\mathbf{Q}$, which can be expressed as:

$$\mathcal{S}(f, \mathcal{D}_{\text{test}}) = \frac{1}{\sqrt[p]{NK}}\|\mathbf{Q}\|_p = \left(\frac{1}{NK}\sum_{i=1}^{N}\sum_{k=1}^{K}|\sigma(\boldsymbol{q}_i)_k|^p\right)^{\frac{1}{p}}, \tag{6.2}$$

As we have $||\mathbf{Q}||_p \leq \sqrt[p]{NK}\max(\mathbf{Q}_{ij}) = \sqrt[p]{NK}$ ($\mathbf{Q}_{ij} \in [0, 1]$), the scaling by $\sqrt[p]{NK}$ leads to $\mathcal{S}(f, \mathcal{D}_{\text{test}}) \in [0, 1]$, providing a standardized metric regardless of variations in the size of the test dataset N and the number of classes K. As p increases, MaNo puts greater emphasis on high-margin terms, focusing on confident classification hyperplanes. In the extreme case where $p \to \infty$, we have $|\mathbf{Q}||_p \to \max(\mathbf{Q}_{ij})$. In

practice, we choose $p = 4$ in all experiments and provide an ablation study on p in Appendix 6.6.1. As the L_p norm is straightforward to compute, our approach is scalable and efficient compared to the current state-of-the-art method Nuclear [5] that requires performing a singular value decomposition.

6.3.3 Theoretical analysis of MaNo

In this section, we provide the theoretical support for the positive correlation between MaNo and test accuracy. More specifically, we reveal that our proposed score is connected with the uncertainty of the neural network's predictions in Theorem 6.3.3. Before presenting this result, we recall below the definition of Tsallis α-entropies introduced in Tsallis [162].

Definition 6.3.2 (Tsallis α-entropies [162]). Let $\alpha > 1$ and $k > 0$. The Tsallis α-entropy is defined as:

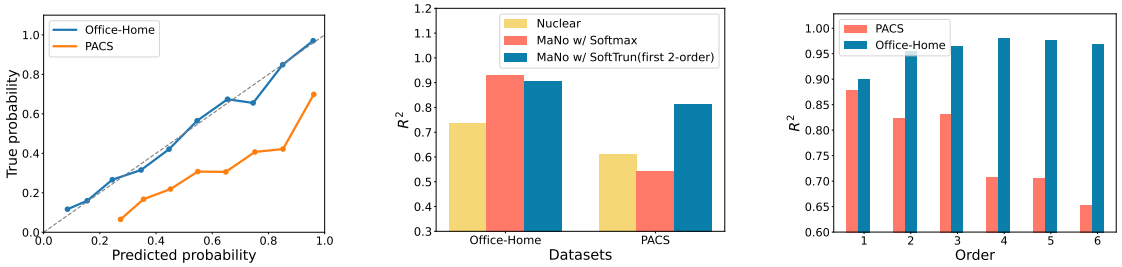
$$\mathbf{H}_\alpha^{\mathrm{T}}(\mathbf{p}) = k(\alpha - 1)^{-1}(1 - \|\mathbf{p}\|_\alpha^\alpha).$$

In this work, we choose $k = \frac{1}{\alpha}$ following Blondel et al. [163]. The Tsallis entropies generalize the Shannon entropy (limit case $\alpha \to 1$) and have been used in various applications [163–165]. More details can be found in Appendix C.3. The following theorem, whose proof is deferred to Appendix C.4.5, states that the estimation score obtained with MaNo is a function of the average Tsallis entropy of the normalized neural network's logits.

Theorem 6.3.3 (Connection to uncertainty). Let $p > 1$, $a = \frac{p(p-1)}{K}$ and $b = \frac{1}{K}$. Given a test set $\mathcal{D}_{\text{test}} = \{\mathbf{x}_i\}_{i=1}^N$, corresponding logits $\mathbf{q}_i = f(\mathbf{x}_i)$, a normalization function $\sigma\colon \mathbb{R}^K \to \Delta_K$ and $p > 1$, the estimation score $\mathcal{S}(f, \mathcal{D}_{\text{test}})$ provided by MaNo (Algorithm 4) verifies

$$\mathcal{S}(f, \mathcal{D}_{\text{test}})^p = -a\left(\frac{1}{N}\sum_{i=1}^N \mathbf{H}_p^{\mathrm{T}}(\sigma(\mathbf{q}_i))\right) + b. \tag{6.3}$$

As $a > 0$, Theorem 6.3.3 implies that the estimation score provided by MaNo is negatively correlated with the average Tsallis-entropy on the test set. In particular,



(A) Calibration curves.

(B) Type of normalization.

(C) Order n in Eq. equation 6.4.

FIGURE 6.2: **Empirical evidence with Resnet18. (a)** The model is well-calibrated on Office-Home and miscalibrated on PACS. (b) `SoftTrun` is superior to the state-of-the-art Nuclear [5] in all scenarios while the `softmax` heavily fails on PACS. **(c)** Increasing the approximation order n in Eq. equation 6.4 is detrimental on PACS and beneficial on Office-Home. Taking $n \in \{2, 3\}$ is an optimal trade-off in all calibration scenarios.

the less certain the model is on test data, the lower the test accuracy is and the higher the entropy term is in Eq. equation 6.3, resulting in a lower score $\mathcal{S}(f, \mathcal{D}_{\text{test}})$. As the converse sense holds, MaNo provides a score positively correlated to the test accuracy. This follows the findings of Guillory et al. [25], Wang et al. [166] and empirically confirmed in Section 6.5 for various architectures, datasets, and types of shift.

6.4 How to Alleviate Overconfidence Issues of Logit-Based Methods?

The most common normalization technique of existing logit-based approaches is the `softmax` normalization. In this section, we show that the widely used `softmax` is sensitive to prediction bias, which hinders the quality of the estimation in poorly calibrated scenarios. To alleviate this issue, we propose a novel normalization strategy, `SoftTrun`, which balances the information completeness and overconfidence accumulation based on calibration.

6.4.1 The failure of softmax normalization under poorly-calibrated scenarios

Empirical evidence. The `softmax` normalization can suffer from overconfidence issues [159, 167] and saturation of its outputs [168], with one entry close to one while the others are close to zero. Consequently, logit-based accuracy estimation methods using `softmax` are sensitive to prediction bias, leading to low-quality estimations in poorly calibrated scenarios. We illustrate this phenomenon in Figure 6.2a on two datasets, where a pre-trained ResNet18 exhibits more pronounced calibration issues on PACS [169] compared to Office Home [109]. Figure 6.2b shows that using `softmax` in both MaNo and the state-of-the-art method Nuclear [5] negatively impacts their accuracy estimation performance on the poorly calibrated PACS dataset.

Analysis. To alleviate those issues, we first notice that the `softmax` can be decomposed as $\text{softmax}\mathbf{q} = \exp(\mathbf{q})/\sum_{k=1}^{K}\exp(\mathbf{q}_k) = (\phi \circ \exp)(\mathbf{q})$, where $\phi\colon \mathbb{R}_+^K \to \Delta_K$ writes $\phi(\mathbf{u}) = \mathbf{u}/\sum_{k=1}^{K}\mathbf{u}_k = \mathbf{u}/\|\mathbf{u}\|_1$. While ϕ has appealing property for normalization (see Proposition C.4.4), the exponential accumulates prediction errors, leading to the `softmax` overconfidence and a biased accuracy estimation. In particular, assume that the k-th entry of the output of the neural network on a test sample $\mathbf{x}_i$ writes $\mathbf{q}_{i,k}^* + \varepsilon_k$, where $\mathbf{q}_{i,k}^*$ are the ground-truth logits and ε_k is the prediction error. Then, the n-order Taylor polynomial of the exponential writes

$$\exp(\mathbf{q}_{i,k}^* + \varepsilon_k) \approx 1 + (\mathbf{q}_{i,k}^* + \varepsilon_k) + \frac{(\mathbf{q}_{i,k}^* + \varepsilon_k)^2}{2!} + \ldots + \frac{(\mathbf{q}_{i,k}^* + \varepsilon_k)^n}{n!}. \tag{6.4}$$

Figure 6.2c illustrates the impact of truncating Eq. equation 6.4 up to the n-th order. We conclude that a trade-off is needed between *information completeness on true logits and error accumulation* depending on the type of calibration scenario. Specifically, when the model is poorly calibrated on a given dataset (*i.e.*, ε_k large in absolute value), the normalization should focus on avoiding error accumulation and when the model is well calibrated (*i.e.*, ε_k small in absolute value), the normalization should focus on information completeness.

6.4.2 `SoftTrun`: the proposed normalization strategy

The above analysis shows that different calibration scenarios require an emphasis on different information during normalization. Therefore, we propose a normalization strategy called `SoftTrun` that normalizes the model outputs based on the calibration scenario. Given logits $\mathbf{q}_i \in \mathbb{R}^K$ and reusing the function ϕ previously introduced, it takes the general form:

$$\sigma(\mathbf{q}_i) = (\phi \circ v)(\mathbf{q}_i) = \frac{v(\mathbf{q}_i)}{\sum_{k=1}^{K} v(\mathbf{q}_i)_k} \in \Delta_K. \tag{6.5}$$

where $v\colon \mathbb{R}^K \to \mathbb{R}_+^K$ is designed to avoid error accumulation under poorly-calibrated scenarios by truncating the exponential ($n = 2$ in Eq. equation 6.4) and using complete logits information under well-calibrated scenarios. As in practice, the calibration of the model on test data is unknown, `SoftTrun` employs a simple yet effective strategy reminiscent of pseudo-labeling [170, 171]. More specifically, given a test dataset $\mathcal{D}_{\text{test}} = \{\mathbf{x}_i\}_{i=1}^N$ and corresponding logits $\mathbf{q}_i = f(\mathbf{x}_i)$, a criterion $\Phi(\mathcal{D}_{\text{test}})$ is computed at the dataset level and the normalized logits are defined as

$$v(\mathbf{q}_i) = \begin{cases} 1 + \mathbf{q}_i + \frac{\mathbf{q}_i^2}{2!}, & \text{if } \Phi(\mathcal{D}_{\text{test}}) \leq \eta \\ \exp(\mathbf{q}_i), & \text{if } \Phi(\mathcal{D}_{\text{test}}) > \eta \end{cases}. \tag{6.6}$$

We define $\Phi(\mathcal{D}_{\text{test}}) = -\frac{1}{NK}\sum_{j=1}^{N}\sum_{k=1}^{K}\log(\frac{\exp(\mathbf{q}_j)_k}{\sum_{j=1}^{K}\exp(\mathbf{q}_i)_j})$, which is equal, up to a constant, to the average KL divergence between the uniform distribution and the predicted `softmax` probabilities. It follows from Tian et al. [172] that showed that this KL divergence was small when the uncertainty of the model was high and large for confident models. Hence, when uncertainty is high, *i.e.*, $\Phi(\mathcal{D}_{\text{test}}) \leq \eta$, we truncate the exponential to reduce error accumulation, and when the model is certain, *i.e.*, $\Phi(\mathcal{D}_{\text{test}}) > \eta$, complete information is used with the exact exponential (and we recover the `softmax`). In all our experiments, we fix $\eta = 5$. In Appendix C.4, we provide theoretical insights on our choices of η and $\Phi(\mathcal{D}_{\text{test}})$ along with additional discussion.

6.5 Experiments

6.5.1 Experiment setup

Pre-training datasets. For pre-training the neural network, we use a diverse set of datasets including CIFAR-10, CIFAR-100 [125], TinyImageNet [126], ImageNet [145], PACS [169], Office-Home [109], DomainNet [173] and RR1-WILDS [174], and BREEDS [146] which leverages class hierarchy of ImageNet [145] to create 4 datasets including Living-17, Nonliving-26, Entity-13 and Entity-30.

Test datasets. In our comprehensive evaluation, we consider 12 datasets with 3 types of distribution shifts: the synthetic, the natural, and the subpopulation shifts. To verify the effectiveness of our method under the synthetic shift, we use CIFAR-10C, CIFAR-100C, and ImageNet-C [8] that span 19 types of corruption across 5 severity levels, as well as TinyImageNet-C [8] with 15 types of corruption and 5 severity levels. For the natural shift, we use the domains excluded from training from PACS, Office-Home, and DomainNet, RR1-WILDS as the OOD datasets. For the novel subpopulation shift, we consider the BREEDS benchmark with Living-17, Nonliving-26, Entity-13, and Entity-30 which are constructed from ImageNet-C.

Training details. To show the versatility of our method across different architectures, we perform experiments on ResNet18, ResNet50 [105], and WRN-50-2 [127] models. We train them for 20 epochs for CIFAR-10 [125] and 50 epochs for the other datasets. In all cases, we use SGD with a learning rate of 10^{-3}, cosine learning rate decay [128], a momentum of 0.9, and a batch size of 128.

Evaluation metrics We use the coefficient of determination (R^2, Nagelkerke et al., 1991) and the Spearman's rank correlation coefficient (ρ, Kendall, 1948) to evaluate performance. The ρ coefficient measures monotonicity, ranging from -1 to 1, where values close to 1 or -1 indicate strong correlation, and 0 indicates no correlation. The R^2 coefficient measures the linearity and goodness of fit between estimates and accuracy, ranging from 0 to 1, with 1 indicating a perfect fit.

TABLE 6.1: Method comparison on four benchmarks with ResNet18, ResNet50 and WRN-50-2 under the **synthetic shift**, where R^2 refers to coefficients of determination, and ρ refers to the absolute value of Spearman correlation coefficients (higher is better). The best results are highlighted in **bold**. MANO consistently achieves the highest R^2 and ρ values across different datasets and network architectures, indicating its superior performance.

Dataset	Network	Synthetic Shift																					
		Rotation		ConfScore		Entropy		AgreeScore		ATC		Fréchet		Dispersion		ProjNorm		MDE		COT		Nuclear		MANO	
		R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ
CIFAR 10	ResNet18	0.822	0.951	0.869	0.985	0.899	0.987	0.663	0.929	0.884	0.985	0.950	0.971	0.968	0.990	0.936	0.982	0.957	0.987	0.989	0.995	**0.995**	**0.997**	**0.995**	**0.997**
	ResNet50	0.835	0.961	0.935	0.993	0.945	0.994	0.835	0.985	0.946	0.994	0.858	0.964	0.987	0.990	0.944	0.989	0.978	0.963	0.984	0.996	0.994	0.996	**0.996**	**0.997**
	WRN-50-2	0.862	0.976	0.943	0.994	0.942	0.994	0.856	0.986	0.947	0.994	0.814	0.973	0.962	0.988	0.961	0.989	0.930	0.809	0.988	0.994	0.994	0.995	**0.996**	**0.992**
	Average	0.840	0.963	0.916	0.991	0.930	0.992	0.785	0.967	0.926	0.991	0.874	0.970	**0.972**	0.990	0.947	0.987	0.955	0.920	0.987	0.995	0.995	**0.996**	**0.996**	0.995
CIFAR 100	ResNet18	0.860	0.936	0.916	0.985	0.891	0.979	0.902	0.973	0.938	0.986	0.888	0.968	0.952	0.988	0.979	0.980	0.975	0.994	0.991	0.995	0.989	0.995	**0.996**	**0.996**
	ResNet50	0.908	0.962	0.919	0.984	0.884	0.977	0.922	0.982	0.921	0.984	0.837	0.972	0.951	0.985	0.988	0.991	0.988	0.995	0.985	0.996	0.979	0.994	**0.995**	**0.997**
	WRN-50-2	0.924	0.970	0.971	0.984	0.968	0.981	0.955	0.977	0.978	0.993	0.865	0.987	0.980	0.991	0.990	0.991	0.995	0.994	0.987	0.997	0.962	0.988	**0.996**	**0.998**
	Average	0.898	0.956	0.936	0.987	0.915	0.983	0.927	0.982	0.946	0.988	0.864	0.976	0.962	0.988	0.985	0.987	0.986	0.994	0.988	0.996	0.977	0.993	**0.996**	**0.997**
TinyImageNet	ResNet18	0.786	0.946	0.670	0.869	0.592	0.842	0.561	0.853	0.751	0.945	0.826	0.970	0.966	0.986	0.970	0.981	0.941	0.993	**0.985**	0.994	0.983	0.994	0.981	**0.996**
	ResNet50	0.786	0.947	0.670	0.869	0.651	0.892	0.560	0.853	0.751	0.945	0.826	0.971	0.977	0.986	0.979	0.987	0.941	0.993	**0.980**	0.994	0.965	0.994	**0.980**	**0.996**
	WRN-50-2	0.878	0.967	0.757	0.951	0.704	0.935	0.654	0.904	0.635	0.897	0.884	0.984	0.968	0.986	0.965	0.983	0.961	0.996	**0.985**	0.997	0.962	0.988	0.979	**0.997**
	Average	0.805	0.959	0.727	0.920	0.650	0.890	0.599	0.878	0.693	0.921	0.847	0.976	0.970	0.987	0.972	0.984	0.950	0.995	**0.984**	0.995	0.968	0.993	0.980	**0.996**
ImageNet	ResNet18	-	-	0.979	0.991	0.963	0.991	-	-	0.974	0.983	0.802	0.974	0.940	0.971	0.975	0.993	0.924	0.994	**0.996**	**0.998**	0.992	0.997	0.992	0.997
	ResNet50	-	-	0.980	0.994	0.967	0.992	-	-	0.970	0.983	0.855	0.974	0.938	0.968	0.986	0.993	0.886	0.994	**0.993**	0.996	0.985	0.997	0.991	**0.998**
	WRN-50-2	-	-	0.983	0.991	0.963	0.991	-	-	0.983	0.993	0.909	0.988	0.939	0.976	0.978	0.993	0.880	0.997	0.989	0.994	0.987	0.998	**0.996**	**0.998**
	Average	-	-	0.981	0.993	0.969	0.992	-	-	0.976	0.987	0.855	0.979	0.939	0.972	0.980	0.993	0.897	0.995	0.993	0.996	0.988	0.998	**0.993**	**0.998**

Baselines. We consider 11 baselines commonly evaluated in the unsupervised accuracy estimation studies, including *Rotation Prediction* (Rotation) [23], *Averaged Confidence* (ConfScore) [24], *Entropy* [25], *Agreement Score* (AgreeScore) [26], *Averaged Threshold Confidence* (ATC) [27], *AutoEval* (Fréchet) [28], *ProjNorm* [7], *Dispersion Score* (Dispersion) [29], MDE [30], COT [31], and *Nuclear Norm* (Nuclear) [5].

6.5.2 Main observations

MaNo shows competitive performance gains under both synthetic and subpopulation shifts. Tables 6.1 and 6.2 present the numerical results of unsupervised accuracy estimation across 8 datasets using 3 different network architectures, evaluated under synthetic and subpopulation shifts. These shifts are quantified by R^2 and ρ. Empirical results demonstrate that these distribution shifts do not significantly impact calibration (*i.e.*, $S_{cali} \leq \eta$). We observe that MANO consistently outperforms other baselines, achieving state-of-the-art performance. For instance, MANO achieves $R^2 > 0.960$ and $\rho > 0.990$ under subpopulation shift, whereas the performance of other baselines does not reach such consistently high levels.

