UTS UNIVERSITY OF TECHNOLOGY SYDNEY

Learning with Restricted Data via Gradient Manipulations

by Jing Li

Thesis submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy

under the supervision of Ivor W. Tsang

University of Technology Sydney
Faculty of Engineering and Information Technology

October 2022

Certificate of Original Authorship

I, Jing Li, declare that this thesis is submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the Faculty of Engineering and Information Technology at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research is supported by the Australian Government Research Training Program.

Signature: Production Note: Signature removed prior to publication.

Date: 20 Oct 2022

This thesis is lovingly dedicated to my family. I wish I would be always your pride and joy.

Acknowledgements

Foremost, I would like to thank my supervisor Prof. Ivor W. Tsang, for his professional guidance and persistent support. He opened up a window for me to have a closer look at the world of *machine learning* which appears complex but fascinating! Without his leading in the passed four-year academic journey, I could not have overcome the obstacles nor found my way to think and do. I am also grateful that Prof. Ivor W. Tsang could always provide appropriate suggestions on my study plan, from each individual project to internship scheduling and thesis preparation. He is such a caring mentor who helps me become confident, mature and independent. What I have learned from him will have a great impact on my future career path.

Thanks also go to my co-supervisor Prof. Yulei Sui who provided a perspective for my work from a security player. His participation makes my work more practical and understandable for people out of the machine learning community. In particular, I appreciate Prof. Yulei Sui's contribution to polishing my papers.

I would like to sincerely thank Dr. Yuangang Pan for his unselfish assistance on my study. His research passion and attitude has encouraged me to keep exploring the interesting topics. I would also thank my best friend Yinghua Yao, who is not only a good flatmate but also reliable co-worker. From Sydney to Singapore, I am fortunate that people around me are helpful. I appreciate Dr. Yan Zhang's invitation last winter and the contribution of Dr. Yueming Lyu in my recent work. I am also grateful for the assistance of Dr. Yaxin Shi, Xiaowei Zhou, Xingrui Yu on my research. My warmest gratitude also goes the Dr Jiangchao Yao, Dr. Xu Chen, Dr. Xiaofeng Xu, Zhuanghua Liu, Jinliang Deng, Feiyang Ye, Bowen Xing, Peiyao Zhao, Yujie Fang, and Cheng Chen. Thanks also to the co-workers out of my group, Shaojun Shi and Tao Zhang.

Many thanks to the University of Technology Sydney for providing me a good studying environment, and Agency of Science, Technology and Research for offering me a rare internship opportunity.

My special thanks to Yume who has accompanied me for the entire PhD journey. Last, Dad and Mom, the sacrifices you have made for me are beyond any description. Thank you, my beloveds. I wish I would always make you proud.

Abstract

Data has been the fuel that drives modern artificial intelligence. With more and more emerging concerns on data, e.g., data privacy, learning paradigms demand evolving accordingly. In this thesis, I focus on discriminative learning on **restricted data** from the role of learning executors who are in charge of the learning process. That means, my research interest lies in how to design proper learning algorithms while data is considered restricted. Specifically, the following three types of scenarios will be explored.

- *Private data is accessible to learning executors, but the learning process should not expose any data information.* In such a scenario, the learning executors are trusted while any others are restricted from accessing data. Differential Privacy (DP) is a golden principle for this problem which preserves the participation of every data point and thus can defend against strong adversaries. I intend to take a step forward and explore how to ensure a safe learning when data is pairwise labelled, to which DP cannot be directly applied due to the explicit pairwise correlations.

- *Only incomplete data is accessible to learning executors.* Learning on incomplete data is a challenging topic when some data information is restricted from learning executors. For example, not all people would like to answer their demographics in a survey. Suppose a common case where the missing values are from a discrete attribute or the label domain. Inferring them comes to the Semi-Supervised Learning (SSL) problem. I will study how to improve the prediction module for unlabeled data from the aspect of prediction uncertainty.

- *None of data is accessible to learning executors, but some feedbacks are available.* This scenario considers that learning executors cannot access data, which hinders feeding data to the model for the end-to-end back-propagation. However, learning is stilled feasible if some feedbacks from data are provided, e.g., model evaluations. In practice, model tuning tasks are studied in this context, and the tuning efficiency for deep neural networks is particularly investigated.

From top to bottom, one can sense that the restriction on data becomes stricter, which also implies some bigger change should be applied to learning paradigms. Despite the specific requirement in each scenario, I am interested how to deal with them via a general principle. It is known that gradient-based optimization has been popular in machine learning. Given a fixed model structure, the eventually attained model can be attributed to the elaborately designed model gradients (Note that it is not always because of losses). With this insight, I propose to deal with different restricted data by using different meaningful **gradient manipulation** techniques. Concretely, I apply gradient perturbation to compensate for the missing or addition of any interested pairwise data, employ gradient masking to reduce the impact of over-confident unlabeled predictions, adopt gradient estimation to learn from model evaluations. I conclude that although the meaning of restricted data varies across different tasks (which also brings out various challenges), the insight of gradient manipulation constantly offers a good perspective to tackle these problems.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Background

A general learning process can be described to learn from data based on the problem or question that is being asked. For example, training a classification model with labeled data learns to assign category to input, and constructing a task-specific distance from provided sample-wise constraints comes to the distance metric learning [Bellet et al., 2013]. With sufficient observed data, such tasks have been thoroughly studied in the past decades. However, are existing powerful learning paradigms still satisfactory when data are *restricted*?

This question is not new if people simply understand restricted data from the aspect of data quantity. The very early Vapnik-Chervonenkis theory [Vapnik, 1999] has proved that the model generalization is positively correlated with the number of training data under the formulation of empirical risk minimization. In other words, learning on small-scale data may incur an over-fitting problem [Caruana et al., 2000]. The research furthering such a direction indeed analyzes how much data does one need to guarantee a "good model" from the statistical learning point. Alternatively, another branch of studies straightforward target improving the learning performance on limited data, such as transfer learning [Yosinski et al., 2014] or few-shot learning [Fei-Fei et al., 2006, Wang et al., 2020c]. The core idea therein is discovering the prior knowledge that generalizes well on a few observed examples. Other research remedies less data learning from different perspectives which includes data augmentation [Zhang et al., 2017a, Cubuk et al., 2019], ensemble learning [Zhang and Ma, 2012], cross validation [Refaeilzadeh et al., 2009], and so on.

Beyond the learning with insufficient examples, this thesis specially considers the restricted data which emerges in real applications. Fig. 1.1 displays three types of concerns from the aspect of data privacy, from which data is restricted during learning. (1) The first concern indicates that the learning executor is exclusively trusted, which implies all others

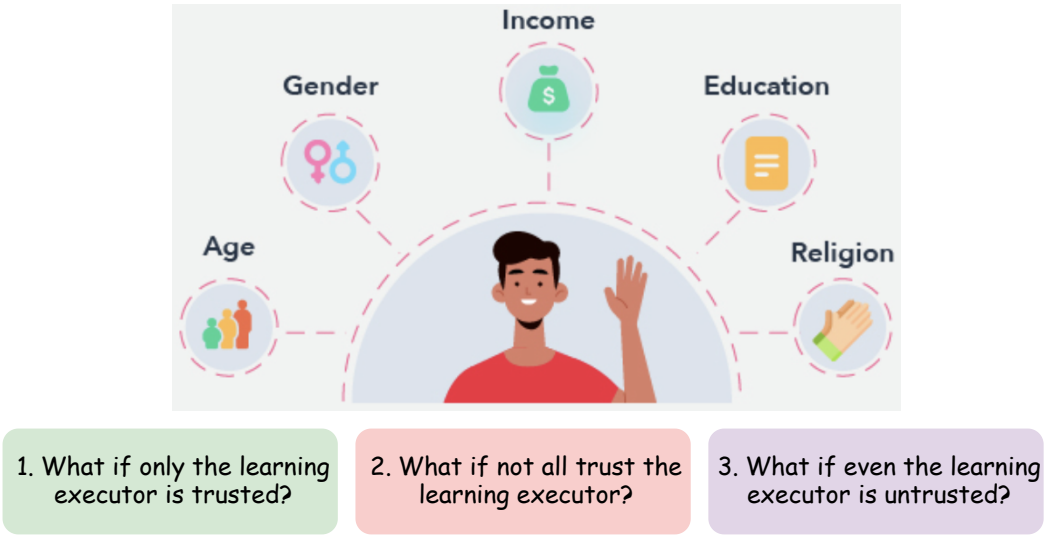


Figure 1.1 Various privacy concerns arise during learning in the real world.

are potential adversaries. Thus, any other parties should be restricted from accessing the data. (2) The second concern implies the complete data is not available for learning executors. For example, not all participants are willing to answer the gender or income in a survey [Fernando et al., 2021]. In such a scenario, private information is hidden in the phase of data collection which is then unavailable to the learning executor. (3) The third concern appears if data is too sensitive to be observed by the learning executor, while data is still possibly used to evaluate a candidate model. This concern is practical especially when data holder is a non-expert and he/she would adopt the safest way to preserve data. As data is kept intact as it is untouched, such a restriction is quite strict. Note that this privacy motivation shares some similarity with Federated Learning [McMahan et al., 2017], but the communication here is not the operational gradient.

Although starting from the privacy concern about data, this thesis is concentrating on learning. That means I am actually playing the role of the learning executor and intend to address these concerns in various machine learning tasks.

1.2 Research Scope

Having the privacy concerns displayed in Fig. 1.1, I will present how they affect the learning process and what specific problems I am going to address, which are shown as Fig. 1.2.

Distance metric learning with private pairwise data. The first concern requires that learning process should not expose any data information. This requirement serves an additive

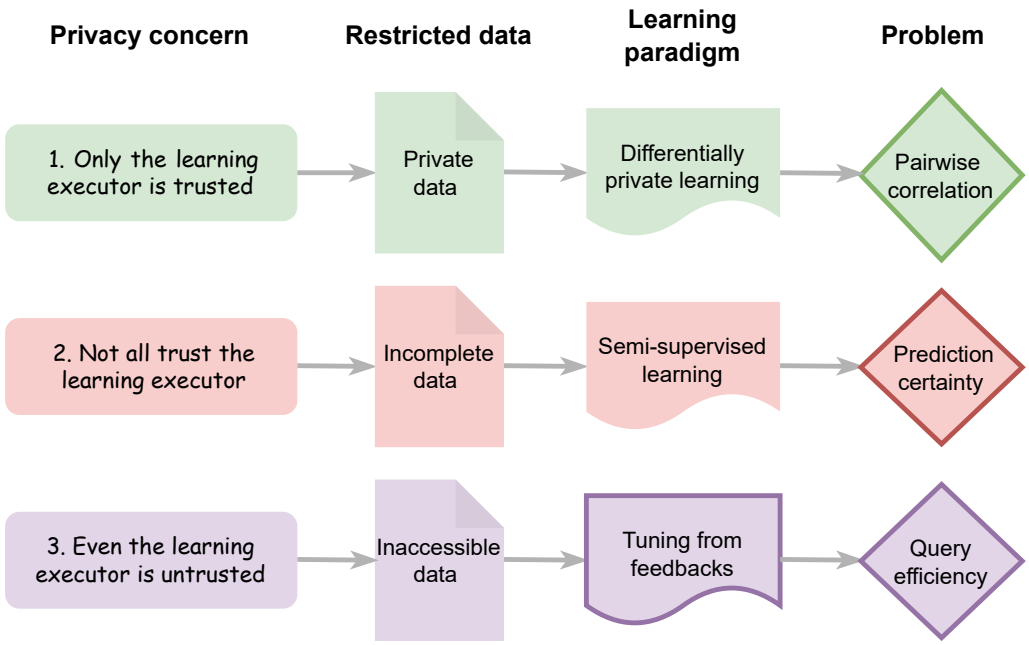


Figure 1.2 Privacy concerns in terms of learning executors motivate the different forms of learning on restricted data. The unit decorated with thicker edges highlights the novel points of this thesis. Note that I specially consider the cases where a single discrete attribute or the label domain is incomplete in the second concern, which can be remedied by SSL then.

constraint compared with a standard learning process. Given an algorithm, the learned model can be viewed as a projection of input data. Thus, learning with different data may lead to different models [Koh and Liang, 2017]. This fact suggests that adversaries can use this knowledge to infer the existence of interested data, e.g., membership attack [Shokri et al., 2017]. That means, when only learning executors are trusted, the learning paradigms need to adjust accordingly to prevent private data leakage to potential adversaries. Differential Privacy (DP) [Dwork et al., 2006] is a golden principle which proposes to add tailored randomization to model to compensate for the participant of any datum. It turns out powerful to defend against strong adversaries with provable theory. I specially consider that the private data is pairwise labeled, which are quite common in machine learning, e.g., distance metric learning [Weinberger and Saul, 2009]. Since the pairwise correlations may provide extra clues for inference, DP cannot be directly applied to this case.

Semi-supervised learning for recovering missing labels. With the second concern in data collection phase, data is incomplete to the learning executor. Although one can pre-process such data by deleting incomplete records or filling them by random values, a more

effective way is to recover them. Instead of considering the general incomplete data [Ipsen et al., 2020], I focus on a common and important case where some specific discrete attribute is not complete (e.g., races are not reported by some participants) or classification labels are partially missing. Thus I propose to solve them by the well-known Semi-Supervised Learning (SSL), where I trickily treat discrete attribute values as labels. Although seldom claimed from privacy concern, SSL is treated as a strategy to recover the missing attributes or labels. Please refer to Section 2.2.2 for practical applications. The key of training a SSL model is properly assigning the labels for the unlabeled data. In this thesis, I will revisit this problem from the view of prediction uncertainty.

Model tuning without peeking on target data. When the learning executor is not trusted, data might be inaccessible to the learning executor. To enable learning in this restrictive scenario, I assume that model candidates are allowed to submit to the data's side for evaluation, and only model performance is returned to the learning executor. This is a safe option for non-expert data manager because the data is never shared out and evaluations are conducted locally in an unobserved manner. Model tuning tasks are studied here which allows a warm-start in such a restrictive scenario and is also tolerant from the view of communication cost. In particular, the tuning efficiency is the core problem given a limited query budget in the real world.

1.3 Challenges

From the model learning perspective, the lack of data can be remedied via a number of ways, such as augmenting original data [DeVries and Taylor, 2017, Cubuk et al., 2019], recovering missing values [Ghahramani and Jordan, 1993, Dick et al., 2008], introducing extra data [Yoon et al., 2017], calling better pre-trained models [Li et al., 2019], and so forth. Although they might be are useful for boosting the training model performance, the privacy concern for the given data is not touched.

With the broad and varying meanings of restricted data across different scenarios, as discussed in the previous sections, the primary challenge is how to deal with restricted data in a macro-principle. There might be multiple viewpoints to answer this question. For example, people from the database would understand restricted data from communication or encryption, while statisticians might be interested in how such restrictions are guaranteed. Apart from model performance, once restricted data is stated, the learning executor from the machine learning community needs to figure out how to solve these restrictions. However,

there is seldom any study on whether there exists a general instruction for such learning problems.

The micro challenge is dependent on specific tasks. In the first problem, due to data correlation, preserving privacy for pairwise data is more difficult compared with independent and identically distributed samples. In the second problem, the challenge is how to confidently fit the unlabeled data without relying on heuristic tricks, e.g., data augmentations. In the third problem, the difficulty lies in how an initially provided model (linear models or complex neural networks) can be efficiently tuned according to the model evaluations.

1.4 Thesis Contributions

This thesis proposes to solve the learning problem where different types of restricted data are provided via a general principle – gradient manipulation. For the specific meaning for this term, one can skip to Section 2.3 of Chapter 2. The contributions of each of three learning tasks are listed as follows.

1.4.1 Distance metric learning with private pairwise data

I extend DP to pairwise data by considering the relation transitivity from the graph perspective. Based on the stochastic gradient decent for metric learning, I use gradient perturbation to effectively preserve the pairwise information during learning. The main contributions are as follows.

- I propose Differential Pairwise Privacy (DPP), a new privacy preserving technique for pairwise data in distance metric learning.
- I analyze the gradient sensitivity [Song et al., 2013] via exploring the distribution of pairs within a minibatch during optimization. It helps to enhance the utility of the released distance metric by reducing the amount of injected noise.
- I exploit the connection between DPP and the existing privacy research, and demonstrate the superiority of DPP through comparing them over numerous experiments.

1.4.2 Semi-supervised learning for inferring missing labels

SSL in this thesis is motivated by recovering the missing values of discrete attributes or labels of collected data. The performance of existing SSL methods can be improved if the overconfident prediction issue is properly tackled. My high-level idea here is to stop

greedily encouraging every probability distribution to be one-hot but focus on informative ones instead. To this end, I design a new distillation strategy which tries to learn from the model output more confidently. The contributions of this study are three folds.

- I propose a new distillation strategy for SSL, named ADaptive Sharpening (ADS), to distill low-entropy predictions for unlabeled data. The proposed ADS can be simply used as a plug-in for existing SSL algorithms.
- I analyze that ADS adaptively masks out the overconfident and negligible predictions and promotes informed predictions only which has not been touched by existing works.
- I verify the efficacy of ADS by combining it with SOTA models on SSL bechmarks, and results show that ADS outperforms other distillation strategies by confidently inferring the missing labels.

1.4.3 Model tuning without peeking on target data

To enable tuning without directly feeding the target data to the model, I propose to leverage query-feedback information. Specifically, the learning executor can query by model candidates while model performance is returned as feedback. Imagine that data is put into a black-box and only used for evaluating models. Particularly, black-box optimization technique is adopted to deal with this problem. The contributions of this work are summarized as follows.

- I introduce the learning setting of Earning eXtra PerformancE from restriCTive feEDdbacks (EXPECTED). EXPECTED is not a conventional data-driven optimization problem and thus supplements the existing model tuning regime.
- I propose Performance-guided Parameter Search (PPS) algorithm which resorts to optimizing the distribution of model parameters via gradient estimation. In terms of tuning DNNs, Layerwise Coordinate Parameter Search (LCPS) algorithm is further brought forward to significantly improve tuning efficiency.
- I theoretically justify the soundness of the proposed algorithms and experimentally demonstrate the efficacy of the proposed algorithms on different modal data, including tabular data, text, and images.

1.5 Thesis Outline

In this thesis, I study how to execute learning given different restricted data. Therein, I particularly focus on three tasks, e.g., distance metric learning with private pairwise data, semi-supervised learning for recovering the missing labels, and model tuning with inaccessible target data. Despite specific difficulties in each task, I investigate each of them following the instruction of gradient manipulation. The thesis is organized as follows:

- Chapter 2 restates the problem of each learning task and reviews related works;
- Chapter 3 extends differential privacy for pairwise data with the distance metric learning as a case study;
- Chapter 4 proposes a new distillation strategy for semi-supervised learning, in which the goal is to moderately fit the unlabeled training data;
- Chapter 5 exploits efficient model tuning through limited query-feedbacks;
- Chapter 6 concludes the thesis and points out some future research directions.

The organization of this thesis is shown in Figure 1.3.

1.6 Publications

- **Jing Li**, Yuangang Pan, Yueming Lyu, Yinghua Yao, Yulei Sui, and Ivor W. Tsang. “Earning Extra Performance from Restrictive Feedbacks." Under review by IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022.
- Yinghua Yao, Yuangang Pan, **Jing Li**, Ivor W. Tsang, and Xin Yao. “COUF: Clustering withOut the Unwanted Factor." Under review by International Journal of Computer Vision, 2022.
- **Jing Li**, Yuangang Pan, and Ivor W. Tsang, “Taming Overconfident Predictions on Unlabeled Data from Hindsight." R&R, IEEE Transactions on Neural Networks and Learning Systems, 2022.
- Feiping Nie, Shaojun Shi, **Jing Li**, and Xuelong Li. “Implicit Weight Learning for Multi-View Clustering." IEEE Transactions on Neural Networks and Learning Systems (Early Access), 2021.

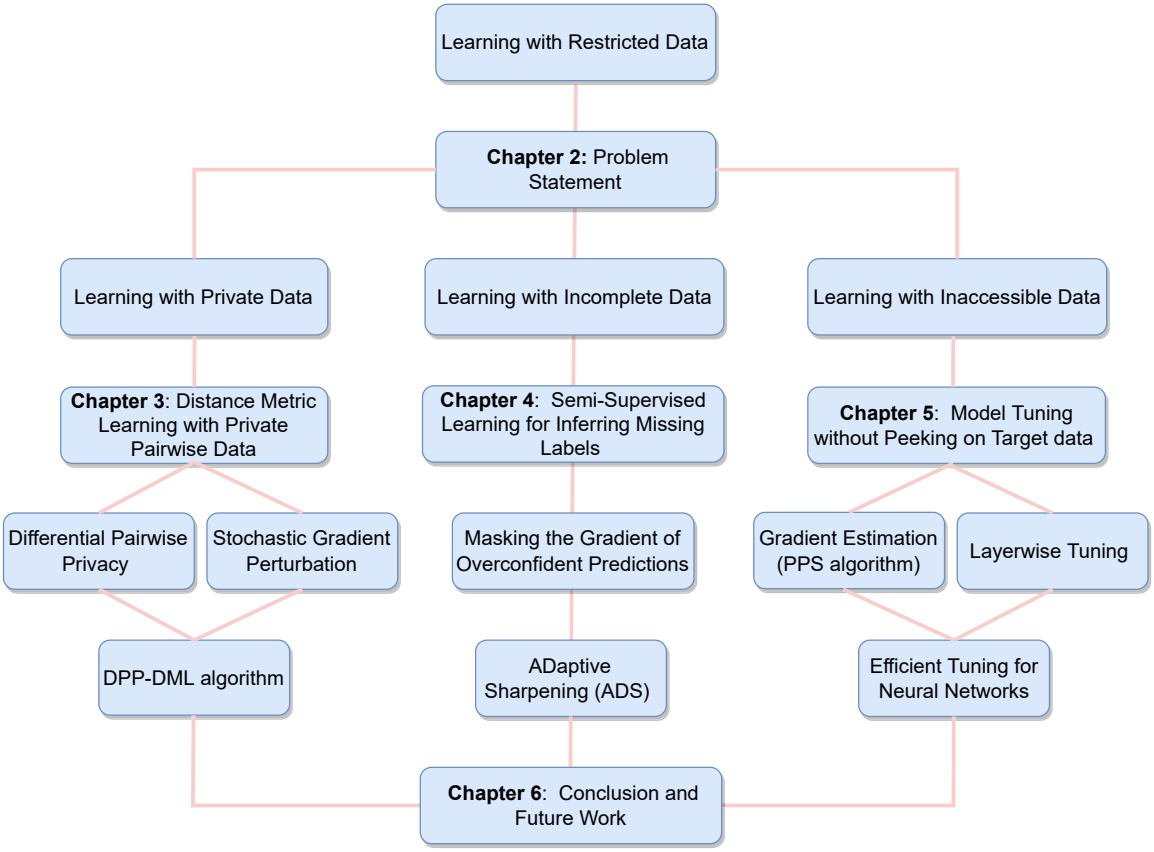


Figure 1.3 The organization of this thesis.

- Tao Zhang, Tianqing Zhu, **Jing Li**, Mengde Han, Wanlei Zhou, and Philip Yu. "Fairness in Semi-Supervised Learning: Unlabeled Data Help to Reduce Discrimination." IEEE Transactions on Knowledge and Data Engineering, vol. 34, no, 4, pp. 1763-1774, 2020.

- **Jing Li**, Yuangang Pan, Yulei Sui, and Ivor W. Tsang. "Secure Metric Learning via Differential Pairwise Privacy." IEEE Transactions on Information Forensics and Security, vol 15, pp. 3640-3652, 2020.

Chapter 2

Problem Statement

In this chapter, firstly I introduce that restricted data is factually common in practice when different concerns are raised. Then I present the potential problems and related literature about learning with such kind of data. Finally, I propose a general principle that provides instructions of how to design proper solutions for restricted data learning despite specific tasks.

2.1 Restricted Data

Data has been very important in machine learning. Without data, one cannot train a model and much modern research will go in vain. With an accelerating pace of producing data in our world, a huge number of datasets have been established in the past decades which provide the fuel of modern machine learning research. The researchers who work in computer vision or machine learning would probably start with the well-known MNIST dataset [LeCun et al., 1998], where zero to nine handwritten digits are simple and convenient to verify a model's performance, from classification tasks to generation tasks. For large-scale datasets, ImageNet [Deng et al., 2009] is often used to verify the scalability of a model.

However, data is never free to use in the real world. In the machine learning community, researchers nowadays are perhaps authorized the use of many public data under the agreement of licenses. Those who want to create their dataset may resort to crowd-sourcing which is usually expensive and suffers from unintentional human errors [Han et al., 2018]. For example, big enterprises usually invest heavily in gathering as much valuable data as possible. That means, because of the great efforts required in data collection or data pre-processing, many datasets are not always on hand for everyone. Such facts inspire me to consider the *restricted data* which has not been broadly studied in the machine learning community.

As mentioned in Section 1.1, one of the early understandings towards the restricted data is from the view of data quantity, i.e., less data. Learning with less data means that one probably has no chance to acquire sufficient data that can perfectly reflect the data distribution. There does exist a gap between the empirical data distribution estimated from finite observed samples and the ground-truth data distribution especially when the sample size is small. Generally, more observations mean one can describe the distribution more accurately. Recent research[1] also mentions restricted-use data from the view of sensitive information, inference risk, or confidential promises. Such interpretations are close to what I will show next, but they focus on applying access to restricted data rather than using them for downstream learning.

In this thesis, I am trying to characterize restricted data with the motivation of real-world concerns about data in the context of machine learning tasks. To unify different forms of restricted data, I will pick data privacy as the standpoint and bring forward more motivations as well as applications in later chapters.

Before starting, I would like first name the person who uses data for model learning as *learning executor*. For convenience, I also suppose there is a *data owner* who owns data and a group of *participants* whose information is collected in case it is necessary.

- Data owner offers private data to learning executor but concerns data privacy during the learning process. In this scenario, the learning executor is trusted and he/she can see the entire dataset. By contrast, all others are not trusted, which means that the private data should be restricted from accessing by any other parties/individuals. As data has been shared out, the learning executor is now responsible to guarantee such restrictions, which comes to the first type of learning with restricted data.

- Data owner offers incomplete data to learning executor without additional requirements because some participants have intentionally hidden their sensitive information out of personal privacy concerns. In this scenario, the incomplete data that learning executor obtains is free to use, and learning executor is actually restricted by the integrity of the original data. Thus, the learning executor who wants to achieve a promising model would infer the missing values beforehand, which comes to the second type of learning with restricted data.

- Data owner keeps the inaccessible data locally and only offers limited query chances to learning executor. This scenario happens when data owner concerns that data are too sensitive (or valuable) to be shared out even to learning executor, and thus learning

[1] https://www.icpsr.umich.edu/web/pages/DSDR/restricted-data.html

executor cannot peek at any data point. Since it allows communications between data and learning executor, learning is still achievable by extracting some useful information therein, which comes to the third type of learning with restricted data.

I will focus on these three types of privacy-motivated restricted data in this thesis. The readers who are interested in more restricted data, please refer to Section 6.2 for my future potential explorations.

2.2 Learning with Restricted Data

When data is said restricted, learning executor is supposed to response such restrictions, which turns out different model learning processes. In this section, I will review the related research which has been developed to confront the similar topics. Based on these studies, I will clarify the problems targeted in my thesis.

2.2.1 Learning with private pairwise data

In this part, private data is abused to particularly denote the restricted data which is kept intact for learning executor while it cannot be leaked to any others, i.e., the referred first type restricted data in Section 2.1. Fig. 2.1 briefly exhibits this task where pairwise data is restricted.

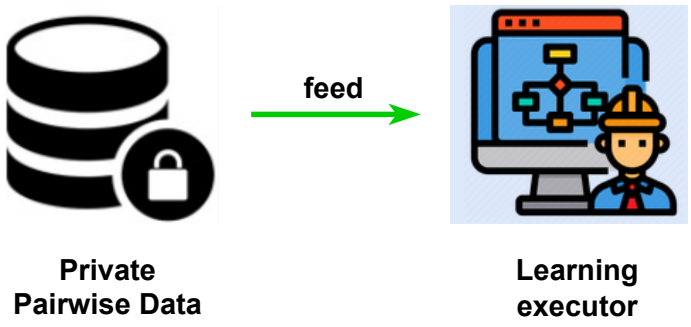


Figure 2.1 Private pairwise data is sent to learning executor with the privacy requirement during the learning process.