TABLE 6.2: Method comparison on four benchmarks using ResNet18, ResNet50, and WRN-50-2 under **subpopulation shift** with R^2 and ρ metrics (the higher the better). The best results are in **bold**.

Subpopulation Shift																									
Dataset	Network	Rotation		ConfScore		Entropy		AgreeScore		ATC		Fréchet		Dispersion		ProjNorm		MDE		COT		Nuclear		MANO	
		R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ
Entity-13	ResNet18	0.927	0.961	0.795	0.940	0.794	0.935	0.543	0.919	0.823	0.945	0.950	0.981	0.937	0.968	0.952	0.981	0.927	0.995	0.960	0.985	0.978	0.991	**0.992**	**0.996**
	ResNet50	0.932	0.976	0.728	0.941	0.698	0.928	0.901	0.964	0.783	0.950	0.903	0.959	0.764	0.892	0.944	0.974	0.912	0.993	0.935	0.971	0.989	0.996	**0.993**	**0.998**
	WRN-50-2	0.939	0.983	0.930	0.977	0.919	0.973	0.871	0.935	0.936	0.980	0.906	0.958	0.815	0.905	0.950	0.977	0.925	0.995	0.944	0.979	0.989	0.995	**0.992**	**0.996**
	Average	0.933	0.973	0.817	0.953	0.804	0.945	0.772	0.939	0.847	0.958	0.920	0.966	0.948	0.977	0.839	0.922	0.921	0.995	0.947	0.979	0.985	0.994	**0.993**	**0.996**
Entity-30	ResNet18	0.964	0.979	0.570	0.836	0.553	0.832	0.542	0.935	0.611	0.845	0.849	0.978	0.929	0.968	0.952	0.987	0.931	0.994	0.971	0.993	0.980	0.993	**0.991**	**0.996**
	ResNet50	0.961	0.980	0.878	0.969	0.838	0.956	0.914	0.975	0.924	0.973	0.835	0.956	0.783	0.914	0.937	0.986	0.918	0.995	0.958	0.982	0.978	0.994	**0.988**	**0.997**
	WRN-50-2	0.940	0.978	0.897	0.974	0.878	0.970	0.826	0.955	0.936	0.984	0.927	0.973	0.927	0.973	0.959	0.986	0.925	0.995	0.944	0.979	0.985	0.996	**0.988**	**0.997**
	Average	0.955	0.978	0.781	0.926	0.756	0.919	0.728	0.956	0.823	0.934	0.871	0.969	0.880	0.952	0.949	0.987	0.925	0.995	0.970	0.988	0.981	0.994	**0.989**	**0.996**
Living-17	ResNet18	0.876	0.973	0.913	0.973	0.898	0.970	0.586	0.736	0.940	0.973	0.768	0.950	0.900	0.958	0.923	0.970	0.927	0.985	0.972	0.984	0.975	0.987	**0.980**	**0.991**
	ResNet50	0.906	0.956	0.880	0.967	0.853	0.961	0.633	0.802	0.938	0.976	0.771	0.926	0.851	0.929	0.903	0.924	0.914	0.985	0.953	0.973	0.967	0.976	**0.975**	**0.997**
	WRN-50-2	0.909	0.957	0.928	0.980	0.921	0.977	0.652	0.793	0.966	0.984	0.931	0.967	0.931	0.966	0.915	0.970	0.914	0.983	0.965	0.990	0.951	0.978	**0.961**	**0.996**
	Average	0.933	0.974	0.907	0.973	0.814	0.969	0.623	0.777	**0.948**	**0.978**	0.817	0.949	0.894	0.951	0.913	0.969	0.918	0.984	0.963	0.982	0.964	0.980	**0.972**	**0.995**
Nonliving-26	ResNet18	0.906	0.955	0.781	0.925	0.739	0.909	0.543	0.810	0.854	0.939	0.914	0.980	0.958	0.981	0.939	0.978	0.929	0.989	**0.982**	**0.992**	0.970	0.989	0.978	0.991
	ResNet50	0.916	0.970	0.832	0.942	0.776	0.918	0.638	0.837	0.893	0.960	0.848	0.950	0.805	0.907	0.873	0.972	0.907	0.993	0.962	0.984	0.956	0.985	**0.975**	**0.995**
	WRN-50-2	0.917	0.977	0.932	0.971	0.912	0.959	0.676	0.861	0.945	0.969	0.885	0.942	0.893	0.939	0.924	0.973	0.909	0.991	0.962	0.979	0.960	0.988	**0.978**	**0.992**
	Average	0.913	0.967	0.849	0.946	0.809	0.929	0.618	0.836	0.897	0.956	0.882	0.957	0.913	0.974	0.886	0.943	0.915	0.991	0.969	0.985	0.962	0.987	**0.977**	**0.992**

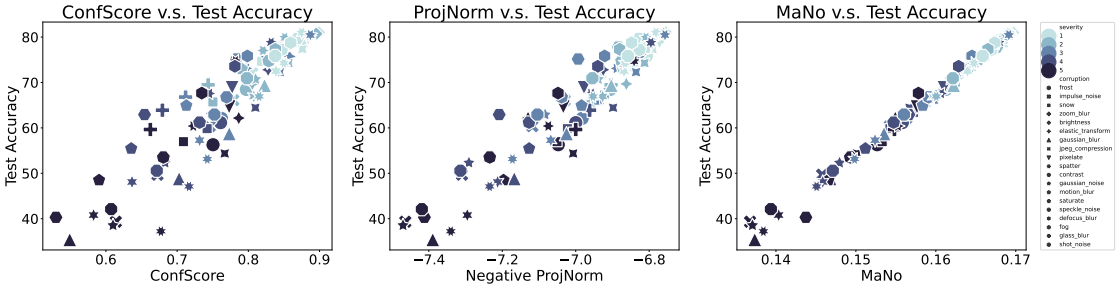


FIGURE 6.3: OOD error prediction versus True OOD error on Entity-13 with ResNet18. This scatter plot compares the performance of MANO with Dispersion Score and ProjNorm. Each point represents one dataset under a specific type and severity of corruption. Different shapes indicate different types of corruption, while darker colors indicate higher severity levels.

MaNo with an appropriate normalization technique significantly boosts estimation performance under the natural shift. Table 6.3 illustrates the results of accuracy estimation under the natural shift on a total of four datasets. Since calibration performance affected by the natural shift is more complex than the other distribution shifts, we balance ground-truth logits and error accumulation via the value of $\Phi(\mathcal{D}_{\text{test}})$. From Table 6.3, we observe a significant improvement compared with the other benchmarks. For instance, our method achieves the best performance on all four datasets on average. To visualize the estimation performance, we provide the scatter plots for *Dispersion Score*, *ProjNorm* and MANO in Figure 6.3 on Entity-18 with ResNet18. We find that MANO presents a robust linear relationship between the designed scores and gound-truth OOD errors, while the other state-of-the-art baselines tend to exhibit a biased estimation of high test errors.

TABLE 6.3: Method comparison on four benchmarks with ResNet18, ResNet50 and WRN-50-2 under **natural shift** with R^2 and ρ metrics (the higher the better). The best results are highlighted in **bold**.

		Natural Shift																					
Dataset	Network	Rotation		ConfScore		Entropy		AgreeScore		ATC		Fréchet		Dispersion		ProjNorm		MDE		COT		Nuclear		MANO	
		R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ
PACS	ResNet18	0.822	0.895	0.594	0.755	0.624	0.755	0.613	0.832	0.514	0.650	0.624	0.804	0.832	0.825	0.161	0.419	0.003	0.153	0.790	0.783	0.609	0.874	**0.827**	**0.909**
	ResNet50	0.860	0.923	0.070	0.069	0.061	0.055	0.463	0.622	0.192	0.265	0.463	0.622	0.073	0.167	0.244	0.587	0.059	0.104	0.891	0.790	0.611	0.888	**0.923**	**0.958**
	WRN-50-2	0.865	0.902	0.646	0.678	0.629	0.671	0.377	0.858	0.752	0.832	0.558	0.832	0.111	0.167	0.474	0.650	0.072	0.244	0.890	0.888	0.607	0.867	**0.924**	**0.972**
	Average	0.849	0.906	0.437	0.501	0.438	0.494	0.488	0.770	0.486	0.582	0.548	0.337	0.338	0.275	0.293	0.552	0.045	0.065	0.857	0.820	0.609	0.876	**0.891**	**0.946**
Office-Home	ResNet18	0.822	0.930	0.795	0.909	0.761	0.881	0.054	0.146	0.571	0.615	0.605	0.755	0.453	0.664	0.064	0.202	0.331	0.650	0.863	0.874	0.692	0.783	**0.926**	**0.930**
	ResNet50	**0.851**	**0.944**	0.769	0.895	0.742	0.853	0.026	0.216	0.487	0.734	0.607	0.685	0.383	0.727	0.169	0.475	0.342	0.622	0.762	0.846	0.731	0.895	0.838	0.916
	WRN-50-2	0.823	**0.958**	0.741	0.874	0.696	0.846	0.132	0.405	0.383	0.643	0.589	0.706	0.456	0.713	0.172	0.531	0.342	0.650	**0.863**	0.874	0.766	0.874	0.800	0.895
	Average	0.832	**0.944**	0.768	0.892	0.733	0.860	0.071	0.256	0.480	0.664	0.601	0.715	0.431	0.702	0.135	0.403	0.339	0.650	0.781	0.855	0.730	0.850	**0.854**	0.913
DomainNet	ResNet18	0.568	0.692	0.670	0.736	0.423	0.609	0.326	0.668	0.429	0.597	0.704	0.903	0.202	0.516	0.219	0.443	0.358	0.445	0.897	0.910	0.758	0.789	**0.902**	**0.937**
	ResNet50	0.588	0.703	0.570	0.706	0.344	0.573	0.455	0.697	0.245	0.404	0.746	0.872	0.002	0.041	0.220	0.430	0.379	0.527	0.903	0.927	0.809	0.879	**0.910**	**0.950**
	WRN-50-2	0.609	0.712	0.774	0.874	0.711	0.845	0.437	0.698	0.846	0.918	0.585	0.831	0.003	0.034	0.363	0.466	0.520	0.713	0.885	0.935	0.850	0.911	**0.893**	**0.978**
	Average	0.588	0.702	0.671	0.722	0.493	0.676	0.406	0.688	0.507	0.639	0.678	0.869	0.069	0.197	0.234	0.446	0.419	0.562	0.894	0.919	0.805	0.895	**0.899**	**0.949**
RR1-WILDS	ResNet18	0.821	1.000	0.951	1.000	0.836	1.000	0.929	1.000	0.342	0.500	0.936	1.000	0.843	1.000	0.859	1.000	0.927	1.000	0.969	1.000	0.885	1.000	**0.983**	**1.000**
	ResNet50	0.740	1.000	0.918	1.000	0.819	1.000	0.938	1.000	**0.986**	**1.000**	0.935	1.000	0.737	1.000	0.867	1.000	0.938	1.000	0.960	1.000	0.906	1.000	0.978	**1.000**
	WRN-50-2	0.031	0.500	0.941	1.000	0.846	1.000	0.946	1.000	**0.988**	**1.000**	0.922	1.000	0.824	1.000	0.878	1.000	0.954	1.000	0.934	1.000	0.840	1.000	0.969	**1.000**
	Average	0.530	0.833	0.937	1.000	0.833	1.000	0.938	1.000	0.779	0.833	0.931	1.000	0.801	0.833	0.868	1.000	0.940	1.000	0.953	1.000	0.877	1.000	**0.977**	**1.000**

Robustness enhancement achieved by MaNo. Figure 6.4 presents a box plot showing the estimation robustness across different distribution shifts on all datasets except ImageNet, using ResNet18. Results for ImageNet are excluded due to the lack of *Rotation* and *AgreeScore* data for this dataset, as these two methods require retraining the networks. We observe that the estimation performance of MANO is more stable than other baselines across three types of distribution shifts. Additionally, MANO achieves the highest median estimation performance.

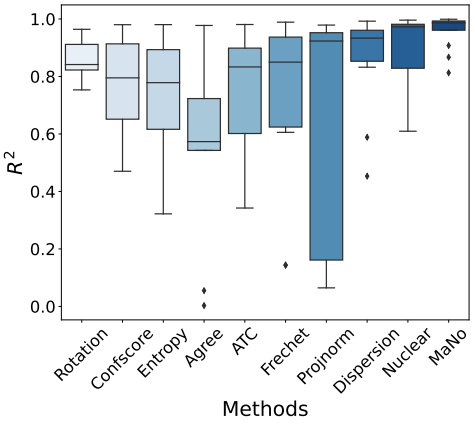


FIGURE 6.4: Robustness comparison between MANO and other baselines across three types of distribution shifts using ResNet18. MANO achieves the best and most robust overall estimations.

6.5.3 Discussion

Generalization capabilities of MaNo. To verify the generalization capabilities of MANO, we utilize designed scores calculated from ImageNet-C and their corresponding accuracy to fit a linear regression model. This model is then used to predict the test accuracy on ImageNet-V2-$\bar{C}$, which is generated using the 10 new corruptions provided by [6] on ImageNet-V2 [177]. These new corruptions are perceptually dissimilar from those in ImageNet-C, including warps, blurs, color

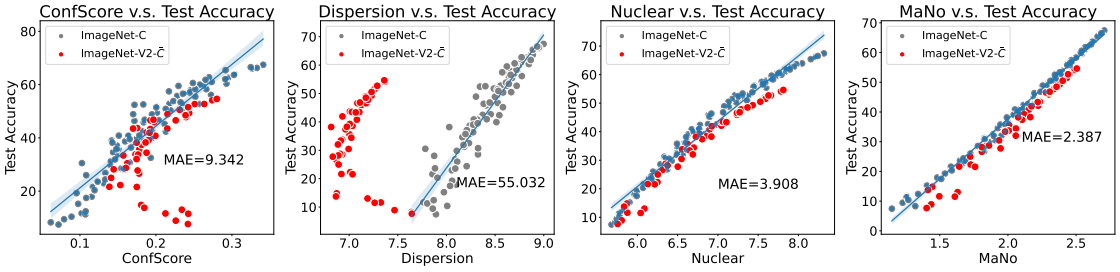


FIGURE 6.5: Comparison of generalization capability across four methods. Each subplot displays a linear regression model fitted on ImageNet-C, which is used to predict the accuracy on ImageNet-V2-$\bar{C}$. The mean absolute error (MAE) is reported. All experiments are conducted using ResNet18.

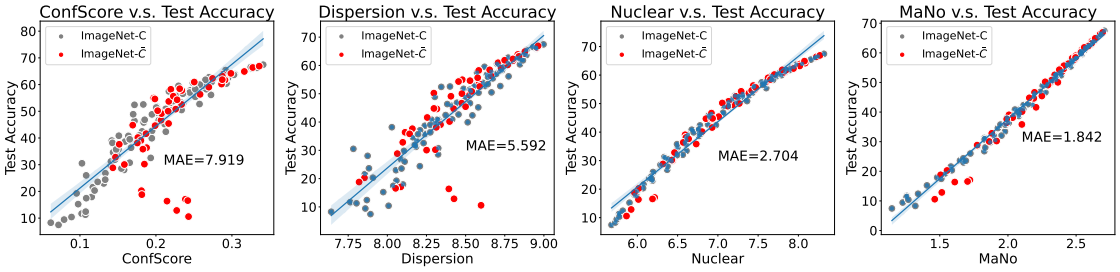
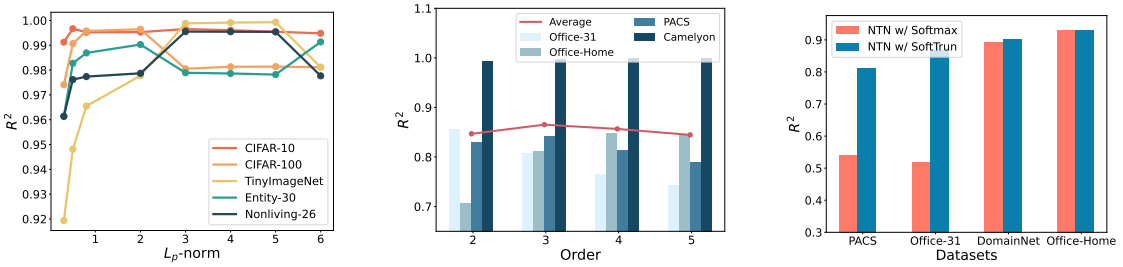


FIGURE 6.6: Comparison of generalization capability across four methods. Each subplot shows a linear regression model fitted on ImageNet-C to predict accuracy on ImageNet-$\bar{C}$. Mean absolute error (MAE) is calculated on ImageNet-$\bar{C}$ [6]. All experiments use ResNet18.

distortions, noise additions, and obscuring effects. Figure 6.5 shows that ConfScore and Dispersion give two distinct trends, while Nuclear exhibits some deviations for ImageNet-V2-$\bar{C}$. In comparison, our MANO exhibits a consistent prediction pattern for both ImageNet-C and ImageNet-V2-$\bar{C}$, aligning well with the linear regression model trained on ImageNet-C. Additionally, experimental results on ImageNet-C and ImageNet-$\bar{C}$, generated from the validation set of ImageNet, are provided in Figure 6.6, further demonstrating the superiority of MANO.

Can `SoftTrun` enhance other logit-based methods? To study this, we conducted an ablation study by equipping `SoftTrun` with *Nuclear* [5], *ConfScore* [24], and our MANO. In Table 6.4, we observe that `SoftTrun` significantly enhances the estimation performance R^2 of Nuclear. For example, Nuclear is improved from 0.692 to 0.826 on poorly-calibrated Office-Home.

Limitations. Despite its soundness and strong empirical performance, our method has potential areas for improvement. One of these is the dependence on η



(A) Impact of the L_p norm.

(B) Approximation order n.

(C) Benefits of `SoftTrun`.

FIGURE 6.7: Sensitivity analysis with Resnet18. **(a)** Effect of the L_p norm types. **(b)** Impact of the Taylor approximation order, *i.e.*, the number of terms in Eq 6.4. For instance, an order of 3 means that 3 terms are taken, which corresponds to the default setting in Eq. equation 6.6 and is used in all our experiments. **(c)** Type of normalization function.

in Eq. equation 6.6. In future work, we will explore a smoother way to automatically select the optimal normalization function without requiring hyperparameters. Additionally, if multiple validation sets are provided, as in [23, 28], we could select η based on those sets.

6.6 Sensitivity Analysis and Ablation Study

6.6.1 Choice of L_p norm.

To reflect the impact of different L_p norms on estimation performance, we conduct a sensitivity study on 5 datasets with ResNet18, whose results are shown in Figure 6.7a. The performance for $p = 1$ is ignored as in this case, $\|\mathbf{Q}\|_1 = 1$ because $\mathbf{Q}$ is right-stochastic. We can see that when we choose $p \in [2, 5]$, the results fluctuate within a satisfying range. This can be explained by the fact that within this range, we emphasize adequately the large positive feature-to-boundary distances without ignoring the other comparatively small distances.

TABLE 6.4: Effect of `SoftTrun` on other logit-based methods. `SoftTrun` significantly boosts the performance of Nuclear. The metric used is R^2.

Dataset	ConfScore		MaNo		Nuclear	
	w/o	w/	w/o	w/	w/o	w/
PACS	**0.594**	0.574	0.541	**0.827**	0.609	**0.851**
Office-Home	0.795	**0.829**	**0.929**	0.926	0.692	**0.826**

6.6.2 Choice of Taylor approximation order.

In Figure 6.7b, we verify the impact of Taylor formula approximation on final accuracy estimation performance. It should be noted that for orders higher than in the default setting in Eq. equation 6.6, the positivity is lost. To alleviate this issue, we consistently remove for all orders the minimum value of the obtained vector to each of its entries to ensure having an output in $\mathbb{R}_+^K$. This extends de Brébisson and Vincent [178] to orders higher than 2. From this figure, we can see that when we reserve the first three terms in the Taylor formula, the average estimation performance is optimal. For well-calibrated datasets such as Office-Home and WILDS, there exists an increased trend of estimation performance when we reserve more Taylor formula terms. As for suboptimal-calibrated datasets such as PACS and Office-31, their performance rises when fewer terms are reserved. It empirically certifies that the normalization technique is a trade-off tool between the ground-truth logits' information and error accumulation. In addition, the optimal choice is to keep 3 terms in Eq. equation 6.4 which motivates our default setting in Eq equation 6.6.

6.6.3 Superiority of `SoftTrun`.

To verify the effectiveness of our proposed normalization technique, `SoftTrun`, we conduct an ablation study by replacing our normalization with the softmax function under the natural shift. In Figure 6.7c, we observe that our proposed normalization significantly enhances the estimation performance of datasets from the natural shift. Especially, R^2 for poorly-calibrated datasets such as Office-31 is improved from 0.51 to 0.86.

To further demonstrate the generalization capability of MaNo, we provide a similar experiment with that in Section 6.5.3 on ImageNet-C and ImageNet-$\bar{C}$ [6] in Figure 6.6. In particular, we fit a linear regression function on ImageNet-C and use the linear function to predict the accuracy of ImageNet-$\bar{C}$. This figure shows that MaNo has better estimation performance than the other baselines when meeting different corruption types.

6.7 Chapter Summary

In this chapter, we introduce MaNo, a simple yet effective training-free method to estimate test accuracy in an unsupervised manner using the Matrix Norm of neural network predictions on test data. Our approach is inspired by the LDS assumption that optimal decision boundaries should lie in low-density regions. To mitigate the negative impact of different distribution shifts on estimation performance, we first demonstrate the failure of `softmax` normalization under poor calibration, due to the accumulation of overconfident errors. We then propose a normalization strategy based on Taylor polynomial approximation, balancing logits information and error accumulation. Extensive experiments show that MaNo consistently outperforms previous methods across various distribution shifts. This work highlights that logits imply the feature-to-boundary distance and considers the impact of calibration on estimation performance. We hope our insights inspire future research to further explore the relationship between model outputs and generalization.

Chapter 7

Conclusion and Future Directions

7.1 Conclusions

Generalization in deep learning is an increasingly important topic for AI deployment safety in the real world, which has obtained a surge of attention recently. In this thesis, we explored two advanced topics in the generalization of deep learning, i.e., generalization performance enhancement and generalization performance estimation. In the following, we conclude the designed algorithms presented in previous chapters.

In Chapter 3, we focus on generalization performance enhancement, and propose a universal and effective approach called GearNet to enhance many existing robust training methods under the existence of noisy labels and distribution shifts. It is the first to explore the benefit of pseudo-supervision knowledge from the target domain in improving the robustness capability against both noisy labels from the source domain and distribution shifts between the source and the target domain. In this chapter, we demonstrate that usually overlooked target-domain information benefits in enhancing generalization performance even under the attack of noisy labels.

From Chapter 4 to Chapter 6, we explore the task of generalization performance estimation, where we need to predict the ground-truth accuracy of test sets in an unsupervised manner. The three chapters demonstrate the strong linear relationship between feature separability, gradients as well as low-density separation and true accuracy during testing.

In Chapter 4, we propose a simple but effective indicator called Dispersion score for predicting the generalization performance on OOD in an unsupervised manner. We show that Dispersion score is strongly correlated to the OOD error and achieves consistently better performance than previous methods under various distribution shifts. Even when the OOD datasets are class imbalanced or have limited number of instances, our method maintains a high prediction performance, which demonstrates the strong flexibility of Dispersion score. Experimental results demonstrate the effectiveness of Dispersion score in unsupervised accuracy estimation.

In Chapter 5, we explore the strong linear relationship between the magnitude of gradients and the model performance under distribution shifts. Based on the observation, we proposed a simple yet efficient method called GdScore to estimate the ground-truth test accuracy by measuring the gradient magnitude of the last classification layer. Our method performs more consistently across diverse types of distribution shifts compared with the state-of-the-art methods.

In Chapter 6, we provide an efficient training-free method called MaNo to estimate test accuracy under distribution shifts. It addresses two questions: what explains the correlation between logits and test accuracy, and how to alleviate the overconfidence issue of logit-based methods. This method is inspired by the LDS assumption that optimal decision boundaries should lie in low-density regions. We propose a normalization strategy based on Taylor polynomial approximation to balance logits information and error accumulation. Then we quantify the density around decision boundaries via the matrix norm of logits. Extensive experiments show that MaNo consistently outperforms previous methods across various distribution shifts.

The three chapters demonstrate the strong correlation between hidden features, gradients, logits and ground-truth accuracy. On the one hand, those approaches provide an unsupervised way to estimate models' generalization performance, which can be used for finding or annotating underperformed datasets. On the other hand, those works inspire the further exploration about understanding the arcanum of generalization.

7.2 Future Directions

In this section, we provide several potentially interesting directions of generalization in machine learning for future exploration.

7.2.1 Fully test time adaptation

In practice, data from the source domain are usually infeasible due to confidential requirement. Therefore, it is important for a model to adapt itself to new and different datasets during the test process, which is known as *fully test time adaptation*. Thus, in the future work, we will also focus on this field to enhance the generalization capability of models via test-time fine-tuning.

7.2.2 Generalization problems in foundation models

Foundation models such as GPT-3 [179], BERT [180], and CLIP [181] have obtained increasing attention in the deep learning community. Since they are trained on broad data rather than task-specific data, their generalization capability has been significantly improved. However, measuring the task-specific accuracy of foundation models across different domain tasks is also very interesting and increasingly popular [182]. Thus, in the future, we will explore more generalization problem settings under foundation models.

7.2.3 Theoretical understanding of generalization capability

In Chapter 4, 5, and 6, we have demonstrated several necessary conditions of distribution shifts. When there exist a distribution shift between the training and the test data, their hidden feature separability, magnitude of gradients as well as L_p norm of the logit matrix vary linearly with the severity of distribution shifts. However, there is no research work to explore the sufficient conditions as well as sufficient and necessary conditions of distribution shifts, which implies the limitation of understanding the generalization capability of deep learning. Thus, in the future, we need to construct a solid theoretical framework with empirical evidence to explain the generalization performance of different models.

Appendix A

Appendix for Chapter 4

A.1 The Calculation of Dispersion Score

Our proposed approach, Dispersion Score, can be calculated as shown in Algorithm 2.

Algorithm 2: OOD Error Estimation via Dispersion Score

Input: OOD test dataset $\tilde{\mathcal{D}} = \{\tilde{x}_i\}_{i=1}^{m}$, a trained model f (feature extractor f_g and classifier f_ω)
Output: The dispersion score
for each OOD instance $\tilde{x}_i$ **do**
  Obtain feature representation via $\tilde{\boldsymbol{z}}_i = f_g(\tilde{\boldsymbol{x}}_i)$.
  Obtain pseudo labels via $\tilde{y}_i' = \arg\max f_\omega(\boldsymbol{z}_i)$
end for
Calculate cluster centroids $\{\tilde{\boldsymbol{\mu}}_j\}_{j=1}^{k}$ using pseudo labels $\tilde{y}_i'$, with $\tilde{\boldsymbol{\mu}}_j = \frac{1}{m_j}\sum_{i=1}^{m_j} \boldsymbol{z}_i \cdot \mathbb{1}\{\tilde{y}_i' = j\}$
Calculate the feature center of all instances by $\bar{\boldsymbol{\mu}} = \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{z}_i$
Calculate Dispersion Score $S(\tilde{\mathcal{D}})$ via Equation (5.8)

A.2 Related Work

Predicting generalization. Since the generalization capability of deep networks under distribution shifts is a mysterious desideratum, a surge of researches pay attention to estimate the generalization capability from two directions.

1) Some works aim to measure generalization gap between training and test accuracy with only training data [70–75]. For example, the model-architecture-based method [70] summarizes the persistent topological map of a trained model to formulate its inner-working function, which represents the generalization gap. Margin distribution [71] measures the gap by gauging the distance between training examples and the decision boundary. However, those methods are designed for the identical distribution between the training and test dataset, being vulnerable to distribution shift.

2) Some studies try to estimate generalization performance on a specific OOD test dataset without annotation during evaluation. Many of them utilize softmax outputs of the shifted test dataset to form a quantitative indicator of OOD error [25, 25–27]. However, those methods are unreliable across diverse distribution shifts due to the overconfidence problem [111]. Another popular direction considers the negative correlation between distribution difference and model's performance in the space of features [28] or parameters [7]. Nevertheless, common distribution distances practically fail to induce stable error estimation under distribution shift [25], and those methods are usually computationally expensive. Unsupervised loss such as agreement among multiple classifiers [26, 77–79] and data augmentation [23] is also employed for OOD error prediction, which requires specific model structures during training. In this work, we focus on exploring the connection between feature separability and generalization performance under distribution shift, which is training-free and does not have extra requirements for datasets and model architectures.

Exploring Feature distribution in deep learning. In the literature, feature distribution has been widely studied in domain adaptation [59, 60, 64, 183], representation learning [112, 184–186], OOD generalization [187–189], and noisy-label learning [190, 191]. Domain adaptation methods usually learn a domain-invariant feature representation by narrowing the distribution distance between the two domains with certain criteria, such as maxinum mean discrepancy (MMD) [59], Kullback-Leibler (KL) divergence [60], central moment discrepancy (CMD) [61], and Wasserstein distance [62]. InfoNCE [186] shows a key factor of contrastive learning that the distance between class centers should be large enough. In learning with noisy labels, it has been shown that the feature clusterability can be used to estimate the transition matrix [190]. To the best of our knowledge, we are the first

to analyze the connection between feature separability and the final accuracy on OOD data.

Appendix B

Appendix for Chapter 5

B.1 Related Work

Unsupervised accuracy estimation. Unsupervised accuracy estimation is a vital topic in practical applications due to frequent distribution shifts and the unavailability of ground-truth labels for test samples. To comprehensively understand this field, we introduce two main existing settings which are related to this topic.

1. Some works aim to estimate the test accuracy or gauge the accuracy discrepancy between the training and the test set only via the training data [70–75]. For example, the model-architecture-based algorithm [70] derives plenty of persistent topology properties from the training data, which can identify when the model learns to generalize to unseen datasets. However, those algorithms are deployed under the assumption that the training and the test data are drawn from the same distribution, which means they are vulnerable to distribution shifts.

2. Our work belongs to the second setting, which aims to estimate the classification accuracy of a specific test dataset during evaluation using unlabeled test samples and/or labeled training datasets. The main research direction is to explore the negative relationship between the distribution discrepancy and model performance from the space of features [28], parameters [7] and labels [76]. Another popular direction is to design an estimation score via the

softmax outputs of the test samples [25, 25–27], which heavily relies on model calibration. Some works also learn from the field of unsupervised learning, such as agreement across multiple classifiers [26, 77–79] and image rotation [23]. In addition, the property of the test datasets presented during evaluation has also been studied recently [29]. To the best of our knowledge, our work is the first to study the linear relationship between the gradients and model performance.

Gradients in generalization. The role of gradients in generalization has attracted increasing attention recently. To gauge the generalization performance of the hypothesis learned from the training data on unseen samples, known as the out-of-sample error [192–194], many studies try to provide a tight upper bound for generalization error from the view of gradient descent theoretically, indicating that gradients correlate with the discrepancy between the empirical loss and the population loss [133, 134, 195, 196]. However, those works assume that seen to unseen data are from the identical distribution, while unsupervised accuracy estimation discusses a more complex and realistic issue that they come from different distributions. Under distribution shift, gradients are also explored. For example, Mansilla et al. [137] clips the conflicting gradients emerging in domain shift scenarios and promotes gradient agreement across multiple domains via the gradient agreement strategy. Similarly, Zhao et al. [139] designs a gradient-based regularization term to make the optimizer find the flat minima. In out-of-distribution (OOD) detection which goal is to determine whether a given sample is in-distribution (ID) or out-of-distribution [18, 24, 197–199], [1] finds that ID data usually have higher gradient magnitude than OOD data from current source distribution to a uniform distribution. However, despite their empirical success, the relationship between gradients and generalization is still unclear.

B.2 Pseudo-code of GdScore

Our proposed GdScore for unsupervised accuracy estimation can be calculated as shown in Algorithm 3.

Algorithm 3: Unsupervised Accuracy Estimation via GDSCORE

Input: Test dataset from unseen domains $\tilde{\mathcal{D}} = \{\tilde{\mathbf{x}}_i\}_{i=1}^m$, a pre-trained model $f_{\boldsymbol{\theta}} = f_{\mathbf{g}} \circ f_{\boldsymbol{\omega}}$ (feature extractor $f_{\mathbf{g}}$ and classifier $f_{\boldsymbol{\omega}}$), a threshold value τ.
Output: The GDSCORE.
for each test instance $\tilde{\mathbf{x}}_i$ **do**
  Obtain the maximum softmax probability via $\tilde{r}_i = \max_k \mathrm{s}_{\boldsymbol{\omega}}^{(k)}(f_{\mathbf{g}}(\tilde{\mathbf{x}}_i))$.
  if $\tilde{r}_i > \tau$ **then**
    Obtain pseudo labels via $\tilde{y}_i' = \arg\max_k f_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_i)$,
  else
    Obtain random labels via $\tilde{y}_i' \sim U[1, K]$.
  end if
end for
Calculate the cross-entropy loss using assigned labels $\tilde{y}_i$ via Eq. 5.6.
Calculate gradients of the weights in the classification layer via Eq. 5.7.
Calculate GDSCORE $S(\tilde{\mathcal{D}})$ via Eq. 5.8.

B.3 Baselines

Rotation. [23] By rotating images from both the training and the test sets with different angles, we can obtain new inputs and their corresponding labels y_i^r which indicate by how many degrees they rotate. During pre-training, an additional classifier about rotation degrees should be learned. Then the *Rotation Prediction* (Rotation) metric can be calculated as:

$$S_r(\mathcal{D}_{\text{test}}) = \frac{1}{m}\sum_{i=1}^{m}(\frac{1}{4}\sum_{r\in\{0°,90°,180°,270°\}}(\mathbb{1}(\hat{y}_i^r \neq y_i^r))),$$

where $\hat{y}_i^r$ denotes the predicted labels about rotation degrees.

ConfScore. [24] This metric directly leverages the average maximum softmax probability as the estimation of the test error, which is expressed as:

$$S_{cf}(\mathcal{D}_{\text{test}}) = \frac{1}{m}\sum_{i=1}^{m}\max(\mathrm{s}_{\boldsymbol{\omega}}(f_{\boldsymbol{g}}(\tilde{\mathbf{x}}_i))).$$

Entropy. [25] This metric estimates the test error via the average entropy loss:

$$S_e(\mathcal{D}_{\text{test}}) = \frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}\mathrm{s}_{\boldsymbol{\omega}}^{(k)}(f_{\boldsymbol{g}}(\tilde{\mathbf{x}}_i))\log\mathrm{s}_{\boldsymbol{\omega}}^{(k)}(f_{\boldsymbol{g}}(\tilde{\mathbf{x}}_i)).$$

AgreeScore. [26] This method trains two independent neural networks simultaneously during pre-training, and estimates the test error via the rate of disagreement across the two models:

$$S_{ag}(\mathcal{D}_{\text{test}}) = \frac{1}{m}\sum_{i=1}^{m} \mathbb{1}(\tilde{y}'_{1,i} \neq \tilde{y}'_{2,i}),$$

where $\tilde{y}'_{1,i}$ and $\tilde{y}'_{2,i}$ denote the predicted labels by the two models respectively.

ATC. [27] It measures how many test samples have a confidence larger than a threshold that is learned from the source distribution. It can be expressed as:

$$S_{atc}(\mathcal{D}_{\text{test}}) = \frac{1}{m}\sum_{i=1}^{m} \mathbb{1}(\sum_{k=1}^{K} \mathrm{s}_{\boldsymbol{\omega}}^{(k)}(f_{\boldsymbol{g}}(\tilde{\mathbf{x}}_i)) \log \mathrm{s}_{\boldsymbol{\omega}}^{(k)}(f_{\boldsymbol{g}}(\tilde{\mathbf{x}}_i)) < t),$$

where t is the threshold value learned from the validation set of the training dataset.

Fréchet. [28] This method utilizes Fréchet distance to measure the distribution gap between the training and the test datasets, which serves the test error estimation:

$$S_{fr}(\mathcal{D}_{\text{test}}) = ||\mu_{train} - \mu_{test}|| + Tr(\Sigma_{train} + \Sigma_{test} - 2(\Sigma_{train}\Sigma_{test})^{\frac{1}{2}}),$$

where μ_{train} and μ_{test} denote the mean feature vector of $\mathcal{D}$ and $\mathcal{D}_{test}$, respectively. Σ_{train} and Σ_{test} refer to the covariance matrices of corresponding datasets.

Dispersion. [29] This chapter estimates the test error by gauging the feature separability of the test dataset in the feature space:

$$S_{dis}(\mathcal{D}_{\text{test}}) = \log \frac{\sum_{k=1}^{K} m_k \cdot \|\bar{\boldsymbol{\mu}} - \tilde{\boldsymbol{\mu}}_k\|_2^2}{K-1},$$

where $\boldsymbol{\mu}$ denotes the center of the whole features, and $\boldsymbol{\mu}_k$ denotes the mean of k^{th}-class features.

TABLE B.1: Method property summary including whether this method belongs to self-training or training-free approaches, and if this method requires training data or specific model architectures.

Method	Self-training	Training-free	Training-data-free	Architecture-requirement-free
Rotation	✗	✓	✓	✗
ConfScore	✗	✓	✓	✓
Entropy	✗	✓	✓	✓
Agreement	✗	✓	✓	✗
ATC	✗	✓	✗	✓
Fréchet	✗	✓	✗	✓
Dispersion	✗	✓	✓	✓
Nuclear	✗	✓	✓	✓
ProjNorm	✓	✗	✓	✓
Ours	✓	✗	✓	✓

Nuclear Norm. [5] This chapter estimates the test error by computing the nuclear norm of final softmax probabilities, which can be expressed as:

$$S_{nu}(\mathcal{D}_{\text{test}}) = ||P||_*,$$

where Nuclear norm $||P||_*$ is defined as the sum of singular values of P.