Privacy strategy overview

Although in the first scenario where I have assumed raw data is directly sent to learning executor, there exist some alternative privacy strategies in the database community from

literature. Let's say private data is about patients' records which are often tabular data. One may ask why not directly hide individual identity because it will immediately solve the privacy issues. Actually, this so-called anonymization trick has been argued as unsafe and inadequate in many previous works, such as [Narayanan and Shmatikov, 2008] and [Sweeney, 2013]. The deficiency of anonymization is being vulnerable to auxiliary information. For example, the information extracted from a social medium without name annotation is still possibly recognized from some other social media data with name annotation by comparison. Back to the exhibited first type of restricted data, others may argue that learning process can be put into a safe environment [Zhai et al., 2016] which cannot communicate with the outside world except by outputting a trained model. However, it is still vulnerable for membership attack [Shokri et al., 2017] through the output model. Differential Privacy (DP) [Dwork et al., 2006] has been a golden principle for private data, which preserves the participation of every sample under a worst-case formulation.

Differentially private learning schemes

Having a specific differential privacy definition, current research suggests implementing privacy in the following three ways.

- Output perturbation. This approach adds noise to the output of the algorithm, which can be easily understood when the algorithm is as simple as a database operator, like averaging the salary of the employees in a company [Dwork et al., 2014]. For some machine learning tasks like classification or regression, possible solutions are given in [Chaudhuri et al., 2011, Zhang et al., 2017b] .

- Objective perturbation. By adding an extra perturbed term in the objective, [Chaudhuri and Monteleoni, 2009, Chaudhuri et al., 2011, Kifer et al., 2012, Talwar et al., 2014, Acar et al., 2017] stated this strategy can work with a guarantee for the better model utility. The perturbed term is usually not general for the other objectives and needs some special considerations for different models.

- Gradient perturbation. This line of works manage to perturb the gradient during the optimization, which has been widely used in [Song et al., 2013, Bassily et al., 2014, Talwar et al., 2015, Abadi et al., 2016, Zhang et al., 2017b, Wang et al., 2017]. The key advantage of this approach is to better adapt to the stochastic learning process.

Preserving privacy for pairwise data

Since differential privacy for independent and identically distributed data has been thoroughly studied [Dwork et al., 2014, Chaudhuri et al., 2011], I would like to take a step forward and focus on another typically used data – pairwise data. In many real applications, samples are labeled by comparing with each other, yielding the pairwise labeled data. Pairwise data has been used in many machine learning tasks, such as distance metric learning [Bellet et al., 2013], semi-supervised clustering [Lu, 2007], pairwise ranking [Chen et al., 2013], and so on. In this part, I will review the possible works that can preserve privacy for pairwise data.

A feasible choice to preserve pairwise data is to leverage Local Differential Privacy (LDP) [Duchi et al., 2013]. For example, one can adopt the coin-flipping style approach [Warner, 1965] to probabilistically change the relationship of any pair. In this case, any subsequently surmised relationship for a data pair can be possibly denied, such as [Hay et al., 2017, Joy and Gerla, 2017, Qin et al., 2017, Sun et al., 2018]. If the privacy for an individual feature is needed, one can resort to the input perturbation like [Fukuchi et al., 2017]. Unfortunately, LDP suffers from the performance degeneration [Sun et al., 2018] because it fails to take the subsequent application into account.

A more related branch of research is studying the privacy of correlated data. One pioneer work by [Kifer and Machanavajjhala, 2011] pointed out that the correlated records degrade the privacy level if not specially treated. They then proposed a customizable privacy framework Pufferfish [Kifer and Machanavajjhala, 2012] that requires the whole algorithm to change an attacker's prior distribution as less as possible. Following this framework, Song et al. [Song et al., 2017b] proposed a so-called Wasserstein mechanism. Recent graph based privacy research [Karwa et al., 2011, Hay et al., 2009, Rastogi et al., 2009, Zhang et al., 2015] constructed the relations among individuals but mainly focused on the graphical statistics. Some other works [Zhu et al., 2014, Liu et al., 2016, Zhao et al., 2017] in the data release community [Nissim et al., 2007, Lee et al., 2019, Zhao et al., 2019] also noticed the detriment that data correlation brings in and formalized diverse DP variants. However, their claimed data correlation is virtually different from pairwise data. For instance, [Song et al., 2017b] included the time series data while [Zhu et al., 2014] took the records distance into consideration. This means correlation in existing research varies from individual to individual. Instead, the correlation of pairwise data is caused by one individual's participation in multiple pairs. Therefore, their schemes cannot be freely extended to my problem.

My research attention

Based on the above statement, my goal is to extend the concept of DP to pairwise labeled data. That means the participation of pairwise data (including pairwise relationships and pairwise features sometimes) is considered as private information for all others except the learning executor. To this end, the learning algorithm should satisfy such a requirement just like what it is in differentially private learning [Chaudhuri et al., 2011]. In particular, distance metric learning is used as a study case which is typically fed with pairwise data. But note that the concept of differential privacy for pairwise data is not dependent on the downstream learning tasks.

2.2.2 Learning with incomplete data

In this part, the agnostic complete data is viewed as the restricted data, and the incomplete data is then sent to the learning executor as the individual privacy concern is "addressed" by not sharing the sensitive information, i.e., the referred second type of restricted data in Section 2.1. Fig. 2.2 simply displays this task where incomplete data is sent to learning executor.

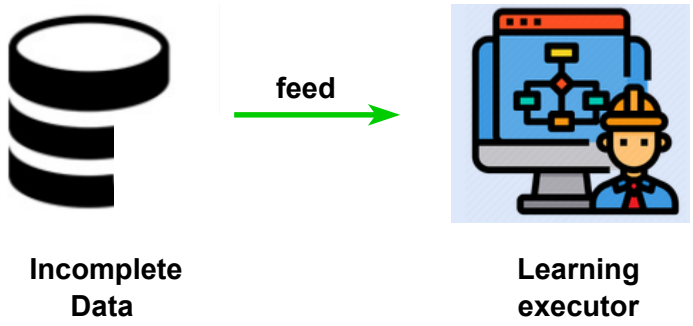


Figure 2.2 Incomplete data is sent to learning executor with sensitive attributes or labels having been hidden by some participants.

Strategy for general incomplete data

There are multiple ways to handle incomplete data. Intuitively, one can simply delete the deficient records if such deletions are not harmful. By contrast, imputation is a good idea that allows more data information to be used. A very direct approach for imputation is to randomly fill the missing values, which is then sometimes connected with the noisy data [Zhu et al., 2012]. A more sophisticated way to do imputation is resorting to reasonable prior knowledge about data. For example, by assuming that data has a clear structure, e.g., low rank [Liu

et al., 2012], the missing values are effectively recovered via matrix factorization [Zhang et al., 2006]. A high-level understanding of recovering the missing values is ensuring the recovered data points sampled from the data manifold [Jakobsen et al., 2017]. Note that the aforementioned research has formulated it as a pre-processing-like problem because the learning task is not involved with data imputation. Other work also aimed to do joint learning, namely learning with recovering the missing values [Ipsen et al., 2020].

Handling missing attributes/labels via semi-supervised learning

Instead of considering general incomplete data, I focus on the case where the missing values happen at some specific dimension to which discrete categories are assigned. Learning with such incomplete data can be solved by Semi-Supervised Learning[2] (SSL) [Chapelle and Scholkopf, 2006]. Here are two practical examples that support this idea.

- Gender parity on the job search platform Seek[3]. The research scientists from Responsible AI department of Seek are in charge of gender parity during recruitment, also known as fairness learning [Chouldechova, 2017] in the academic domain. A practical problem is that gender parity typically relies on an instance-wise gender indicator which is often unavailable from resumes. This is quite an interesting topic because resumes without explicit gender information avert gender discrimination to a certain degree while it hinders the evaluation of fairness. One possible workaround is to set human-recognized genders as reliable labels for resumes and use SSL to predict the rest. After that, different fairness metrics then can be applied.

- Invariant representation learning without complete domain indices. With full domain labels, domain invariant representation can be learned by various techniques, such as adversarial training [Ganin and Lempitsky, 2015], or mutual information [Moyer et al., 2018]. However, domain labels are not always available during data collection while human labeling is often expensive. Learning with partially labeled data only cannot achieve a satisfactory performance because the invariant modeling on labeled instances might be insufficient. In practice, it has been found that using SSL to recover the missing domain labels is powerful to improve the learning performance [Wu et al., 2022].

[2]Consider the following fact $p(x) = p(x_o, x_m) = p(x_m|x_o)p(x_o)$, where the complete sample x is the concatenation of observed features x_o and missing values x_m. From the aforementioned likelihood view for incomplete data, one can approximate the right-hand expression by a prediction model from x_o to x_m plus a proper regularizer, under the assumption that x_o and x_m are dependent.

[3]https://www.seek.com.au

Note that learning with incomplete data is *doing attack via inference*. Please see a further discussion about this concern in Section 6.2.

The challenge of predicting for unlabeled data in SSL

Learning to infer the missing attributes or labels via an SSL paradigm is an interesting research topic and many works have been developed [Van Engelen and Hoos, 2020]. The main challenge of SSL is how to induce the model to confidently fit the unlabeled data though the true labels are agnostic. In other words, how does one ensure whether a model has correctly found the true attribute values or true labels? One can trickily rely on an extra labeled validation set to do model selection [Sohn et al., 2020] or to conduct a bi-level optimization [Ren et al., 2020]. However, beyond such techniques, encouraging a classification model to have confident decisions is more crucial. A very important observation is that existing models gave over-certain predictions for unlabeled data, and their non-satisfactory performance on test data suggests its careless incorrectness over unlabeled learning samples. Then the challenge converts to how to remedy such certain but wrong predictions.

My research attention

Although transductive SSL is able to tell us the missing labels of training records, learning a classifier that predicts the labels for unseen samples is targeted in this task. For example, in the gender parity problem during recruitment, the research scientists may expect the model to confidently predict the gender information for future resumes as well. In addition, my research attention for SSL is to derive confident and correct predictions for unlabeled examples. To better align with the current SSL research, I am intending to straightforward dive into the SSL study in Chapter 4 and leave the discussion for addressing practical concerns into Section 6.2.

2.2.3 Learning with inaccessible data

In this part, the entire data is viewed as restricted data to learning executor (and to any others as well) and learning is relying on the agreed two-way communications between data and model (Please recall the third type of restricted data in Section 2.1). Fig. 2.3 illustrates a general pipeline of this learning problem.

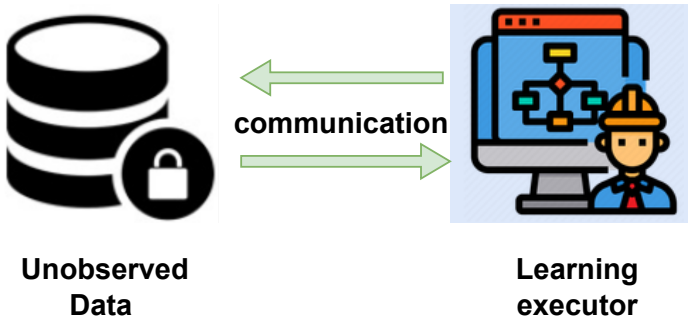


Figure 2.3 Learning executor cannot access data but receives some feedbacks instead.

Learning from query-feedbacks

According to Fig. 2.3, one of the appealing properties of this setting is that data is preserved locally, which shares a similar motivation with Federated Learning (FL) [Konečnỳ et al., 2016]. Despite the regular one-to-many framework, FL enables a local actor to contribute to the global model without sharing its data, thus allowing it to address a series critical data security issues. Nevertheless, the success of FL is entrusting the gradient computation in terms of local data to local devices. In a one-to-one framework, FL reduces to a standard model learning process with the full-batch gradient. In particular, the gradient used in FL is argued too informative to be safe by recent attack study [Yin et al., 2021].

Another property is that two-way communication can be viewed as a query-feedback process. Learning executor submits a query and obtains a corresponding feedback which encodes some statistics of unobserved data. If the query is a model candidate then the feedback can be some evaluations on the unobserved data. That suggests such a problem is hopefully solved by evolutionary algorithms [Emmerich and Deutz, 2018]. However, searching for the optimal model with heuristic evolutionary algorithms is less efficient, especially when the dimension of model parameters is extremely high.

Task clarification

In practice, the budget of communications is limited, which means some learning tasks are not suitable for such a restrictive scenario. Based on this insight, I present two points to clarify the task.

- Model tuning tasks are targeted instead of training from scratch. Compared with training from a randomly initial model, tuning tasks commonly need fewer efforts to reach the optimum. In addition, many of the model parameters are frozen during tuning which reduces the search space. Without loss of generality, if one confronts a

training task, he/she can do pre-training on a similar task beforehand and then conduct model tuning.

- The queries are some (partial) model parameters in practice. If the bandwidth of communication is also limited, only the changed parameters are sent to the target data, without the need of sending the whole model. Regarding the feedbacks, they are usually as simple as scalar scores, e.g., classification accuracy.

My research attention

Tuning a model with query-feedback information only is a difficult but interesting task. Imagine in a Kaggle competition, every team has a number of chances to submit the learned model to the organizer, and the best performance will be reported on the ranking board. Although one can modify the model structure, reorganize the training data, or tune the hyper-parameters based on feedbacks, the proposed learning task tries to use the feedback information to tune the model parameters directly without the need for re-training (For example, training is time-consuming or relies on many computation resources). In other words, the goal of such a learning task is to greedily fit the target data instead of concentration on the source task anymore. In particular, to make it practical to use, I consider the tuning for deep neural networks in the main body of this thesis (See Chapter 5).

2.3 Gradient Manipulation

Gradient-based optimization has been demonstrated powerful in machine learning. For an optimized minimization problem, if a closed-form solution is not available, gradient-based optimization is then thought of as a useful candidate[4]. For a convex function, gradient-based optimization can reach the global optimum by carefully scheduling the learning rate, and it tweaks its parameters iteratively to a local minimum if the function is non-convex. From this perspective, one can say

It is gradient that shapes the eventually learned model.

Recall that the macro challenge of this thesis (See Section 1.3) is to create a general principle for instructing the learning with different types of restricted data. As gradient is crucial to model learning, it inspires me to manipulate corresponding gradients to confront different

[4]The closed-form solutions which suffer from high computational costs are also replaced by gradient-based optimization in practice.

restricted data. Note that unlike the literature where gradient manipulation [Chen et al., 2021] typically refers to some specific operature over gradients, I use this term here to inspire the analyses to the challenge when restricted data is taken into consideration.

2.3.1 Gradient perturbation

Gradient is mathematically a slope of a function which is calculated over input data given a specific model. With different needs, one can post-process gradients by different approaches before they update the model parameters. For example, gradient clipping [Zhang et al., 2019] is used to avert exploding gradients which is thus accelerating the training process, and gradient penalty [Gulrajani et al., 2017] in generative models is to stabilize the learning of discriminators.

As presented in Section 2.2.1, stochastic gradients can be intentionally perturbed to ambiguate the participation of any data point, in a manner of post-processing gradient manipulation. Intuitively, the goal of gradient perturbation in DP is

$$g^{per}(\mathscr{B}) \approx g^{per}(\mathscr{B}'), \quad \mathscr{B}' = \mathscr{B} / \ x, \forall x \in \mathscr{B}, \tag{2.1}$$

where $g^{per}(\mathscr{B})$ denotes the perturbed gradient w.r.t. a batch of samples $\mathscr{B}$, and $\mathscr{B}'$ is any subset of $\mathscr{B}$ with the size of $|\mathscr{B}| - 1$. As the perturbation is with some randomness, the approximation of Eq. (2.1) is factually over a statistical measurement. More strictly, the maximum discrepancy therein is required to be upper bounded by a privacy budget.

Based on the idea of implementing DP with gradient perturbation, my work for studying private pairwise data in distance metric learning is then understood by investigating the gradient influence of a pair of data. However, since the wide existence of data correlation in pairwise data, the major concern here is to investigate whose gradients matter in such a framework.

2.3.2 Gradient masking

Unlike post-processing gradient manipulations where stochastic gradients are ready to use, SSL applications demand reliable gradients by model designing (The gradients for unlabeled samples are possibly unreliable due to the incorrect pseudo labels). For example, the local smoothness [Miyato et al., 2018] is proposed to enhance the gradient reliability of unlabeled samples. As mentioned in Section 2.2.2, one of the deficiencies of existing SSL methods is their over-confident fitting on hard unlabeled data. Back to the well-known pseudo-labelling technique [Lee et al., 2013], it masks the gradient of hard unlabeled samples by a human-

designed threshold τ on the model's output which selects reliable unlabeled samples to update in every iteration, i.e.,

$$g(x) = \begin{cases} g(x), & \text{if } \max p_i(x) > \tau, \\ 0, & \text{otherwise.} \end{cases} \tag{2.2}$$

However, such strategies are found that the predictions for all the unlabeled samples become certain after training (See Section 4.5.2), even some are incorrectly predicted (because the actual training error is non-zero, especially when the label proportion is quite low). Thus, my work for addressing the overconfident predictions follows the similar idea but has an important expectation. That is, the proposed method is supposed to leave the hard unlabeled data under-fitted if the model is not sufficiently confident. In particular, instead of doing sample-wise masking like PL [Lee et al., 2013], our method resorts to class-wise discovery.

2.3.3 Gradient estimation

Learning with inaccessible data keeps data and models isolated with each other and only allows controllable two-way communications (Refer to Fig. 2.3). Thus, target data cannot be used to calculate gradients like that of Section 2.3.1 and 2.3.2. To take advantage of gradient-based optimization, learning under such a restrictive setting is to construct gradients for tuning, known as gradient estimation [Mohamed et al., 2020] from literature. With this understanding, gradient manipulation in this context is interpreted as extracting gradient information by utilizing query-feedback information. Let $g(\mathscr{D})$ be the gradient function with the inaccessible target data $\mathscr{D}$ as input. Gradient estimation is then described as

$$g^e(\{q_i, f_i\}_{i=1}^b) \xrightarrow{\text{approximate}} g(\mathscr{D}), \tag{2.3}$$

where g^e is the constructed estimator whose input are query q_i and feedback f_i, and b is the number of query times needed for each gradient estimation.

By instancing query as model candidate and feedback as model performance, model tuning is viewed as an optimum search problem with a warm start. The difficulty lies in two points. One is how to do efficient estimation, i.e., the construction of $g^e(\cdot)$, and the other is how to make such an estimation practical even if the tuned model is with complex structure.

Chapter 3

Distance Metric Learning with Private Pairwise Data

This chapter first analyzes how pairwise information is leaked to attackers in Distance Metric Learning (DML). By studying the limitation of Differential Privacy (DP) for preserving the privacy of pairwise data, I reformulate the worst-case assumption for pairwise data preservation by introducing an extended privacy definition, namely Differential Pairwise Privacy (DPP). Compared with the traditional DML algorithm, the novelty of the proposed DML lies in the tailor-designed randomness which is used to compensate for the difference between the original dataset and the prior knowledge of attackers. Therefore, the randomized DML is able to output hardly distinguishable distance metrics no matter which one is the actual input dataset, the original dataset or the possible prior knowledge of the attacker for the worst case. As a result, even provided with infinite query chances, no more useful information can be obtained by attackers.

3.1 Problem Understanding

3.1.1 Pairwise data leakage in distance metric learning

The distance/similarity between two samples (e.g., euclidean distance) is the base of many applications [Bellet et al., 2013], such as clustering, classification, information retrieval, etc. Distance Metric Learning (DML) is a fundamental tool that learns a distance metric over the data to support these applications. It expects that in the projected space the similar samples would be better grouped; while the dissimilar ones would be appropriately separated. Such a principle is properly spoken of in [Xing et al., 2003] and much subsequent research [Tsang and Kwok, 2003, Kwok and Tsang, 2003, Weinberger and Saul, 2009, Guillaumin et al.,

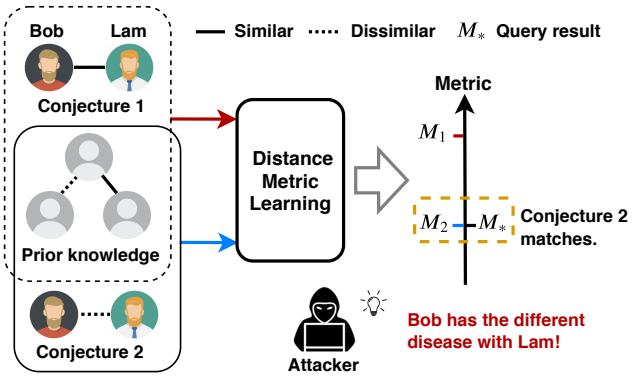


Figure 3.1 Leakage of pairwise relationship. An attacker with all the prior knowledge of the dataset except the target relationship between Bob and Lam, is able to infer their real relationship by matching the conjecture and query results.

2009, Ying and Li, 2012, Hu et al., 2014, Sohn, 2016, Xie et al., 2018] has followed this criterion.

The training data fed to DML model is often pairwise labelled[1] which naturally encodes some secrets when the data is collected from humans. A popular application of learning distance metric would be healthcare data [Wang et al., 2011, Huai et al., 2018, Suo et al., 2019]. A pairwise relationship in this application might be, for example, two patients have the same/different disease(s). However, the relationships in the training data can be leaked to attackers through a deterministic and resilient DML model. I give a white-box attack example to explain a scenario that a traditional DML model may unfortunately leak the pairwise relationship to external attackers.

A privacy leakage scenario. Assume that one has a set of pairwise medical records as the training data, a DML model is trained over the entire pairwise data and returns a static metric M_* (e.g., oracle). Suppose that a powerful attacker understands the DML model and has some prior knowledge, i.e., knowing partial pairwise data of the entire training set (e.g. $K-1$, where K is the total number of pairs). The attacker would like to exploit a particular relationship (e.g., Bob and Lam) which the attacker does not know. First, she/he combines the prior knowledge by checking two possible conjectures: (1) Bob and Lam have the same disease, and (2) Bob and Lam have different diseases. Then, the attacker feeds them to the DML model separately and obtains corresponding conjecture results M_1 and M_2. Lastly, she/he compares the two results with the oracle M_*. The matched conjecture will expose the correct pairwise relationship between Bob and Lam, leaking the private relationship to the attacker. Fig. 3.1 depicts this process.

[1]This setting is called weakly supervised DML in some literature but is overwhelming in metric learning community.

3.1.2 Differential privacy and its limitation for pairwise data

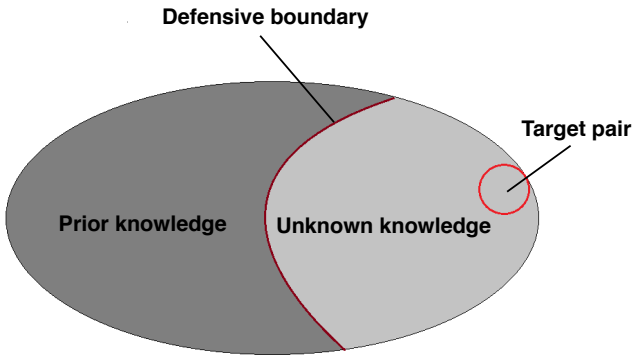


Figure 3.2 Knowledge diagram. The prior knowledge is supposed to be smaller than the whole data deducting the target pair because of the data correlation. For a given target pair, there always exists a corresponding defensive boundary which restricts the volume of prior knowledge in practice.

Differential Privacy (DP) [Dwork et al., 2006] has become a popular and widely accepted privacy framework in recent years. It is initially used in data mining and data release research community and recently moved to the machine learning community [Ji et al., 2014]. It aims to ensure that the participation of an individual sample never changes the probability of any possible outcome by much. This goal is usually achieved by adding perturbation during modeling. As a result, the model is able to protect against the powerful attackers who have access to the whole dataset except the targeted sample. This conclusion is derived from the DP's assumption that all the samples are independent, so that DP can defend against the worst case scenario, i.e., except the target sample, all the remaining data (K-1 samples) are the prior knowledge of an attacker. Unfortunately, this assumption is broken in pairwise data, even if a pair of data is treated as a single sample. For example, the disease relation between Alice and Bob in Fig. 3.3 can be inferred by Jim since Jim has the same disease as both Alice and Bob. Therefore, attackers does not need K-1 samples to get the same prior knowledge as the worst case. In the pairwise data setting, the worst case assumption (K-1 samples in an attacker's prior knowledge) in DP is not valid anymore. The DP assumption, which can be seen as a defensive boundary for securing pairwise privacy, needs to be redefined to understand the defence capability of DP for pairwise data. This thought is depicted as Fig. 3.2, where the defensive boundary would be shrunk to the target pair's boundary if data are independent.

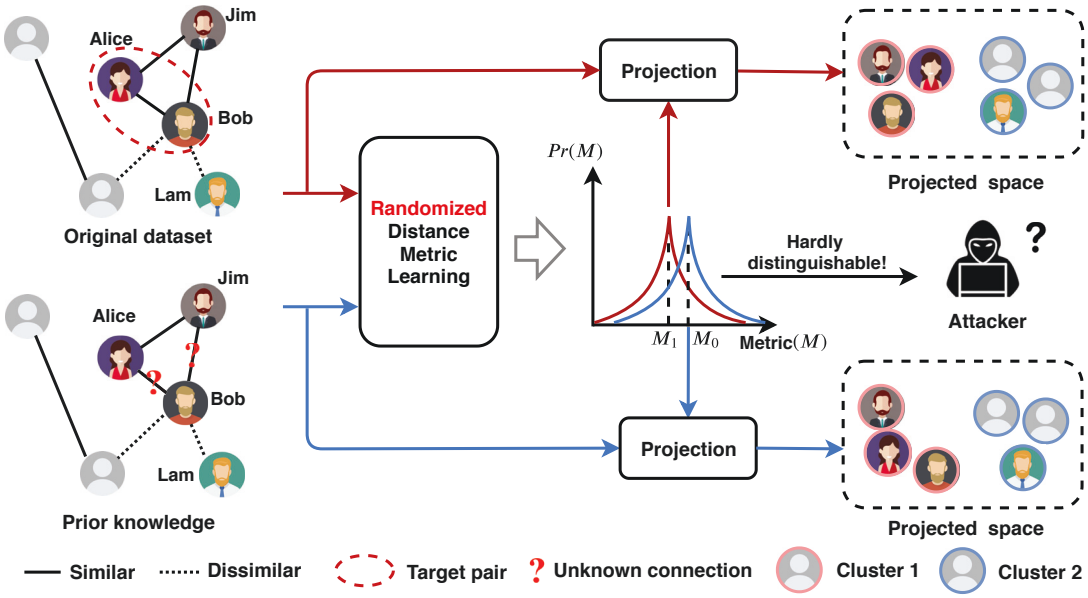


Figure 3.3 Preserving privacy of pairwise relationship. Suppose the relationship between Alice and Bob is the target. The attacker may have the prior knowledge that excludes edges with question mark. This provides one of the worst cases, where the relationship of Alice and Bob cannot be inferred from prior knowledge. DPP ensures that the prior knowledge of the attacker for the worst case has the hardly indistinguishable output with the original dataset. Particularly, the obtained metric M_0 is expected to group training data as M_1 does.

3.2 Preliminaries

Notation: A pairwise datum is a tuple $z_{ij} = (\Delta x_{ij}, y_{ij})$, where $\Delta x_{ij} = x_i - x_j \, (i \neq j, \Delta x_{ij} \in \mathbb{R}^d)$ is the feature difference between the individual i and j, and pairwise label y_{ij} is a binary variable that encodes their relationship. Plus, x_i could also be any representation learned by deep embedding. Omitting the subscript of z_{ij} whenever it clearly indicates a single pair, and then $Z = \{z^1, z^2, ..., z^K\}$ is a dataset of K pairs which are the input of a DML algorithm. For a vector v, I use $||v||$, $||v||_2$ to denote its ℓ_1-norm and ℓ_2-norm, respectively. In a graph, $\langle , \rangle$ denotes a pair of nodes and $(,)$ denotes an edge or a path.

Definition 1 *(Edge-disjoint s-t paths) Given an undirected graph and two nodes in it, source s and destination t, two paths from s to t are said edge-disjoint if they do not share any edge.*

Definition 2 *(Lipschitz function over model parameter θ) A loss function $f : \mathscr{C} \times \mathscr{Z} \to \mathbb{R}$ ($\mathscr{C}$, $\mathscr{Z}$ are parameter space and input space separately.) is h-Lipschitz (under ℓ_1- norm) over θ, if for any $z \in \mathscr{Z}$ and $\theta_1, \theta_2 \in \mathscr{C}$, then $|f(\theta_1, z) - f(\theta_2, z)| \leq h||\theta_1 - \theta_2||$.*

Definition 3 *(ε-Differential Privacy [Dwork et al., 2006]) A randomized algorithm $\mathscr{A}$ is said to guarantee ε-differentially private if for all datasets $\mathscr{Z}$ and $\mathscr{Z}'$ satisfying $Dist(\mathscr{Z}, \mathscr{Z}') = 1$ ($\mathscr{Z}'$ is called the neighboring dataset of $\mathscr{Z}$, and $Dist(,)$ means the number of records two datasets differ.) and for any possible algorithm output o the following holds:*

$$Pr[\mathscr{A}(\mathscr{Z}) = o] \leq e^{\varepsilon} \cdot Pr[\mathscr{A}(\mathscr{Z}') = o], \tag{3.1}$$

where $Pr[\cdot]$ is w.r.t. the randomness in $\mathscr{A}$, and the non-negtive parameter ε is known as privacy budget.

Sensitivity based methods for ε-DP leverage the Laplace Mechanism [Dwork et al., 2014] which has the following definitions.

Definition 4 *(ℓ_1-sensitivity) The ℓ_1-sensitivity of a function $g : \mathscr{Z} \to \mathbb{R}^d$ is:*

$$\Delta g = \max_{Dist(\mathscr{Z}, \mathscr{Z}')=1} \|g(\mathscr{Z}) - g(\mathscr{Z}')\|. \tag{3.2}$$

Definition 5 *(Laplace Mechanism) Given a function $g : \mathscr{Z} \to \mathbb{R}^d$, the Laplace Mechanism $\mathscr{M}_{Lap}$ is defined as:*

$$\mathscr{M}_{Lap}(\mathscr{Z}, g, \varepsilon) = g(\mathscr{Z}) + Y, \tag{3.3}$$

where Y is drawn from Laplace distribution $Lap(0, b)$ with $b = \frac{\Delta g}{\varepsilon}$. Laplace Mechanism preserves ε-differentially private.

Remark 1 *Definition 3 can be relaxed to the* approximate *DP according to [Dwork et al., 2014] if Definitions 2-5 are properly modified. The proposed* DPP *can be also relaxed with the same style. I leave these analyses to Appendix A.3 for a supplemental discussion.*

Distance Metric Learning. DML aims to learn a representative distance between i and j defined by $\text{Dist}_M(i,j) = \sqrt{\Delta x_{ij}^T M \Delta x_{ij}}$, where $M \in \mathbb{R}^{d\times d}$ is the distance metric, a.k.a., a symmetric positive semidefinite matrix. The original form of M refers to the case where i and j are drawn from the same distribution with covariance matrix Σ, with $M = \Sigma^{-1}$. According to the literature [Weinberger and Saul, 2009, Guillaumin et al., 2009, Ying and Li, 2012, Hu et al., 2014, Sohn, 2016, Xie et al., 2018, Li et al., 2018], this metric can be better learned once the pairwise labels are provided. Practically, pairwise label is often denoted as a binary variable $y_{ij} \in \{0,1\}$ to indicate category. For instance, if two samples i and j are from the same class, then $y_{ij} = 0$, and if they are from different classes, $y_{ij} = 1$. Let $M = W^T W$, where the transformation matrix $W \in \mathbb{R}^{d'\times d}$ $(1 \le d' \le d)$ is free of any constraint. As a result, W can be optimized by minimizing the projected distance between similar samples while maximizing the distance between dissimilar ones.

Recipe of gradient perturbation. Gradient based perturbation can be applied to most machine learning models including deep neural networks. Here is a recipe of DP in machine learning via the gradient perturbation approach. It mainly consists of 4 components.

- Privacy principle satisfying some requirements.
- Loss function designed for a learning task.
- Upper bound for the gradient sensitivity.
- Noise injection during optimization.

3.3 Pairwise Relation in Distance Metric Learning

3.3.1 Privacy investigation

Since the pairwise data encodes the interpersonal relationship, preserving the privacy of involved pairwise relationship is the core problem in this work. The algorithm designer is thought as the trusted party, and thus the whole dataset can be directly fed into the algorithm, i.e., DML. The user is returned an available distance metric after they submit a query to the system (DML algorithm). To prevent any potential attacks in this setting, the algorithm

designer is responsible to develop a DML algorithm which guarantees the privacy of every pairwise relationship.

Practically, I am playing the role of the algorithm designer. Inspired by [Kifer and Machanavajjhala, 2012], I adopt the following principle to preserve pairwise relationship. **If an attacker fails to infer the target relationship through her/his prior knowledge, she/he cannot obtain more information by querying the DML algorithm either.** Based on this principle, I show three key insights.

- Defending against the attacks that need query. If a targeted sample can be inferred through its relations with other samples in the prior knowledge of the attacker, privacy preserving techniques, e.g., DP, would never be possible to defend against such attacks. Therefore, as a algorithm designer, I am supposed to recognize and formulate the maximum prior knowledge of an attacker, and prevent them knowing more about data during their interactions with the DML algorithm. In contrast, the users who can already infer the target relationship through their prior knowledge is outside of my scope, because they do not need to query a DML algorithm output.

- Preserving both feature difference and pairwise label. The task is motivated by preserving the privacy of pairwise relationship, i.e., y_{ij}, the last element of the tuple z_{ij}, similar to *Attribute DP* introduced in [Kifer and Machanavajjhala, 2011]. Unfortunately, it is actually not adequate to solely preserve y_{ij}. Recalling the goal of DML, the distance change between two individuals reflects their relationship. For instance, two samples close to each other but far away in the projected space are likely labeled as dissimilar, i.e., $||\Delta x_{ij}||$ is small while $\text{Dist}_M(i, j)$ is large. As DML is to return an available distance metric, hiding the information of Δx_{ij} can avert this kind of leakage. Therefore, it is concluded that to achieve the privacy of any pairwise relationship between i and j, the privacy concern for Δx_{ij} and y_{ij} are both needed. Please note it is not equal to preserving a single tuple z_{ij}, and see Section 3.4.1 for more details.

- Enhancing the utility by introducing randomness as less as possible. Existing works [Chaudhuri et al., 2011, Chaudhuri and Monteleoni, 2009, Kifer et al., 2012, Song et al., 2013, Bassily et al., 2014, Talwar et al., 2015, Zhang et al., 2017b, Wang et al., 2017, Talwar et al., 2014] applying DP in machine learning algorithms have shown introducing randomness to the machine learning model is provable to preserve data privacy. Compared to DP in which randomness is only used to compensate for the change of any single sample, the privacy cost for pairwise data is apparently higher because there are more than one pair's change needs compensating. As a result, the utility of output distance metric is decreased. This problem is alleviated by using

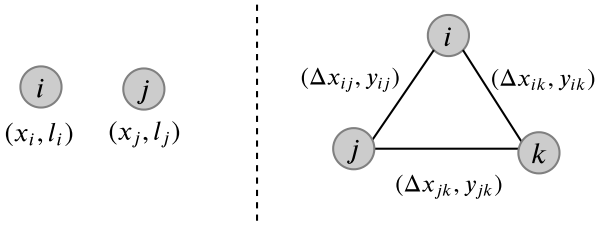


Figure 3.4 Comparison between the samples fed to classification or regression models and pairwise data fed to DML algorithms. **Left:** Any two samples composed of the feature x_i (x_j) and its label l_i (l_j) are independent in existing ERM-based works. **Right:** Pairwise data are correlated with each other because an individual may participate in multiple pairs.

a sensitivity reduction technique (See Section 3.5.2) which reduces the amount of injected noises in stochastic learning.

3.3.2 Clarification

Two peculiar properties of DML are clarified in this part.

Transitive vs intransitive relationship. Pairwise labels have different semantics in different context, and it is noticed that they are not identical for different cases. Summarily, there are mainly two types of pairwise relationship; one is transitive and the other is intransitive. Transitive relationship is like having the same disease or working in the same company, and intransitive relationship is like being friends or hanging out together. In the main body of this paper, I focus on the transitive relationship and attribute intransitive relationship as its special case. It is reasonable because transitive relationship requires more privacy concern for the transitivity risks. Please note the naive edge differential privacy (edge DP) [Hay et al., 2009, Rastogi et al., 2009] is not applicable even for the intransitive relationship. The analysis of intransitive relationship is left to Section A.2.

DML vs classification/regression. A formulated DML loss function can be seen as an instance of ERM-based [Vapnik, 1992] model. From this view, the transitivity property of pairwise data makes DML distinct from existing privacy works for classification and regression [Chaudhuri et al., 2011, Song et al., 2013, Abadi et al., 2016, Wang et al., 2017]. Fig. 3.4 exhibits their comparison over input data. Concretely, any two data points fed to existing ERM-based models are usually independent with each other, which naturally matches the DP's definition. For pairwise data, the dependence happens due to the correlation both on feature differences and pairwise labels indicated as edges on the right of Fig. 3.4. Please note although two types of correlation are caused by the same reason, they are virtually not the identical problem. The specific details are discussed in Section 3.4.

3.4 Differential Pairwise Privacy from Graph Perspective

Consider each individual as a node and every pairwise data z_{ij} as an edge, which comes to an *undirected* graph $G = (V, E)$. Please note the node set V only keeps the identity of individuals while the edge set contains all the information used for DML training, i.e., $E = \{z_{ij} = (\Delta x_{ij}, y_{ij}) | (i, j \in V^2 \wedge i \neq j\}$. Thus, all the private information used to learning is on edges. Based on this graph, Differential Pairwise Privacy (DPP) is presented in this section. I will show this definition is a nontrivial extension of DP (a.k.a. edge DP in the context of graph data) over the pairwise data.

3.4.1 Privacy concern on edge

Let $\langle s, t \rangle$ denote the target pair whose pairwise relationship is the interest of the attacker. According to the statement in Section 3.3.1, given the original graph G, I need to formulate the prior knowledge G' of the attacker.

Pairwise relationship correlation. For a binary category case, there are three basic relationship inference patterns[2]. Let $\sim, \nsim$ denote that two individuals are from identical and different categories respectively, and C is the category number. I have

- $s \sim i, i \sim t \Rightarrow s \sim t$
- $s \sim i, i \nsim t \Rightarrow s \nsim t$
- $s \nsim i, i \nsim t \Rightarrow s \sim t$ (iff $C = 2$)

Thus, it is concluded that where there is a path between two nodes there might be a possible inference exposing their relationship. That is to say, for a target pair $\langle s, t \rangle$, all the paths between s and t provide useful inferences. According to Menger's Theorem [Göring, 2000], the least cost of preventing these inferences is properly breaking down $|P_{st}|$ edges, where P_{st} is the set of all edge-disjoint s-t paths. Fig. 3.5 shows an example to state this thought. Fig. 3.5 (I) is the derived graph G formed by the given pairs. In terms of the nodes s and t, there are totally three paths from s to t, i.e., (s, t), (s, c, e, t), and (s, c, u, t). Two of them are edge-disjoint paths which are indicated by two arrows respectively in Fig. 3.5 (II). Obviously, an instant way to prevent the inference is to break the key edges, i.e., (s, c) and (s, t), shown as the dashed edges in Fig. 3.5 (III).

Feature difference correlation. According to linear algebra, when an individual is included in a circle, its feature can be calculated since the degree of freedom equals to the

[2]The number of inference patterns are actually determined by the category number, but even so I can still come to the consistent conclusion if similar pairs are dominant in the correlations.

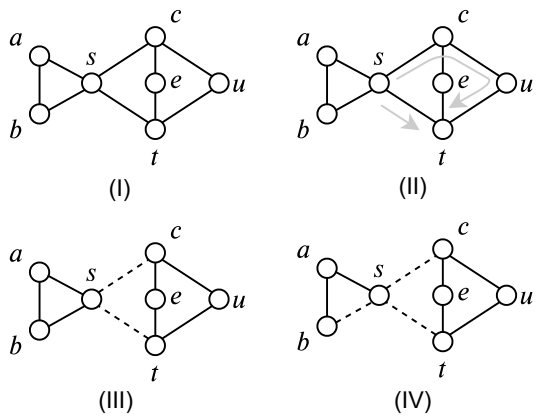


Figure 3.5 Construction of neighboring graph w.r.t. the pair $\langle s,t\rangle$. (I) The graph encoding all the pairwise data. (II) Disjoint-edge identification. (III) Two key edges (s,c) and (s,t) determining the relationship inference. (IV) Edge (b,s) exposing the feature of the individual s.

number of constraints (i.e., edges). As shown in Fig. 3.5 (IV), the feature of s is exposed once the information of edges (s,a), (a,b) and (b,s) is known. As hiding the feature of either s or t is adequate to privatize Δx_{st}, one of the possible options is to delete the edge (b,s) in this example. It is observed that breaking down the edge-disjoint paths sometimes help decrease the cycles used for feature inference, e.g., s is also in the cycle (s,c,e,t,s). Thus, the edges determining Δx_{st} inference are only searched in the subgraph $G-P_{st}$, i.e., removing all edges belonging to P_{st} from G. Let c_s, c_t denote the number of edges that isolate the nodes s and t from the possible cycles over $G-P_{st}$, respectively. The minimum number of edges that should be deleted is $\min\{c_s,c_t\}$.

Summarily, it is concluded that the attacker who targets the relationship between s and t should at least miss $|P_{st}|+\min(c_s,c_t)$ edges in G. This quantitative measure actually generalizes the attacker's prior knowledge but makes it easy to do formulation. As any pair is the potential target pair, the attacker's prior knowledge G' should satisfy

$$\mathrm{Dist}(G,G') \geq \kappa, \tag{3.4}$$

where $\mathrm{Dist}(,)$ means the number of edges two graph differs, and the introduced variable κ is calculated by traversing the entire graph

$$\kappa = \max_{\forall s,t\in V, s\neq t} \{|P_{st}|+\min(c_s,c_t)\}. \tag{3.5}$$

Particularly, G' is dubbed as κ-neighboring graph of G if the equality exactly holds in Eq. (3.4) following the convention of DP.

3.4.2 Differential Pairwise Privacy (DPP)

For a graph G, the attacker is called κ-*Att* if her/his prior knowledge is exactly a κ-neighboring graph of G. An attacker with fewer edges prior, i.e., $\text{Dist}(G,G') > \kappa$, is unable to know more than κ-*Att*, while an attacker knowing more edges, i.e., $\text{Dist}(G,G') < \kappa$, is likely to have known the target pair without any need of querying the DML algorithm. Thus, κ-*Att* defines the worst case for pairwise data.

From Eq. (3.5), κ is only determined by the given pairwise data. Although searching the exact value of κ is not an interactive component of the DML algorithm, it is very challenging to compute in reality (for full batch gradient descent). To overcome this dilemma, an alternative approach to effectively calculate the value of $\kappa$is provided which is shown in Appendix A.1.

Suppose κ is known, based on the above analyses, the definition of differential pairwise privacy is then described as follows.

Definition 6 *(ε-Differential Pairwise Privacy (*DPP*)). A randomized DML algorithm $\mathscr{A}_{DML}$ is said to guarantee ε-differentially pairwise privacy if for all datasets G, G' satisfying $Dist(G,G') = \kappa$ and for any possible output o_M the following holds:*

$$Pr[\mathscr{A}_{DML}(G) = o_M] \leq e^{\varepsilon} \cdot Pr[\mathscr{A}_{DML}(G') = o_M]. \tag{3.6}$$

DPP inherits good advantages of DP but works on pairwise data. Here presents two valuable peculiarities of DPP.

- Bearing the individual participation. DPP potentially allows an attacker to have prior knowledge about the participation of the target individuals; it de facto privatizes the participation of the involved pairwise relationship, i.e., the connected edge. This is consistent with common sense. For instance, if an attacker wants to know whether Alice and Bob have the same disease, he/she must ensure they are contained in the dataset beforehand.

- Defending against implicit attacks. No assumption is made over the given pairwise data, because data is often collected and labelled by some domain experts (or labors) who are agnostic to the downstream privacy issues. By the proposed DPP, the learning executor guarantees that pairwise data is restricted to any others as one cannot acquire extra knowledge from the private learning process.

Implement DPP in DML is challenging, because simply using off-the-shelf approaches will reduce the utility of the training model. For example, a large κ will result in a heavy noise injection, which is the key problem we are focusing on later.

3.5 Private Distance Metric Learning

In this section, the proposed DPP is applied to DML with the contrastive loss [Chopra et al., 2005, Guillaumin et al., 2009, Xie et al., 2018] as a study case. Following the recipe of DP in Section 3.2, gradient perturbation is used to implement DPP. In addition, under the framework of stochastic gradient descent, the utility of the proposed DML algorithm is further improved by the sensitivity reduction.

3.5.1 Differential pairwise privacy with contrastive loss

Contrastive loss is a classic DML model which measures pairwise data in a projected space for better fitting the pairwise labels. For simplicity, $z_{ij} = (\Delta x_{ij}, y_{ij})$ is denoted by $z = (\Delta x, y)$, and the loss function is written as

$$L(W,z) = \frac{1}{2}(1-y)D_W^2 + \frac{1}{2}y\{\max(0, m - D_W)\}^2, \tag{3.7}$$

where W is the transformation matrix, $D_W = \|W\Delta x\|_2$ is the distance of data point i, j in the projected space, and $m > 0$ is a margin threshold used for avoiding collapsed solutions.

It is noted that gradient computation is the unique component of DML algorithm that interacts with the input data. As gradient is an aggregated information which indicates the membership, gradient perturbation [Song et al., 2013, Bassily et al., 2014, Abadi et al., 2016] has been used to implement data privacy by sacrificing some model performance.

As each row of W is independent, the gradient w.r.t $W_r (r = 1, 2, ..., d')$ is

$$g_r(z) = \begin{cases} W_r \Delta x \Delta x^T & y = 0 \\ \frac{D_W - m}{D_W} W_r \Delta x \Delta x^T & y = 1, D_W < m \\ \mathbf{0} & y = 1, D_W \geq m \end{cases}. \tag{3.8}$$

Before further computing the gradient sensitivity, I introduce the following lemma.

Lemma 1 *Given a pair $(\Delta x, y)$, the gradient function $\|g_r(\cdot)\|$ is infinite if only if $y = 0$ and $\|W_r\|$ is unbounded.*

Proof: If $y = 0$ and $\|W_r\| \to \infty$, clearly $\|g_r(\cdot)\|$ will be infinite.
If $y = 1$ and $D_W \geq m$, $\|g_r(\cdot)\| = 0$.

If $y = 1$ and $D_W < m$, then I have

$$\begin{aligned} \|g_r(\cdot)\| &= |1 - \frac{m}{\|W\Delta x\|_2}| \cdot \|W_r \Delta x \Delta x^T\| \\ &\overset{①}{\leq} \frac{m\|W_r \Delta x \Delta x^T\|}{\frac{1}{\sqrt{d'}}\|W\Delta x\|} - \|W_r \Delta x \Delta x^T\| \\ &\overset{②}{\leq} \frac{m\sqrt{d'}|W_r \Delta x| \cdot \|\Delta x^T\|}{\|W\Delta x\|} \\ &\overset{③}{\leq} 2m\sqrt{d'}. \end{aligned} \tag{3.9}$$

where ① follows the fact that for any vector $u \in \mathbb{R}^{d'}$ $\|u\| \leq \sqrt{d'}\|u\|_2$, ② utilizes the facts that every induced norm is submultiplicative, i.e., $\|W_r \Delta x \Delta x^T\| \leq \|W_r \Delta x\| \cdot ||\Delta x^T||$, and second term is non-negative, and ③ follows the fact $\|W_r \Delta x\| \leq \|W\Delta x\|$ and triangle inequality, i.e., $\|\Delta x\| \leq \|x_i - x_j\| \leq 2\|x_j\| \leq 2$ where x_i is ℓ_1 normalized.

Summarizing above results completes the proof. ■

According to Lemma 1, $g_r(\cdot)$ is bounded if $\|W_r\|$ is smaller than a const. A tender option is to impose some restriction on W. For example, Jin et al. [Jin et al., 2009] employed a regularizer to measure the complexity of W. Yet, it fails to bound the gradient during the entire optimization. As a result, the sensitivity of gradient $g_r(\cdot)$ will be infinite according to Definition 4. Chaudhuri et al. [Chaudhuri et al., 2011] averted this difficulty by constraining the objective to be h-Lipschitz, which is simply achieved by the gradient clipping technique [Abadi et al., 2016]. Following their schemes and I formulate a new theorem.

Theorem 1 *If the objective function in Eq.* (3.7) *is h-Lipschitz w.r.t. W_r, the ℓ_1 gradient sensitivity Δg_r on a minibatch $\mathscr{B}$ is at most $\frac{2\kappa h}{|\mathscr{B}|}$, where κ is specified by Eq.* (3.4).

Theorem 1 is a direct extension to the Corollary 8 of [Chaudhuri et al., 2011].

Remark 2 *With this gradient sensitivity,I can subsequently make DML algorithm satisfy Definition 6 by adding the noise sampled from the distribution $Lap(0, \frac{2\kappa h}{|\mathscr{B}|\varepsilon})$ to the iterative batch gradient. Particularly, according to [Song et al., 2013], the whole DML algorithm will satisfy ε pure* DPP *if every batch is disjoint.*

3.5.2 Improvement by sensitivity reduction

The amount of injected noise to gradient is determined by the gradient sensitivity. To improve the utility of optimization algorithm, an instant option is to reduce the gradient sensitivity

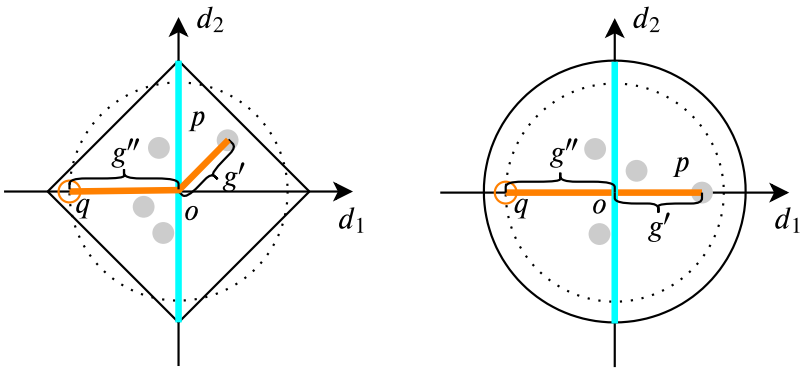


Figure 3.6 Gradient sensitivity reduction w.r.t. minibatch data. **Left:** Individuals are ℓ_1 normalized. The cyan line segment denotes the factor $2h$ specified by Theorem 1. p_{max} is one of all batch members whose gradient value is the largest, q denotes the possible counterpart of p_{max} in the neighboring batch that satisfies Eq. (3.12). The orange line segment connecting p_{max} and q is likely shorter than the cyan one. **Right:** Individuals are ℓ_2 normalized, and representation are consistent with the left. This subfigure is also specified by Corollary 2 in Section A.3.

value. Keeping this thought in mind, I propose a gradient sensitivity reduction approach by exploring the distribution of pairs within a minibatch during the optimization.

Since noise is added to every local batch gradient, it does not cost much to adjust the sensitivity in a minibatch. Specifically, the sensitivity based method aims to find the maximum difference between the given batch and its any possible neighbor over the gradient value. If $\kappa = 1$, it is equivalent to find out which pair is the most sensitive among a group of batch members. Fig. 3.6 presents an example in a two-dimensional space, where p_{max} denotes the pair (label information is folded) which has the maximum gradient value. Without loss of generality, I suppose the larger distance between two pairs, the more different their gradients will be. Thus, Theorem 1 actually takes the biggest sensitivity value represented by cyan segment. In the presented case, p_{max} is the most sensitive pair because it has the largest possible distance to the ℓ_1 norm boundary compared with other pairs. Pair q represents the farthest pair possibly occurring in the neighboring batch. That means the sensitivity value is determined by the distance between p_{max} and q. Inspired by this observation, the gradient sensitivity reduction approach is proposed. In the followings part, p is abused as an input pair of gradient function for convenience.

Theorem 2 *If the objective function in Eq.* (3.7) *is h-Lipschitz w.r.t.* W_r*, the* ℓ_1 *gradient sensitivity* Δg_r *on any batch* $\mathscr{B}$ *is at most* $\frac{\kappa(g'_r+g''_r)}{|\mathscr{B}|}$*, where* κ *is specified by Eq.* (3.4)*, the batch gradient peak* $g'_r = \max(||g_r(p_1)||, ..., ||g_r(p_{|\mathscr{B}|})||)$*, and its possible counterpart* $g''_r = \min\{h, \max(4||W_r||, 2m\sqrt{d'})\}$.

Algorithm 1 DPP-DML Algorithm

Input: Dataset Z containing K tuples
Parameter: Reduced dimension d', margin threshold m, Lipschitz constant h, the distance κ, batch size $|\mathscr{B}|$, privacy budget ε, epoch number $T_{\max}$
Output: Distance metric M

Initialize transformation matrix W randomly, step size $\eta = 1$, counter $\tau = 0$, batch index $it_{bat} = \frac{K}{|\mathscr{B}|}$, privacy budget for an epoch $\varepsilon\prime = \frac{\varepsilon}{T_{\max}}$
for $T = 1, 2, ..., T_{\max}$ **do**
  for $it = 1, 2..., it_{bat}$ **do**
    $\tau \leftarrow \tau + 1$
    $\eta \leftarrow \frac{\eta}{\sqrt{\tau}}$
    for $r = 1, 2, ..., d'$ **do**
      Compute gradient $g_r(\cdot)$ w.r.t each $p_j \in \mathscr{B}_{it}$ by Eq. (3.8) and then do gradient clipping $\overline{g}_r(p_j) = g_r(p_j)/\max(1, \frac{\|g_r(p_j)\|}{h})$
      Compute $g'_r = \max(\overline{g}_r(p_1), \overline{g}_r(p_2), ..., \overline{g}_r(p_{|\mathscr{B}|}))$
      Compute $g''_r = \min\{h, \max(4||W_r||, 2m\sqrt{d'})\}$
      Add noise $g^*_r = \frac{1}{|\mathscr{B}|}\sum_j \overline{g}_r(p_j) + Lap(\frac{\kappa \Delta g_r}{\varepsilon'})$, where $\Delta g_r = \frac{g'_r + g''_r}{|\mathscr{B}|}$
      $W_r \leftarrow W_r - \eta g^*_r$
    end for
  end for
end for
$M = W^T * W$

Proof: The worst case is that two neighbouring sets exactly differ by the constraint $\|\mathscr{B} - \mathscr{B}'\| = \kappa$. Since the pair satisfying $y = 1$ and $D_W \geq m$ has no contribution to the batch gradient, ℓ_1 sensitivity of $g_r(\cdot)$ can be written as

$$\begin{aligned}
\Delta g_r &= \max ||\frac{1}{|\mathscr{B}|}(g_r(\mathscr{B}) - g_r(\mathscr{B}'))|| \\
&\leq \max_j \frac{\kappa}{|\mathscr{B}|}||(g_r(p_j) - g_r(p_j{}'))|| \ (j = 1, 2, ..., |\mathscr{B}|) \\
&\leq \frac{\kappa}{|\mathscr{B}|}(||(g_r(p_{max})|| + ||g_r(q))||).
\end{aligned} \tag{3.10}$$

Let $g'_r := \max(||g_r(p_1)||, ..., ||g_r(p_{|\mathscr{B}|})||)$, and I have

$$||g_r(p_{max})|| = g'_r \leq h. \tag{3.11}$$

Meanwhile, according to the proof of Lemma 1 I have $||\frac{D_W-m}{D_W}W_r\Delta x\Delta x^T|| \leq 2m\sqrt{d'}$. As $||W_r\Delta x\Delta x^T|| \leq 4||W_r||$, the following inequality must hold

$$||g_r(q)|| \leq \min\{h, \max(4||W_r||, 2m\sqrt{d'})\} := g_r''. \tag{3.12}$$

Therefore, I can arrive at

$$\Delta g_r \leq \frac{\kappa(g_r' + g_r'')}{|\mathscr{B}|}, \tag{3.13}$$

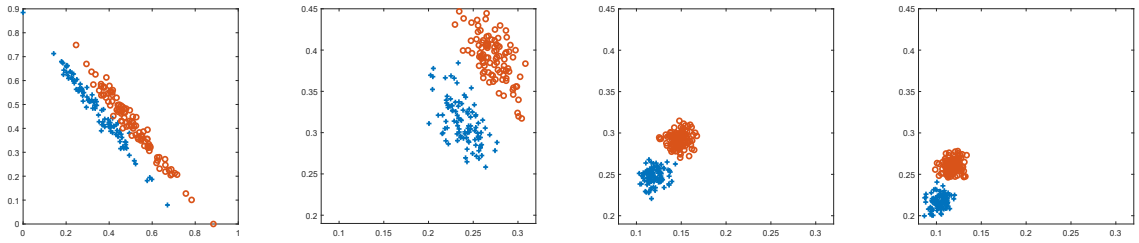
which completes the proof. ■

Remark 3 *Unlike Theorem 1, Theorem 2 is dataset dependent. However, it is easily verified through the worst case Theorem 2 will not expose the participation of any batch member. Specifically, I have $g_r' = h$ if full batch gradient descent is applied. Otherwise, the proposed approach takes advantage of the difference between g_r' and h. Please note a minibatch is usually collected from a component of G (G could be* disconnected*) according to [Hu et al., 2014, Sohn, 2016], instead of globally sampling over the entire dataset D. Thus, g_r' is diverse across different batches. It is noted that sensitivity reduction trick also benefits from the difference between g_r'' and h. The reason is that the gradient function of contrastive loss is piecewise different. This point is attributed to the property of the DML loss function.*

Combining with the steps of optimizing the contrastive loss, I summarize the entire process into Algorithm 1 which naturally satisfies Definition 6. For Laplacian mechanism, it is known that $Var(||g_r||) \propto (\frac{\Delta g_r}{\varepsilon'})^2$, where ε' is the fixed privacy budget for each minibatch. This means the smaller sensitivity factually implies the better utility of gradient. From the composition theory [McSherry, 2009], the privacy budget for an epoch is still ε'. Particularly, once T_{max} epochs are needed for better convergence, then the accumulated privacy budget should be $\varepsilon = \varepsilon' T_{\max}$.