ProjNorm. [7] This method fine-tunes the pre-trained model on the test dataset with pseudo-labels, and measures the distribution discrepancy between the training and the test datasets in the parameter level:

$$S_{pro}(\mathcal{D}_{\text{test}}) = ||\tilde{\boldsymbol{\theta}}_{ref} - \tilde{\boldsymbol{\theta}}||_2,$$

where $\boldsymbol{\theta}_{ref}$ denotes the parameters of the pre-trained model, while $\boldsymbol{\theta}$ denotes the parameters of the fine-tuned model.

Those algorithms mentioned in this chapter can be summarized as Table B.1.

B.4 Formulation of the Entropy Loss for Low-confidence Samples

In the ablation study, we explore the impact of loss selection on the performance of unsupervised accuracy estimation. In particular, the detail about the entropy loss for samples with low confidence is expressed as follows: In particular, the entropy loss can be expressed as follows:

$$\begin{aligned}\mathcal{L}(f_{\boldsymbol{\theta}}(\tilde{\mathbf{x}})) = &-\frac{1}{m_1}\sum_{i=1}^{m_1}\sum_{k=1}^{K}\tilde{y}_{i,con>\tau}^{(k)}\log \mathrm{s}_{\boldsymbol{\omega}}^{(k)}(f_{\boldsymbol{g}}(\tilde{x}_i^{con>\tau})) \\ &-\frac{1}{m_2}\sum_{i=1}^{m_2}\sum_{k=1}^{K}\mathrm{s}_{\boldsymbol{\omega}}^{(k)}(f_{\boldsymbol{g}}(\tilde{\mathbf{x}}_i^{con\leq\tau}))\log \mathrm{s}_{\boldsymbol{\omega}}^{(k)}(f_{\boldsymbol{g}}(\tilde{\mathbf{x}}_i^{con\leq\tau})),\end{aligned}$$

where the first term denotes the cross-entropy loss calculated for samples with confidence larger than the threshold value τ, the second term denotes the entropy loss for samples with lower confidence than τ, and *con* means the sample confidence, m_1 and m_2 denote the total number of samples with higher confidence and lower confidence than τ, respectively.

B.5 Comparison to Nuclear Norm

Here, we compare our method to Nuclear Norm [5] across 3 types of distribution shifts with ResNet18, ResNet50, and WRN-50-. Results are shown in Table B.2. From this table, we observe our method outperforms Nuclear Norm under synthetic shift and natural shift.

B.6 Connection to Huang et al. [1]

A current work, GradNorm [1], employs gradients to detect OOD samples whose labels belong to a different label space from the training data. It gauges the magnitude of gradients in the classification layer, backpropagated from a KL divergence between the softmax probability and uniform distribution. Compared with GradNorm, our method bears three critical differences, in terms of the problem

TABLE B.2: Performance comparison on 11 benchmark datasets with ResNet18, ResNet50 and WRN-50-2, where R^2 refers to coefficients of determination, and ρ refers to the absolute value of Spearman correlation coefficients (higher is better). The best results are highlighted in **bold**.

Method	Network	CIFAR 100		TinyImageNet		Office-Home		Camelyon17		Entity-13		Entity-30	
		R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ
Nuclear	ResNet18	**0.989**	0.995	**0.983**	**0.994**	0.692	0.783	0.858	**1.000**	**0.978**	**0.991**	**0.980**	0.993
	ResNet50	0.979	**0.994**	0.965	0.994	0.731	0.895	0.849	**1.000**	**0.989**	**0.996**	**0.978**	0.994
	WRN-50-2	0.962	0.988	0.956	0.992	0.766	0.874	0.983	**1.000**	**0.989**	**0.995**	**0.985**	**0.996**
	Average	0.977	0.993	0.968	0.993	0.730	0.850	0.916	**1.000**	**0.989**	**0.995**	**0.989**	**0.995**
Ours	ResNet18	0.987	**0.996**	0.971	**0.994**	**0.876**	**0.909**	**0.996**	**1.000**	0.969	**0.991**	0.970	**0.995**
	ResNet50	**0.991**	**0.994**	**0.980**	**0.995**	**0.829**	**0.944**	**0.999**	**1.000**	0.960	0.995	0.957	**0.996**
	WRNt-50-2	**0.995**	**0.998**	**0.975**	**0.996**	**0.809**	**0.916**	**0.997**	**1.000**	0.968	**0.995**	0.949	0.994
	Average	**0.991**	**0.997**	**0.976**	**0.995**	**0.837**	**0.923**	**0.998**	**1.000**	0.966	0.994	0.959	0.995

setting, methodology, and theoretical insights. We also empirically demonstrate the superiority of our method in Table B.4.

1) *Problem setting*: GradNorm focuses on OOD detection, which aims to determine whether a given sample is in-distribution (ID) or out-of-distribution [18, 24, 197–199], while our method aims to estimate the test accuracy without ground-truth test labels. It requires GradNorm should be an instance-level score for classification, but our method is a dataset-level score for linear regression. Furthermore, in OOD detection, the label spaces of OOD data and training data are disjoint, while in unsupervised test accuracy, the training and the OOD label spaces are shared. The two are also evaluated differently: AUROC score for OOD detection and correlation coefficients for error estimation. Those differences are summarized in Table B.3.

TABLE B.3: Main differences between OOD detection and OOD error estimation.

Learning Problem	Goal	Scope	Metric
OOD detection	Predict ID/OOD	$\tilde{\mathbf{x}}_i$	AUROC
OOD error estimation	Proxy to test error	$\mathcal{D}_{\text{test}}$	R^2 and ρ

2) *Methodology*: GradNorm obtains the magnitude of gradients via a KL-divergence loss measuring the distribution distance from the training distribution to a uniform distribution, while our method obtains them using a standard cross entropy loss with the specifically designed label generation strategy. GradNorm assumes that OOD data should have a lower magnitude of gradients, but in our cases, test data are certified to have a higher magnitude of gradients. Table B.4 presents the performance comparison of the two methods in unsupervised accuracy estimation

on 7 datasets across 3 types of distribution shifts with ResNet18. It illustrates that GradNorm is inferior to our approach for unsupervised accuracy estimation suggesting that the two problems cannot be tackled with the same tools.

3) *Theoretical insights*: GradNorm is certified to capture the joint information between features and outputs to detect OOD data from the oncoming dataset. However, the reason why OOD data have a lower magnitude of gradients is still unclear. Our method provides clearer theoretical insights to demonstrate the relationship between gradients and test accuracy even under distribution shift, which further inspires future work to address generalization issues from the view of gradients

TABLE B.4: Performance comparison between Huang et al. [1] and our methods on 7 datasets with ResNet18. The metric used in this table is the coefficient of determination R^2. The best results are highlighted in **bold**.

Method	CIFAR 10	CIFAR 100	TinyImageNet	Office-31	Office-Home	Entity-30	Living-17
[1]	0.951	0.978	0.894	0.596	0.848	0.964	0.942
Ours	**0.972**	**0.983**	**0.971**	**0.675**	**0.876**	**0.970**	**0.949**

B.7 Proofs

B.7.1 Proof of Theorem 5.3.1

We start by proving the following lemma.

Lemma B.7.1. For any convex function $f : \mathbb{R}^D \to \mathbb{R}$ and any $p, q \geq 1$ such that $\frac{1}{p} + \frac{1}{q} = 1$, we have:

$$\forall \mathbf{a}, \mathbf{b} \in \mathrm{dom}(f), \quad |f(\mathbf{a}) - f(\mathbf{b})| \leq \max_{\mathbf{c} \in \{\mathbf{a}, \mathbf{b}\}} \{\|\nabla f(\mathbf{c})\|_p\} \cdot \|\mathbf{a} - \mathbf{b}\|_q .$$

Proof. Using the fact that f is convex, we have:

$$\begin{aligned} f(\mathbf{a}) - f(\mathbf{b}) &\leq \langle \nabla f(\mathbf{a}), \mathbf{a} - \mathbf{b} \rangle \\ &\leq |\langle \nabla f(\mathbf{a}), \mathbf{a} - \mathbf{b} \rangle| \\ &\leq \sum_{i=1}^{p} |\nabla f(\mathbf{a})_i \, (\mathbf{a}_i - \mathbf{b}_i)| \\ &\leq \|\nabla f(\mathbf{a})\|_p \|\mathbf{a} - \mathbf{b}\|_q, \end{aligned}$$

where we used Hölder's inequality for the last inequality. The same argument gives:

$$f(\mathbf{b}) - f(\mathbf{a}) \leq \|\nabla f(\mathbf{b})\|_p \|\mathbf{b} - \mathbf{a}\|_q.$$

Using the absolute value, we can combine the two previous results and obtain the desired inequality. □

The proof of Theorem 5.3.1 follows from applying Lemma B.7.1 to the convex function $\mathcal{L}_T$.

B.7.2 Proof of Theorem 5.3.2

Proof. The proof follows from Theorem 5.3.1 by noting that $\|\boldsymbol{\omega}_s - \mathbf{c}\|_q = \eta \|\nabla \mathcal{L}_T(\boldsymbol{\omega}_s)\|_q$. □

B.7.3 Proof of Theorem 5.3.3

We start by introducing some notations. We denote $\mathcal{L}_{\mathbf{x},y}$ the loss evaluated on a specific data-point $(\mathbf{x}, y) \sim P_T(\mathbf{x}, \mathbf{y})$. We can then decompose the expected loss as $\mathcal{L}_T = \mathbb{E}_{P_T(\mathbf{x},y)} \mathcal{L}_{\mathbf{x},y}$. It follows by linearity of the expectation that

$$\nabla \mathcal{L}_T = \mathbb{E}_{P_T(\mathbf{x},y)} \nabla \mathcal{L}_{\mathbf{x},y}.$$

Then, we prove the following lemma that gives the formulation of the gradient of the cross-entropy.

Lemma B.7.2. The gradient of the cross-entropy loss with respect to $\boldsymbol{\omega} = (\mathbf{w}_k)_{k=1}^K$ writes

$$\nabla \mathcal{L}_{\mathbf{x},y}(\boldsymbol{\omega}) = \left(-y^{(k)} \mathbf{x} (1 - \mathrm{s}_{\boldsymbol{\omega}}^{(k)}(\mathbf{x})\right)_{k=1}^K.$$

Proof. First, let's compute the partial derivative of the softmax w.r.t. $\mathbf{w}_k$ for any $k \in \{1, \dots, K\}$. We have:

$$\begin{aligned}\frac{\partial \mathrm{s}_{\boldsymbol{\omega}}^{(k)}(\mathbf{x})}{\partial \mathbf{w}_k} &= \frac{\mathbf{x} e^{\mathbf{w}_k^\top \mathbf{x}} \left(\sum_{\tilde{k}} e^{\mathbf{w}_{\tilde{k}}^\top \mathbf{x}} - e^{\mathbf{w}_k^\top \mathbf{x}}\right)}{\left(\sum_{\tilde{k}} e^{\mathbf{w}_{\tilde{k}}^\top \mathbf{x}}\right)^2} \\ &= \mathbf{x}\left(\mathrm{s}_{\boldsymbol{\omega}}^{(k)}(\mathbf{x}) - \left[\mathrm{s}_{\boldsymbol{\omega}}^{(k)}(\mathbf{x})\right]^2\right).\end{aligned}$$

Using the chain rule, the partial derivative of the loss w.r.t. $\mathbf{w}_k$ writes:

$$\begin{aligned}\frac{\partial \mathcal{L}_{\mathbf{x},y}(\boldsymbol{\omega})}{\partial \mathbf{w}_k} &= \frac{\partial \mathcal{L}_{\mathbf{x},y}(\boldsymbol{\omega})}{\partial s(\boldsymbol{\omega}, \mathbf{x})} \cdot \frac{\partial s(\boldsymbol{\omega}, \mathbf{x})}{\partial \mathbf{w}_k} \\ &= \begin{cases} -\frac{1}{\mathrm{s}_{\boldsymbol{\omega}}^{(k)}(\mathbf{x})} \cdot \mathbf{x}\left(\mathrm{s}_{\boldsymbol{\omega}}^{(k)}(\mathbf{x}) - \left[\mathrm{s}_{\boldsymbol{\omega}}^{(k)}(\mathbf{x})\right]^2\right), & \text{if } y^{(k)} = 1 \\ 0, & \text{otherwise} \end{cases} \\ &= -y^{(k)} \mathbf{x}\left(1 - \mathrm{s}_{\boldsymbol{\omega}}^{(k)}(\mathbf{x})\right)\end{aligned}$$

As the $\frac{\partial \mathcal{L}_{\mathbf{x},y}(\boldsymbol{\omega})}{\partial \mathbf{w}_k}$ are the coordinates of $\nabla \mathcal{L}_{\mathbf{x},y}(\boldsymbol{\omega})$, we obtain the desired formulation. □

We now proceed to the proof of Theorem 5.3.3.

Proof. Using the convexity of $\|\cdot\|_p$ and the Jensen inequality, we have that

$$\begin{aligned}\|\nabla \mathcal{L}_T(\boldsymbol{\omega})\|_p &= \|\mathbb{E}_{P_T(\mathbf{x},y)} \nabla \mathcal{L}_{\mathbf{x},y}(\boldsymbol{\omega})\|_p \\ &\leq \mathbb{E}_{P_T(\mathbf{x},y)} \|\mathcal{L}_{\mathbf{x},y}(\boldsymbol{\omega})\|_p \qquad \text{(Jensen inequality)} \\ &= \mathbb{E}_{P_T(\mathbf{x},y)} \left(\sum_{i=1}^{D} \sum_{k=1}^{K} |-y^{(k)} \mathbf{x}_i (1 - \mathrm{s}_{\boldsymbol{\omega}}^{(k)}(\mathbf{x})|^p\right)^{1/p} \\ &= \mathbb{E}_{P_T(\mathbf{x},y)} \left(\sum_{k=1}^{K} y^{(k)} \left(1 - \mathrm{s}_{\boldsymbol{\omega}}^{(k)}(\mathbf{x})\right)^p\right)^{1/p} \left(\sum_{i=1}^{D} |\mathbf{x}_i^p|\right)^{1/p} \\ &= \mathbb{E}_{P_T(\mathbf{x},y)} \left(\left(1 - \mathrm{s}_{\boldsymbol{\omega}}^{(k_y)}(\mathbf{x})\right)^p\right)^{1/p} \left(\sum_{i=1}^{D} |\mathbf{x}_i^p|\right)^{1/p} \qquad (k_y \text{ such that } y^{(k_y)} = 1) \\ &= \mathbb{E}_{P_T(\mathbf{x},y)} \alpha(\boldsymbol{\omega}, \mathbf{x}, y) \|\mathbf{x}\|_p,\end{aligned}$$

where $\alpha(\boldsymbol{\omega}, \mathbf{x}, y) = \left(1 - \mathrm{s}_{\boldsymbol{\omega}}^{(k_y)}(\mathbf{x})\right)$, with k_y such that $y^{(k_y)} = 1$. We used the fact that $\mathbf{y}$ is a one-hot vector so it has only one nonzero entry. □

B.7.4 Proof of Remark 5.5.1

Proof. Using the reverse Minkowski inequality, as $0 < p < 1$, we have that

$$\begin{aligned} \|\boldsymbol{\omega}_s\|_p = \|\mathbf{c} + \eta \cdot \nabla \mathcal{L}_T(\boldsymbol{\omega}_s)\|_p \geq \|\mathbf{c}\|_p + \eta \cdot \|\nabla \mathcal{L}_T(\boldsymbol{\omega}_s)\|_p \\ \implies \|\boldsymbol{\omega}_s\|_p - \|\mathbf{c}\|_p \geq \eta \cdot \|\nabla \mathcal{L}_T(\boldsymbol{\omega}_s)\|_p. \end{aligned}$$

In the same fashion, we have that

$$\begin{aligned} \|\mathbf{c}\|_p = \|\boldsymbol{\omega}_s - \eta \cdot \nabla \mathcal{L}_T(\boldsymbol{\omega}_s)\|_p \geq \|\boldsymbol{\omega}_s\|_p + \eta \cdot \|\nabla \mathcal{L}_T(\boldsymbol{\omega}_s)\|_p \\ \implies \|\mathbf{c}\|_p - \|\boldsymbol{\omega}_s\|_p \geq \eta \cdot \|\nabla \mathcal{L}_T(\boldsymbol{\omega}_s)\|_p. \end{aligned}$$

We obtain the desired upper bound by combining those results. □

Appendix C

Appendix for Chapter 6

C.1 Pseudo-Code of MaNo

Algorithm 4 summarizes MaNo introduced in Section 6.3.2, which is a lightweight, training-free method for unsupervised accuracy estimation using the neural network's outputs.

Algorithm 4: Our proposed algorithm, MaNo, for unsupervised accuracy estimation.

Input: Model f pre-trained on $\mathcal{D}_{\text{train}}$, test dataset $\mathcal{D}_{\text{test}} = \{\mathbf{x}_i\}_{i=1}^{N}$.
Parameters: Hyperparameter $p > 1$.
Initialization: Empty prediction matrix $\mathbf{Q} \in \mathbb{R}^{N \times K}$.
Criterion: compute threshold $\Phi(\mathcal{D}_{\text{test}})$ and choose σ following Eq. equation 6.6.
for $i \in [\![1, N]\!]$ **do**
  Inference: recover logits $\mathbf{q}_i = f(\mathbf{x}_i) \in \mathbb{R}^K$.
  Normalization: obtain normalized logits $\sigma(\mathbf{q}_i) \in \Delta_K$.
  Update: fill the prediction matrix $\mathbf{Q}_i \leftarrow \sigma(\mathbf{q}_i)$ following Eq. equation 6.1.
Output Estimation score $\mathcal{S}(f, \mathcal{D}_{\text{test}}) = \frac{1}{\sqrt[p]{NK}} \|\mathbf{Q}\|_p$ following Eq. equation 6.2.

C.2 Related Work

Unsupervised accuracy estimation. This task aims to estimate model generalization performance on unlabeled test sets. To achieve this, several directions have been proposed. (1) Utilizing model outputs: One popular research direction is to

use the model outputs on distribution-shifted data to construct a linear relationship with the test accuracy [5, 24, 25, 27]. For example, a recent work [5] introduced the nuclear norm of the softmax probability matrix as the accuracy estimator. However, current approaches following this direction significantly suffer from the overconfidence issues [111], leading to fluctuating estimation performance across natural distribution shifts. Our work focuses on addressing this issue by balancing logit-information completeness and overconfidence-information accumulation. (2) Considering distribution discrepancy: another direction examines the negative relation between test accuracy and the distribution discrepancy between the training and test datasets [7, 28, 76]. However, commonly-used distribution distances do not guarantee stable accuracy estimation under different distribution shifts [25, 29], and some of these methods are time-consuming on large-scale datasets due to the requirement of training data [28]. (3) Constructing unsupervised losses: methods such as data augmentation and multiple-classifier agreement have also been introduced [26, 77–79]. However, they usually require special model architectures, undermining their practical applicability.

Distance to decision boundaries in deep learning. Decision boundaries of deep neural networks have been studied in various contexts. For example, some works explore the geometric properties of deep neural networks either in the input space [200–203] or in the weight space [204–208]. Some works apply the properties of decision boundaries to address practical questions, such as adversarial defense [209–211], OOD detection [212], and domain generalization [213, 214]. Similar to Li et al. [215], MANO discusses the distance between the learned intermediate feature to each decision boundary in the last hidden space.