3.6 Experiment

This section contains four parts. In the first part, the efficacy of the proposed method is validated on a synthetic dataset. In the second part, based on the same privacy mechanism, i.e., Laplace mechanism, DPP is further compared with other two methods on four real-world benchmarks. Their performances are evaluated by classification accuracy on the test set. By replacing the based privacy mechanisms, I demonstrate the effectiveness of sensitivity reduction in the third part. In the last part, the effects of different parameters are investigated. Through all the experiments, the ℓ_1-norm of every sample is preprocessed smaller than 1 before used to compute the feature difference. To guarantee the convergence, all the stochastic



(a) Original (b) DPP (c) DPP-S (d) NonPriv

Figure 3.7 DML projects original data into a new space. (a) A synthetic dataset containing 200 data points drawn from two aligned strips. (b)-(d) Data distribution after applying the metric learned by contrastive loss with DPP, DPP-S (with sensitivity reduction), and NonPriv concern, respectively.

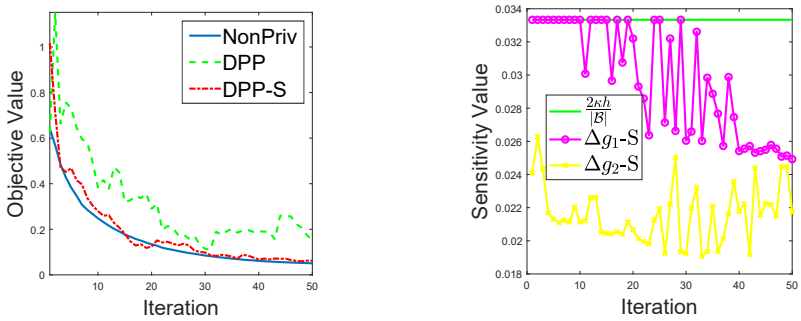
algorithms follow the step size update rule in Algorithm 1, which typically enforces the conditions $\sum_\tau \eta(\tau) = \infty$ and $\sum_\tau \eta^2(\tau) \leq \infty$.

3.6.1 Toy example

Synthetic dataset introduced in [Nguyen et al., 2017] is used here as a toy example for DML. As shown in Fig. 3.7(a), raw data are composed of two classes, each of which is single-Gaussian distributed and contains 100 points. By trickily selecting 50 intra-class pairs within each class and 50 inter-class pairs, I obtain an undirected acylic graph. In this experiment, contrastive loss is optimized with NonPriv, DPP, and DPP-S (DPP with sensitivity reduction) concern separately. To verify their efficacy, I suppose the original data is always accessible in this toy example.

During the optimization, I empirically set $\varepsilon = 2$, $m = 1$, $h = 0.5$, $|\mathscr{B}| = 30$, and $T_{\max} = 10$. Fig. 3.7(b)-(d) show the transformed data using the distance metrics learned by three privacy schemes, respectively. For the convenient comparison, the transformed samples are exhibited in the same range map, $[0.08, 0.32]$ horizontally, and $[0.19, 0.45]$ vertically. Apparently, each of them now has the clearer structure than original data. It is noticed that optimization with DPP-S is very close to NonPriv result. That means the sensitivity reduction technique improves fidelity of output distance metric.

To better understand the above observations, I present the convergence curves of each solution shown as Fig. 3.8(a). It is seen that NonPriv converges steadily among three methods as its every iterative step is assigned with a clean gradient. Compared to DPP that uses the vanilla gradient sensitivity, the objective value of DPP-S decreases faster and has slighter fluctuation. Particularly, DPP seems to need more rounds to converge to a steady point. It is consistent with my expectation because more noises are injected in this privacy



(a) Convergence (b) Sensitivity

Figure 3.8 (a) The objective values of Eq. (3.7) versus iteration number with NonPriv, DPP and DPP-S, respectively. (b) The sensitivity value $\frac{2\kappa h}{|\mathcal{B}|}$ specified by Theorem 1 and reduced sensitivity specified by Theorem 2 (exhibited by each dimension) versus iteration number.

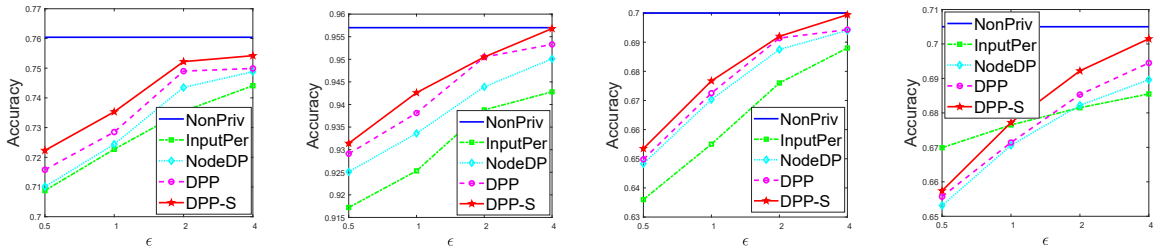
Table 3.1 Statistics of datasets

Dataset	Records (n)	Dime.	Pos./Neg. Pairs
Adult	48,842	124	18,700/18,700
Bank	45,211	33	8,464/8,464
IPUMS-BR	38,000	53	30,016/30,016
IPUMS-US	40,000	58	31,104/31,104

scheme. In addition, it is noticed that NonPriv objective value is not strictly lower than DPP-S. The reason is that the perturbation to the gradient sometimes provides diverse exploration directions. However, the optimization cannot benefit from random perturbations in a long period. Moreover, I investigate the sensitivity value of each row of the transformation matrix W, shown as Fig. 3.8(b). During a quantity of iterations, the sensitivity value after reduction is dramatically lower than the standard one referred in Lemma 1. Overall, the experimental results demonstrate the sensitivity reduction mitigates the utility degeneration caused by a huge amount of redundant noises.

3.6.2 Comparison on real-world datasets

Four datasets about humans are employed here following the setting of recent work [Lee and Kifer, 2018]: (i) Adult [Dua and Karra Taniskidou, 2017] is extracted from the 1994 Census database which contains the instances of personal income information. (ii) Bank [Dua and Karra Taniskidou, 2017] is related with direct marketing campaigns of a Portuguese banking institution. (iii) IPUMS-BR and (iv) IPUMS-US datasets are also about Census data collected from IPUMS-International [Ruggles et al., 2018]. For imbalanced dataset (Adult



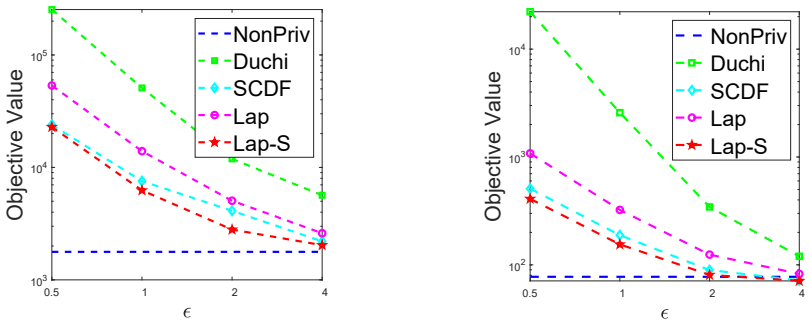
(a) Adult (b) Bank (c) IPUMS-BR (d) IPUMS-US

Figure 3.9 Classification accuracy of compared methods versus privacy budget ε over four real-world datasets.

and Bank), downsampling the majority class is simply adopted during training. Table 3.1 briefly summarizes the statistics of these datasets.

Since there is not any direct solution which is applicable to the proposed private metric learning problem, I borrow the idea of *Node DP* [Kasiviswanathan et al., 2013] and ERM-based input perturbation (denoted by InputPer for short) [Fukuchi et al., 2017] thought as two competing methods. The former defends against the attacker who is unknown as many as the maximum node degree edges, which is shown as the worst case of DPP according to the analysis in Section A.1. The latter assumes that data collector is also not reliable that guarantees a stronger privacy. The advantage of InputPer for pairwise data preserving is that no special consideration is needed for the pair correlations, because any inference is now unreliable. Throughout all the involved randomized algorithm, Laplace mechanism is used to extract noise. In particular, for InputPer method, the Laplacian noise is added to every dimension over feature, while the label is randomly flipping by Warner's model [Warner, 1965]. To prevent κ being too large, I empirically label the pairwise data with the fixed density of constructed graph, i.e., $\frac{|E|}{|V|} = 2$. Batch size and Lipschiz const are set as $|\mathscr{B}| = 50$ and $h = 0.5$ respectively. For different datasets, the margin m is preset as the average distance of dissimilar pairs, i.e., $m = \frac{1}{K_N}\sum||\Delta x||$, where K_N is the number of dissimilar pairs. The output distance metric is then evaluated by classification accuracy. Specifically, the distance metric M is firstly decomposed into transformation matrix W, and W projects original data into a new space. Then a kNN classifier ($k = 5$) is trained on the samples that are only employed in training pairs, and all the remaining samples are regarded as the test set. Last, every method is repeated for 20 times and the average classification accuracy is eventually reported.

Fig. 3.9 shows the classification results comparison versus different privacy budgets. At the first glance, the classification accuracy of every randomized method rises steadily with the increment of privacy budget. It is apparent because less noise is injected to the



(a) Adult (b) Bank

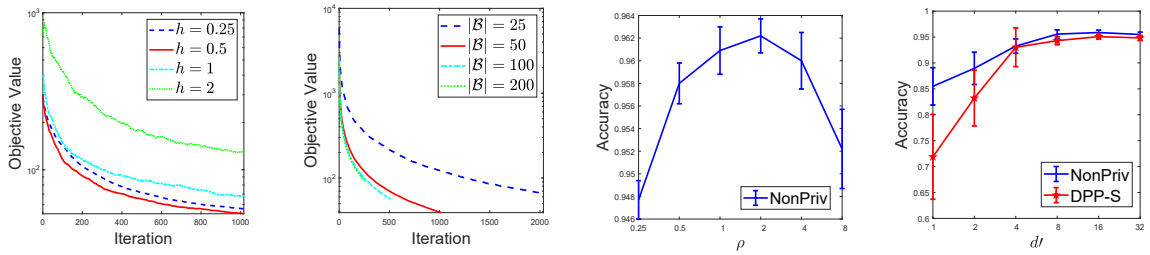
Figure 3.10 Different ε-DP mechanisms comparison in implementing DPP through their objective values.

gradient and the precision of distance metric is improved with the same iterative steps. As Node DP injects more noise than the proposed DPP during each iteration, one can find that its accuracy is always lower than DPP. Furthermore, DPP with sensitivity reduction, i.e. DPP-S, effectively enhances the accuracy, especially on Adult and IPUMS-US datasets. It is observed that in most cases, InputPer shows the worst performance, because InputPer is agnostic to the downstream application. Interestingly, it outperforms other competitors on IPUMS-US dataset when ε is not large. A reasonable explanation is that the dataset distance κ or maximum node degree is sometimes large (determined by randomly labelled pairwise data) in other three methods, which results in a great amount of injected noise in optimization, while InputPer is not influenced by these factors.

3.6.3 Privacy mechanisms comparison

Besides Laplace mechanism (and its sensitivity reduction version), I employ two more ε-DP mechanisms as the alternatives during implementing DPP. Soria-Comas and Domingo-Ferr [Soria-Comas and Domingo-Ferrer, 2013] proposed a staircase-like mechanism which is a variant of Laplace mechanism. Duchi et al. [Duchi et al., 2018] proposed a method to perturb multidimensional numeric tuples. For convenience, these four mechanisms are orderly denoted by Lap, Lap-S, SCDF, and Duchi separately. They are compared by replacing the gradient perturbation component and reporting their objective values after an epoch optimization. Obviously, the smaller objective values indicates better convergence. The experimental results on Adult and Bank datasets are as representatives. The involved parameters setting follows Section 3.6.2.

Fig. 3.10 presents the objective values of Eq. (3.7) when different privacy mechanisms are employed. On both two datasets, it is noted that SCDF converges faster than Lap. This is



(a) LipCons (b) Batch size (c) Margin threshold (d) DimRed

Figure 3.11 Effects of several key parameters on Bank dataset. (LipCons stands for Lipschitz constant and DimRed stands for Dimension reduction.)

because SCDF benefits from an adjustable parameter controlling staircase width in Laplace mechanism. Staircase width is a function of both sensitivity and privacy budget. Thus, it has the potential to add less noise than Lap that enhances the utility. In addition, with the use of reduced sensitivity trick, Lap-S is found converging fastest in all mechanism members. Duchi performs worst in this experiment compared with other sensitivity based mechanisms. Specifically, all the mechanisms are extracting unbiased noise, but the variance of Duchi for one-dimension data is $(\frac{e^{\varepsilon}+1}{e^{\varepsilon}-1})^2$ while the variance is $\frac{4h^2}{|\mathscr{B}|^2\varepsilon^2}$ for Lap. If I take $h = 0.5$, $|\mathscr{B}| = 50$ and $\varepsilon = 1$, Duchi's variance is around 10,000 times larger than Lap's!

3.6.4 Effects of parameters

The effects of involved parameters Lipschitz constant h, batch size $|\mathscr{B}|$, margin threshold m, and reduced dimension d' on are carefully investigated on Bank dataset.

To properly clip the gradient in each step, a good Lipschitz constant h is desired, because either a smaller or larger h would cause the slower convergence of optimization. On the one hand, if h is too small, then most of gradients will be clipped, and thus the objective converges slowly. On the other hand, if h is too large, the sensitivity value will be large according to Theorem 2, and consequently the amount of noise is increased in each step optimization. In this experiment, let $\varepsilon = 4$ and $T_{\max} = 3$, I investigate the objective curves under different values of h. The experimental results are shown as Fig. 3.11(a). It is observed that the objective converges faster when $h = 0.5$. Particularly, when $h = 2$, the objective value initially increases within the a few of steps and then decreases steadily. It is because the step size is large at the beginning of the optimization, which amplifies the imprecise gradient caused by a relatively large h.

Similar to Lipschitz constant h, according to Theorems 1 and 2, the batch size is also related to the convergence rate of the algorithm. Grid search is applied to $|\mathscr{B}|$ and all of them

are conducted for same epochs, i.e., $T_{\max} = 3$. The results are shown as Fig. 3.11(b). When the batch size is larger, the total optimized steps are smaller, and vice versa. As $|\mathscr{B}| = 50$ reaches the lowest objective value within the limited epochs, I empirically accept it as the default setting in other experiments.

The margin threshold m is a hyperparameter for contrastive loss. Instead of manually fixing it, I connect it with the average distance of dissimilar pairs, i.e., $m = \frac{\rho}{K_N} \sum ||\Delta x||$, where ρ serves as a ratio factor. In this experiment, m is tuned by changing the value of ρ for NonPriv. Since the objective value is a function of m, ρ is then searched by evaluating the corresponding testing accuracy. Fig. 3.11(c) shows the accuracy comparison. Although $\rho = 2$ obtains the best performance, the accuracy of $\rho = 1$ is only slightly lower than the best result. Thus, I simply use the naive average distance in all of other experiments.

A distance metric can be decomposed into transformation matrix, which projects the original data into different dimensional space, a.k.a. dimension reduction. Fig. 3.11(d) presents the testing accuracy result versus different reduced dimensions. The highest accuracy is obtained when $d' = 16$, and the projection without dimension reduction ($d' = 32$) is close to the best performance. Interestingly, it is observed that with the decrease of dimension, the variance of testing accuracy increases significantly. Thus, I conclude most of features are contributive in this dataset and each category data is likely discriminative when more dimensions are kept.

3.7 Summary

In this chapter, I have exploited distance metric learning with restricted pairwise data. As pairwise data is said only trusted to the learning executor, the proposed metric learning algorithm is to ensure the learning process does not expose any extra data information. This problem is addressed following the principle of differential privacy. My achievement here is extending this principle to pairwise data and carefully discussing the connections with existing works from both theoretical analyses and experimental demonstrations. Please note the proposed DPP is discussed in the context of pairwise labelled data in machine learning tasks. For more complex cases where casual relations exist [Tschantz et al., 2020], we need further efforts to figure out a reliable privacy notion. In addition, it is worth mentioning there exist other parallel studies or subsequent research when I was preparing the thesis. Differently, they focus on theoretical privacy guarantee on convex [Huai et al., 2020, Xue et al., 2021] or non-convex objective [Kang et al., 2021a], stability on non-smooth loss [Yang et al., 2021, Kang et al., 2021b], and so on.

Chapter 4

Semi-Supervised Learning for Inferring Missing Labels

In this chapter, I focus on the learning with incomplete data in which some specific discrete attribute (e.g., race) is not filled in every record, or a proportion of members are unlabeled. To deal with this kind of data, I propose to infer the missing values through Semi-Supervised Learning (SSL). Unlike most of current SSL research, my attention here is to properly fit unlabeled data from the view of prediction uncertainty. To this end, a novel distillation method is developed, which is motivated by the thought of learning from the model output. By comparing with related distillation-based SSL studies from both theoretical analyses and experimental results, I demonstrate the efficacy of the proposed method.

4.1 Problem Understanding

Learning with partially labeled data implicitly requires the model to exploit the missing labels on its own. This form of learning paradigm, known as semi-supervised learning [Chapelle and Scholkopf, 2006] (SSL), is of practical significance as labeling costs are now passed on to the subsequent algorithm design.

4.1.1 Semi-supervised learning paradigm

The success of modern SSL algorithms is essentially attributed to two indispensable components designed for unlabeled data. (1) *Consistency regularization*. It assumes that each unlabeled sample u should have a consistent prediction with its transformed counterpart $T(u)$. Such a self-supervised constraint on unlabeled data indeed extends the conventional similarity regularization constructed over neighboring samples [Zhu et al., 2003, Hadsell et al., 2006,

Wang et al., 2020b], providing a favourable optimization direction for classification, which thus is greatly boosted by recent data augmentation techniques [DeVries and Taylor, 2017, Cubuk et al., 2019]. (2) *Prediction distillation*. The predictions on unlabeled data are often of high uncertainty due to the absence of supervised information. Prediction distillation tries to enhance model prediction based on current output, e.g., following the principle of Minimum Entropy [Grandvalet and Bengio, 2005]. As a consequence, the decision boundary is encouraged to deviate from the region where ambiguous predictions reside. Many SSL models [Grandvalet and Bengio, 2005, Niu et al., 2014, Miyato et al., 2018, Oliver et al., 2018] in literature have benefited from this component.

Generally, given a labeled set $\mathscr{L}$ and an unlabeled set $\mathscr{U}$, an SSL model parameterized by θ typically minimizes the following objective

$$\mathscr{J}(\mathscr{L},\mathscr{U};\theta) = \mathscr{J}_{\mathrm{S}} + \alpha\,\mathscr{J}_{\mathrm{C}} + \beta\,\mathscr{J}_{\mathrm{D}}, \tag{4.1}$$

where $\mathscr{J}_{\mathrm{S}}$ denotes a supervised loss on labeled set $\mathscr{L}$, $\mathscr{J}_{\mathrm{C}}$ and $\mathscr{J}_{\mathrm{D}}$ denote the consistency loss and distillation loss on unlabelled data $\mathscr{U}$, respectively. α and β are non-negative weight factors for balancing three losses. Despite the great boost of consistency regularization to SSL, extra expertise is usually required to customize the effective data augmentation w.r.t. different data types [Karras et al., 2020]. Conversely, prediction distillation manages to learn from the model output, serving as a more general and practical tool in the machine learning community. Besides the aforementioned Minimum Entropy (ME), many other distillation strategies and their variants have been developed lately, such as Sharpening (SH) [Xie et al., 2019, Berthelot et al., 2019b,a], Pseudo-Labeling (PL) [Lee et al., 2013, Sohn et al., 2020, Arazo et al., 2020], Negative Sampling (NS) [Chen et al., 2020b]. From the optimization perspective, although many of them progressively encourage partial unlabeled data to reach the low-entropy state, existing strategies unavoidably introduce incorrect distillations because neural networks may not be well calibrated during the training process [Guo et al., 2017]. The corresponding predictions are then dubbed *overconfident predictions*[1] in this study. That means for unlabeled predictions, not all distillations are beneficial.

4.1.2 Taming overconfident predictions

Many recent studies can be interpreted as workarounds to this problem. By discriminating the importance of unlabeled data, [Ren et al., 2020] learned the sample-wise weights for unlabeled data through an additive labeled validation set. A more direct thought is leveraging the self-paced technique which selects the promising unlabeled samples [Cascante-Bonilla

[1]This problem is called over-confirmation for Pseudo-Labeling in [Arazo et al., 2020].

et al., 2020] to do optimization in each iteration. Lately, [Rizve et al., 2021] proposed to reduce the bias of PL by approximating the calibration criterion from the overall uncertainty. These methods practically require a lot of extra effort, albeit some improvements are made. Specifically, the labelled validation set needed in [Ren et al., 2020] is not always allowed in real applications. Self-paced strategy [Cascante-Bonilla et al., 2020] may slow down the convergence rate, whose reported performance cannot rival the state-of-the-arts yet in their paper. The approximated calibration executed in [Rizve et al., 2021] is implemented by Monte Carlo dropout [Gal and Ghahramani, 2016] which demands multiple times inferences more than standard SSL training. Therefore, instead of following this branch of work, my research scope restricts to remedying overconfident predictions via tailor-designing a more intelligent prediction distillation strategy which is expected to be less computational and more extendable.

4.2 Preliminaries

In this section, I first present the formulation of existing distillation strategies and then analyze how they work from an entropic view.

4.2.1 Formulation

In a K-way classification problem, the softmax function produces strictly positive predictions (probabilities) for a logits vector $z \in \mathbb{R}^K$ by componentwise computing

$$\mathrm{softmax}_i(z) = \frac{\exp(z_i)}{\sum_j \exp(z_j)}. \tag{4.2}$$

Following a common fashion, I further denote $p = \mathrm{softmax}(z)$ for a compact statement. Then different distillation strategy can be formulated as follows.

ME The loss for minimum entropy of an unlabelled sample is written as

$$\mathscr{J}_{\mathrm{D}}^{\mathrm{ME}}(p) = -\sum_{i=1}^{K} p_i \log p_i. \tag{4.3}$$

Note that ME is originally defined on unlabeled data $\mathscr{U}$, and it also applies to augmented unlabelled data, i.e., $T(\mathscr{U})$.

SH Sharpening function [Heaton, 2018, Berthelot et al., 2019b] constructs the target label by adjusting the temperature of the current categorical distribution p,

$$\mathrm{SH}_i(p,\lambda) = \frac{p_i^{1/\lambda}}{\sum_j^K p_j^{1/\lambda}}, \tag{4.4}$$

where λ ($\lambda > 0$) is the temperature that controls how sharp the output distribution looks like. Then the corresponding distillation loss is formulated as

$$\mathscr{J}_{\mathrm{D}}^{\mathrm{SH}}(p) = \mathrm{Dist}(\mathrm{SH}(p,\lambda), p), \tag{4.5}$$

where $\mathrm{Dist}(\cdot)$ represents any feasible distance measure over distributions, such as Kullback-Leibler (KL) divergence or Wasserstein distance. I use $\mathrm{Dist}(\cdot)$ as a general distance in the rest of this chapter when there are multiple alternatives.

PL Naive Pseudo-Labeling [Lee et al., 2013] could be described as a hard version of sharpening, and it picks the class which has the maximum predicted probability to be the target label,

$$\mathrm{PL}_i(p) = \begin{cases} 1, & \text{if } i = \arg\max_j p_j \\ 0, & \text{otherwise.} \end{cases} \tag{4.6}$$

In practice, a stricter condition applies [Sohn et al., 2020] that requires the maximum probability to exceed a predefined threshold τ_{PL}. This means that PL does sample selection in each round of optimization. For each selected sample, the distillation loss is written in a similar manner to SH,

$$\mathscr{J}_{\mathrm{D}}^{\mathrm{PL}}(p) = \mathrm{Dist}(\mathrm{PL}(p), p). \tag{4.7}$$

NS Picking negative classes by a threshold τ_{NS} and minimizing their probabilities comes to the distillation loss of

$$\mathscr{J}_{\mathrm{D}}^{\mathrm{NS}}(p) = -\log(1 - \sum_{i=1}^{K} \mathbb{I}(p_i < \tau_{\mathrm{NS}}) p_i), \tag{4.8}$$

where $\mathbb{I}(\cdot)$ is the indicator function. This loss term is proved to better approximate the true likelihood of unlabeled data in [Chen et al., 2020b] and it is treated as one of the distillation baselines.

Distillation Strategies	Candidates selection	Activation in target label	Uncertainty minimization
Minimum Entropy (ME)	None	Single	I, II, III
Sharpening (SH)	None	Full	I, II, III
Pseudo-Labeling (PL)	Hard threshold	Single	I
Negative Sampling (NS)	Hard threshold	Partial	III
ADaptive Sharpening (ADS)	Soft threshold	Partial	II

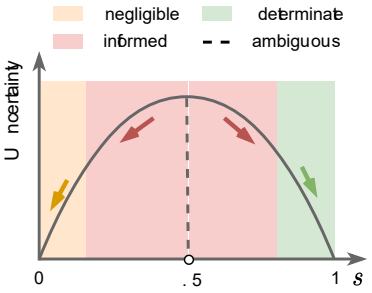


Figure 4.1 **Left:** Comparison among various distillation strategies, each of which is viewed as a two-stage process by first selecting candidate classes and then aligning their predictions with the categorical target label distribution. (I) enhance determinate predictions; (II) promote informed predictions; (III) suppress negligible predictions. **Right:** Different strategies turn out having different fashions to minimize prediction uncertainty. Binary classification is showcased here for simplicity where $p = (s, 1-s)$ where $0 \leq s \leq 1$.

4.2.2 An entropic view of distillation

It is observed that existing distillation strategies are in pursuit of low-entropy predictions via post-processing the model predictions in different ways. According to their formulations, I summarize them into the left panel of Fig. 4.1 where they are compared in terms of candidate classes selection, the sparsity of activation, and the way of uncertainty minimization. Before diving into more details, I need the following definitions.

The labeled data can be viewed as anchor points which initialize the model and induce the unlabeled prediction. Given $0 < \theta_1 \ll 0.5 \ll \theta_2 < 1$, I adhere the philosophy of this thought and partition the intact probability space into four intervals:

- **negligible:** $\{p_i | p_i < \theta_1, i = 1, 2, \ldots, K\}$;
- **ambiguous:** $\{p_i | p_i = \frac{1}{K}, i = 1, 2, \ldots, K\}$;
- **informed:** $\{p_i | \theta_1 \leq p_i \leq \theta_2,\ p_i \neq \frac{1}{K},\ i = 1, 2, \ldots, K\}$;
- **determinate:** $\{p_i | p_i > \theta_2, i = 1, 2, \ldots, K\}$.

The above definitions make it feasible to categorize each prediction as one of the four types. Thus, one can see that ME and SH consider all classes as the candidates and aggressively minimize uncertainties over all categories except ambiguous ones which provide no useful information for distillation. By simply setting $\theta_1 = \tau_{\text{NS}}$ and $\theta_2 = \tau_{\text{PL}}$, it is concluded that PL and NS resort to heuristic thresholds to ensure that distillation acts on relatively certain (determinate or negligible) predictions only. As a result, ME, SH, and PL suffer the problem of overconfident predictions [Arazo et al., 2020, Ren et al., 2020] as they are enhancing the determinate predictions (type I in Fig. 4.1). Notice that NS tries to penalize the negligible

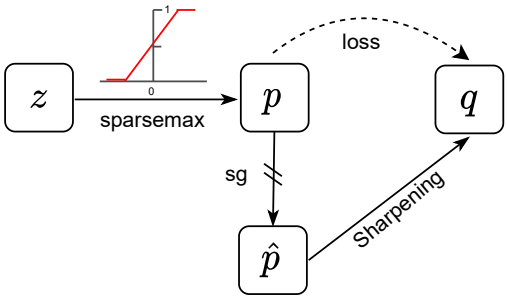


Figure 4.2 Distillation architecture of ADS.

predictions, which seems to be safer. However, it factually has a similar effect to PL for binary class classification as further discussed in Section 4.4.1.

For the sake of a better understanding to the different distillation strategies, I display the four types of prediction in a binary-class classification case on the right panel of Fig. 4.1. As all distillation strategies have been interpreted as uncertainty minimization, each distillation strategy can be decomposed in this probability space. It is observed ambiguous prediction is special and it only takes a point. θ_1 and θ_2 do not coexist in existing methods, and they are not always empirically satisfying $\theta_1 + \theta_2 = 1$. In my study, leveraging the informed predictions only is the goal, which remedies the overconfident predictions by doing nothing to the extremely certain predictions (More benefits can be seen in Section 4.4.2). Particularly, as in different training stages and for different unlabeled samples, the informed predictions should be relatively different and dynamic. Thus, the informed predictions are not quite aligned with the ones defined by hard thresholds used in PL and NS.

4.3 ADS Based SSL Model

According to the understanding of prediction distillation indicated by the left panel of Fig. 4.1, I propose ADaptive Sharpening (ADS), a dual distillation mechanism that only sharpens the informed predictions to avert the overconfident issue. An architecture of ADS from logits to losses is showcased in Fig. 4.2. I also analyze how other loss terms benefit from ADS, which completes the presentation of concrete ADS-based SSL models.

4.3.1 ADaptive Sharpening (ADS)

Adaptive selection via sparsemax

Properly selecting partial classes to do distillation assists the classification model to concentrate on the most confusing classes for each unlabeled sample [Chen et al., 2020b].

Existing prediction distillation strategies, such as PL and NS, are heuristically overlaid on the probabilistic distribution produced by softmax function, formulating a hard threshold-based post-processing approach [Chen et al., 2020d]. Despite their popularity in recent research [Sohn et al., 2020, Chen et al., 2020b], the major drawback of them is the adoption of a predefined threshold, which is not robust enough to account for individual variability and model variability at different training stages, dwindling its effectiveness in practical implementations.

Therefore, a flexible sampler is expected, which can automatically determine a partial set of promising classes as candidates for the true label, being adaptive to each unlabelled sample at different training stages. In this section, I study this problem from neural networks' emission, i.e., logits vector z. Instead of applying a truncation-based approach, I apply sparsemax [Martins and Astudillo, 2016] to logits, which is written as

$$\text{sparsemax}_i(z) = [z_i - \tau(z)]_+, \tag{4.9}$$

where $[u]_+ := \max(0, u)$.

Proposition 1 *The form of $\tau(z)$ in Eq.* (4.9) *is expressed as follows. Let $z_{(1)} \geq z_{(2)} \geq ... \geq z_{(K)}$ be the sorted coordinates of z, and define $\kappa(z) := \max\{k \in \{1, ..., K\} | 1 + kz_{(k)} > \sum_{j \leq k} z_{(j)}\}$. Then, $\tau(z) = \frac{(\sum_{j \leq \kappa(z)} z_{(j)}) - 1}{\kappa(z)}$.*

Note that $\kappa(z)$ and $\tau(z)$ in Proposition 1 is not manually designed because of the fact that sparsemax is the closed-form solution of the Euclidean projection problem [Duchi et al., 2008]. One can find more details about the optimization in [Martins and Astudillo, 2016].

In fact, I am interested in the following properties of the sparsemax: (1) $\kappa(z)$ implies the support of sparsemax(z). Given a logits vector z, the form of $\kappa(z)$ determines the smallest $K - \tau(z)$ coordinates will produce zero probabilities. (2) $\sum_j [z_j - \tau(z)]_+ = 1$ always holds where $\tau(z)$ is dependent on z. Compared with the hard thresholds on predictions, $\tau(z)$ can be viewed an adaptive soft-threshold on logits. I am in favor of these two properties as sparsemax is expected to gradually rule out the unrelated classes in SSL, which serves as an elegant sampler for distillation.

Selecting proper candidate classes motivates the employment of sparsemax. This transformation also implies the informed predictions needed by the distillation component. We focus on the completeness of our method here and leave the analyses of how ADS fulfills our hindsight in Section 4.4.1.

Sharpening on sparse probabilities

Prediction distillation acts as a self-training process. To iteratively refine the classification model, I define the distillation loss of ADS as a distance, e.g., KL-divergence, between the raw prediction p and the constructed target label distribution q:

$$\mathscr{J}_{\mathrm{D}} = \sum_{u \in \mathscr{U}} \mathrm{KL}(q||p) = \sum_{u \in \mathscr{U}} \sum_{i} q_i \log \frac{q_i}{p_i}, \tag{4.10}$$

where $p_i = \mathrm{sparsemax}_i(f(u;\theta))$. The choice of target label distribution q is crucial for ADS's performance. Particularly, the target label distribution q is expected to have the following properties: (1) strengthen confident predictions, (2) exclude the contributions of negative class predictions.

According to the comparison in Fig. 4.1, Sharpening is a feasible option to satisfy the above requirements. Having a sparse prediction p in hand, I aim to construct its pseudo target label distribution q. The first step is exerting $\hat{p} = \mathrm{sg}(p)$, where $\mathrm{sg}(\cdot)$ stands for the stop-gradient operator that is defined as identity during forward computation and has zero partial derivatives from q during back-propagation. This operation enables that minimizing $\mathscr{J}_{\mathrm{D}}$ induces p to approach q. Similar to Eq. (4.4), the second step is calculating each element of q, which is a normalized probability raised from $\hat{p}_i$ with the power of r:

$$q_i = \frac{\hat{p}_i^r}{\sum_j \hat{p}_j^r}. \tag{4.11}$$

Obviously, if $\hat{p}_i = 0$, I have $q_i = 0$, and if $\hat{p} = \frac{1}{K}\mathbf{1}$, I have $q = \hat{p}$. Otherwise, as $r \to \infty$, the categorical distribution q will approach a delta distribution. It is emphasized that although ADS borrows the idea of SH for constructing the target label distribution, they have different functionalities during distillation. Particularly, I will show ADS does not aggressively distill relatively certain predictions in Section 4.4.1, which mitigates the overconfident risk.

Fig. 4.2 shows the process of my distillation design starting from derived logits. Note that Eq. (4.10) is not restrictive to the original unlabeled data $\mathscr{U}$; it is applicable to augmented unlabeled data $T(\mathscr{U})$ as well if data augmentation techniques are employed.

4.3.2 In conjunction with other loss

As the supervised loss and the consistency regularization are both constructed over predictions, unlike other post-processing distillation strategies, ADS indeed has a direct impact on these two losses as well.

In terms of the consistency regularization, it restricts each unlabeled sample to reach a consistent prediction with its transformed counterpart. Thus for ADS, the consistency loss $\mathscr{J}_{\mathrm{C}}$ could be generally formulated as

$$\mathscr{J}_{\mathrm{C}} = \mathrm{Dist}(\mathrm{sparsemax}(z), \mathrm{sparsemax}(z')) \tag{4.12}$$

where z and z' are logits vectors of the inspected unlabeled sample and its counterpart, and $\mathrm{Dist}(\cdot)$ could be any proper distance function. Suppose $\mathrm{sparsemax}_i(z) = \mathrm{sparsemax}_i(z') = 0$ holds for the i-th class. In this case, class i will not contribute to the consistency loss. With more classes like this, the consistency loss of ADS will focus on a few of confusing classes only. From Eq. (4.2), the condition of equality of $\mathrm{softmax}_i(z) = \mathrm{softmax}_i(z')$ is much stricter. That means softmax-based consistency will be distracted by the unrelated classes during model training. Taking image classification as an example, an unlabeled image is categorized to *leopard* only if all its transformed counterparts reach a consensus on *leopard*. In practice, the improvement of consistency regularization essentially lies in its constantly confirmed decision among the confusing classes. Intuitively, an augmented *leopard* image might be somehow close to a *cat* while it should less similar to a *dog*. Thus, putting more attentions on confusing classes are expected to achieve better performance which is also referred in previous research [Gal and Ghahramani, 2016].

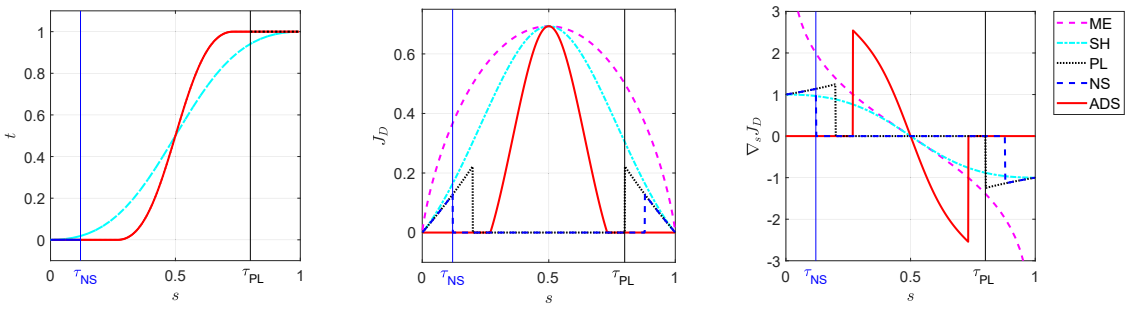
Particularly, I also allow the networks' output of labeled data to pass the sparsemax. However, if the model assigns zero probability to the gold label, the entire training sample would be ruled out [Martins and Astudillo, 2016]. Although there exists some possible workaround like adding a small constant to the probabilities and then do re-normalization, I instead use the following supervised loss as an alternative of cross entropy proposed by the recent work [Blondel et al., 2020]

$$\mathscr{J}_{\mathrm{S}} = \frac{1}{2}(||y - z||_2^2 - ||\mathrm{sparsemax}(z) - z||_2^2), \tag{4.13}$$

where y is the groundtruth encoded in a one-hot format. Notice that the ℓ_2-norm based loss trickily averts the aforementioned optimization dilemma, and the derived gradient w.r.t. z has a closed form, making a friendly backward propagation.

4.4 Theoretical Analyses

In this section, I theoretically justify the principle of ADS and then provide evidence of its superiority from the view of the transformation function.



(a) Target Probability (b) Distillation Loss (c) Distillation Gradient

Figure 4.3 Comparison of different distillation strategies in terms of target probability, distillation loss, and gradient. Note that the distillation gradient of SH and ADS shown in the subfigure (c) is corresponding to the reduced losses (See Appendix B.1).

4.4.1 ADS promotes informed predictions

It is highlighted that ADS does distillation via promoting informed predictions only, showing an efficient distillation mechanism compared with existing strategies. Regarding a two-class case, the neural networks' output is denoted as $z = (u, 0)$. In terms of same z, suppose $\text{softmax}_1(z) = s$ and $\text{sparsemax}_1(z) = s'$. In particular, I have $s \neq s'$ if $u \neq 0.5$. Since existing distillation strategies work over softmax output, one can consider s' as a function of s and rewrite the distillation loss $\mathscr{J}_{\text{D}}$ (refer to the detailed derivation in Appendix B.1), making a clear and direct comparison for all methods in the same probability space. Fig. 4.3 plots the target probability, the prediction distillation loss, and the corresponding gradient versus the first dimensional probability s in terms of each distillation strategy. I remind readers to read this figure from the shape and trend of every curve other than the scale as they can be equipped with different weights during formulations. The main conclusions are as follows:

- ADS is the unique strategy focusing on informed predictions only. According to Fig. 4.3(b), it is observed that the proposed ADS has zero penalties when s is negligible or determinate. Notably, ADS is viewed as a corrective SH by masking out the determinate predictions to avoid introducing any overconfident risk.

- PL and NS are consistent strategies for the binary classification case. PL and NS have the complementary philosophy from Figure 4.3(a). Given proper thresholds, they are shown to have the same form of loss and gradient according to Fig. 4.3(b) and (c).

- ME has the unbounded gradients. ME takes the highest loss value but produces zero gradient on ambiguous predictions according to Fig. 4.3(b) and (c). Meanwhile, when

s is relatively small or large, the gradient of ME is large, exposed to the risk of unstable optimization.

From the above analyses, ADS will outperform other distillation strategies if the assumption holds that leveraging informed predictions only is sufficient for distillation. From Fig. 4.3 it is seen that the terminology "informed" is factually specified by a corresponding threshold in the probability space of softmax output. Now a more general case is explored where data could be categorized into one of the multiple classes.

Theorem 3 *For an unlabeled sample u whose softmax output is $p \in \mathbb{R}^K (K > 2)$, the distillation loss $\mathscr{J}_D \equiv 0$ (then zero gradient) for* ADS *in Eq.* (4.10) *holds if $p_{(1)} > e p_{(2)}$, where $p_{(1)}, p_{(2)}$ are the first two largest coordinate of p and e is Euler number.*

Proof: Let $z \in \mathbb{R}^K$ denote the logits of u, i.e., $z = f(u;\theta)$, and $p = \text{softmax}(z)$. According to the definition of softmax, i.e., Eq. (4.2), I have $p_i = \frac{e^{z_i}}{C}$ for $i = 1, 2, ..., K$, where $C = \sum_k e^{z_i}$. Then I can rewrite $z_i = \ln(C p_i)$, for $i = 1, 2, ..., K$. Let $p_{(1)}, p_{(2)}$ are the first two largest coordinates of p, and I have element expression

$$z_{(1)} = \ln(C p_{(1)}), \quad z_{(2)} = \ln(C p_{(2)}), \tag{4.14}$$

where $z_{(1)}, z_{(2)}$ denote the corresponding first two largest coordinates of z. Apart from the trivial case of $p_{(1)} = p_{(2)} = ... = p_{(K)} = \frac{1}{K}$, $\mathscr{J}_\text{D} = 0$ holds iff $\text{sparsemax}(z)$ reaches one-hot. According to the closed-form solution of sparsemax in Section 4.3.1, I obtain

$$1 + 2z_{(2)} \leq z_{(1)} + z_{(2)}. \tag{4.15}$$

Taking Eq. (4.14) into Eq. (4.15), I have

$$1 + \ln(C p_{(2)}) \leq \ln(C p_{(1)}) \Rightarrow p_{(1)} \geq e p_{(2)}, \tag{4.16}$$

which completes the proof. ■

From Theorem 3, one can see that the "informed" predictions spoken of in the output space of softmax refer to the categorical probability p which satisfies $p_{(1)} < e p_{(2)}$ and $p \neq \frac{1}{K}\mathbf{1}$. **That means, unlike existing distillation strategies,** ADS **does not encourage the relatively certain predictions to further become extremely certain since the distillation loss $\mathscr{J}_\mathbf{D} \equiv 0$ holds as long as $p_{(1)} \geq e p_{(2)}$, and thus the issue of overconfident predictions is mitigated.** The following corollary quantifies the soft threshold used by ADS.

Corollary 1 *For a K-way semi-supervised classification problem, the determinate predictions and negligible predictions for* ADS *are masked out by the sample dependent threshold* $\theta_1 \in [\frac{e}{e+K-1}, \frac{e}{e+1}]$ *and* $\theta_2 \in [\frac{e^{\rho}}{\rho+e^{\rho}(K-\rho)}, \frac{e^{\rho}}{\rho+e^{\rho}}]$ *in the corresponding softmax output space, respectively, where e is Euler number and* ρ *is the population of non-zero predictions.*

The proof of this corollary is left to Appendix B.2. Notably, I clarify that the determinate predictions are spoken of in the context of multi-class classification where only a single class is the groundtruth. In other words, a prediction is said determinate in terms of its potential to be the real label. In addition, the negligible predictions are handled similarly, but its population for a single unlabelled sample is at least 1. In particular, for ADS the negligible predictions do not coexist with the determinate prediction.

4.4.2 ADS facilitates entropy minimization

Entropy minimization has been demonstrated effective in the existing SSL research [Grandvalet and Bengio, 2005, Miyato et al., 2018, Oliver et al., 2018]. In this section, I point out this principle is fundamentally related to the probability transformation function used in neural networks, e.g., softmax activation. It typically involves projecting a logits vector z on the probability simplex with an optimized problem of

$$p^{\star} = \underset{p \in \Delta^{K-1}}{\arg\min} \{-\langle z, p\rangle - H(p)\}, \tag{4.17}$$

where Δ^{K-1} is the probability simplex with freedom of $K-1$, and $-H(p)$ is a convex function, serving as a regularizer.

When $H(p)$ is implemented using Shannon entropy, i.e., $H(p) = -\sum_k p_i \log p_i$, the closed-form solution of Eq. (4.17) is the softmax transformation, a.k.a. the maximum entropy transformation. When $H(p)$ is replaced with Gini entropy, namely $H(p) = \frac{1}{2}\sum_k p_i(1-p_i)$, Eq. (4.17) becomes the aforementioned Euclidean projection, and its solution is also known as the sparsemax transformation, i.e., Eq (4.9).

Back to the minimum entropy principle, existing distillation methods built on softmax can be viewed as minimizing the prediction entropy in a post-processing manner [Chen et al., 2020d], which conflicts with the function of the regularizer $H(p)$ in Eq. (4.17). As Shannon entropy is a stronger penalty than Gini entropy, softmax would intensify contradictions compared with sparsemax [Blondel et al., 2020]. Therefore, I argue that the proposed ADS facilitates entropy minimization since it looks for a balance between probability transformation and predication distillation.

4.4.3 ADS introduces a lightweight computation

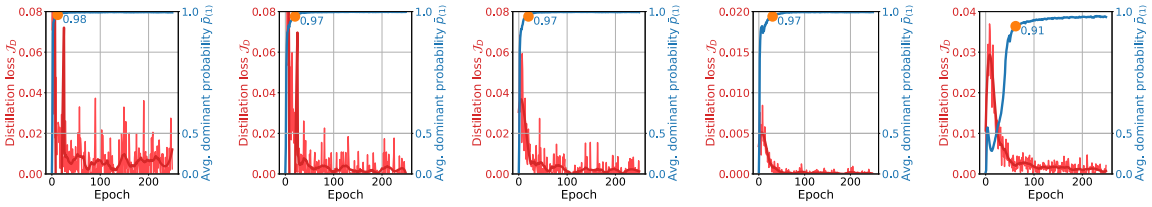
Compared with other distillation strategies exhibited in the left panel of Fig. 4.1, the bottleneck of computing ADS is the involved sparsemax, i.e., Eq. (4.9). It is because all there distillation losses are built on output probabilities and they have a consistent formulation (See Eqs. (4.3), (4.5), (4.7), (4.8), (4.10)). During the forward propagation, computing sparsemax probabilities incurs a the complexity of $\mathcal{O}(K \log K)$ due to the operation of sorting logits. By using the median pivot and partitioning, the expected complexity can be reduced to $\mathcal{O}(K)$ [Laurent, 2016], which is practically comparable with the componentwise computation of softmax (Eq. (4.2)). Regarding the backward propagation, the product between the Jacobian and a given vector is required. In case of sparsemax, it is computed with $\mathcal{O}(|\kappa(z)|)$, where $\kappa(z)$ is the number of non-zero probabilities pre-computed in the forward propagation [Martins and Astudillo, 2016]. This sublinear time complexity is advantageous once many non-gold classes are excluded.

To sum up, ADS only introduces a lightweight extra computation when plugged in existing SSL algorithms. The wall-time validation about its time efficiency is empirically investigated in Section 4.5.6.

4.5 Experiment

4.5.1 Experimental setup

Empirical comparisons are performed following the setting of [Chen et al., 2020b] except FixMatch for which I adopt the codebase from [Sohn et al., 2020]. For a fair comparison, the default network architecture for all datasets except MNIST and ImageNet is Wide ResNet-28-2 [Zagoruyko and Komodakis, 2016] with 1.5M parameters. Regarding MNIST, I employ a 7-layer convolutional neural network, and for ImageNet I use a ResNet-50 network architecture. The batch size for unlabeled data is 64. In terms of the labeled data, the batch size is set as the number of labeled data if it is smaller than 64, and set as 64 otherwise. The computation resource for all methods is set up with a classical rtx2080ti GPU. I do not exhaustively adjust the network parameters for different benchmarks, such as the number of scales and filters, so as to best reproduce the results. The mean and variance of five independent running under different random seeds are reported. Throughout all the experiments involving ADS, $r = 2$ is as a default setting as I will show this value achieves stable performance for different datasets.



(a) ME (b) SH (c) PL (d) NS (e) ADS

Figure 4.4 Converged curves of distillation loss $\mathscr{J}_{\mathrm{D}}$ and average dominant probability $\bar{p}_{(1)}$ for unlabeled training samples on MNIST datset. The loss values are smoothed for a better visualization. For ADS, $\bar{p}_{(1)}$ is collected and calculated by replacing sparsemax with softmax which does not change the training process.

Table 4.1 Test error (%) of various distillation strategies based on VAT. The best results are marked in bold.

Methods	MNIST (20 labels)	CIFAR-10 (4,000 labels)
VAT	23.76±1.18	14.72±0.23
VAT+ME	20.64±1.28	14.34±0.18
VAT+SH	18.45±1.09	12.90±0.25
VAT+PL	19.72±1.35	14.15±0.14
VAT+NS	19.36±1.17	13.94±0.10
VAT+ADS	**14.52**±1.03	**12.40**±0.31

In particular, "X+Y" is used to dub a method by adding a distillation strategy Y to an SSL algorithm X, and "X-Y" is used to to dub an SSL algorithm X whose distillation component is replaced by Y.

4.5.2 Study on VAT

Virtual Adversarial Training [Miyato et al., 2015] (VAT) serves as a powerful SSL algorithm without requiring manual data augmentation [Oliver et al., 2018]. It contains a consistency loss which has a similar form with Eq. (4.12), but optimizes the adversarial perturbation $v_{\mathrm{adv}} = \varepsilon \frac{g}{||g||}$, where ε is a predefined perturbation scale and g is approximated by the gradient on a randomly sampled unit vector r, that is

$$\nabla_v \mathrm{Dist}(\mathrm{softmax}(f(x;\theta)), \mathrm{softmax}(f(x+v;\theta))). \tag{4.18}$$

VAT is demonstrated to be improved if it is added to a minimum entropy according to [Miyato et al., 2018]. Based on this fact, I implement various distillation strategies on VAT for a direct and fair comparison. To this end, each baseline is derived by replacing the distillation

loss $\mathscr{J}_{\mathrm{D}}$ of Eq. (4.1) with differential distillation strategies. Experiments are conducted on MNIST and CIFAR-10 with 20 and 4,000 labeled training examples, respectively.

Table 4.1 presents the test error of various distillation strategies based on VAT. It is observed that each distillation loss helps semi-supervised learning and obtains the better performance. More importantly, ADS achieves the lowest test error compared with both vanilla VAT and other variants. In particular, the improvement to the second-best on the two datasets is around 4.7% and 1.5%, respectively.

To have a close look at the distillation component for MNIST dataset on which a better improvement has been achieved, for each method, the distillation loss $\mathscr{J}_{\mathrm{D}}$ and average dominant probability are visualized. Average dominant probability is calculated over all unlabeled training samples $\bar{p}_{(1)} = \frac{1}{|\mathscr{U}|}\sum_{\boldsymbol{x}\in\mathscr{U}} p_{(1)}$ which reflects the certainty of unlabeled predictions. Two measures are recorded at each epoch and the results are shown as Fig. 4.4.

In terms of ADS, the distillation loss decreases stably with the increase of average dominant probability. The curve of $\bar{p}_{(1)}$ reaches a cusp and levels off, with more modest increases after $\bar{p}_{(1)} = 0.91$. For the other four strategies, the distillation loss decreases dramatically, and the average dominant probability surges within a small number of epochs. It is observed that their values of $p_{(1)}$ locate at the cusp which are all much larger than that of ADS. That is to say, they apt to optimize the learning model to an extremely certain state but inevitably introduce overconfident distillations. Instead, ADS gradually optimizes the model by using informed predictions, averting the overconfident risks naturally. Let $\mathscr{U}_{+}$ and $\mathscr{U}_{-}$ denote the correctly and incorrectly classified unlabeled training examples, and the average dominant probability can be rewritten as $\bar{p}_{(1)} = \frac{1}{|\mathscr{U}|}(\sum_{\boldsymbol{x}\in\mathscr{U}_{+}} p_{(1)} + \sum_{\boldsymbol{x}\in\mathscr{U}_{-}} p_{(1)})$. For the correct ones, they are expected to be more certain but are not supposed to be extremely certain in practice. A piece of evidence for this claim is that in a supervised task, penalizing the low-entropy prediction has been demonstrated to be beneficial to model generalization. Label smoothing [Pereyra et al., 2017] is exactly one simple trick following this philosophy. For the incorrect ones, their dominant predictions should be small as they provide the wrong optimization direction. To sum up, the relatively smaller $\bar{p}_{(1)}$ than competitors suggests better performance by considering two aspects. This is another hindsight by reviewing existing methods.

4.5.3 Improvement on advanced SSL algorithms

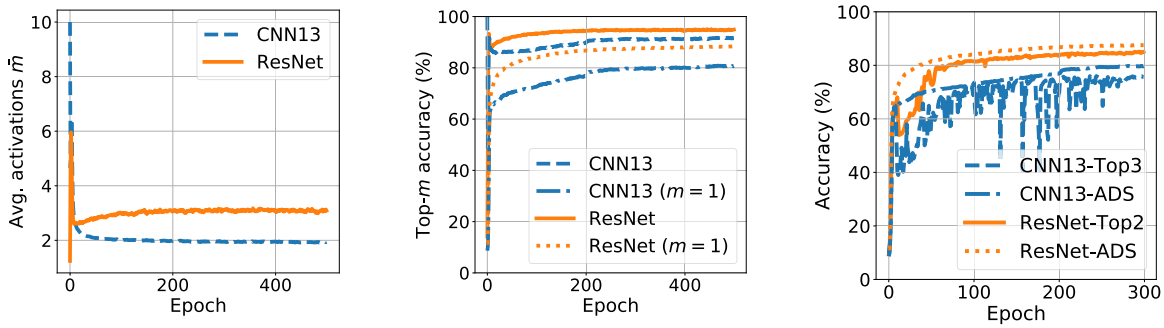
To evaluate the efficacy of the proposed ADS on the SSL algorithm which benefits from data augmentations, ADS is plugged in two state-of-the-art models MixMatch [Berthelot et al., 2019b] and FixMatch [Sohn et al., 2020]. Since vanilla MixMactch has already incorporated SH and vanilla FixMatch has adopted PL as their distillation component respectively, I do

Table 4.2 Performance comparison on four benchmarks. The best performance is marked as bold in two separate blocks. "MM" is short for "MixMatch", and "FM" is short for "FixMatch".