Normalization in deep learning. Normalization is a crucial technique extensively utilized across various fields in deep learning, including domain generalization [166, 216, 217], metric learning [218–220], face recognition [221–225] and self-supervised learning [226]. For example, TENT [166] normalizes features of test data using the mean value and standard deviation estimated from the target data. L_2-constrained softmax [221] introduces L_2 normalization on features. These normalization techniques are primarily employed to adapt new samples to familiar domains, calculate similarity, and speed up convergence. However, our proposed

normalization focuses on reducing the negative implications of poorly calibrated scenarios.

C.3 Background on Tsallis Entropies

The definition of Tsallis α-entropies [162] is given below.

Definition C.3.1 (Tsallis α-entropies). Let $\mathbf{p} \in \Delta_K$ be a probability distribution. Let $\alpha > 1$ and $k \geq 0$. The Tsallis α-entropy is defined as:

$$\mathbf{H}_\alpha^{\mathrm{T}}(\mathbf{p}) = k(\alpha - 1)^{-1}(1 - \|\mathbf{p}\|_\alpha^\alpha).$$

It is common to take $k = 1$ or $k = \frac{1}{\alpha}$ following Blondel et al. [163]. The Tsallis α-entropy generalizes the Boltzmann-Gibbs theory of statistic mechanics to nonextensive systems. It has been used as a measure of disorder and uncertainty in many applications [227–230], including in Machine Learning [163–165]. Moreover, they generalize two widely-known measures of uncertainty. Indeed, the limit case $\alpha \to 1$ leads to the Shannon entropy $\mathbf{H}_S$ [see 231, Appendix A.1], *i.e.*, $\lim_{\alpha \to 1} \mathbf{H}_\alpha^{\mathrm{T}}(\mathbf{p}) = \mathbf{H}_S(\mathbf{p}) = -\sum_{j=1}^K p_j \ln(p_j)$, while taking $\alpha = 2$ leads to the Gini index $\mathbf{G}$, a popular impurity measure fo decision trees [232], *i.e.*, $\mathbf{H}_2^{\mathrm{T}}(\mathbf{p}) = \frac{1}{2}(1 - \|p\|_2^2) = \mathbf{G}(\mathbf{p})$. Tsallis entropies measure the uncertainty: the higher the entropy the greater the uncertainty. From a probabilistic perspective, the entropy will take high values for *uncertain* probability distributions, *i.e.*, close to the uniform distribution. We visualize the evolution of the Tsallis entropy for varying parameters α in Figure C.1, where the case $\alpha = 1$ corresponds to the Shannon entropy.

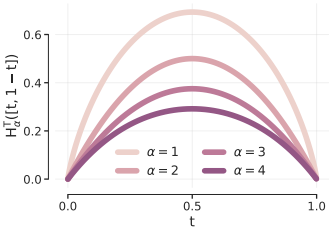


FIGURE C.1: Tsallis α-entropies of $[t, 1-t]$ for $t \in [0, 1]$.

C.4 Theoretical Insights and Proofs

In this section, we provide theoretical insights for Section 6.4 and the proofs of our theoretical results.

Notations. Scalar values are denoted by regular letters (e.g., parameter λ), vectors are represented in bold lowercase letters (e.g., vector $\mathbf{x}$) and matrices are represented by bold capital letters (e.g., matrix $\mathbf{A}$). The i-th row of the matrix $\mathbf{A}$ is denoted by $\mathbf{A}_i$, its j-th column is denoted by $\mathbf{A}_{\cdot,j}$ and its elements are denoted by $\mathbf{A}_{ij}$. The trace of a matrix $\mathbf{A}$ is denoted by $\mathrm{Tr}(\mathbf{A})$ and its transpose by $\mathbf{A}^\top$. The L_p norm of a vector $\mathbf{x}$ is denoted by $\|\mathbf{x}\|_p$, and by abuse of notation we denote it by $\|\mathbf{A}\|_p$ for a matrix $\mathbf{A}$ with $\|\mathbf{A}\|_p^p = \sum_i \|\mathbf{A}_i\|_p^p = \sum_{ij} |\mathbf{A}_{ij}|^p$. Let $\Delta_K \coloneqq \{\mathbf{p} \in [0,1]^K | \sum_{i=1}^K \mathbf{p}_i = 1\}$ be the K-dimensional probability simplex.

C.4.1 Impact of prediction errors

Let $\mathbf{x} \in \mathcal{D}_{\text{test}}$ be a test sample with ground-truth label $y \in \{1, \dots, K\}$. In multi-class classification, the `softmax` operator is used to approximate the posterior probability $p(y|\mathbf{x})$ [see 233, chap.4, p.198]. Reusing the notations of Section 6.4, because of the distribution shifts between source and target, logits are subject to a prediction bias $\boldsymbol{\varepsilon} = (\varepsilon_k)_k$ and write $f(\mathbf{x}) = \mathbf{q}^* + \boldsymbol{\varepsilon}$ where $\mathbf{q}^*$ are ground-truth logits. In this section, we study the impact of such bias on the approximation of the posterior $p(y|\mathbf{x})$.

Impact on the posterior approximation. Proposition 6.3.1 shows the impact of the prediction bias on the KL divergence between the true class posterior probabilities, assumed modeled as $\mathbf{p} = \mathrm{softmax}\mathbf{q}^* \in \Delta_K$, and the estimated ones $\mathbf{s} = \mathrm{softmax} f(\mathbf{x}) \in \Delta_K$. In particular, it states that

$$0 \leq \mathrm{KL}(\mathbf{p}||\mathbf{s}) \leq \boldsymbol{\varepsilon}_+^T \mathbf{p}, \tag{C.1}$$

where $\boldsymbol{\varepsilon}_+ = (\max_l\{\varepsilon_l\} - \varepsilon_k)_k \in \mathbb{R}_+^K$. The proof is given below.

Proof. We denote $\mathbf{q} = f(\mathbf{x}) \in \mathbb{R}^K$ the neural network's outputs on a given test sample $\mathbf{x}$. We first remark that

$$\begin{aligned}
\mathrm{KL}(\mathbf{p}||\mathbf{s}) &= \sum_k \mathbf{p}_k \ln\left(\frac{\mathbf{p}_k}{\mathbf{s}_k}\right) \\
&= \sum_k \mathbf{p}_k \ln\left(\frac{\exp(\mathbf{q}_k^*)}{\sum_{j=1}^K \exp(\mathbf{q}_j^*)} \cdot \frac{\sum_{j=1}^K \exp(\mathbf{q}_j^* + \varepsilon_j)}{\exp(\mathbf{q}_k^* + \varepsilon_k)}\right) \\
&= \sum_k \mathbf{p}_k \ln\left(\exp(-\varepsilon_k) \cdot \frac{\sum_{j=1}^K \exp(\mathbf{q}_j^* + \varepsilon_j)}{\sum_{j=1}^K \exp(\mathbf{q}_j^*)}\right).
\end{aligned} \tag{C.2}$$

To obtain the upper-bound, we notice that

$$\exp(\mathbf{q}_j^* + \varepsilon_j) = \exp(\mathbf{q}_j^*)\exp(\varepsilon_j) \leq \exp(\mathbf{q}_j^*) \cdot \max_l \{\exp(\varepsilon_l)\}.$$

This leads to

$$\frac{\sum_{j=1}^K \exp(\mathbf{q}_j^* + \varepsilon_j)}{\sum_{j=1}^K \exp(\mathbf{q}_j^*)} \leq \frac{\max_l\{\exp(\varepsilon_l)\}\sum_{j=1}^K \exp(\mathbf{q}_j^*)}{\sum_{j=1}^K \exp(\mathbf{q}_j^*)} = \max_l \{\exp(\varepsilon_l)\}. \tag{C.3}$$

Using the fact that all the terms are positive and that ln and exp are increasing functions, we obtain from Eq. equation C.2 that

$$\begin{aligned}
\sum_k \mathbf{p}_k \ln\left(\exp(-\varepsilon_k) \cdot \frac{\sum_{j=1}^K \exp(\mathbf{q}_j^* + \varepsilon_j)}{\sum_{j=1}^K \exp(\mathbf{q}_j^*)}\right) &\leq \sum_k \mathbf{p}_k \ln(\exp(-\varepsilon_k) \cdot \max_l\{\exp(\varepsilon_l)\}) \\
&\leq \sum_k \mathbf{p}_k [\ln(\exp(\max_l\{\varepsilon_l\})) - \varepsilon_k] \\
&\leq \sum_k \mathbf{p}_k [\max_l\{\varepsilon_l\} - \varepsilon_k] \\
&= \boldsymbol{\varepsilon}_+^T \mathbf{p}.
\end{aligned} \tag{C.4}$$

Combining Eq. equation C.2 and Eq. equation C.4 gives the upper bound. □

The quantities $\boldsymbol{\varepsilon}_+$ is a linear transformation of the prediction bias $\boldsymbol{\varepsilon} \in \mathbb{R}^K$ and has nonnegative entries, which means each class is overestimated, representing an *overconfident* model. Proposition 6.3.1 shows that the discrepancy between the error approximation of the posterior probabilities is controlled by the alignment between the posterior and this extreme prediction bias. In addition, by a simple application

of Cauchy-Schwartz in Eq. equation C.1 and using the fact that $|\mathbf{p}\| = \sum_{k=1} \mathbf{p}_k^2 \leq \sum_{k=1} \mathbf{p}_k = 1$, we have $\mathrm{KL}(\mathbf{p}||\mathbf{s}) \leq \|\boldsymbol{\varepsilon}_+\|_2$. In particular, in the perfect situation where $\boldsymbol{\varepsilon} = \mathbf{0}$, $\boldsymbol{\varepsilon}_+$ is equal to 0 and the `softmax` probabilities perfectly approximate the posterior. In summary, Proposition 6.3.1 indicates that not only the norm of the prediction bias but also its alignment to the posterior is responsible for the approximation error of the posterior. In our setting, it means that logits-methods need a low prediction bias on classes on which the model is confident such that `softmax` probabilities can be reliably used to estimate accuracy. This follows our analysis and empirical verification from Section 6.4.

A real-world example. Although we usually tend to think that a high prediction bias shifts the predicted posterior towards the uniform distribution, in the general case, other situations may happen that hinder the quality of the accuracy estimation. For example, one may think of a letter recognition task with a neural network pre-trained on the Latin alphabet and tested on the Cyrillic one. In this case, some prediction probabilities will be adversarial as the neural network will not be aware of the semantic differences between the Latin "B" and the Cyrillic "B", therefore predicting a wrong class with high probability.

C.4.2 Choice of Threshold $\Phi(\mathcal{D}_{\text{test}})$ in Eq. equation 6.6

To automatically select the appropriate normalization function in MaNo (Algorithm 4), we introduced a decision threshold $\Phi(\mathcal{D}_{\text{test}})$ equal, up to a constant, to the average KL divergence between the uniform distribution and the predicted `softmax` probabilities. This stems from Tian et al. [172] that showed that this KL divergence was a measure of the model's uncertainty. In this section, we provide theoretical insights into this formulation by establishing the connection between $\Phi(\mathcal{D}_{\text{test}})$ and the misclassification error made on the test data. We first introduce the following lemma that is inspired from Lemma 4 in Seldin and Tishby [234].

Lemma C.4.1 (Change of measure inequality [234]). Let Z be a random variable on $\{1, \ldots, K\}$ and $\boldsymbol{\mu} = (\mu_k)_k \in \Delta_K$ and $\boldsymbol{\nu} = (\nu_k)_k \in \Delta_K$ be two probability distributions. For any measurable function $\psi \colon \mathcal{Z} \to \mathbb{R}$, the following inequality

holds:

$$\sum_{k=1}^{K} \mu_k \psi(k) \leq \mathrm{KL}(\boldsymbol{\mu} \parallel \boldsymbol{\nu}) + \ln\left(\sum_{k=1}^{K} \nu_k \exp(\psi(k))\right).$$

Proof. We have

$$\begin{aligned}
\sum_{k=1}^{K} \mu_k \psi(k) &= \sum_{k=1}^{K} \mu_k \ln\left(\frac{\mu_k}{\nu_k} \exp(\psi(k)) \frac{\nu_k}{\mu_k}\right) \\
&= \sum_{k=1}^{K} \mu_k \ln\left(\frac{\mu_k}{\nu_k}\right) + \sum_{k=1}^{K} \mu_k \ln\left(\exp(\psi(k)) \frac{\nu_k}{\mu_k}\right) \\
&= \mathrm{KL}(\mu \parallel \nu) + \sum_{k=1}^{K} \mu_k \ln\left(\exp(\psi(k)) \frac{\nu_k}{\mu_k}\right) \quad \text{(Definition of KL}(\cdot \parallel \cdot)\text{)} \\
&\leq \mathrm{KL}(\mu \parallel \nu) + \ln\left(\sum_{k=1}^{K} \mu_k \exp(\psi(k)) \frac{\nu_k}{\mu_k}\right) \quad \text{(Jensen inequality)} \\
&= \mathrm{KL}(\mu \parallel \nu) + \ln\left(\sum_{k=1}^{K} \nu_k \exp(\psi(k))\right).
\end{aligned}$$

□

Let $\mathbf{u} = \frac{1}{K}\mathbb{1}_K \in \Delta_K$ be the uniform probability. The test dataset writes $\mathcal{D}_{\text{test}} = \{\mathbf{x}_i\}_{i=1}^N$ with corresponding logits $\mathbf{q}_i = f(\mathbf{x}_i)$ and ground-truth labels $\mathcal{Y}_{\text{test}} = \{y_i\}_{i=1}^N$ (unavailable in practice). We denote the `softmax` probabilities by $\mathbf{s}^i = \mathrm{softmax}\mathbf{q}_i = \exp(\mathbf{q}_i)/\sum_{k=1}^K \exp(\mathbf{q}_i)_k \in \Delta_K$. We introduce the entropy of a probability vector as $\mathrm{H}(\mathbf{p}) = -\frac{1}{K}\sum_{k=1}^K \mathbf{p}_k \ln(\mathbf{p}_k)$. In particular, it is a measure of uncertainty and takes a high value on uncertain probabilities, *i.e.*, close to the uniform. We establish in the following proposition the connection between the criterion $\Phi(\mathcal{D}_{\text{test}})$, the miscalibration error, the model's confidence, and its entropy.

Proposition C.4.2 ($\Phi(\mathcal{D}_{\text{test}})$, misclassification error, confidence and entropy). We have

$$\underbrace{\xi(\mathcal{D}_{\text{test}}, \mathcal{Y}_{\text{test}})}_{\text{misclassification}} + \underbrace{\mathcal{U}(\mathcal{D}_{\text{test}})}_{\text{confidence}} + \underbrace{\mathcal{H}(\mathcal{D}_{\text{test}})}_{\text{entropy}} \leq \underbrace{\Phi(\mathcal{D}_{\text{test}})}_{\text{criterion}} + \ln(e + \frac{1}{K}),$$

where $\xi(\mathcal{D}_{\text{test}}, \mathcal{Y}_{\text{test}}) = \frac{1}{N}\sum_{i=1}^N (1 - \mathbf{s}^i_{y_i})$, $\mathcal{U}(\mathcal{D}_{\text{test}}) = \frac{1}{N}\sum_{i=1}^N \mathrm{KL}(\mathbf{u} \parallel \mathbf{s}^i)$, and $\mathcal{H}(\mathcal{D}_{\text{test}}) = \frac{1}{N}\sum_{i=1}^N \mathrm{H}(\mathbf{s}^i)$.

Proof. For a given test sample $\mathbf{x}_i \in \mathcal{D}_{\text{test}}$, we first notice that

$$\mathrm{KL}(\mathbf{u} \parallel \mathbf{s}^i) = \sum_{k=1}^{K} \mathbf{u}_k \ln\left(\frac{\mathbf{u}_k}{\mathbf{s}_k^i}\right) = \frac{1}{K}\sum_{k=1}^{K} \ln(\frac{1}{K}) - \ln(\mathbf{s}_k^i) = -\ln(K) - \frac{1}{K}\sum_{k=1}^{K} \ln(\mathbf{s}_k^i).$$

Similarly, we obtain $\mathrm{KL}(\mathbf{s}^i \parallel \mathbf{u}) = \sum_{k=1}^{K} \mathbf{s}_k^i \ln(\mathbf{s}_k^i) + \ln(K)$. Combining those results leads to

$$\begin{aligned} &\mathrm{KL}(\mathbf{u} \parallel \mathbf{s}^i) + \mathrm{KL}(\mathbf{s}^i \parallel \mathbf{u}) = -\frac{1}{K}\sum_{k=1}^{K} \ln(\mathbf{s}_k^i) + \sum_{k=1}^{K} \mathbf{s}_k^i \ln(\mathbf{s}_k^i) \\ \iff &\mathrm{KL}(\mathbf{s}^i \parallel \mathbf{u}) = -\mathrm{KL}(\mathbf{u} \parallel \mathbf{s}^i) - \frac{1}{K}\sum_{k=1}^{K} \ln(\mathbf{s}_k^i) + \sum_{k=1}^{K} \mathbf{s}_k^i \ln(\mathbf{s}_k^i). \end{aligned}$$

Consider the function $\psi(k) = \mathbb{I}(y_i \neq k)$ that takes the value 1 when $y_i \neq k$ and 0 otherwise. Using Lemma C.4.1 with the measures $\boldsymbol{\mu} = \mathbf{s}^i, \boldsymbol{\nu} = \mathbf{u}$ and ψ, and the previous equation, we obtain

$$\begin{aligned} &\sum_{k=1}^{K} \mathbf{s}_k^i \mathbb{I}(y_i \neq k) \leq +\mathrm{KL}(\mathbf{s}^i \parallel \mathbf{u}) + \ln\left(\sum_{k=1}^{K} \mathbf{u}_k \exp(\mathbb{I}(y_i \neq k))\right) \\ \iff &\sum_{k=1}^{K} \mathbf{s}_k^i \mathbb{I}(y_i \neq k) \leq -\mathrm{KL}(\mathbf{u} \parallel \mathbf{s}^i) - \frac{1}{K}\sum_{k=1}^{K} \ln(\mathbf{s}_k^i) + \sum_{k=1}^{K} \mathbf{s}_k^i \ln(\mathbf{s}_k^i) + \ln\left(\sum_{k=1}^{K} \mathbf{u}_k \exp(\mathbb{I}(y_i \neq k))\right) \\ \iff &1 - \mathbf{s}_{y_i}^i \leq -\mathrm{KL}(\mathbf{u} \parallel \mathbf{s}^i) - \frac{1}{K}\sum_{k=1}^{K} \ln(\mathbf{s}_k^i) - \mathrm{H}(\mathbf{s}^i) + \ln\left(\frac{1}{K}(Ke+1)\right) \\ &\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (\textstyle\sum_{k=1}^{K} \mathbf{s}_k^i = 1) \\ \iff &1 - \mathbf{s}_{y_i}^i + \mathrm{KL}(\mathbf{u} \parallel \mathbf{s}^i) + \mathrm{H}(\mathbf{s}^i) \leq -\frac{1}{K}\sum_{k=1}^{K} \ln(\mathbf{s}_k^i) + \ln\left(e + \frac{1}{K}\right). \end{aligned}$$

Summing over all the test samples and dividing by N leads to

$$\frac{1}{N}\sum_{i=1}^{N}(1-\mathbf{s}_{y_i}^i) + \frac{1}{N}\sum_{i=1}^{N} \mathrm{KL}(\mathbf{u} \parallel \mathbf{s}^i) + \frac{1}{N}\sum_{i=1}^{N} \mathrm{H}(\mathbf{s}^i) \leq \underbrace{-\frac{1}{NK}\sum_{i=1}^{N}\sum_{k=1}^{K} \ln(\mathbf{s}_k^i)}_{=\Phi(\mathcal{D}_{\text{test}})} + \ln\left(e + \frac{1}{K}\right),$$

which concludes the proof by using the notations introduced in Proposition C.4.2. □

Interpretation. The term $\xi(\mathcal{D}_{\text{test}}, \mathcal{Y}_{\text{test}})$, dubbed misclassification error, is the expected error on the test set between the optimal probability on the true label

(*i.e.*, 1) and the predicted probability $\mathbf{s}_y^i$. It takes high values when the model makes a lot of mistakes and low values otherwise. $\mathcal{U}(\mathcal{D}_{\text{test}})$ is the expected KL divergence on the test set between the predicted probabilities and the uniform distribution and it measures the model's confidence [172]. It takes high values when the predicted probabilities are far from the uniform (confidence) and low values when they are close to the uniform (uncertain). $\mathcal{H}(\mathcal{D}_{\text{test}})$ is the expected entropy on the test set of the predicted probabilities. It takes high values when predicted probabilities are close to the uniform and low values otherwise. Finally, $\Phi(\mathcal{D}_{\text{test}})$ is the criterion used in Eq. 6.6 to select the proper normalization function. Proposition C.4.2 implies $\Phi(\mathcal{D}_{\text{test}})$ can take *high values* in three situations (the case of few mistakes but an uncertain model is ignored as it rarely happens in practice on such a challenging task):

1. $\xi(\mathcal{D}_{\text{test}}, \mathcal{Y}_{\text{test}})$ is high, *i.e.*, the model makes a lot of mistakes while being confident so $\mathcal{U}(\mathcal{D}_{\text{test}})$ is high and $\mathcal{H}(\mathcal{D}_{\text{test}})$ is low;
2. $\xi(\mathcal{D}_{\text{test}}, \mathcal{Y}_{\text{test}})$ is high, *i.e.*, the model makes a lot of mistakes while being uncertain so $\mathcal{U}(\mathcal{D}_{\text{test}})$ is low and $\mathcal{H}(\mathcal{D}_{\text{test}})$ is low;
3. $\xi(\mathcal{D}_{\text{test}}, \mathcal{Y}_{\text{test}})$ is low, *i.e.*, the model makes few mistakes and is confident so $\mathcal{U}(\mathcal{D}_{\text{test}})$ is high and $\mathcal{H}(\mathcal{D}_{\text{test}})$ is low.