Dataset		MM	MM+NS	MM-ADS	FM	FM-ADS
CIFAR-10	40 labels	47.54±9.80	46.32±9.90	**42.86**±9.32	15.39±3.18	**14.84**±2.14
	250 labels	14.49±1.60	12.48±1.21	**10.87**±0.92	**6.81**±0.42	7.18±0.37
	4,000 labels	7.05±0.10	6.92±0.12	**6.46**±0.21	5.92±0.14	**5.21**±0.18
CIFAR-100	400 labels	67.64±1.36	67.50±1.54	**66.12**±1.20	54.21±2.28	**52.83**±2.53
	2,500 labels	39.94±0.37	39.74±0.21	**39.10**±0.25	37.42±0.38	**37.28**±0.42
	10,000 labels	33.72±0.33	33.45±0.19	**33.18**±0.54	29.57±0.11	**27.96** ±0.13
SVHN	40 labels	42.58±9.42	41.92±7.80	**40.82**±8.17	7.63±2.58	**2.37**±2.27
	250 labels	3.75±0.09	3.38±0.08	**2.63**±0.06	2.64±0.64	**2.15**±0.45
	1,000 labels	3.28±0.11	3.14±0.11	**2.42**±0.14	2.36±0.19	**2.07**±0.15
STL-10	1,000 labels	22.20±0.89	21.74±0.33	**19.20**±0.31	11.12±0.63	**10.43**±0.47

not conduct the corresponding ablation study. Notably, ME does not construct an explicit target label, and thus cannot be applied to data augmentations based SSL algorithms, which requires distilling explicit labels for augmented unlabeled data. Following [Sohn et al., 2020, Han et al., 2020], the remaining baselines are evaluated on four benchmarks CIFAR-10, CIFAR-100, SVHN, and STL-10, where different numbers of labeled data are used.

Table 4.2 shows that ADS based methods achieve significant improvement over vanilla Mixmatch, Fixmatch, and their variants, demonstrating the superiority of ADS over other distillation strategies, i.e., SH, PL, NS. In addition, it is observed that MixMatch+NS also achieves some improvements over vanilla Mixmatch. Interestingly, this method can be viewed as a simple compensation for overusing extremely certain predictions (See the distillation loss of SH and NS in Fig. 4.3(b)). From this aspect, ADS alone is functionally similar to this combination and MixMatch-ADS is shown to outperform the two baselines given different numbers of labeled data. FixMatch-ADS also improves FixMatch except for CIFAR-10 with 250 labels. This failure case may result from the occasion that PL may deny the strong augmentations that are out of data distribution once its predictions on weak augmentation are not very satisfactory. In particular, it is found that the improvement of ADS is not reliable to the amount of labeled data, i.e., it improves the baseline methods no matter the labelled data size is small or not, showing a good property in practical scenarios. In a word, all these observations are consistent with my claim that promoting informed predictions are beneficial for SSL model training to distill confident and reliable output.



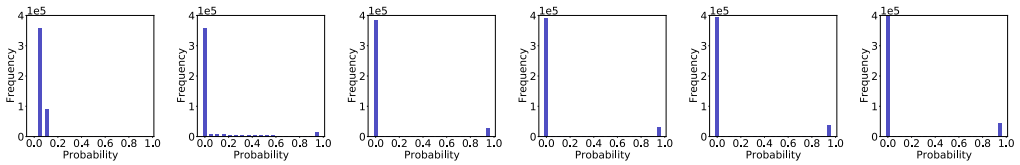
(a) Sparsity (b) Top-m accuracy (c) Top-m selection

Figure 4.5 The safety study of candidates selection in terms of two backbones ResNet and CNN13. (a) The average sparse activations $\bar{m}$ on unlabeled training data. (b) Top-m accuracy comparison where m is example-wise sparsity. (c) Standard accuracy comparison with globally fixed Top-m selection.

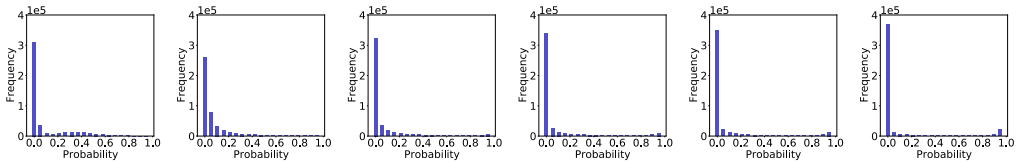
4.5.4 Safety with different backbone structures

Intuitively, it should be safe to do candidates selection when every class is initialized to be evenly activated. Nevertheless, a concern arises when neural networks are biasedly initialized which prefer some specific classes at the very beginning but unfortunately fail to cover the true label. To address this concern, I apply two different neural network architectures on CIFAR-10 with 250 training labels as an example. One is CNN13 [Tarvainen and Valpola, 2017] which is typically initialized with the uniform distribution, and the other is the default ResNet which is initialized with the normal distribution. The average activations $\bar{m}$ and Top-m accuracy are recorded over all unlabeled training data in Fig. 4.5, where example-wise m is the number of non-zero outputs. Note that "Top-m accuracy" denotes the success rate of partial activations including the true label, and it degenerates to the standard accuracy if $m = 1$.

Fig. 4.5(a) shows that CNN13 activates all the classes at the beginning ($\bar{m} = 10$) and then rules out unrelated classes till convergence. Contrastingly, the number of activations for ResNet surges from the one-hot state ($\bar{m} = 1$) in the first few epochs, then drops to a low value, and gradually reaches a steady value afterward. Fig. 4.5(b) shows that ResNet achieves better performance than CNN13 in terms of both two measurements although it starts with a biased initialization (Fig. 4.5(a)), experimentally demonstrating that initialization is not an issue for the candidates selection of ADS. In addition, since the standard accuracy is upper bounded by Top-m accuracy as one can expect, there exists a considerable gap between CNN13 (ResNet) and its corresponding Top-1 accuracy. This observation implies that the selected candidates of ADS are always meaningful.



(a) epoch=1 (b) epoch=10 (c) epoch=20 (d) epoch=30 (e) epoch=40 (f) epoch=100



(g) epoch=1 (h) epoch=10 (i) epoch=20 (j) epoch=30 (k) epoch=40 (l) epoch=100

Figure 4.6 Numerical distribution of prediction values of VAT+ADS on unlabeled training data. Top row shows the result on MNIST and the bottom row shows the result on CIFAR-10. The initialization of the network is used as their default.

Notably, neither ResNet nor CNN13 converges to the one-hot format in the above experiment; the converged average activations $\bar{m}$ of them are around 3 and 2, respectively. A possible reason for this phenomenon is that the distillation loss of MixMatch is explicitly defined on the manually augmented unlabeled data instead of the natural ones. For example, if an unlabeled sample is generated via some strong corruption, ADS-based model might *reject* to yield confident predictions. It is rational because not all the augmentations are meaningful for classification. A further question is if they are equivalent to the counterparts in which only Top-3 and Top-2 predictions are picked (with the remaining truncated) and distilled during the entire training stage. This thought is also known as set-valued classification [Chzhen et al., 2021], and I compare them as a new group of experiments.

Fig. 4.5(c) presents the accuracy comparison between ADS and fixed Top-m selection. It is observed that both two backbones have better accuracy when ADS is used, where their activations are dynamically changed indicated by Fig. 4.5(a). In addition, it is also noticed that Top-m selection suffers the intense fluctuation on the accuracy, which is rooted in the unstable optimization of the plain truncation. Particularly, the accuracy drop for CNN13 is severer than ResNet, which suggests another benefit of residual structure.

4.5.5 Observation of prediction histograms

I further verify the mechanism of ADS by recording the prediction frequency of unlabeled training data on MNIST and CIFAR-10 dataset. To this end, I evenly partition the prediction space into 10 intervals and let each of them serve as a bin. Each bin is a half-closed interval except the last one. For example, the first bin is defined as $[0, 0.1)$ and the last is $[0.9, 1]$.

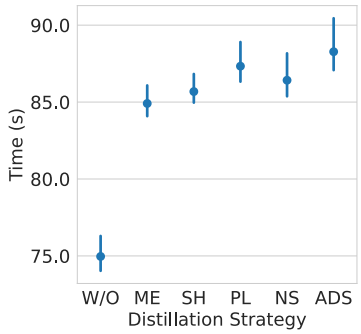
Each prediction of an unlabeled sample, i.e., $p_i(1 \leq i \leq K)$, will fall in a bin based on its value. The experiments are executed following the settings of Section 4.5.3 where VAT serves as the base model.

Fig. 4.6 presents the numerical distribution of prediction values over a series of epochs, where the maximum epoch is 100 because they are found not clearly altered after 100 epochs optimization. For MNIST, the prediction values are around 0.1 for every sample at the beginning. Then they gradually shift to other bins with the process of optimization. Eventually, almost all the predictions fall into the first and last bin, which means the predictions are distilled to be sufficiently certain. Regarding CIFAR-10, the results are a bit different. The initialization shows the model has some preference as mentioned in Section 4.5.4. In addition, the predictions grouped into the last bin are relatively fewer than that of MNIST. This is because the discriminative information for CIFAR-10 images is harder to capture via unlabeled data. In other words, given a semi-supervised model and training data, ADS aims to utilize the informed predictions and leaves the unreliable samples underfitted (This will not incur a low test confidence, please refer to Appendix B.3 for an example).

4.5.6 Running time comparison

The wall-time cost per epoch of different methods is recorded to demonstrate the analyses in Section 4.4.3. The first group of experiments collect the execution time of VAT with different distillation strategies on CIFAR-10 with 4,000 labels. The second and third group collects the time cost with the baseline of MixMatch and FixMatch on CIFAR-100 with 10,000 labels, respectively. For a fair comparison, each method runs for 250 epochs and all of them are provided with the same computation resources. The results are collated and shown as Fig. 4.7.

From the left of Fig. 4.7, it is observed that: (1) All the distillation components introduce an extra computation to the vanilla VAT (i.e., "W/O"); they commonly increase around 10 seconds per epoch. (2) Compared with other four distillation strategies, ADS only incurs slightly more time, i.e., averagely smaller than 3 seconds per epoch. Again, from the right of Fig. 4.7, one can also see that ADS is lightweight to compute, which agrees with the analyses in Section 4.4.3.



Method	Time (s)
MixMatch	82.6±1.5
MixMatch+NS	84.5±1.3
MixMatch-ADS	86.3±1.5
FixMatch	218.4±23.0
FixMatch-ADS	231.1±22.3

Figure 4.7 Runing time (seconds) comparisons over epochs. **Left:** VAT (i.e., "W/O") with different distillation strategies on CIFAR-10 (4,000 labels). **Right:** MixMatch and FixMatch based methods on CIFAR-100 (10,000 labels).

4.5.7 Scalability to ImageNet

I also evaluate ADS on ImageNet to verify its scalability. FixMatch is employed here due to its superior performance. As a convention of [Xie et al., 2020], 10% training data are used as labeled data and the rest are treated as unlabeled samples. For a fair comparison, I do not exhaustively modify model hyper-parameters. The reported top-1 and top-5 error rate of FixMatch are 28.54±0.52% and 10.87±0.28%, respectively. By conducting FixMatch-ADS on this task, the classification performance is improved on both two metrics to a certain degree. FixMatch-ADS achieves 26.90±0.58% top-1 error rate and 10.25±0.24% top-5 error rate. Meanwhile, by recording the running time, I find out that compared with the naive FixMatch, the extra time cost per epoch for FixMatch-ADS is averagely less than 20 seconds, demonstrating that ADS is a lightweight plug-in distillation strategy for SSL methods in practice.

4.5.8 Ablation study

In this section, VAT and FixMatch are used as baselines which are executed on CIFAR-10 with all but 4,000 labeled data to deconstruct ADS and explore the impact of the sharpening power r.

Irreplaceable sharpening in ADS

ADS consists of two main components: *sparsemax* for candidates selection, and *sharpening* for enhancing the sparse predictions. I justify their dependence by replacing the sharpening of ADS with other distillation strategies. The experimental results are shown as Table 4.3, where the average test errors are reported.

Table 4.3 Test error on CIFAR-10 with 4,000 labels, where "Sp" is short for "Sparsemax". Note that Sparsemax+ME does not apply to FixMatch.

Baseline	None	Sp	Sp+ME	Sp+PL	Sp+NS	ADS
VAT	14.72	13.15	13.59	14.26	13.02	**12.40**
FixMatch	5.92	5.75	-	5.75	5.44	**5.21**

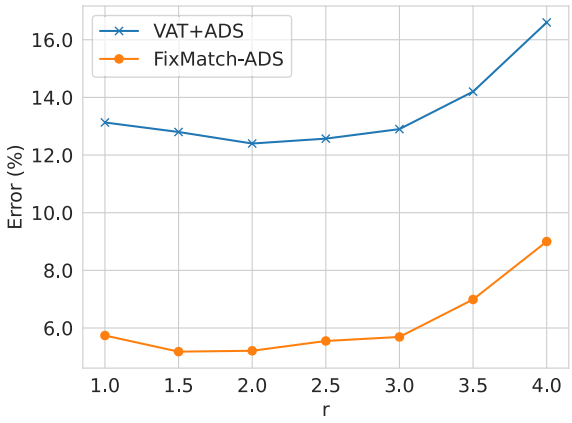


Figure 4.8 Impact of different values for the power r. The best r is in the range of $[1.5, 2.5]$ for VAT+ADS, and $[1, 2]$ for FixMatch-ADS.

Table 4.3 exhibits that: (1) ADS achieves the best performance compared with other different combinations. (2) Sparsemax solely improves the baseline methods because it encourages the consistency loss to focus on the confusing classes only. (3) However, combining sparsemax with other distillation approaches, such as Sparsemax+ME and sparsemax+PL, may hurt the performance. This is because both ME and PL expect the single activation, failing to take the advantages in candidates selection of sparsemax. (4) Sparsemax+NS seems a good combo, but getting the proper thresholds is not easy as NS is found threshold sensitive [Chen et al., 2020b]. (5) Finally it is concluded that sparsemax and sharpening both contribute to ADS, and this combination is always better than the others.

Lazy choice for the value of power r

The default value of the power r follows the convention of the MixMatch [Berthelot et al., 2019b], i.e., $r = 2$. I justify this lazy choice for ADS by studying its impact with different r values. To this end, VAT+ADS and FixMatch-ADS on CIFAR-10 (4,000 labels) are executed by traversing r from 1 to 4 with the step size of 0.5. The test errors are eventually reported in Fig. 4.8.

From Fig. 4.8, it is observed that: (1) A larger r, e.g., $r > 3$, often leads to an unsatisfactory performance. This is rational because a class that is not the actual true label but with a probability slightly greater than the others will be over-encouraging by the large r.. (2) Although $r = 2$ is not always the best choice for ADS (FixMatch-ADS achieves the best performance when $r = 1.5$.), the test errors are consistently smaller when r is in the range of $[1, 2.5]$. Hence, I conclude that setting $r = 2$ across different methods is lazy but effective for ADS.

4.6 Summary

Motivated by using semi-supervised learning (SSL) to infer the missing discrete attribute values or labels, I have presented a distillation technique in this chapter. Realizing that minimizing prediction uncertainty on unlabeled data is a key factor to achieve good performance in SSL, I have investigated most related works which distill low-entropy prediction by either accepting the determining class (with the largest probability) as the true label or suppressing subtle predictions (with the smaller probabilities). Unarguably, these distillation strategies are heuristic and less informative for model training. From this discernment, I propose a dual mechanism, named ADaptive Sharpening (ADS), which first applies a soft-threshold to adaptively mask out determinate and negligible predictions, and then seamlessly sharpens the informed predictions, distilling certain predictions with the informed ones only. I have theoretically analyzed the traits of ADS by comparing it with various distillation strategies. Numerous experiments have been done to verify that ADS can significantly improve the state-of-the-art SSL methods by making it a plug-in. ADS forges a cornerstone for future distillation-based SSL research.

Chapter 5

Model Tuning without Peeking on Target Data

Learning without accessing target data is a challenging but interesting task because the entire dataset is restricted to the learning executor. This chapter studies how a learning executor conducts tuning from only limited feedbacks about target data. With model candidates as queries and model performances as feedbacks, the core problem is then converted to how to make use of query chances to achieve a better model. Since many models are commonly optimized by gradients, estimating gradients from query-feedbacks is a promising way to remedy such a problem. In addition, in the main body of this chapter, I explore a practical strategy for the scenario where a deep model needs tuning without peeking on target data.

5.1 Problem Understanding

5.1.1 Model tuning without back-propagation

Plainly using a provided model usually cannot fulfill the requirement of downstream tasks, probably on account of the distribution shift on data [Popov et al., 2018, Wang et al., 2020a], or altered evaluation metrics [Adel et al., 2019]. For example, after the deployment of a language model on user devices, model updates are often needed to enable a stronger performance on personalized data [Popov et al., 2018]. These necessary model updates, known as model tuning in the machine learning community, compensate for the potential discrepancy between upstream and downstream tasks. With accessible target data to the deployed model, the end-to-end back-propagation based model tuning has been demonstrated as a powerful technique in a wide range of fields, such as computer vision [Donahue et al., 2014, Chen et al., 2020c], natural language [Howard and Ruder, 2018, Devlin et al., 2019],

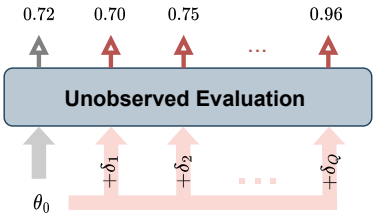
and medicals [Tajbakhsh et al., 2016]. However, there remain many model tuning applications that cannot be dealt with in this manner.

Example. (Model tuning with hided target data) Alice possesses a model developed on a collection of data (public or private), and she would only send out an application instead of the source model (e.g., to protect intellectual property). Bob is a user who owes local data which might be different from Alice's. He is also unwilling to share his data (e.g., personal concerns) but wishes Alice could update the source model so that the corresponding application eventually achieves a pleasant performance for him. This requirement is rational because the users who experiences unsatisfied performance will become discouraged and more likely to quit using the application [Hashimoto et al., 2018]. Typically, Bob returns some feedback about how the candidate model performs, assisting Alice in doing meaningful product updates. Such interactions are normally included in mutually agreed protocols in the real world [Rashidi and Cook, 2009].

Beyond the privacy-motivated situation, there are some other applications for which the standard model tuning is not applicable either. For example, in a user-centric system, the perceptual evaluation that reflects personal and situational characteristics is invaluable to boost the system [Knijnenburg et al., 2012]. These subjective evaluations (e.g., user rating) show the desire for model efficacy and cannot be naturally treated as input data for the tuning purpose. Despite the specific scenarios, I summarize the common trait of all the involved applications. That is, although the model provider cannot access target data for standard model tuning, the model is still hopefully to be improved by merely utilizing some downstream feedback information.

5.1.2 EXPECTED setting

The model tuning problems stated in Section 5.1.1 are abstracted as a new form of learning setting, that is, *Earning eXtra PerformancE from restriCTive feEDdbacks*, dubbed by EXPECTED. Each feedback in EXPECTED is an evaluation result of a legal candidate model, thus the tuning objective is to achieve a satisfactory evaluation result on the target task. In this sense, "earning extra performance" means the improvement of evaluation performance over the initially provided model after multiple queries. "Restrictive" has twofold meanings here. On one hand, the model evaluation result should be uncomplicated, because the common evaluation metric like inference accuracy is typically a score and the user subjective evaluation is probably a star-rating. On the other hand, the number of evaluations is supposed to be limited due to the practical requirement for communication cost or efficiency. Fig. 5.1(a) depicts the EXPECTED problem, where the given model θ_0 is to be tuned so as to achieve a performance



(a) EXPECTED problem

	Supervised Tuning	Unsupervised Tuning	Federated Learning	EXPECTED
$\mathcal{D}$	$\mathcal{X}$ $\mathcal{Y}$	$\mathcal{X}$ $\mathcal{Y}$	$\mathcal{X}$ $\mathcal{Y}$	$\mathcal{X}$ $\mathcal{Y}$
∇_θ	$\nabla\mathcal{L}(\mathcal{X},\mathcal{Y})$	$\nabla\mathcal{L}(\mathcal{X})$	∇_θ	$\nabla\mathcal{L}(\{E_{1:Q}\})$

(b) Comparison of model tuning settings

Figure 5.1 Overview of EXPECTED. (a) Given a deployed model parameterized by θ_0, EXPECTED aims to adapt it to the target task with limited query-feedbacks (budget Q) through the unobserved evaluation. (b) The unobserved evaluation is instanced by the inaccessibility of target data. In this case, EXPECTED is compared with other three model tuning settings from the aspects of (1) how much information about target data $\mathscr{D}$ is accessible and (2) how the gradient information ∇_θ is attained. The grey filling indicates the object is unobserved to the learning executor. In term of the federated learning, although local data $\mathscr{X},\mathscr{Y}$ is inaccessible to the global model, the true gradient ∇_θ is actually returned. Note that E_i is informally short for $E(\mathscr{D};(\theta_0+\delta_i))$.

as high as possible with Q queries. To make it more understandable, "unobserved evaluation" is used to absorb different cases of the aforementioned applications.

EXPECTED challenges the existing research and poses a new and difficult model tuning task. Without the explicit target data which has been treated as an indispensable ingredient for learning, model tuning cannot be executed through the standard back-propagation implemented in different software repositories. In addition, although one can design some heuristic strategies to guess what a better model for the target task looks like from feedbacks, conducting a gradient-based optimization for model tuning remains troublesome, especially for modern Deep Neural Networks (DNNs) which are in high dimensional space and often with complex structures.

5.1.3 Comparison with other model tuning settings

Adapting a pre-trained model to the related target tasks motivates the model tuning setting. For a better statement, I assume the inaccessible target data only differs source data by (input) distribution shift (Refer to the case of altered metrics in Section 5.4.3), and thus the network architecture does not need modifying. Fig. 5.1 (b) compares EXPECTED with the three most related model tuning settings from the following two aspects.

1) Model tuning setting evolves with more restrictive target information accessible. If sufficient target data including features and labels are accessible during tuning, this is literally the supervised learning paradigm, i.e., fine-tuning [Donahue et al., 2014]. Once

the label information is absent, it comes to the unsupervised tuning, which is also known as the test-time training [Sun et al., 2020, Wang et al., 2020a] or source-free unsupervised domain adaptation [Sahoo et al., 2020, Li et al., 2020a, Liang et al., 2020]. Federated learning [McMahan et al., 2017] lets the decentralized global model fit local data without sharing them out. While studied in a one-to-many context, it can be viewed as a model tuning process from global to local[1]. However, one can see that federated learning preserves local data by bringing model training to the device, which is in fact not applicable to the scenarios where intellectual property is also concerned as referred in the previous example. Uniquely, the proposed EXPECTED neither accesses features nor labels of target data, and it only allows limited two-way communications, i.e., querying with model candidates and receiving model performances as feedbacks.

2) More restrictive target information implies a harder gradient-based optimization. Through the above statement, I notice that both supervised tuning and federated learning actually take the sufficient gradient information because they are empirically derived on the labelled target data. In terms of unsupervised tuning where only $\mathscr{X}$ is observed, model gradient is computed from the self-supervision formulation [Sun et al., 2020] or in a fashion of self-training [Yarowsky, 1995]. Therefore, their gradients might be biased (See CIFAR-10-C experimental results in Section 5.4.2). It is seen all these three settings can be conclusively categorized to the data-driven model tuning, as their gradients can be sample-wise decomposed. Things for EXPECTED are indirect by contrast, because the ingredients of EXPECTED for computing gradients are query models and their feedbacks. By understanding every feedback E_i as a summary statistic of the target data in terms of the i-th query model, EXPECTED is consequently interpreted as a model-driven tuning problem. In particular, the proposed algorithms are then expected to achieve the compared performance with the data-driven model tuning methods, even when the query budget is not very generous.

5.2 Preliminaries

Since a great number of symbols will be used in the rest of this chapter, I first prepare the common mathematical notations as Table. 5.1 for the convenience of reading.

Problem Formulation. Let F_{θ_0} denote the initially provided (pre-trained) model which is parameterized with θ_0, $\mathscr{D}$ denote the inaccessible target data that the model aims to adapt

[1]The involved global-local interaction strictly becomes a model tuning process when a collaborator has data changes and aims to acquire a fine-tuned model from the global model, referred by the recent work [Mazumder et al., 2021].

Table 5.1 Common mathematical notations in this chapter.

Notation	Explanation
$\mathscr{D} = \{\mathscr{X}, \mathscr{Y}\}$	target data with features $\mathscr{X}$ and labels $\mathscr{Y}$
Q	query budget
H	layer number of tuned parameters
θ_i	model parameter for query index i
θ^t	model parameter updated after t-th iteration
$F_\theta(\cdot)$	model F parameterized with θ
$E(\cdot)$	evaluation metric
$G(\cdot)$	performance gain by tuning
$\pi(\cdot)$	distribution of model parameters
$l_h \in \theta$	parameters of the h-th layer
p_h	probability of the h-th layer to be sampled
u	the number of queries for a unit update
b	batch size of samplings
σ	standard variance
$\varepsilon \sim \mathscr{N}(0, I)$	standard Gaussian noise
$\lvert\cdot\rvert$	dimension of a vector
$\lvert\lvert\cdot\rvert\rvert$	ℓ_2-norm of a vector

to, and Q denote query budget, i.e., the tolerant number of model evaluations. For a probing candidate model $F_{\theta_i}(1 \leq i \leq Q)$, evaluation function $E(\cdot)$ measures its performance over target data $\mathscr{D}$ and returns a score s_i [2] as feedback. That is, $s_i = E(\mathscr{D}; F_{\theta_i})$. Supposing a larger score is preferred, e.g., accuracy, EXPECTED aims to solve the following problem,

$$\theta_* = \arg\max_{\theta} E(\mathscr{D}; F_\theta), \;\; s.t. \; \#queries \leq Q. \tag{5.1}$$

Please also see Fig. 5.1(a) for this example. Note that I will use an alternative form $E(\mathscr{D}; \theta)$ or $E(\theta)$ to replace $E(\mathscr{D}; F_\theta)$ for the convenient expression in the rest of the paper when it does not cause any ambiguity.

A Naive Approach – Random Search. With query chance budgeted by Q, one can randomly perturb the deployed model and ask for its evaluation on the target data. Afterwards, the model that achieves the best performance is selected as the optimal approximation of θ_*. That is,

$$\theta_* \approx \theta_0 + \arg\max_{\delta_i} \{E(\mathscr{D}; \theta_0 + \delta_i)\}_{i=1}^{Q}, \tag{5.2}$$

[2] When multiple evaluation metrics are used, a tuple might be returned of which s_i could be as an element. One can refer to Section 5.4.3 for a case study about this scenario.

where δ_i represents the difference between the initially provided model F_{θ_0} and the tuned model F_{θ_i}. If each δ_i is derived independently, solving Eq. (5.2) comes to a Random Search [Bergstra and Bengio, 2012] game, which may not meet the need of aforementioned restrictive conditions as the parameter space is too large to do a search in this way (See Section 5.4.2 for experimental demonstrations).

Notice. Transferring other emerging hyperparameter searching techniques [Buczak and Horn, 2021] or advanced evolution algorithms [Opara and Arabas, 2019] to the model tuning scenario is beyond my scope. This study will focus on how to effectively solve the EXPECTED problem based on gradient estimation, especially when modern DNNs are to be tuned.

5.3 Tuning from Restrictive Feedbacks

This section starts with a general case in which I aim to optimize θ by taking advantage of the feedback information. With the consideration of model structures, the second part focuses on tuning DNNs under EXPECTED.

5.3.1 Gradient-based optimization from query-feedbacks

The constraint about the query number in objective (5.1) can be simply eliminated by applying a stopping criterion about the performance gain. For convenience, I will treat it as an unconstrained problem by still running the full Q queries.

Learning the distribution of model parameters

If the evaluation function $E(\cdot)$ of EXPECTED denotes the classification accuracy, its specific form is then written as

$$E(\mathscr{D};\theta) = 1 - \frac{1}{|\mathscr{D}|} \sum_{(x_i,y_i)\in\mathscr{D}} \mathbb{I}(F_\theta(x_i) \neq y_i), \tag{5.3}$$

where $\mathbb{I}(\cdot)$ is the sign function, a.k.a. zero-one loss. The right hand term is the negative expression of empirical risk, which suggests that problem (5.1) can be decomposed over the target samples, i.e.,

$$\max_\theta \mathbb{E}_{x,y\sim p_{(x,y)}} E(x,y;\theta). \tag{5.4}$$

Problem (5.4) can be viewed as the standard tuning paradigm with the indifferentiable loss. As $p(x,y)$ is agnostic, I alternatively consider the following form via introducing the

distribution of θ,

$$E(\mathscr{D};\theta) = E(\mathscr{D};\mathbb{E}_{\theta\sim\pi(\theta)}(\theta)) \approx \mathbb{E}_{\theta\sim\pi(\theta)}E(\mathscr{D};\theta), \tag{5.5}$$

where θ is assumed sampled from the parameter distribution $\pi(\theta)$, the equality holds by defining θ as its expectation over $\pi(\theta)$, and the later approximation follows [Wierstra et al., 2014]. Since the optimal θ_* is intended to obtain, solving problem (5.1) can be written as

$$\begin{aligned} \theta_* &\sim \pi_*(\theta), \\ \pi_*(\theta) &= \arg\max_{\pi(\theta)} \mathbb{E}_{\theta\sim\pi(\theta)}E(\mathscr{D};\theta), \end{aligned} \tag{5.6}$$

where $\pi_*(\theta)$ represents the best estimation of $\pi(\theta)$, and the expectation is taken over all candidate models. One can see that this proxy objective relaxes the optimization to θ into characterizing its distribution $\pi(\theta)$.

To make it practical, I parameterize π by the density probability $\pi(\theta|\omega)$, where ω denotes the distribution parameters. As a result, solving problem (5.6) requires the maximization of the following objective,

$$J(\omega) = \mathbb{E}_{\theta\sim\pi(\theta|\omega)}[E(\mathscr{D};\theta)] = \int E(\mathscr{D};\theta)\pi(\theta|\omega)\,d\theta. \tag{5.7}$$

The gradient of Eq. (5.7) w.r.t. ω can be computed by

$$\begin{aligned} \nabla_\omega J(\omega) &\overset{①}{=} \mathbb{E}_{\theta\sim\pi(\theta|\omega)}[E(\mathscr{D};\theta)\nabla_\omega \log\pi(\theta|\omega)] \\ &\overset{②}{\approx} \frac{1}{b}\sum_{i=1}^{b} E(\mathscr{D};\theta_i)\nabla_\omega \log\pi(\theta_i|\omega), \end{aligned} \tag{5.8}$$

where ① uses the so-called log-likelihood trick [Wierstra et al., 2014] which enables the gradient decoupled from the evaluation function $E(\cdot)$, ② adopts the Monte Carlo approximation by empirically conducting b samplings from $\pi(\theta|\omega)$, i.e., $\theta_1,...,\theta_b$. From Eq. (5.8), it is noticed that the involved samplings θ_i for estimating the gradient of ω can be properly achieved by query chances of EXPECTED. Concretely, in every iteration, I consume b queries to draw from the current $\pi(\theta|\omega)$ which are used to estimate the gradient of ω. Then I update ω by the gradient ascent and obtain a new $\pi(\theta|\omega)$ for the next round of samplings. This process will not terminate until Q queries run out or some candidate is already satisfactory.

Implementation with Gaussian prior

Recall that the canonical form of training a supervised model is i.i.d. log-likelihoods plus a log prior. It is known that the widely used weight regularizer from literature is the weight decay, which corresponds to a centered Gaussian prior. Following this convention, I instance the distribution of the model parameter $\pi(\theta|\omega)$, i.e., $\omega = [\mu, \Sigma]$. Optimizing ω requires the natural gradient for scale conformity [Wierstra et al., 2014] which generally involves the inverse of the Fisher information matrix with the size of $|\theta| \times |\theta|$ in my case. To reduce the burden of heavy computation, I assume that $\Sigma \approx \sigma^2 I$ will not hurt the tuning performance by much, which enables me to treat σ^2 as a hyperparameter and estimate the gradient of θ only. For example, at the first iteration, μ is initialized by θ^0 ($\theta^0 := \theta_0$), i.e., $\pi(\theta|\omega) = \mathcal{N}(\theta|\theta^0, \sigma^2 I)$. By leveraging the reparameterization technique, the candidate models are sampled around the current model θ^0, i.e., $\theta_i = \theta^0 + \sigma\varepsilon_i$ ($1 \leq i \leq b$), where $\varepsilon_i \sim \mathcal{N}(0, I)$. Taking such samplings into Eq. (5.8) yields the gradient estimation w.r.t. ω. As updating ω is equivalently updating θ, I can directly give the gradient estimation w.r.t. θ,

$$\nabla \mathbb{E}[E(\theta)] \approx \frac{1}{\sigma b} \sum_{i=1}^{b} \varepsilon_i E(\theta + \sigma\varepsilon_i), \tag{5.9}$$

where $\mathscr{D}$ is dropped from now on for simplicity. Although starting from a surrogate objective (5.6), the above implementation helps optimize model parameters θ directly.

Before taking Eq.(5.9) into a gradient ascent update, I exhibit two techniques to facilitate this gradient estimation. First, I adopt antithetic sampling [Geweke, 1988] which is demonstrated to stabilize the update. That is, in each round, I only independently sample $b/2$ Gaussian points ε_j and let the rest be the negative copies, i.e., $\varepsilon_{b-j+1} = -\varepsilon_j$. Second, to reduce the impact of the scale of model performance, I normalize the feedbacks by subtracting the mean and then dividing by the standard deviation before using them. Such a normalization step has been demonstrated to maintain a constant learning rate η [Li et al., 2019] and also provides an important condition for some fundamental facts used in Appendix C.1. Alg. 2 summarizes the whole procedure, which is named Performance-guided Parameters Search (PPS).

Quality analysis of the estimated gradient

I first show that applying the antithetic sampling on Eq. (5.9) can explicitly build the connections between the estimated gradient and the true gradient (data-driven).

Algorithm 2 Performance-guided Parameter Search (PPS)

Require: Initially provided model F_{θ_0}, query budget Q, learning rate η, batch size b, variance σ^2.

1: **for** $t = 0, ..., \lfloor Q/b \rfloor$ **do**
2: Sample $\{\varepsilon_j\}_{j=1}^{b/2} \sim \mathcal{N}(0, I)$, and for each j get $\varepsilon_{b-j+1} = -\varepsilon_j$.
3: Generate candidate models $\{\theta_i\}_i^b$ as queries where $\theta_i = \theta^t + \sigma\varepsilon_i$. $\#\theta^t = \theta_0$ if $t = 0$
4: Collect and normalize feedbacks $\{E(\mathscr{D}; \theta_i)\}_{i=1}^b$.
5: $\theta^{t+1} \leftarrow \theta^t + \frac{\eta}{\sigma b} \sum_{i=1}^b \varepsilon_i E(\mathscr{D}; \theta^t + \sigma\varepsilon_i)$.
6: **end for**

Ensure: $\theta^{\lfloor Q/b \rfloor + 1}$.

Proposition 2 *If σ is small, any estimated gradient $\nabla\mathbb{E}[E(\theta)]$ derived by Alg. 2 can be seen as a projection of the corresponding true gradient $\nabla E(\theta) \in \mathbb{R}^{|\theta|}$ onto a lower dimensional space with $b/2$ independent random Gaussian vectors being bases.*

Proof: When antithetic sampling is used, the expression of Eq. (5.9) can be written as:

$$
\begin{aligned}
\nabla\mathbb{E}[E(\theta)] &\approx \frac{1}{\sigma b} \sum_{i=1}^{b} E(\theta + \sigma\varepsilon_i)\varepsilon_i \\
&\overset{①}{=} \frac{1}{b/2} \sum_{i=1}^{b/2} \frac{E(\theta + \sigma\varepsilon_i) - E(\theta - \sigma\varepsilon_i)}{2\sigma} \varepsilon_i \\
&\overset{②}{\approx} \frac{1}{b/2} \sum_{i=1}^{b/2} \left(\nabla_{\varepsilon_i} E(\theta) \cdot ||\varepsilon_i||\right) \varepsilon_i \\
&\overset{③}{=} \frac{1}{b/2} \sum_{i=1}^{b/2} \langle \nabla E(\theta), \varepsilon_i \rangle \varepsilon_i \\
&\overset{④}{=} \frac{1}{b/2} \sum_{i=1}^{b/2} \mathrm{Proj}_{\varepsilon_i}(\nabla E(\theta)) \cdot ||\varepsilon_i|| \cdot \varepsilon_i,
\end{aligned}
$$

where ① follows the step 2 of Alg 2, ② uses the definition of directional derivative when $\sigma \to 0$, ③ rewrites the directional derivation into a form of the dot product, and ④ is a natural reformulation to align with the definition of vector projection. Taking all the independent ε_i as bases, the coordinate value of $\nabla\mathbb{E}[E(\theta)]$ onto each base is $\mathrm{Proj}_{\varepsilon_i}(\nabla E(\theta)) \cdot ||\varepsilon_i||$, which completes the proof. ■

Then I introduce the following theorem which quantifies how well a random projection preserves the length information of a vector.

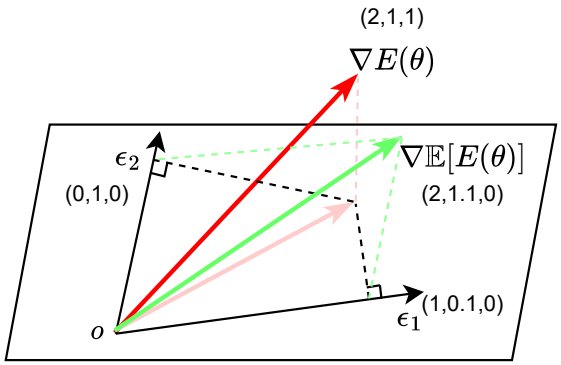


Figure 5.2 An example of how the estimated gradient $\nabla\mathbb{E}[E(\theta)]$ approximates the true gradient $\nabla E(\theta)$. The pink arrow denotes the projection of $\nabla E(\theta)$ onto selected finite bases ε_1 and ε_2. One can easily verify that a true gradient $(2,1,1)$ under this decomposition corresponds to an estimated gradient $(2,1.1,0)$.

Theorem 4 *Let $M \in \mathbb{R}^{|\theta|\times\frac{b}{2}}$ denote the random projection matrix with $||\varepsilon_i|| \cdot \varepsilon_i \quad (i = 1,2,...,\frac{b}{2})$ being the columns. For the true gradient $\nabla E(\theta)$ at any θ, I have*

$$\Pr\left\{(1-\xi)||\nabla E(\theta)||^2 \leq ||M^T\nabla E(\theta)||^2 \leq (1+\xi)||\nabla E(\theta)||^2\right\} > 1-2e^{-C\xi^2 b},$$

where $\xi \in (0,1)$ and $C > 0$ is a constant. ■

Theorem 4 is a direct application of the Johnson-Lindenstrauss Lemma [Matoušek, 2013] but uses the unnormalized Gaussian bases. The norm of projected coordinates is lower and upper bounded, which means the length of true gradient $\nabla E(\theta)$ is almost preserved after the projection M. Let $a \in \mathbb{R}^{\frac{b}{2}}$ denote the coefficient vector with the i-th entry being $\text{Proj}_{\varepsilon_i}(\nabla E(\theta)) \cdot ||\varepsilon_i||$. As any $\varepsilon_i, \varepsilon_j$ are nearly orthogonal [Gorban et al., 2016], according to Proposition 2, I have

$$||\nabla\mathbb{E}[E(\theta)]||^2 \approx ||a||^2. \tag{5.10}$$

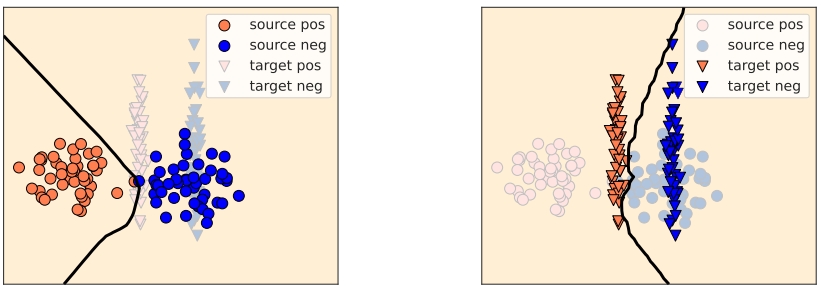
Since $a = M^T\nabla E(\theta)$, from Theorem 4, it is concluded that the estimated gradient almost preserves the length of corresponding true gradients. Here I present an example in Fig. 5.2 to depict the involved connections.

Remark 4 *The approximation in Eq.* (5.10) *hinders the strict comparison between $||\nabla E(\theta)||^2$ and $||\nabla\mathbb{E}[E(\theta)]||^2$. However, generally speaking, when b increases, this approximation is more accurate as the fidelity is better preserved, and the bound of Theorem 4 becomes tighter as well.*

Toy example for PPS

I present a toy example to verify the efficacy of Alg. 2. Three-layer perceptron networks (3-MLP) are firstly pre-trained on source data (two Gaussians with the variance of $[0.7, 0.7]$) and are then tuned following Alg. 2 on target data – another two Gaussians with the variance of $[0.1, 1.5]$). In this experiment, the last layer of the 3-MLP is tuned. The evaluation error on half of randomly selected target data is used as the feedback, and the query budget is set as 80.

Fig. 5.3(a) shows that the classifier is able to correctly classify two classes of source data after pre-training, but fails on target data. Fig. 5.3(b) shows that the pre-trained classifier finally adapts to the target data successfully after conducting Alg. 2. In particular, as most model parameters are frozen during tuning, it is observed that the tuned model eventually maintains a good classification performance on source data as well.



(a) Pre-training on source (b) PPS for target

Figure 5.3 Example of EXPECTED optimized by PPS. (a) Pre-training on source data delivers the initially provided model. (b) The given model successfully adapts to target data through PPS within 80 queries.

5.3.2 Extension to complex models

Typically, many tasks only tune a proportion of model parameters for better generalization (I actually have followed this idea in the example of Section 5.3.1). In the rest of this chapter, I abuse θ as the tuned parameters and present how they are efficiently updated when they are with a complex structure.

Limitations of PPS and my strategy

In many applications, θ consist of parameters distributed across different layers of DNNs. For example, researchers adjust all the batch normalization layers to narrow the source-target discrepancy [Li et al., 2016, Wang et al., 2020a]. However, I point out that directly applying

PPS in such a scenario suffers from two limitations: (1) PPS is not sufficiently efficient if $|\theta|$ is large because the gradient estimation in a high-dimensional space is found less stable [Li et al., 2019]. (2) Treating θ as an entirety overlooks the fact that different layers should have different levels of impact [Li et al., 2020b], which means PPS may waste query chance on less contributive layers.

My strategy for overcoming the above limitations consists of two techniques. **(1) Layer-wise tuning.** The first limitation of PPS can be understood that it fails to consider the inner connections among the chained parameters. More concretely, parameters on later layers cannot immediately capture the change happening in the former layers at each iteration. To remedy this problem, I partition the tuned parameters layerwise, i.e., $\theta = \{\ell_1, \ell_2, ..., \ell_H\}$, where $\ell_h (1 \leq h \leq H)$ represents the parameters of the h-th layer. Then I propose to tune parameters θ more naturally; every time I only focus on updating a single layer's parameters ℓ_h while freezing the remaining layers, i.e., $\theta - \{\ell_h\}$. The basic idea is related to the sequential training on neural networks [Belilovsky et al., 2019]. However, they try to scale the end-to-end training to large size datasets while I focus on query-efficient tuning from feedbacks. **(2) Query budget reassignment.** I model the importance of different layers by inspecting their performance improvements. Instead of leaning on the static weights from the prior knowledge [Kirkpatrick et al., 2017], I propose to dynamically assign more queries to the layers which receive bigger pay-offs.

Query-efficient layerwise tuning

Let $\alpha \in \mathbb{R}^H$ be a layer importance vector. I intend to map it to a $(H-1)$-dimensional simplex, based on which a layer l_h is sampled to be updated or not. As all the tuned parameters are deemed useful, a base probability $\frac{\gamma}{H}$ is maintained for each layer, and the sampling distribution p is then written component-wise

$$p_h = (1-\gamma)\frac{\exp(\alpha_h)}{\sum_{i=1}^{H}\exp(\alpha_i)} + \frac{\gamma}{H}. \tag{5.11}$$

In practice, I make the base probability deterministic to guarantee a least update for each layer. That is, a unit execution is conducted for every layer before samplings, which equals to give u queries[3] to each layer beforehand. As the least update tells us which layers are more contributive, their respective performance improvements will be used to measure the layer importance. Specifically, for the h-th layer at the $t+1$-th iteration, the update rule for

[3]As the number of parameters in different layers varies, u is not identical for different layers in practice. I use the same u for the convenient statement here.

α is written as:

$$\alpha_h^{t+1} = \alpha_h^t + \beta \underbrace{\max\{0, \bar{E}_h^{t+1}(\theta) - \hat{E}_{h-1}^{t+1}(\theta)\}}_{\text{Average improvement } I_h^{t+1}}, \tag{5.12}$$

where β represents how much the observed improvement is relied on from the least update. The involved average improvement I_h^{t+1} is obtained by comparing with the last layer update. That is, $\bar{E}_h^{t+1}(\theta)$ denotes the average evaluation result of the candidates models that perturb h-th layer at the iteration of $t+1$,

$$\bar{E}_h^{t+1}(\theta) = \mathbb{E}_{\ell_h \sim \mathcal{N}(\ell_h^t, \sigma^2)} E\left(\{\ell_1^{t+\frac{1}{2}}, ..., \ell_{h-1}^{t+\frac{1}{2}}, \ell_h^t, ..., \ell_H^t\}\right),$$

and $\hat{E}_{h-1}^{t+1}(\theta)$ denotes the evaluation after $(h-1)$-th layer is updated by unit queries during the iteration of $t+1$,

$$\hat{E}_{h-1}^{t+1}(\theta) = E\left(\{\ell_1^{t+\frac{1}{2}}, ..., \ell_{h-1}^{t+\frac{1}{2}}, \ell_h^t, ..., \ell_H^t\}\right).$$

Particularly, $\hat{E}_{h-1}^{t+1}(\theta) = \hat{E}_H^t(\theta)$ if $h = 1$.

Note that the above statement essentially suggests to split a single iteration into two stages. The first stage is in charge of the least update for every layer which yields the importance factors used for the queries reassignment during the second half. I emphasize that the first stage is indispensable because the goal to inspect the response of every layer with the fact that only a few layers selected in the second stage. The complete algorithm, named Layerwise Coordinate Parameter Search (LCPS), is formally summarized into Alg. 3.

Regret analysis

Reassigning different numbers of queries to different layers can be viewed as an exploration-exploiting game. Concretely, tuning without layer importance equals a pure exploration process, which is not a good option when the limited queries are given but layer discrimination does exist. By contrast, barely updating the most important layer corresponds to an exploiting strategy, which is not wise unless the query number is very small. If tuning a specific layer is regarded as selecting a slot machine to play, my strategy can be understood to solve a multi-armed bandits problem [Seldin et al., 2013]. In this sense, I aim to minimize the following expected regret,

$$G_{\max}(T) - \mathbb{E}[G_{\mathscr{A}}(T)], \tag{5.13}$$

Algorithm 3 Layerwise Coordinate Parameter Search (LCPS)

Require: Initially provided model F_{θ_0}, query budget Q, learning rates η, β, batch size b, variance σ^2, unit size u.

for $t = 0, ..., \lfloor Q/b \rfloor$ **do**
  for $h = 1, ..., H$ **do**
    Update $\ell_h^{t+\frac{1}{2}}$ with u queries following Alg. 2.
    Compute average improvement I_h^{t+1} through Eq. (5.12).
    $\alpha_h^{t+1} \leftarrow \alpha_h^t + \beta I_h^{t+1}$.
  end for
  Compute p^{t+1} by Eq. (5.11) with $\gamma = 0$.
  for $j = 1, ..., \lfloor (b - Hu)/u \rfloor$ **do**
    Sample a layer h with p_h^{t+1}.
    Update ℓ_h^{t+1} with u queries following Alg. 2.
  end for
end for

Ensure: $\theta^{\lfloor Q/b \rfloor + 1}$.

where $T = 1, 2, ..., \lfloor Q/b \rfloor$ is the horizon time, $G_{\max}(T)$ denotes the performance gain by picking the unknown optimal layer sequence, $G_{\mathscr{A}}(T)$ stands for the performance gain achieved by a designed algorithm, and the expectation is taken over the sampled layer sequences.

Straightforward optimizing Eq. (5.13) comes to an intractable problem, but this expected regret serves as a measurement to evaluate how the algorithm $\mathscr{A}$ approaches the oracle performance. Here I present an expected regret bound for the proposed LCPS.

Theorem 5 *Given a deep model whose tuned parameters are $\theta = \{\ell_1, \ell_2, ..., \ell_H\}$, for any $\beta > 0$, I have that*

$$G_{\max} - \mathbb{E}[G_{LCPS}] \leq (\beta c(e-2) + 1) G_{\max} + \frac{c}{\beta} \ln H$$

holds for any $T > 0$, where $c = \frac{b - Hu}{u}$, (b is the batch size, u is the unit size), and e is Euler's number. ■

The proof of Theorem 5 follows the sketch of Exp3 algorithm [Seldin et al., 2013], which is left to Appendix C.1 for the readers who are interested in the differences. From Theorem 5, it can be seen that: (1) A smaller c implies less expected regret, which suggests computing the query reassignment more frequently. However, by assuming the performance gain hardly change in a few updates, I can attribute this problem to the selection of batch size b. (2) This weak regret bound is also a function of step-size β. From the Karush–Kuhn–Tucker (KKT) conditions, I can obtain the regret reaches its minimum if I set $\beta = \sqrt{\frac{\ln H}{(e-2)G}}$, where G is the predicted maximum performance gain.

Remark 5 *This regret bound is related to the scale of performance gain; the accumulative performance gain $G_{\max}$ could be very large if $T \to \infty$. Expect for some evaluation measurements like accuracy which naturally makes $G_{\max}$ upper bounded, one can do re-normalization to the immediate reward so that the bound of Theorem 5 remains meaningful.*

5.4 Experiment

I remind readers two rules that all the experiments will obey. First, the experiments include how the initially provided model is produced, i.e., pre-training. But once the pre-training is completed, the source data is no longer used during the tuning, following the convention of standard model tuning. Second, I assume providers would only tune the parameters, which enables a lightweight modification on the users' side and also a tolerant communication cost.

5.4.1 Experimental setup

Datasets. Adult [Kohavi, 1996] is a tabular dataset for categorizing the annual income of different groups of citizens. As personal information is recorded, it is also a benchmark for fairness studies. Amazon review [McAuley and Leskovec, 2013] is a text dataset that contains user comments about various products. The corrupted CIFAR-10/CIFAR-100 [Hendrycks and Dietterich, 2019] are visual datasets where different type/level of corruptions simulate the real-world data noises. STS-B [Cer et al., 2017] predicts the semantic similarity between pairs of sentences which are extracted from different sources.

Except for the particular restatement, the main usages of the above datasets are described as follows. Adult has 14 properties such as country, age, work class, education, etc, and I predict whether income exceeds $50K$ per year. I pre-train a binary classifier on the records with the country of "U.S" and take "non-U.S" records as unobserved target data. Amazon dataset is constructed from Amazon review data with two categories of products selected, i.e., "Electronics" and "Watches". In the experiments, the data-rich category "Electronics" is used to pre-train a prediction model which maps user comments to the rating score ranging from one to five, and "Watches" is treated as the target data. The settings of Adult and Amazon follows the work [Chen et al., 2020a]. In terms of CIFAR-10-C/CIFAR-100-C, the initial provided model is built on clean images, and it is then tuned to fit the disjoint corrupted images following the unsupervised tuning research [Wang et al., 2020a], which mimics the unexpected distribution shift in the real world. Last, I aim to tune BERT [Devlin et al., 2018] and its variants on the STS-B task under EXPECTED. Following the research [Hendrycks et al., 2020], the models are firstly trained on the sentence pairs from the genre of MSRvid

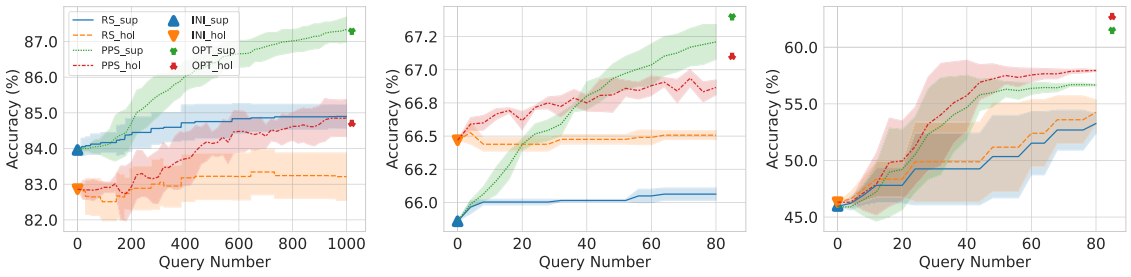
and then tuned to fit the unknown target data which are extracted from Images, where the evaluation metric is Pearson's correlation coefficient.

On the task of corrupted image classification, all the corrupted images (target data) are treated as the tuning data for a fair comparison with the unsupervised tuning methods. Throughout all the remaining datasets, the target data are split into two sets. I do randomly equal splitting for Adult and Amazon and use the default splitting for STS-B. One is the *support set* that is used for evaluating the query efficiency of tuning algorithms, and the other is the *holdout set* on which the model generalization is assessed. The corresponding performances are denoted by "sup" and "hol", respectively.

Models. In respect of Adult, I use a 3-MLP with the penultimate layer of 80 neurons. Following the fashion of [Kristiadi et al., 2020], I simply perturb the weights of the last layer which thus contains 80 parameters for binary classification. For score prediction on Amazon, my implementation is based on the `torchtext` library, in which the first layer is a mapping from the vocabulary to a latent representation, followed by three convolutional layers and a linear transformation to label space. The weights of the first layer are set as the tune parameters (with the size of $250K$) because the remaining layers are found less sensitive to the change of domains according to experiments[4]. In terms of corrupted CIFAR-10/CIFAR-100, I use residual networks [He et al., 2016] with 26 blocks which are implemented by `pycls` library [Radosavovic et al., 2019]. I modulate features for target data by estimating normalization statistics and then update transformation parameters channel-wise. This setting is consistent with the recent research [Sun et al., 2020, Liang et al., 2020], which turns out that the tuned parameters make up a small proportion of the whole model. Similarly, I resort to tuning the layer normalization of BERT and its variants for predicting sentence similarity, managing to earn more performance improvement by tuning on target data.

Baselines. The naive baselines for EXPECTED include the initially provided model by pre-training and supervised tuning, which are dubbed as "INI" and "OPT", respectively. In addition, Random Search (RS) [Bergstra and Bengio, 2012] is also borrowed here in a similar manner for the hyperparameter tuning. All the other baselines are used for specific comparisons, whose results are retrieved from the literature or recomputed when it is possible. For the methods involving randomness, the experiments will be repeated for 10 times for a convincing comparison.

[4]Freezing partial parameters of the provided model is common but empirical in the model tuning community. Although LCPS is able to tease out the useful parameters, this would consume a plenty of queries. In experiments, I would prefer employing LCPS only for "certainly useful parameters".



(a) Adult (b) Amazon (good VOC) (c) Amazon (bad VOC)

Figure 5.4 Performance comparison on Adult and Amazon. Throughout all the experiments, the accuracy on the support set is monotonically non-decreasing, since I display the historically best at every iteration. Note that "good VOC" and "bad VOC" correspond to the different selections of vocabulary. The line shadow represents the standard deviation.

5.4.2 EXPECTED on shifted data distribution

Distribution shift is one of the common roots for model tuning. This group of experiments justify the EXPECTED setting over different tasks under this configuration.

Income classification. Alg. 2 is conducted on Adult with the query budget $Q = 1K$. Fig. 5.4(a) exhibits the classification performance on support set and holdout set, respectively. It is observed that (1) PPS significantly improves the performance of INI and closely approaches OPT at $Q = 1K$. (2) RS only achieves a subtle improvement to the initial model with the same number of queries and thus is less efficient for tuning model parameters.