In the normalization strategy `SoftTrun` used in MaNo, we assume that a high $\Phi(\mathcal{D}_{\text{test}})$ indicates that the model is confident, and vice-versa. Indeed, we consider that when $\Phi(\mathcal{D}_{\text{test}}) \leq \eta$, uncertainty is high and we truncate the exponential to reduce error accumulation, while when $\Phi(\mathcal{D}_{\text{test}}) > \eta$, the model can be trusted and complete information is used with the exact exponential (and we recover the `softmax`). This corresponds to the last situation and follows the empirical evidence from from Tian et al. [172] that showed that this KL divergence was small when the uncertainty of the model was high and large for confident models. While this does not cover all the situations, we experimentally show the benefits of $\Phi(\mathcal{D}_{\text{test}})$ and `SoftTrun` in Section 6.5 where MaNo achieves superior performance against 11 commonly used baselines for various architectures and types of shifts on 12 datasets.

C.4.3 Choice of η in Eq. equation 6.6

In all our experiments, we take $\eta = 5$ for the selection criterion in Eq. equation 6.6. We motivate this choice in what follows. In our setting, we consider test samples $\mathbf{x}_i \in \mathcal{D}_{\text{test}}$ drawn i.i.d. from the test distribution p_T. As the model f pre-trained on $\mathcal{D}_{\text{train}}$ is a deterministic function, the logits $\mathbf{q}_i$ are i.i.d. random variables and the decision threshold $\Phi(\mathcal{D}_{\text{test}}) = -\frac{1}{NK}\sum_{i=1}^{N}\sum_{k=1}^{K}\ln\Big(\exp(\mathbf{q}_i)_k / \sum_{j=1}^{K}\exp(\mathbf{q}_i)_j\Big)$ is a random variable with mean μ and variance ν. Applying the Chebyshev's inequality leads to

$$\mathbb{P}(|\Phi(\mathcal{D}_{\text{test}}) - \mu| > \nu\eta) \leq \frac{1}{\eta^2}. \tag{C.5}$$

The threshold $\Phi(\mathcal{D}_{\text{test}})$ is used to determine how calibrated the model is on a given test dataset $\mathcal{D}_{\text{test}}$. Figure 6.2b shows that our proposed normalization is optimal in poorly calibrated datasets and performs slightly below the `softmax` in calibrated situations. Hence, we can afford to be conservative and we want to consider the model calibrated only for *extreme* values of $\Phi(\mathcal{D}_{\text{test}})$. From Eq. equation C.5, taking $\eta = 5$ ensures that the probability that $\Phi(\mathcal{D}_{\text{test}})$ deviates from its mean by several standard deviations with probability smaller than 5% ($\frac{1}{25} < 0.05$). It should be noted that we do not claim the optimality of this choice nor the optimality of our automatic selection in Eq. equation 6.6. However, it is particularly difficult to define decision rules in unsupervised and semi-supervised settings [235]. Moreover, using Eq. equation 6.6, MaNo remains suitable even when test labels are not available which is often the case in real-world applications, and we demonstrate state-of-the-art performance for various architecture and types of shifts in Section 6.5.

C.4.4 Distance to the hyperplane

Lemma C.4.3 (Dohmatob [236]). Let $\boldsymbol{\omega} \in \mathbb{R}^n$ be non zero and $b \in \mathbb{R}$. The distance between any point $\mathbf{z} \in \mathbb{R}^n$ and the hyperplane $\{\mathbf{x}|\boldsymbol{\omega}^\top\mathbf{x} + b = 0\}$ writes $\mathrm{d}(\boldsymbol{\omega}, \mathbf{z}) = |\boldsymbol{\omega}^\top\mathbf{x} + b| / \|\boldsymbol{\omega}\|$.

Proof. The proof follows the geometric intuition from Dohmatob [236]. We recall it here for the sake of self-consistency. The distance between $\mathbf{z}$ and the hyperplane $\mathcal{H} = \{\mathbf{x} \in \mathbb{R}^n | \boldsymbol{\omega}^\top\mathbf{x} + b = 0\}$ is equal to the distance between $\mathbf{z}$ and its orthogonal projection on $\mathcal{H}$. We consider the line $L = \{\mathbf{z} + t\boldsymbol{\omega} | t \in \mathbb{R}\}$ that is orthogonal to $\mathcal{H}$ and passes through $\mathbf{z}$. The desired orthogonal projection is simply the point

$\mathbf{z} + t^*\boldsymbol{\omega}$ such that L and $\mathcal{H}$ intersects, *i.e.*, such that

$$\begin{aligned}\boldsymbol{\omega}^\top(\mathbf{z} + t^*\boldsymbol{\omega}) + b = 0 &\Leftrightarrow \boldsymbol{\omega}^\top\mathbf{z} + b = -t^* \|\boldsymbol{\omega}\|^2 \\ &\Leftrightarrow t^* = -\frac{\boldsymbol{\omega}^\top\mathbf{z} + b}{\|\boldsymbol{\omega}\|^2}. \qquad (\|\boldsymbol{\omega}\| \neq 0)\end{aligned}$$

It follows that the distance between $\mathbf{z}$ and $\mathcal{H}$ writes

$$\mathrm{d}(\boldsymbol{\omega}, \mathbf{z}) = \|\mathbf{z} - \mathbf{z} + t^*\boldsymbol{\omega}\| = \left\| -\frac{\boldsymbol{\omega}^\top\mathbf{z} + b}{\|\boldsymbol{\omega}\|^2} \times \boldsymbol{\omega} \right\| = \frac{|\boldsymbol{\omega}^\top\mathbf{z} + b|}{\|\boldsymbol{\omega}\|}.$$

□

C.4.5 Proof of Theorem 6.3.3

Proof. Reusing the notations introduced in Section 6.3.2 and Algorithm 4, we have that

$$\begin{aligned}\mathcal{S}(f, \mathcal{D}_{\text{test}})^p = \frac{1}{NK}\|\mathbf{Q}\|_p^p &= \frac{1}{NK}\sum_{i=1}^{N}\|\mathbf{Q}_i\|_p^p && \text{(Definition of } \|\mathbf{Q}\|_p\text{)} \\ &= \frac{1}{NK}\sum_{i=1}^{N}\|\sigma(\mathbf{q}_i)\|_p^p && \text{(Definition of } \mathbf{Q}_i \text{ in Algorithm 4)} \\ &= \frac{1}{NK}\sum_{i=1}^{N} 1 - (1 - \|\sigma(\mathbf{q}_i)\|_p^p) \\ &= \frac{1}{K} - \frac{p(p-1)}{NK}\sum_{i=1}^{N}\frac{1}{p(p-1)}(1 - \|\sigma(\mathbf{q}_i)\|_p^p) \\ &= \frac{1}{K} - \frac{p(p-1)}{NK}\sum_{i=1}^{N}\mathbf{H}_p^{\mathrm{T}}(\sigma(\mathbf{q}_i)) && \text{(Definition of } \mathbf{H}_p^{\mathrm{T}}\text{)} \\ &= b - a\left(\frac{1}{N}\sum_{i=1}^{N}\mathbf{H}_p^{\mathrm{T}}(\sigma(\mathbf{q}_i))\right),\end{aligned}$$

where $a = \frac{p(p-1)}{K} > 0$ and $b = \frac{1}{K}$. Rearranging the terms concludes the proof. □

C.4.6 Properties of ϕ

The `softmax` can be decomposed as $\text{softmax}\mathbf{q} = \exp(\mathbf{q})/\sum_{k=1}^{K}\exp(\mathbf{q})_k = (\phi \circ \exp)(\mathbf{q})$, where $\phi\colon \mathbb{R}_+^K \to \Delta_K$ writes $\phi(\mathbf{u}) = \mathbf{u}/\sum_{k=1}^{K}\mathbf{u}_k = \mathbf{u}/\|\mathbf{u}\|_1$. We extend the domain of ϕ to $\mathbb{R}_+^K$ by setting $\phi(\mathbf{0}) = \frac{1}{K}\mathbb{1}_K$. The following proposition states the properties of ϕ.

Proposition C.4.4 (Properties of ϕ).

1. **Generalized injectivity.** $\forall \mathbf{u}, \mathbf{v} \in \mathbb{R}_+^K \setminus \{\mathbf{0}\}, \quad \phi(\mathbf{u}) = \phi(\mathbf{v}) \iff \exists \alpha \in \mathbb{R}^*, \text{ s.t. } \mathbf{u} = \alpha\mathbf{v}$

2. **Evaluation on constant inputs.** Let $\mathbf{u} = \alpha\mathbb{1}_K$ with $\alpha \geq 0$. Then, we have $\phi(\mathbf{u}) = \frac{1}{K}\mathbb{1}_K$.

Proof. We start by proving the first part of Proposition C.4.4. Let $\mathbf{u}, \mathbf{v} \in \mathbb{R}_+^K \setminus \mathbf{0}$. We have

$$\phi(\mathbf{u}) = \phi(\mathbf{v}) \iff \frac{\mathbf{u}}{\|\mathbf{u}\|_1} = \frac{\mathbf{v}}{\|\mathbf{v}\|_1} \iff \mathbf{u} = \underbrace{\frac{\|\mathbf{u}\|_1}{\|\mathbf{v}\|_1}}_{\alpha>0} \times \mathbf{v}.$$

Then, we prove the second part of the proposition. Let $\alpha \geq 0$ and consider $\mathbf{u} = \alpha\mathbb{1}_K \in \mathbb{R}_+^K$. If $\alpha = 0$, then $\mathbf{u} = \mathbf{0}$ and by definition, $\phi(\mathbf{u}) = \phi(\mathbf{0}) = \frac{1}{K}\mathbb{1}_K$. Assuming $\alpha > 0$, we have

$$\phi(\mathbf{u}) = \frac{\mathbf{u}}{\|\mathbf{u}\|_1} = \frac{\mathbf{u}}{\sum_{k=1}^{K}\mathbf{u}_k} = \frac{\alpha}{\sum_{k=1}^{K}\alpha} \times \mathbb{1}_K = \frac{1}{K}\mathbb{1}_K.$$

□

The first part of the proposition is dubbed "generalized injectivity" as the injectivity can be retrieved by fixing $\alpha = 1$ in Proposition C.4.4. It ensures that ϕ only has *equal* outputs if the inputs are *similar*. To illustrate that, consider the logits $\mathbf{q}, \boldsymbol{\delta} \in \mathbb{R}^K$. From Proposition C.4.4, having $\text{softmax}\mathbf{q} = \text{softmax}\boldsymbol{\delta}$ is equivalent to having $\exp(\mathbf{q}) = \alpha\exp(\boldsymbol{\delta})$ for some $\alpha \neq 0$. By positivity of both sides, it implies $\alpha > 0$, and taking the logarithm leads to $\mathbf{q} = \boldsymbol{\delta} + \ln\alpha$. It means that $\mathbf{q}$ is equal to $\boldsymbol{\delta}$ up to a fixed constant. From a learning perspective, those logits will thus have

the same predicted label and the same normalized logits. Proposition C.4.4 shows that using ϕ preserves the information from the neural network. In addition, if the neural network's output is not informative,*i.e.*, all entries are equal, then the link function gives equal probability to all classes.

List of Author's Awards, Patents, and Publications[1]

Journal Articles

- Hongxin Wei, **Renchunzi Xie**, Lei Feng, Bo Han, Bo An. Deep Learning from Multiple Noisy Annotators as A Union. *IEEE Transactions on Neural Networks and Learning Systems*, accepted.
- **Renchunzi Xie**, Mahardhika Pratama. Automatic online multi-source domain adaptation. *Information Sciences*, accepted.

Conference Proceedings

- **Renchunzi Xie**, Ambroise Odonnat, Vasilii Feofanov, Weijian Deng, Jianfeng Zhang, Bo An. MANO: Exploiting Matrix Norm for Unsupervised Accuracy Estimation Under Distribution Shifts. *Proceedings of the 37th Annual Conference on Neural Information Processing Systems*, accepted, 2024.
- **Renchunzi Xie**, Hongxin Wei, Yuzhou Cao, Lei Feng, Bo An. On the Importance of Feature Separability in Predicting Out-Of-Distribution Error. *Proceedings of the 36th Annual Conference on Neural Information Processing Systems*, accepted, 2023.
- Hongxin Wei, Lue Tao, **Renchunzi Xie**, Bo An. Open-set Label Noise Can Improve Robustness Against Inherent Label Noise. *Proceedings of the 35th Annual Conference on Neural Information Processing Systems*, pp.7978-7992, 2021.

[1]The superscript * indicates joint first authors

- **Renchunzi Xie**, Hongxin Wei, Lei Feng, Bo An. GearNet: Stepwise Dual Learning for Weakly Supervised Domain Adaptation. *Proceedings of the 36th AAAI Conference on Artificial Intelligence*, pp.8717-8725, 2022.

- Hongxin Wei, Lue Tao, **Renchunzi Xie**, Lei Feng, Bo An. Open-Sampling: Exploring Out-of-Distribution Data for Re-balancing Long-tailed Datasets. *Proceedings of the 39th International Conference on Machine Learning*, pp.23615-23630, 2022.

- Hongxin Wei, **Renchunzi Xie**, Hao Cheng, Lei Feng, Bo An, Yixuan Li. Mitigating Neural Network Overconfidence with Logit Normalization. *Proceedings of the 39th International Conference on Machine Learning*, pp.23631-23644, 2022.

- Huiping Zhuang, Zhenyu Weng, Hongxin Wei, **Renchunzi Xie**, Toh Kar-Ann, Zhiping Lin. Analytic Class-Incremental Learning with Absolute Memorization and Privacy Protection. *Proceedings of the 36th Annual Conference on Neural Information Processing Systems*, accepted, 2022.

- Hongxin Wei, Huiping Zhuang, **Renchunzi Xie**, Lei Feng, Gang Niu, Bo An, Yixuan Li. Mitigating Memorization of Noisy Labels by Clipping the Model Prediction. *Proceedings of the 40th International Conference on Machine Learning*, accepted, 2023.

Bibliography

[1] Rui Huang, Andrew Geng, and Yixuan Li. On the importance of gradients for detecting distributional shifts in the wild. *Advances in Neural Information Processing Systems*, 34:677–689, 2021. xvii, xxv, 54, 66, 67, 98, 102, 104

[2] DC Dowson and BV666017 Landau. The fréchet distance between multivariate normal distributions. *Journal of Multivariate Analysis*, 12(3):450–455, 1982. xix, 35, 36

[3] Arthur Gretton, Karsten Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex Smola. A kernel method for the two-sample-problem. *Advances in Neural Information Processing Systems*, 19, 2006. xix, 35, 36

[4] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pages 5637–5664. PMLR, 2021. xx, 1, 12, 31, 53, 62, 66, 71

[5] Weijian Deng, Yumin Suh, Stephen Gould, and Liang Zheng. Confidence and dispersity speak: Characterising prediction matrix for unsupervised accuracy estimation. *arXiv preprint arXiv:2302.01094*, 2023. xx, 3, 65, 71, 74, 77, 78, 79, 82, 85, 101, 102, 110

[6] Eric Mintun, Alexander Kirillov, and Saining Xie. On interaction between augmentations and corruptions in natural corruption robustness. *Advances in Neural Information Processing Systems*, 34:3571–3583, 2021. xxi, 84, 85, 87

[7] Yaodong Yu, Zitong Yang, Alexander Wei, Yi Ma, and Jacob Steinhardt. Predicting out-of-distribution error with the projection norm. *arXiv preprint arXiv:2202.05834*, 2022. xxiii, 3, 13, 32, 34, 35, 39, 41, 42, 43, 44, 45, 47, 53, 54, 61, 64, 65, 82, 94, 97, 101, 110

[8] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019. xxiii, 1, 32, 40, 44, 62, 81

[9] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014. 1, 5

[10] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience*, 2018, 2018. 1, 9

[11] Daniel W Otter, Julian R Medina, and Jugal K Kalita. A survey of the usages of deep learning for natural language processing. *IEEE transactions on neural networks and learning systems*, 32(2):604–624, 2020. 1, 9

[12] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 1

[13] Nathan Drenkow, Numair Sani, Ilya Shpitser, and Mathias Unberath. Robustness in deep learning for computer vision: Mind the gap? *arXiv preprint arXiv:2112.00639*, 2021. 1

[14] Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. Learning from noisy labels with deep neural networks: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 2022. 1

[15] Justin M Johnson and Taghi M Khoshgoftaar. Survey on deep learning with class imbalance. *Journal of big data*, 6(1):1–54, 2019. 1, 5

[16] Chuanxing Geng, Sheng-jun Huang, and Songcan Chen. Recent advances in open set recognition: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 43(10):3614–3631, 2020. 1

[17] Kenji Kawaguchi, Leslie Pack Kaelbling, and Yoshua Bengio. Generalization in deep learning. *arXiv preprint arXiv:1710.05468*, 1(8), 2017. 2, 9

[18] Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. Energy-based out-of-distribution detection. *Advances in Neural Information Processing Systems*, 33:21464–21475, 2020. 2, 98, 103

[19] Yang Shu, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. Transferable curriculum for weakly-supervised domain adaptation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 4951–4958, 2019. 2, 11, 15, 17, 19, 23, 25, 28

[20] Feng Liu, Jie Lu, Bo Han, Gang Niu, Guangquan Zhang, and Masashi Sugiyama. Butterfly: A panacea for all difficulties in wildly unsupervised domain adaptation. In *NeurIPS LTS Workshop*, 2019. 2, 11, 15, 17, 19, 25, 28

[21] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International conference on machine learning*, pages 1989–1998. Pmlr, 2018. 2, 10, 17, 24

[22] Konstantinos Kamnitsas, Christian Baumgartner, Christian Ledig, Virginia Newcombe, Joanna Simpson, Andrew Kane, David Menon, Aditya Nori, Antonio Criminisi, Daniel Rueckert, et al. Unsupervised domain adaptation in brain lesion segmentation with adversarial networks. In *Information Processing in Medical Imaging: 25th International Conference, IPMI 2017, Boone, NC, USA, June 25-30, 2017, Proceedings 25*, pages 597–609. Springer, 2017. 2, 10, 17

[23] Weijian Deng, Stephen Gould, and Liang Zheng. What does rotation prediction tell us about classifier accuracy under varying testing environments? In *International Conference on Machine Learning*, pages 2579–2589. PMLR, 2021. 3, 13, 41, 64, 82, 86, 94, 98, 99

[24] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016. 34, 41, 53, 64, 71, 82, 85, 98, 99, 103, 110

[25] Devin Guillory, Vaishaal Shankar, Sayna Ebrahimi, Trevor Darrell, and Ludwig Schmidt. Predicting with confidence on unseen distributions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1134–1144, 2021. 13, 31, 34, 41, 53, 64, 71, 74, 78, 82, 94, 98, 99, 110

[26] Yiding Jiang, Vaishnavh Nagarajan, Christina Baek, and J Zico Kolter. Assessing generalization of sgd via disagreement. *arXiv preprint arXiv:2106.13799*, 2021. 13, 41, 64, 82, 94, 98, 100, 110

[27] Saurabh Garg, Sivaraman Balakrishnan, Zachary C Lipton, Behnam Neyshabur, and Hanie Sedghi. Leveraging unlabeled data to predict out-of-distribution performance. *arXiv preprint arXiv:2201.04234*, 2022. 7, 13, 31, 41, 53, 54, 64, 71, 74, 82, 94, 98, 100, 110

[28] Weijian Deng and Liang Zheng. Are labels always necessary for classifier accuracy evaluation? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15069–15078, 2021. 12, 13, 31, 34, 35, 41, 53, 54, 64, 71, 82, 86, 94, 97, 100, 110

[29] Renchunzi Xie, Hongxin Wei, Yuzhou Cao, Lei Feng, and Bo An. On the importance of feature separability in predicting out-of-distribution error. *arXiv preprint arXiv:2303.15488*, 2023. 13, 64, 82, 98, 100, 110

[30] Ru Peng, Heming Zou, Haobo Wang, Yawen Zeng, Zenan Huang, and Junbo Zhao. Energy-based automated model evaluation. *arXiv preprint arXiv:2401.12689*, 2024. 82

[31] Yuzhe Lu, Yilong Qin, Runtian Zhai, Andrew Shen, Ketong Chen, Zhenlin Wang, Soheil Kolouri, Simon Stepputtis, Joseph Campbell, and Katia Sycara. Characterizing out-of-distribution error via optimal transport. *Advances in Neural Information Processing Systems*, 36, 2024. 3, 82

[32] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015. 5

[33] Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the ACM*, 50(4): 506–519, 2003. 5

[34] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in Neural Information Processing Systems*, pages 8527–8537, 2018. 5, 11, 17, 19, 23, 25

[35] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3(1):1–40, 2016. 5

[36] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009. 5, 10, 15, 17

[37] Fadi Thabtah, Suhel Hammoud, Firuz Kamalov, and Amanda Gonsalves. Data imbalance in classification: Experimental evaluation. *Information Sciences*, 513:429–441, 2020. 5

[38] Peng Cui and Jinjia Wang. Out-of-distribution (ood) detection based on deep learning: A review. *Electronics*, 11(21):3500, 2022. 5

[39] Yiyou Sun, Yifei Ming, Xiaojin Zhu, and Yixuan Li. Out-of-distribution detection with deep nearest neighbors. In *International Conference on Machine Learning*, pages 20827–20840. PMLR, 2022. 5

[40] Aad W Van der Vaart. *Asymptotic statistics*, volume 3. Cambridge university press, 2000. 6

[41] Masashi Sugiyama, Matthias Krauledat, and Klaus-Robert Müller. Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research*, 8(5), 2007. 7

[42] Zhi-Hua Zhou. A brief introduction to weakly supervised learning. *National Science Review*, 5(1):44–53, 2018. 7

[43] Jesper E Van Engelen and Holger H Hoos. A survey on semi-supervised learning. *Machine Learning*, 109(2):373–440, 2020. 8

[44] Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. *arXiv preprint arXiv:1902.07379*, 2019. 8

[45] Jessa Bekker and Jesse Davis. Learning from positive and unlabeled data: A survey. *Machine Learning*, 109(4):719–760, 2020. 8

[46] Lei Feng, Jiaqi Lv, Bo Han, Miao Xu, Gang Niu, Xin Geng, Bo An, and Masashi Sugiyama. Provably consistent partial-label learning. *Advances in Neural Information Processing Systems*, 33:10948–10960, 2020. 8

[47] Min-Ling Zhang, Fei Yu, and Cai-Zhi Tang. Disambiguation-free partial label learning. *IEEE Transactions on Knowledge and Data Engineering*, 29(10): 2155–2167, 2017. 8

[48] Takashi Ishida, Gang Niu, Aditya Menon, and Masashi Sugiyama. Complementary-label learning for arbitrary losses and models. In *International Conference on Machine Learning*, pages 2971–2980. PMLR, 2019. 8

[49] Prissadang Suta, Xi Lan, Biting Wu, Pornchai Mongkolnam, and Jonathan H Chan. An overview of machine learning in chatbots. *International Journal of Mechanical Engineering and Robotics Research*, 9(4):502–510, 2020. 8

[50] Claudine Badue, Rânik Guidolini, Raphael Vivacqua Carneiro, Pedro Azevedo, Vinicius B Cardoso, Avelino Forechi, Luan Jesus, Rodrigo Berriel, Thiago M Paixao, Filipe Mutz, et al. Self-driving cars: A survey. *Expert Systems with Applications*, 165:113816, 2021.

[51] Jitendra Parmar, Satyendra Singh Chouhan, Vaskar Raychoudhury, and Santosh S Rathore. Open-world machine learning: applications, challenges, and opportunities. *ACM Computing Surveys (CSUR)*, 2021. 8

[52] Yan Yan, Zhongwen Xu, Ivor Tsang, Guodong Long, and Yi Yang. Robust semi-supervised learning through label aggregation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016. 9

[53] Kaidi Cao, Maria Brbic, and Jure Leskovec. Open-world semi-supervised learning. *arXiv preprint arXiv:2102.03526*, 2021. 9

[54] Ling Shao, Fan Zhu, and Xuelong Li. Transfer learning for visual categorization: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 26(5):1019–1034, 2014. 10, 17

[55] Judy Hoffman, Sergio Guadarrama, Eric Tzeng, Ronghang Hu, Jeff Donahue, Ross Girshick, Trevor Darrell, and Kate Saenko. Lsda: Large scale detection through adaptation. *arXiv preprint arXiv:1407.5035*, 2014.

[56] Mohsen Ghafoorian, Alireza Mehrtash, Tina Kapur, Nico Karssemeijer, Elena Marchiori, Mehran Pesteie, Charles RG Guttmann, Frank-Erik de Leeuw, Clare M Tempany, Bram Van Ginneken, et al. Transfer learning for domain adaptation in mri: Application in brain lesion segmentation. In *International Conference on Medical Image Computing and Computer-assisted Intervention*, pages 516–524. Springer, 2017.

[57] Dong Wang and Thomas Fang Zheng. Transfer learning for speech and language processing. In *2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, pages 1225–1237. IEEE, 2015.

[58] John Blitzer, Ryan McDonald, and Fernando Pereira. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 120–128, 2006. 10, 17

[59] Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210, 2010. 10, 11, 17, 94

[60] Fuzhen Zhuang, Xiaohu Cheng, Ping Luo, Sinno Jialin Pan, and Qing He. Supervised representation learning: Transfer learning with deep autoencoders. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, pages 4119–4125, 2015. 10, 17, 94

[61] Werner Zellinger, Thomas Grubinger, Edwin Lughofer, Thomas Natschläger, and Susanne Saminger-Platz. Central moment discrepancy (cmd) for domain-invariant representation learning. *arXiv preprint arXiv:1702.08811*, 2017. 10, 17, 94

[62] Jaeho Lee and Maxim Raginsky. Minimax statistical learning with wasserstein distances. *arXiv preprint arXiv:1705.07815*, 2017. 10, 17, 94

[63] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016. 11, 17, 23, 25

[64] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7167–7176, 2017. 11, 17, 34, 35, 94

[65] Yining Chen, Colin Wei, Ananya Kumar, and Tengyu Ma. Self-training avoids using spurious features under domain shift. *arXiv preprint arXiv:2006.10032*, 2020. 11, 17

[66] Yang Zou, Zhiding Yu, Xiaofeng Liu, BVK Kumar, and Jinsong Wang. Confidence regularized self-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5982–5991, 2019. 11, 17

[67] Xiyu Yu, Tongliang Liu, Mingming Gong, Kun Zhang, Kayhan Batmanghelich, and Dacheng Tao. Label-noise robust domain adaptation. In *International Conference on Machine Learning*, pages 10913–10924. PMLR, 2020. 11, 15, 17, 19

[68] Joaquin Quinonero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence. *Dataset shift in machine learning*. Mit Press, 2008. 12, 31, 53, 71

[69] Renchunzi Xie, Ambroise Odonnat, Vasilii Feofanov, Weijian Deng, Jianfeng Zhang, and Bo An. Mano: Exploiting matrix norm for unsupervised accuracy estimation under distribution shifts. *arXiv preprint arXiv:2405.18979*, 2024. 12

[70] Ciprian A Corneanu, Sergio Escalera, and Aleix M Martinez. Computing the testing error without a testing set. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2677–2685, 2020. 13, 94, 97

[71] Yiding Jiang, Dilip Krishnan, Hossein Mobahi, and Samy Bengio. Predicting the generalization gap in deep networks with margin distributions. In *International Conference on Learning Representations*, 2019. 39, 94

[72] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. *Advances in Neural Information Processing Systems*, 30, 2017.

[73] Thomas Unterthiner, Daniel Keysers, Sylvain Gelly, Olivier Bousquet, and Ilya Tolstikhin. Predicting neural network accuracy from weights. *arXiv preprint arXiv:2002.11448*, 2020.

[74] Scott Yak, Javier Gonzalvo, and Hanna Mazzawi. Towards task and architecture-independent generalization gap predictors. *arXiv preprint arXiv:1906.01550*, 2019.

[75] Charles H Martin and Michael W Mahoney. Heavy-tailed universality predicts trends in test accuracies for very large pre-trained deep neural networks. In *Proceedings of the 2020 SIAM International Conference on Data Mining*, pages 505–513. SIAM, 2020. 13, 94, 97

[76] Yuzhe Lu, Yilong Qin, Runtian Zhai, Andrew Shen, Ketong Chen, Zhenlin Wang, Soheil Kolouri, Simon Stepputtis, Joseph Campbell, and Katia Sycara. Characterizing out-of-distribution error via optimal transport. *arXiv preprint arXiv:2305.15640*, 2023. 13, 53, 71, 97, 110

[77] Omid Madani, David Pennock, and Gary Flake. Co-validation: Using model disagreement on unlabeled data to validate classification algorithms. *Advances in Neural Information Processing Systems*, 17, 2004. 13, 94, 98, 110

[78] Emmanouil Antonios Platanios, Avinava Dubey, and Tom Mitchell. Estimating accuracy from unlabeled data: A bayesian approach. In *International Conference on Machine Learning*, pages 1416–1425. PMLR, 2016.

[79] Emmanouil Platanios, Hoifung Poon, Tom M Mitchell, and Eric J Horvitz. Estimating accuracy from unlabeled data: A probabilistic logic approach. *Advances in Neural Information Processing Systems*, 30, 2017. 13, 94, 98, 110

[80] Remi Tachet des Combes, Han Zhao, Yu-Xiang Wang, and Geoff Gordon. Domain adaptation with conditional distribution matching and generalized label shift. *arXiv preprint arXiv:2003.04475*, 2020. 15

[81] Kun Zhang, Bernhard Schölkopf, Krikamol Muandet, and Zhikun Wang. Domain adaptation under target and conditional shift. In *International Conference on Machine Learning*, pages 819–827. PMLR, 2013.

[82] Jiahua Dong, Yang Cong, Gan Sun, Zhen Fang, and Zhengming Ding. Where and how to transfer: Knowledge aggregation-induced transferability perception for. unsupervised domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[83] Jiahua Dong, Yang Cong, Gan Sun, Bineng Zhong, and Xiaowei Xu. What can be transferred: Unsupervised domain adaptation for endoscopic lesions segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4022–4031, June 2020. 15

[84] Benoît Frénay and Michel Verleysen. Classification in the presence of label noise: a survey. *IEEE Transactions on Neural Networks and Learning Systems*, 25(5):845–869, 2013. 15

[85] Aritra Ghosh, Himanshu Kumar, and PS Sastry. Robust loss functions under label noise for deep neural networks. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 15, 18

[86] Zhen Fang, Jie Lu, Feng Liu, Junyu Xuan, and Guangquan Zhang. Open set domain adaptation: Theoretical bound and algorithm. *IEEE Trans. Neural Networks Learn. Syst.*, pages 4309–4322, 2021. 17

[87] Jiahua Dong, Zhen Fang, Anjin Liu, Gan Sun, and Tongliang Liu. Confident-anchor-induced multi-source-free domain adaptation. In *NeurIPS*, 2021. 17

[88] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016. 17, 19

[89] Jiangfan Han, Ping Luo, and Xiaogang Wang. Deep self-learning from noisy labels. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5138–5147, 2019.

[90] Aditya Menon, Brendan Van Rooyen, Cheng Soon Ong, and Bob Williamson. Learning from corrupted binary labels via class-probability estimation. In *International Conference on Machine Learning*, pages 125–134, 2015.

[91] Zhen Fang, Jie Lu, Anjin Liu, Feng Liu, and Guangquan Zhang. Learning bounds for open-set learning. In *ICML*, pages 3122–3132, 2021. 17

[92] Jacob Goldberger and Ehud Ben-Reuven. Training deep neural-networks using a noise adaptation layer. In *Proceedings of the 5th International Conference on Learning Representation*, 2016. 18

[93] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1944–1952, 2017. 18, 26

[94] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning*, pages 2304–2313. PMLR, 2018. 18

[95] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. *arXiv preprint arXiv:1803.09050*, 2018. 18

[96] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. In *Advances in Neural Information Processing Systems*, pages 8778–8788, 2018. 18

[97] Lei Feng, Senlin Shu, Zhuoyi Lin, Fengmao Lv, Li Li, and Bo An. Can cross entropy loss be robust to label noise? IJCAI, 2020. 18

[98] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4320–4328, 2018. 19

[99] Fuli Luo, Peng Li, Jie Zhou, Pengcheng Yang, Baobao Chang, Zhifang Sui, and Xu Sun. A dual reinforcement learning framework for unsupervised text style transfer. *arXiv preprint arXiv:1905.10060*, 2019. 19

[100] Zhongyi Han, Xian-Jin Gui, Chaoran Cui, and Yilong Yin. Towards accurate and robust domain adaptation under noisy environments. *arXiv preprint arXiv:2004.12529*, 2020. 19, 25, 28

[101] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951. 22

[102] Guanglei Yang, Haifeng Xia, Mingli Ding, and Zhengming Ding. Bi-directional generation for unsupervised domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 6615–6622, 2020. 24

[103] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2223–2232, 2017. 24

[104] Sen Wu, Hongyang R Zhang, and Christopher Ré. Understanding and improving information transfer in multi-task learning. *arXiv preprint arXiv:2005.00944*, 2020. 24

[105] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 25, 41, 63, 81

[106] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International Conference on Machine Learning*, pages 97–105. PMLR, 2015. 25

[107] Hongxin Wei, Lei Feng, Xiangyu Chen, and Bo An. Combating noisy labels by agreement: A joint training method with co-regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2020. 25

[108] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *European Conference on Computer Vision*, pages 213–226. Springer, 2010. 26, 62

[109] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5018–5027, 2017. 26, 62, 79, 81

[110] Zizhao Zhang, Han Zhang, Sercan O Arik, Honglak Lee, and Tomas Pfister. Distilling effective supervision from severe label noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9294–9303, 2020. 26

[111] Hongxin Wei, Renchunzi Xie, Hao Cheng, Lei Feng, Bo An, and Yixuan Li. Mitigating neural network overconfidence with logit normalization. *arXiv preprint arXiv:2205.09310*, 2022. 31, 72, 94, 110

[112] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013. 32, 94

[113] Rui Gao and Anton Kleywegt. Distributionally robust stochastic optimization with wasserstein distance. *Mathematics of Operations Research*, 2022. 35

[114] Aman Sinha, Hongseok Namkoong, Riccardo Volpi, and John Duchi. Certifying some distributional robustness with principled adversarial training. *arXiv preprint arXiv:1710.10571*, 2017. 35

[115] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014. 35

[116] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning*, pages 1180–1189. PMLR, 2015. 35

[117] Tzay Y Young and Thomas W Calvert. *Classification, estimation, and pattern recognition*. Elsevier Publishing Company, 1974. 37

[118] Pierre A Devijver and Josef Kittler. *Pattern recognition: A statistical approach*. Prentice hall, 1982. 37

[119] Andrew R Webb and Keith D Copsey. Introduction to statistical pattern recognition. *Statistical Pattern Recognition*, pages 1–31, 1990. 37

[120] Richard O Duda, Peter E Hart, et al. *Pattern classification*. John Wiley & Sons, 2006. 37

[121] Kagan Tumer and Joydeep Ghosh. Bayes error rate estimation using classifier ensembles. *International Journal of Smart Engineering System Design*, 5(2): 95–109, 2003. 37

[122] Mahalanobis Prasanta Chandra et al. On the generalised distance in statistics. In *Proceedings of the National Institute of Sciences of India*, volume 2, pages 49–55, 1936. 37

[123] Keinosuke Fukunaga. *Introduction to statistical pattern recognition*. Elsevier, 2013. 37

[124] Frederick D Garber and Abdelhamid Djouadi. Bounds on the bayes classification error based on pairwise risk functions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(2):281–288, 1988. 38

[125] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 40, 41, 62, 63, 81

[126] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015. 40, 41, 62, 81

[127] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016. 41, 63, 81

[128] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 41, 63, 81

[129] Stuart Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982. 46

[130] J MacQueen. Classification and analysis of multivariate observations. In *5th Berkeley Symp. Math. Statist. Probability*, pages 281–297, 1967. 46

[131] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016. 49

[132] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018. 53, 71

[133] Jian Li, Xuanyuan Luo, and Mingda Qiao. On generalization error bounds of noisy gradient methods for non-convex learning. *arXiv preprint arXiv:1902.00621*, 2019. 54, 98

[134] Haozhe An, Haoyi Xiong, Xuhong Li, Xingjian Li, Dejing Dou, and Zhanxing Zhu. Can we use gradient norm as a measure of generalization error for model selection in practice? 2020. 54, 98

[135] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, 2017. 54

[136] Yuge Shi, Jeffrey Seely, Philip HS Torr, N Siddharth, Awni Hannun, Nicolas Usunier, and Gabriel Synnaeve. Gradient matching for domain generalization. *arXiv preprint arXiv:2104.09937*, 2021. 54

[137] Lucas Mansilla, Rodrigo Echeveste, Diego H Milone, and Enzo Ferrante. Domain generalization via gradient surgery. In *Proceedings of the IEEE/CVF international Conference on Computer Vision*, pages 6630–6638, 2021. 54, 98

[138] Yingxue Zhou, Belhal Karimi, Jinxing Yu, Zhiqiang Xu, and Ping Li. Towards better generalization of adaptive gradient methods. *Advances in Neural Information Processing Systems*, 33:810–821, 2020. 54

[139] Yang Zhao, Hao Zhang, and Xiuyuan Hu. Penalizing gradient norm for efficiently improving generalization in deep learning. In *International Conference on Machine Learning*, pages 26982–26992, 2022. 54, 98

[140] Giulia Denevi, Carlo Ciliberto, Riccardo Grazzi, and Massimiliano Pontil. Learning-to-learn stochastic gradient descent with biased regularization. In *The International Conference on Machine Learning (ICML)*, pages 1566–1575, 2019. 57

[141] Maria-Florina Balcan, Mikhail Khodak, and Ameet Talwalkar. Provable guarantees for gradient-based meta-learning. In *The International Conference on Machine Learning (ICML)*, pages 424–433, 2019. 57

[142] Sebastien Arnold, Shariq Iqbal, and Fei Sha. When maml can adapt fast and how to assist when it cannot. In *Proceedings of International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2021. 57

[143] Vasilii Feofanov, Emilie Devijver, and Massih-Reza Amini. Transductive bounds for the multi-class majority vote classifier. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3566–3573, 2019. 59

[144] Massih-Reza Amini, Vasilii Feofanov, Loic Pauletto, Lies Hadjadj, Emilie Devijver, and Yury Maximov. Self-training: A survey. *arXiv preprint arXiv:2202.12040*, 2023. 61, 65

[145] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255. Ieee, 2009. 62, 81

[146] Shibani Santurkar, Dimitris Tsipras, and Aleksander Madry. Breeds: Benchmarks for subpopulation shift. *arXiv preprint arXiv:2008.04859*, 2020. 62, 81

[147] Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When does label smoothing help? *Advances in Neural Information Processing Systems (NeurIPS)*, 32, 2019. 67

[148] Handing Wang, Yaochu Jin, and Xin Yao. Diversity assessment in many-objective optimization. *IEEE transactions on cybernetics*, 47(6):1510–1522, 2016. 67

[149] Ronald Kemker, Marc McClure, Angelina Abitino, Tyler Hayes, and Christopher Kanan. Measuring catastrophic forgetting in neural networks. In *Proceedings of the Association for the Advancement of Artificial Intelligence conference on artificial intelligence (AAAI)*, 2018. 68

[150] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 596–608, 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/06964dce9addb1c5cb5d6e3d9838f733-Paper.pdf. 69

[151] Jiahua Dong, Zhen Fang, Anjin Liu, Gan Sun, and Tongliang Liu. Confident anchor-induced multi-source free domain adaptation. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 2848–2860, 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/168908dd3227b8358eababa07fcaf091-Paper.pdf.

[152] Yaodong Yu, Zitong Yang, Alexander Wei, Yi Ma, and Jacob Steinhardt. Predicting out-of-distribution error with the projection norm. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 25721–25746. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/yu22i.html. 69

[153] Matthias Minderer, Josip Djolonga, Rob Romijnders, Frances Hubis, Xiaohua Zhai, Neil Houlsby, Dustin Tran, and Mario Lucic. Revisiting the calibration of modern neural networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 15682–15694, 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/8420d359404024567b5aefda1231af24-Paper.pdf. 69

[154] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1321–1330, 2017. 69

[155] Dan Hendrycks and Mantas Mazeika. X-risk analysis for ai research. *arXiv preprint arXiv:2206.05862*, 2022. 71

[156] Pinar Donmez, Guy Lebanon, and Krishnakumar Balasubramanian. Unsupervised supervised learning i: Estimating classification and regression errors without labels. *Journal of Machine Learning Research*, 11(4), 2010. 71

[157] Olivier Chapelle and Alexander Zien. Semi-supervised classification by low density separation. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, pages 57–64, 2005. 72, 74

[158] Vasilii Feofanov, Malik Tiomoko, and Aladin Virmaux. Random matrix analysis to balance between supervised and unsupervised learning under the low density separation assumption. In *Proceedings of the 40th International Conference on Machine Learning*, pages 10008–10033, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/feofanov23a.html. 72

[159] Ambroise Odonnat, Vasilii Feofanov, and Ievgen Redko. Leveraging ensemble diversity for robust self-training in the presence of sample selection bias. In Sanjoy Dasgupta, Stephan Mandt, and Yingzhen Li, editors, *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pages 595–603. PMLR, 02–04 May 2024. URL https://proceedings.mlr.press/v238/odonnat24a.html. 72, 79

[160] David Mickisch, Felix Assion, Florens Greßner, Wiebke Günther, and Mariele Motta. Understanding the decision boundary of deep neural networks: An empirical study. *arXiv preprint arXiv:2002.01810*, 2020. 75

[161] Arthur Mensch, Mathieu Blondel, and Gabriel Peyré. Geometric losses for distributional learning. In *Proceedings of the 36th International Conference on Machine Learning*, pages 4516–4525, 2019. 76

[162] Constantino Tsallis. Possible generalization of Boltzmann-Gibbs statistics. *Journal of Statistical Physics*, 52(1):479–487, Jul 1988. ISSN 1572-9613. doi: 10.1007/BF01016429. URL https://doi.org/10.1007/BF01016429. 77, 111

[163] Mathieu Blondel, Andre Martins, and Vlad Niculae. Learning classifiers with Fenchel-Young losses: Generalized entropies, margins, and algorithms. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, pages 606–615, 16–18 Apr 2019. URL https://proceedings.mlr.press/v89/blondel19a.html. 77, 111

[164] Mathieu Blondel, André F. T. Martins, and Vlad Niculae. Learning with Fenchel-Young losses. *J. Mach. Learn. Res.*, 21(1), jan 2020. ISSN 1532-4435.

[165] Boris Muzellec, Richard Nock, Giorgio Patrini, and Frank Nielsen. Tsallis regularized optimal transport and ecological inference. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 2387–2393, 2017. 77, 111

[166] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=uXl3bZLkr3c. 78, 110

[167] Hongxin Wei, Renchunzi Xie, Hao Cheng, Lei Feng, Bo An, and Yixuan Li. Mitigating neural network overconfidence with logit normalization. In *Proceedings of the 39th International Conference on Machine Learning*, pages 23631–23644, 2022. 79

[168] Binghui Chen, Weihong Deng, and Junping Du. Noisy softmax: Improving the generalization ability of dcnn via postponing the early softmax saturation. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4021–4030, 2017. URL https://api.semanticscholar.org/CorpusID:23578881. 79

[169] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, pages 5542–5550, 2017. 79, 81

[170] Dong-Hyun Lee. Pseudo-Label : The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks. *ICML 2013 Workshop : Challenges in Representation Learning (WREPL)*, 07 2013. 80

[171] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: simplifying semi-supervised learning with consistency and confidence. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, 2020. 80

[172] Junjiao Tian, Yen-Chang Hsu, Yilin Shen, Hongxia Jin, and Zsolt Kira. Exploring covariate and concept shift for out-of-distribution detection. In *NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2021. 80, 114, 117

[173] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF international Conference on Computer Vision*, pages 1406–1415, 2019. 81

[174] J. Taylor, B. Earnshaw, B. Mabey, M. Victors, and J. Yosinski. Rxrx1: An image set for cellular morphological variation across many experimental batches. In *International Conference on Learning Representations (ICLR)*, 2019. 81

[175] Nico JD Nagelkerke et al. A note on a general definition of the coefficient of determination. *Biometrika*, 78(3):691–692, 1991. 81

[176] Maurice George Kendall. Rank correlation methods. *Michigan University*, 1948. 81

[177] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning*, pages 5389–5400, 2019. 84

[178] Alexandre de Brébisson and Pascal Vincent. An exploration of softmax alternatives belonging to the spherical loss family. In *International Conference on Learning Representations, (ICLR)*, 2016. URL http://arxiv.org/abs/1511.05042. 87

[179] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 91

[180] Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 91

[181] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 91

[182] Gauthier Guinet, Behrooz Omidvar-Tehrani, Anoop Deoras, and Laurent Callot. Automated evaluation of retrieval-augmented language models with task-specific exam generation. *arXiv preprint arXiv:2405.13622*, 2024. 91

[183] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. *Advances in Neural Information Processing Systems*, 19, 2006. 94

[184] Jeff Z HaoChen, Colin Wei, Adrien Gaidon, and Tengyu Ma. Provable guarantees for self-supervised deep learning with spectral contrastive loss. *Advances in Neural Information Processing Systems*, 34:5000–5011, 2021. 94

[185] Yifei Ming, Yiyou Sun, Ousmane Dia, and Yixuan Li. How to exploit hyperspherical embeddings for out-of-distribution detection. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023.

[186] Weiran Huang, Mingyang Yi, and Xuyang Zhao. Towards the generalization of contrastive self-supervised learning. *arXiv preprint arXiv:2111.00743*, 2021. 94

[187] Ya Li, Xinmei Tian, Mingming Gong, Yajing Liu, Tongliang Liu, Kun Zhang, and Dacheng Tao. Deep domain generalization via conditional invariant adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 624–639, 2018. 94

[188] Yang Chen, Yu Wang, Yingwei Pan, Ting Yao, Xinmei Tian, and Tao Mei. A style and semantic memory mechanism for domain generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9164–9173, 2021.

[189] Zijian Wang, Yadan Luo, Ruihong Qiu, Zi Huang, and Mahsa Baktashmotlagh. Learning to diversify for single domain generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 834–843, 2021. 94

[190] Zhaowei Zhu, Yiwen Song, and Yang Liu. Clusterability as an alternative to anchor points when learning with noisy labels. In *International Conference on Machine Learning*, pages 12912–12923. PMLR, 2021. 94

[191] Zhaowei Zhu, Zihao Dong, and Yang Liu. Detecting corrupted labels without training a model to predict. In *International Conference on Machine Learning*, pages 27412–27427. PMLR, 2022. 94

[192] Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *International Conference on Machine Learning (ICML)*, pages 1225–1234, 2016. 98

[193] Ben London. A pac-bayesian analysis of randomized learning with application to stochastic gradient descent. *Advances in Neural Information Processing Systems (NeurIPS)*, 30, 2017.

[194] Omar Rivasplata, Emilio Parrado-Hernández, John S Shawe-Taylor, Shiliang Sun, and Csaba Szepesvári. Pac-bayes bounds for stable algorithms with instance-dependent priors. *Advances in Neural Information Processing Systems (NeurIPS)*, 31, 2018. 98

[195] Satrajit Chatterjee. Coherent gradients: An approach to understanding generalization in gradient descent-based optimization. *arXiv preprint arXiv:2002.10657*, 2020. 98

[196] Jeffrey Negrea, Mahdi Haghifam, Gintare Karolina Dziugaite, Ashish Khisti, and Daniel M Roy. Information-theoretic generalization bounds for sgld via data-dependent estimates. *Advances in Neural Information Processing Systems (NeurIPS)*, 32, 2019. 98

[197] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. *arXiv preprint arXiv:1812.04606*, 2018. 98, 103

[198] Jingkang Yang, Kaiyang Zhou, Yixuan Li, and Ziwei Liu. Generalized out-of-distribution detection: A survey. *arXiv preprint arXiv:2110.11334*, 2021.

[199] Shiyu Liang, Yixuan Li, and Rayadurgam Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv preprint arXiv:1706.02690*, 2017. 98, 103

[200] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, Pascal Frossard, and Stefano Soatto. Empirical study of the topology and geometry of deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3762–3770, 2018. 110

[201] Ben Poole, Subhaneil Lahiri, Maithra Raghu, Jascha Sohl-Dickstein, and Surya Ganguli. Exponential expressivity in deep neural networks through transient chaos. *Advances in Neural Information Processing Systems*, 29, 2016.

[202] Guido F Montufar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. On the number of linear regions of deep neural networks. *Advances in Neural Information Processing Systems*, 27, 2014.

[203] Hamid Karimi, Tyler Derr, and Jiliang Tang. Characterizing the decision boundary of deep neural networks. *arXiv preprint arXiv:1912.11460*, 2019. 110

[204] Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. In *Artificial Intelligence and Statistics*, pages 192–204, 2015. 110

[205] Yann N Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *Advances in Neural Information Processing Systems*, 27, 2014.

[206] Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-SGD: Biasing gradient descent into wide valleys. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124018, 2019.

[207] Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In *International Conference on Machine Learning*, pages 1019–1028, 2017.

[208] C Daniel Freeman and Joan Bruna. Topology and geometry of half-rectified network optimization. *arXiv preprint arXiv:1611.01540*, 2016. 110

[209] Warren He, Bo Li, and Dawn Song. Decision boundary analysis of adversarial examples. In *International Conference on Learning Representations*, 2018. 110

[210] Byeongho Heo, Minsik Lee, Sangdoo Yun, and Jin Young Choi. Knowledge distillation with adversarial samples supporting decision boundary. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3771–3778, 2019.

[211] Francesco Croce and Matthias Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. In *International Conference on Machine Learning*, pages 2196–2205, 2020. 110

[212] Dongha Lee, Sehun Yu, and Hwanjo Yu. Multi-class data description for out-of-distribution detection. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1362–1370, 2020. 110

[213] Roozbeh Yousefzadeh. Deep learning generalization and the convex hull of training sets. *arXiv preprint arXiv:2101.09849*, 2021. 110

[214] Jingde Li, Changqing Shen, Lin Kong, Dong Wang, Min Xia, and Zhongkui Zhu. A new adversarial domain generalization network based on class boundary feature detection for bearing fault diagnosis. *IEEE Transactions on Instrumentation and Measurement*, 71:1–9, 2022. 110

[215] Yu Li, Lizhong Ding, and Xin Gao. On the decision boundary of deep neural networks. *arXiv preprint arXiv:1808.05385*, 2018. 110

[216] Xinjie Fan, Qifei Wang, Junjie Ke, Feng Yang, Boqing Gong, and Mingyuan Zhou. Adversarially adaptive normalization for single domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8208–8217, 2021. 110

[217] Seonguk Seo, Yumin Suh, Dongwan Kim, Geeho Kim, Jongwoo Han, and Bohyung Han. Learning to optimize domain specific normalization for domain generalization. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*, pages 68–83. Springer, 2020. 110

[218] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. *Advances in Neural Information Processing Systems*, 29, 2016. 110

[219] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3733–3742, 2018.

[220] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 110

[221] Rajeev Ranjan, Carlos D Castillo, and Rama Chellappa. L2-constrained softmax loss for discriminative face verification. *arXiv preprint arXiv:1703.09507*, 2017. 110

[222] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 212–220, 2017.

[223] Feng Wang, Xiang Xiang, Jian Cheng, and Alan Loddon Yuille. Normface: L2 hypersphere embedding for face verification. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 1041–1049, 2017.

[224] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. ArcFace: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2019.

[225] Xiao Zhang, Rui Zhao, Yu Qiao, Xiaogang Wang, and Hongsheng Li. Adacos: Adaptively scaling cosine logits for effectively learning deep face representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10823–10832, 2019. 110

[226] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, pages 1597–1607, 2020. 110

[227] Murray Gell-Mann and Constantino Tsallis. *Nonextensive Entropy: Interdisciplinary Applications* . 04 2004. ISBN 9780195159769. doi: 10.1093/oso/9780195159769.001.0001. URL https://doi.org/10.1093/oso/9780195159769.001.0001. 111

[228] Renato Negrinho and Andre Martins. Orbit regularization. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27, 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/file/f670ef5d2d6bdf8f29450a970494dd64-Paper.pdf.

[229] Robert Sneddon. The Tsallis entropy of natural information. *Physica A: Statistical Mechanics and its Applications*, 386(1):101–118, 2007. ISSN 0378-4371. doi: https://doi.org/10.1016/j.physa.2007.05.065. URL https://www.sciencedirect.com/science/article/pii/S0378437107006206.

[230] Zeinab Teimoori, Kazem Rezazadeh, and Abasat Rostami. Inflation based on the Tsallis entropy. *The European Physical Journal C*, 84(1):80, Jan 2024. ISSN 1434-6052. doi: 10.1140/epjc/s10052-024-12435-z. URL https://doi.org/10.1140/epjc/s10052-024-12435-z. 111

[231] Ben Peters, Vlad Niculae, and André F. T. Martins. Sparse sequence-to-sequence models. In Anna Korhonen, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1504–1519, July 2019. doi: 10.18653/v1/P19-1146. URL https://aclanthology.org/P19-1146. 111

[232] C. Gini. *Variabilità e mutabilità: contributo allo studio delle distribuzioni e delle relazioni statistiche. [Fasc. I.]*. Studi economico-giuridici pubblicati per cura della facoltà di Giurisprudenza della R. Università di Cagliari. Tipogr. di P. Cuppini, 1912. 111

[233] Christopher M Bishop. *Pattern Recognition and Machine Learning*, page 738. Springer, 2006. 112

[234] Yevgeny Seldin and Naftali Tishby. Pac-bayesian analysis of co-clustering and beyond. *Journal of Machine Learning Research*, 11(117):3595–3646, 2010. URL http://jmlr.org/papers/v11/seldin10a.html. 114

[235] Massih-Reza Amini, Vasilii Feofanov, Loic Pauletto, Lies Hadjadj, Emilie Devijver, and Yury Maximov. Self-training: A survey. *arXiv preprint arXiv:2202.12040*, 2022. 118

[236] Elvis Dohmatob. Distance from a point to a hyperplane. https://math.stackexchange.com/questions/1210545/distance-from-a-point-to-a-hyperplane, 2020. 118