Rating prediction. Alg. 2 is also run on Amazon, where a small query budget $Q = 80$ is used[5]. When a good vocabulary is carefully selected shown as Fig. 5.4(b), the improvement space w.r.t. the initially provided model is found quite limited, capped by the supervised tuning performance, i.e., OPT. In this case, the accuracy implemented by PPS only increases 1.3% and 0.4% on support set and holdout set respectively. By contrast, when a bad vocabulary is unintentionally selected shown as Fig. 5.4(c), it can found that PPS rapidly boosts the performance of INI, and it becomes stable after 50 queries. In summary, the comparison between Fig. 5.4(b)&(c) shows that (1) PPS consistently earns more performance than RS regardless of different equipments of vocabulary. (2) PPS is found sometimes trapped in a local optimum, which probably because the non-smooth evaluation function is insensitive to model perturbations. In other words, the multiple samplings with a fixed variance probably fail to help the model to escape from saddle points.

[5]I empirically find the desired batch size on this task could be very small, and I use $b = 4$ in the experiments. From Proposition 2, I suspect that the derived gradient for the tuning purpose lies in a very low dimensional space.

Table 5.2 Comparison of different model tuning methods on CIFAR-10-C and CIFAR-100-C with the highest severity.

Method	CIFAR-10-C	CIFAR-100-C	Note
INI	40.8	67.2	-
DAN [Ganin and Lempitsky, 2015]	18.3	38.9	Access X
TTT [Sun et al., 2020]	17.5	45.0	
BN [Schneider et al., 2020]	17.3	42.6	
Tent [Wang et al., 2020a]	14.3	37.3	
RS [Bergstra and Bengio, 2012]	16.7	40.5	$3K$ queries
PPS	15.1	38.6	$3K$ queries
LCPS	13.8	35.2	$1K$ queries

Corrupted image classification. I run both Algs. 2&3 on CIFAR-10-C/CIFAR-100-C with the query number of $3K/1K$ to tune batch normalization layers. In this experiment, more baselines are included, such as Domain Adversarial Netowrks (DAN) [Ganin and Lempitsky, 2015], Test-Time Training (TTT) [Sun et al., 2020], test-time Batch Normalization (BN) [Ioffe and Szegedy, 2015], and test-time adaptation work Tent [Wang et al., 2020a]. Table 5.2 summarizes these methods and presents the average errors of tuning performance, where OPT with a supervised end-to-end tuning is omitted here because it can achieve a very low error. The results show that (1) Tent is the most powerful among unsupervised tuning methods which access the entire features of target data, while the average error of LCPS is surprisingly better than Tent with $Q = 1K$. (2) Even offered more queries, RS and PPS cannot compete with LCPS, implying the advantage of Alg. 3 in tuning modern DNNs. Fig. 5.5 exhibits a close look to the specific results over each type of corruption. It is observed that Tent updates towards a wrong direction on some particular corruptions, such as "motion" and "bright", causing even worse performance than BN. However, such performance drops do not happen to LCPS because feedbacks are simple but reliable (as label information is used during evaluations).

Sentences similarity prediction. Alg. 3 is also run on STS-B in which different models including BERT, RoBERTa, and DistilBERT [Hendrycks et al., 2020] are examined. Pearson's correlation coefficient of each pre-trained model on the holdout set, i.e., the test set of Images, is 0.861, 0.907, and 0.849, respectively. Again, I denote them by "INI_hol" in terms of each backbone. Although these results are found comparable with what they behave on the source task [Hendrycks et al., 2020], it is interesting to know whether they can be further tuned to achieve a better performance. By using $Q = 5K$ queries, RS and LCPS are applied to these three models and the corresponding improvements are shown as Fig. 5.6, where the standard tuning denoted by "OPT_hol" is added in as a reference. The

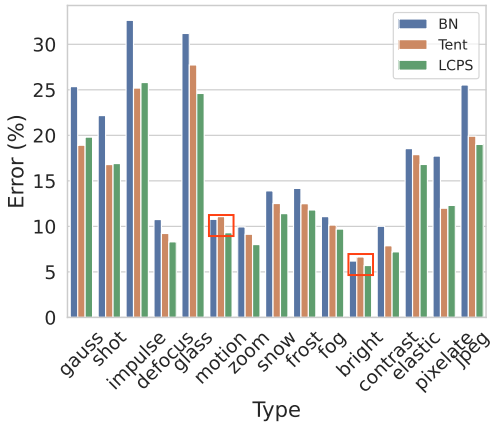


Figure 5.5 Average error (%) over 15 types of corruptions for the highest severity, where RS, PPS, LCPS and Tent are based test-time BN. Red marks denote the failure cases of Tent.

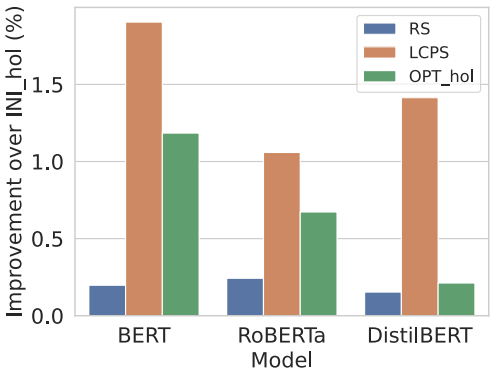


Figure 5.6 Generalization improvement of BERT and its variants after the model tuning on STS-B, which is computed by $\frac{s-s_0}{s_0}$, where s_0 and s represent the model performance before and after tuning, respectively.

experimental results show that LCPS significantly improves the pre-trained model across different backbones, which certainly outperforms the RS strategy as well. Surprisingly, the standard tuning fails to upper bound LCPS as that in other experimental tasks. A reasonable explanation is that the variance of stochastic gradients offsets the minor improvement while the whole-set performance feedbacks (summary statistics) are still useful in this case.

5.4.3 EXPECTED for customized evaluation metrics

In some applications like the machine learning service provision, a customized evaluation metric might be needed for clients. Thus, the provided model which is never trained towards such an objective usually cannot fulfill the downstream expectation. In this part, I study two interesting topics as the representatives of this situation. The first one is the fair classification where not only classification accuracy but also fairness critic is considered. The second one is fault-intolerant learning where the original evaluation metric is replaced by another metric in target tasks. For simplicity, I follow the basic configuration about datasets where the data distribution shift still exists, targeting a more challenging model tuning task.

Fair classification

In this experiment, *demographic parity* [Hardt et al., 2016] is adopted as the fairness metric. Suppose a user requires a classifier which is unbiased on gender z ($z = 1$ denotes male and $z = 0$ is for female) in terms of the high salary ($> \$50k$ per year). The corresponding

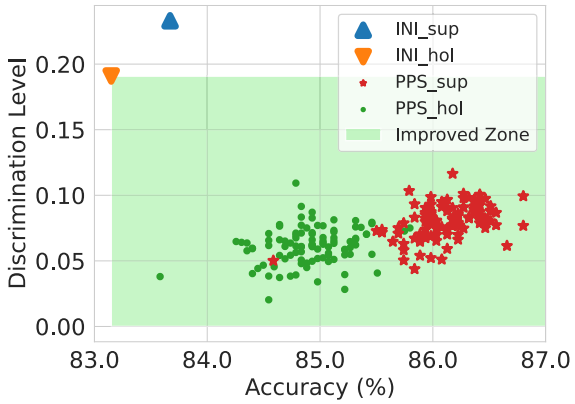


Figure 5.7 Discrimination level reduction for model fairness tuning, where the particles falling in "Improved Zone" represent the models that have been improved in terms of both accuracy and fairness metrics on the holdout set.

Figure 5.8 Evaluation performance (%) of LCPS with top-1 or top-5 error as a tuning metric on two types of corruptions (Gaussian and Impulse noises) over CIFAR-10-C. "Non" represents an initially provided model with the test-time BN is directly evaluated without any tuning efforts. The lowest errors are marked as bold.

Type	Tuning	Error	
		Top-1 (↓)	Top-5 (↓)
Gaussian	Non	25.4	3.0
	w/ Top-1	**17.7**$^{\pm 0.13}$	1.5$^{\pm 0.03}$
	w/ Top-5	19.7$^{\pm 0.18}$	**1.1**$^{\pm 0.04}$
Impulse	Non	33.6	4.6
	w/ Top-1	**21.9**$^{\pm 0.09}$	2.1$^{\pm 0.05}$
	w/ Top-5	24.6$^{\pm 0.14}$	**1.4**$^{\pm 0.03}$

discrimination level of demographic parity then can be defined by $\Gamma(\theta) = |\Pr(F_\theta = 1|z = 1) - \Pr(F_\theta = 1|z = 0)|$. That means every time after a local evaluation, the user will return a two-dimensional tuple with one element for the classification accuracy and the other for the discrimination level, i.e. (E, Γ). Since two metrics commonly compete with each other [Liu and Vicente, 2022], I propose to update their joint gradients as shown in Alg. 4 of Appendix C.2.

Fig. 5.7 shows the results of 100 independent executions of PPS under the above setting. Each green point denotes the model performance of a tuned model on the support set, and red stars are the corresponding performances on the holdout set. The most lower-right is the best. From this figure, one can see that (1) the particles falling in the green zone refer to the models which achieve improvements over the pre-trained model in terms of both classification accuracy and model fairness on the holdout set, making up 100% of the whole trials. (2) The overall tuning accuracy is superior to testing while the discrimination level of testing is slightly better, implying an acceptable discrepancy between tuning and testing. (3) The discrimination level of INI has been dramatically decreased (by more than half) after tuning. Thus the proposed method under EXPECTED factually serves as an efficient fair-tuning approach for inaccessible data.

Fault-intolerant evaluation

One of the common fault-intolerance metrics is top-K accuracy [Chzhen et al., 2021]. Unlike the single output prediction, top-K classification produces lower errors. I take the multi-class classification task over CIFAR-10-C as an example. In the experiment, apart from tuning with top-1 error, top-5 error is used for tuning metric as a comparison. I achieve this by simply replacing the standard top-1 error with the top-5 error during the local evaluation. The experimental results on two corruption types with $2K$ queries, i.e., Gaussian and Impulse noise, are finally reported.

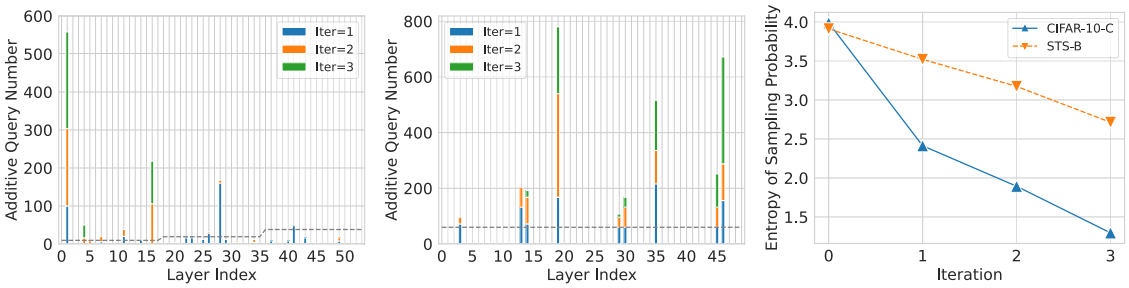
Fig. 5.8 exhibits the results of tuning with top-1 and top-5 metric separately. When any of them is not used for tuning, its corresponding error is computed by evaluating the tuned model on the target task with this metric. For example, regarding images corrupted by Gaussian noises, the model tuned with top-1 metric under EXPECTED eventually achieves about 17.7% error, and I can also obtain its top-5 error by evaluating the tuned model with the top-5 metric whose performance turns out around 1.5%. From Fig. 5.8, one can see that (1) LCPS is efficient for the model tuning under EXPECTED because it has dramatically decreased the classification errors on both metrics. Notably, through $2K$ queries, the top-1 error has been decreased by around 7.7% and 11.7% on two types of corruptions, respectively. (2) Although tuning with the top-1 metric decreases the top-5 error as well, the top-5 error could be reduced to a smaller value when it is directly used as the tuning metric. That means if the user demands a lower top-5 error, LCPS naturally satisfies this requirement by straightforward replacing the top-1 error with top-5 error at the beginning of tuning.

Note. The fairness metric is often hard to optimize since it is a group level measure defined on the entire tuning set. Top-K error is non-differentiable which can be implemented by some extra operation like truncation. That means both of them need some elaborate design in a standard model tuning task. Interestingly, I emphasize that these metrics can be innocently used for my methods as the emitted performance over them is barely collected under EXPECTED.

5.4.4 A close investigation to LCPS

In this part, I further investigate how LCPS works for complex models by visualizing the process of layerwise tuning on CIFAR-10-C (with Gaussian corruption) and STS-B (on BERT). The basic experimental settings follow Section 5.4.2 but I let $Q = 2K$ on CIFAR-10-C for a better comparison.

The experimental results are presented in Fig. 5.9, which show that (1) in each iteration, only partial layers are selected in LCPS for updates, whose additive query numbers turn



(a) CIFAR-10-C (b) STS-B (c) Entropy

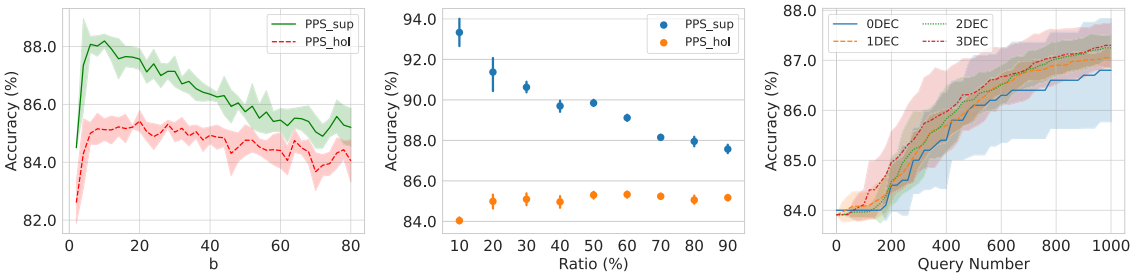
Figure 5.9 Query budget reassignment of LCPS on CIFAR-10-C and STS-B. (a) and (b) are corresponding the results of CIFAR-10-C with Gaussian corruptions and STS-B with BERT being backbones. The grey dashed line indicates the expected query assignment for each layer without the layer importance concern. (c) exhibits the entropy of sampling probability over each iteration for the two experiments.

out much higher than the corresponding expected numbers (which are proportional to $|\ell_h|(h = 1,2,...,H)$ and indicated by the grey dashed lines). (2) A layer selected in the previous iterations would be prone to be selected again later. This is because the sampling probabilities of selected layers are much higher than the remaining ones due to the dominant average improvements (Refer to Eqs. (5.11)&(5.12)). (3) With the increment of iteration number, fewer layers are sampled, which means that the tuning process is towards exploitation given the limited query budget. This observation is also demonstrated by their steadily decreasing entropy of sampling probabilities (See Fig. 5.9(c)). (4) The additive queries which are reassigned during the second stage are dependent on the specific task. Roughly speaking, shallow features update is more crucial to CIFAR-10-C while STS-B prefers deep features. A possible explanation is that Gaussian corruption changes low features significantly while data genre in STS-B is encoded by some high-level information.

5.4.5 Important factors study

Four factors that may have impact on the results are empirically verified in this part.

Batch size. The sampling batch size is varied from 2 to 80 with the step size of 2 on Adult. The results are shown as Fig. 5.10(a). In terms of the support set, the optimal performance is achieved when the batch size is about 10. While it reaches the optima with the batch size being 20 on the holdout set. Hence, I use $b = 20$ throughout all other experiments on Adult. Recall that the sampling batch size for Amazon is quite smaller from Section 5.4.2. Therefore, I remind that this hyperparameter should be carefully selected, especially when



(a) Sampling batch size (b) Support size (c) Precision of feedbacks

Figure 5.10 Ablation study on three factors: sampling batch size, support size, and precision of feedbacks. "XDEC" in (c) means that the feedback value is rounded with X decimals.

the query efficiency is required. One possible workaround to this issue is resorting to an auxiliary validation set before executing tuning.

Support size. The effects of the size of support set is investigated by varying its ratio from 10% to 90% on Adult, and the results are shown as Fig. 5.10(b). With the increase of support size, it becomes harder to fit all the supported samples given the same query budget, but the model generalization, i.e., the accuracy on the holdset set, gradually improves. Additionally, it is found that a smaller support set leads to a larger variance. By contrast, when more than 50% of full support data ($>$ 1000 samples) is used, the model generalization becomes steady with a slight fluctuation only. This observation also suggests that EXPECTED does not demand a big support set in this task, showing a desired trait for some data-scarcity scenarios.

Precision of feedbacks. I also study whether the precision of feedbacks has a direct impact in EXPECTED, which is also important when the back-doors attack [Song et al., 2017a] is concerned (Please refer to Appendix C.3 for more explanations). To this end, I run PPS on Adult by setting the number of decimals for the accuracy values from 0 to 3. The tuning performances on the support set are shown as Fig. 5.10(c), which demonstrates that (1) zero decimal case fails to preserve the quality of feedbacks as the performance drops dramatically compared with the best configuration. (2) The more precise feedbacks usually guarantee the better performance. However, as $\frac{1}{N_{\text{sup}}} > 0.01\%$ on Adult where N_{sup} is the support size, 2-decimal feedback is sufficient to use in this case. Hence, I can attribute the selection of the number of decimals to the side information about the support size.

Layer importance. To verify the necessity of developing LCPS for tuning complex models, PPS and LCPS (with and without layer importance) are compared through running them on CIFAR-10-C in terms of Gaussian and Impulse corruptions. Table 5.3 displays the corresponding results. One can observe that LCPS only needs fewer queries to achieve the preset performances than both PPS and LCPS (w/o), showing a favourable property in tuning DNNs.

Table 5.3 The required query number (K) to achieve the preset tuning performance for two types of corruptions (Gaussian and Impulse) on CIFAR-10-C.

Type (%)	PPS	LCPS (w/o)	LCPS (w/)
Gaussian (22.0)	> 10.0	~ 3.0	~ 0.2
Impulse (20.0)	-	> 8.4	~ 3.5

5.5 Discussion

The affinities of EXPECTED and existing research are discussed in this section to clarify my research contribution.

Model tuning or model adaptation? The previous statement of not changing semantic classes is consistent with the convention of domain adaptation [Ganin and Lempitsky, 2015]. However, I use "tuning" instead of "adaptation" throughout this chapter because of three reasons. (1) Except some source-free studies [Wang et al., 2020a, Liang et al., 2020], most domain adaptation works [Quiñonero-Candela et al., 2009, Ganin and Lempitsky, 2015, Tzeng et al., 2015, Long et al., 2015] are doing the alignment between source and target data, while EXPECTED focuses on tuning a provided model to fit the target data regardless of the performance on the source. (2) The application of handling customized metrics on target data conceptually falls in the model tuning community, because the assumption of source-target distribution shift in domain adaptation is not a necessary requirement in the proposed setting. (3) Standard tuning with the accessible target data usually upper bounds the proposed method. Technically, similar to the standard tuning, the proposed methods can also apply to the case where semantic labels are changed and the classifier's head need renewing. Nevertheless, starting with such a cold status is more difficult to find the optimal solution, especially when a tight query budget is offered.

Black-box optimization or reinforcement learning? The technique used to solve EXPECTED problem actually stems from the idea of Black-box Optimization (BO). Many branches including Bayesian optimization and derivative-free methods have been developed in the past decades. They have a range of applications such as hyperparameter tuning [Turner et al., 2021] and black-box adversarial attacks [Ilyas et al., 2018]. To the best my knowledge, this is a first-of-a-kind work that solves model tuning problem through BO methods. By understanding in this way, the inaccessible data in the proposed setting is now "put into a black box" (Recall the Unobserved Evaluation in Fig. 5.1(a)), and the learning task here is to do model tuning given this restrictive condition. Particularly, I focus on the efficiency issue caused by the high-dimensional structured parameters in DNNs. One may find this technique has also been used in the basic policy gradient derivation in reinforcement learning [Sutton

et al., 1999], especially when the improvement of model performance is viewed as a reward. However, none of them tune model parameters from the literature.

Preserving data content or membership? Preserving data content or membership? Data privacy can be interpreted from different aspects in various scenarios. We argue that in EXPECTED, the risk of data content or membership attacks can be attributed to the control of query budget and precision of feedback. Regarding the data content extraction attacks [Yin et al., 2021], if query models do not have back-doors[6] [Song et al., 2017a], we can see that the amount of data leakage is exclusively dominated by two factors - the query budget and the precision of feedbacks - which we have investigated separately in our experiments. If membership is of interest [Shokri et al., 2017], we can adopt Differential Privacy, introduced in Chapter 3, as a solution. In this case, the participation of each test sample can be offset by adding noise to the feedback. If the feedback is accuracy, we can absorb the noise into the precision problem of feedbacks, as the niose is proportional to a small value i.e., $\frac{1}{N_{\text{sup}}-1} - \frac{1}{N_{\text{sup}}}$. We will explore other types of data privacy with respect to EXPECTED in the future.

5.6 Summary

This chapter has presented a pioneer work of studying how to tune a provided model with only restrictive feedbacks on the target task, which is thought as a strategy for learning with inaccessible data. The main technique is to estimate the distribution of tuned parameters in a similar manner to black-box optimization, where I particularly considered its practicability in tuning modern DNNs. The equipped theoretical analyses supported the utility of the proposed algorithms, and numerous experiments verified their efficacy. I remind readers many interesting research topics can be explored based on this work. Some of those have been mentioned, e.g., tuning a classifier to be fair for the unseen target data, while others may need more investigation, e.g., joint learning with source data when it is available to produce more promising results.

[6] In practice, data owners can ensure this by checking the consistency of the model both before and after testing it on private data.

Chapter 6

Conclusion and Future Work

In this chapter, I briefly conclude three learning tasks and then present some possible trends for future study.

6.1 Conclusion

Learning with restricted data is an appealing research topic if data is thought not free to use. By giving different meanings to restriction, one can arrive at different learning tasks. The three works presented in this thesis are motivated by a practical concern, i.e., data privacy. Therein, I have considered learning with intact data while privacy requirement is applied during a learning process, learning with data in which partial discrete attributes or labels are hidden, and learning with inaccessible data with only feedbacks being available. Actually, beyond privacy, I have also reminded that there might exist other motivations (See Section 5.1.1 for more examples) which come to the same learning tasks. Thus, the concentration of this thesis is how to execute learning under a restricted data context, though sometimes I have provided proper insights to respond to privacy concerns.

Intuitively, one can sense that with more restriction on data, learning becomes more difficult. But what makes me excited is that I have realized and found the key learning difficulty can be always solved from the view of model gradients. Generally speaking, the restricted data prevents us from the informative intact-data-driven gradients. Note that I have to sacrifice precise gradients to preserve privacy of pairwise data, internationally drop the unreliable gradients of overconfident predictions on unlabeled data, and approach the oracle gradients from limited feedbacks. Thus, the idea of gradient manipulation inspires people to think what the property of a specific restriction has brought to learning. And this thesis also aims to encourage deeper understanding towards this direction.

6.2 Future Work

I provide two interesting research topics for future work.

- Rethink the learning with incomplete data with privacy concern. In Chapter 4, I have proposed to use a semi-supervised learning framework to recover the missing attributes or labels, which factually serves as a pre-processing tool for the downstream tasks. That means although the learning executor is not trusted by some participants, their elaborately hidden information can be effectively inferred by SSL algorithms. By contrast, if the learning executor is forced not to "learn" their hidden information, e.g., required by law, how will the learning guarantee such a requirement? This scenario is a bit tricky, as attributes or labels can be collected from these participants but the model is required to do invariant predictions for specific instances w.r.t. the attributes. Note fairness learning [Chouldechova, 2017] has explored the representations invariant to the sensitive attributes but differently, the constraints are applied to the whole dataset.

- A broader concept of restricted data should be explored. With new practical concerns emerging on data, more restricted data should be considered in machine learning. For example, once a model is trained with the entire dataset, how will the model respond if some of the samples are withdrawn? In other words, these samples become restricted to the learning process after training. Of course, one can retrain a new model with the rest of data points. However, this is not economical especially when the withdrawals are frequent and dynamic. There are some works that has touched such problems by considering the influence of each individual sample [Koh and Liang, 2017]. However, there remain many challenges. For example, the computation of the inverse of Hessian matrix is quite high, especially for deep neural networks, or how to properly evaluate if a sample is successfully removed.

Appendix A

Appendix

A.1 Sensitivity Upper Bound for Efficiency

According to Fig. 3.5 (IV), one can see that the privacy for pairwise relationship and feature difference can be both eventually attributed to the 1-hop neighbors of the target node, i.e. s or t. Inspired by this observation, I present the following proposition.

Proposition 3 *For a target pair (s,t), concerning the correlation both on pairwise relationship and feature difference, I have*

$$|\mathscr{P}_{st}| + \min\{c_s, c_t\} \leq De(s) - Co_+(\bar{s}), \tag{A.1}$$

where $De(s)$ means the degree of s and $Co_+(\bar{s})$ represents the increased number of components by removing s from G. Particularly, the equality holds if and only if $c_s = c_t$.

Proof: If all the 1-hop neighbors of s are connected to t, there must exist a node $t' \in V$ that causes $De(s)$ edge-disjoint s-t' paths, i.e. $|\mathscr{P}_{st}| \leq |\mathscr{P}_{st'}| = De(s)$. Otherwise, $|\mathscr{P}_{st}| < De(s)$ always holds. As a result, there are at most $De(s) - |\mathscr{P}_{st}|$ edges that can provide feature inference of s. From the graph theory, the increased number of components $Co_+(\bar{s})$ should satisfy the equality $Co_+(\bar{s}) = De(s) - |\mathscr{P}_{st}| - c_s$. Due to the fact that $\min(c_s, c_t) \leq c_s$, then Eq. (A.1) holds. ■

For all the possible pair, I have

$$\kappa' = \max_{\forall s \in V} \{De(s) - Co_+(\bar{s})\}, \tag{A.2}$$

where κ' is a upper bound of κ in Eq. (3.5). As the component number of a graph can be efficiently calculated, the whole time complexity for searching κ' is $\mathscr{O}(|V|(|V|+|E|))$.

Furthermore, one can greedily search two connected nodes having the maximum sum for the right part of Eq. (A.1) to force $\kappa' = \kappa$.

More importantly, from Proposition 3, I instantly conclude that DPP is upper bounded by the known *Node DP* [Hay et al., 2009]. Deleting a node in graph equals to removing all the edges associated with this node. Thus, I have $\kappa = \max_{\forall s \in V} De(s)$ from the view of Node DP. It is concluded that the proposed DPP is superior to Node DP. Particularly, if the derived graph is a tree with large degree node, κ is 1 for DPP while κ is the maximum node degree for Node DP.

A.2 DPP for Intransitive Relationship Case

The intransitive relationship is a relaxed version of transitive relationship. For the pairwise data with intransitive relationship, I only need to consider the correlation on feature difference. For a target pair $\langle s,t \rangle$, if s and t are mutually 1-hop neighbors, then there should be $1 + \min(c_s, c_t)$ edges that need to concern for the worst case, where c_s and c_t is searched on the subgraph $G - \langle s,t \rangle$. Otherwise, only the number of $\min(c_s, c_t)$ edges need concerning, where c_s and c_t are searched over the entire graph. It is noted that the former case equals to the transitive relationship when $\mathscr{P}_{st} = \{(s,t)\}$. For the latter case, similar to Proposition 3, I have

$$\min(c_s, c_t) \leq De(s) - Co_+(\bar{s}) - 1. \tag{A.3}$$

This inequality follows the fact that $c_s = De(s) - Co_+(\bar{s}) - 1$. One can define κ-neighboring graph of G now by assigning the greater value of two cases to κ. Since the value of κ is determined by the given pairwise data, edge DP cannot handle this case.

A.3 Sensitivity Reduction for Approximate DPP

I first prepare the basic ingredients for approximate DPP. Suppose the individual features are ℓ_2 normalized, i.e. $||x_i||_2 \leq 1$. I modify the ℓ_1-norm in Definitions 2 and 4 into ℓ_2-norm and draw noise Y from Gaussian distribution $\mathscr{N}(0, \sigma^2 I_d)$ with $\sigma \geq \frac{\sqrt{2\ln(1.25/\delta)}\Delta g}{\varepsilon}$, which naturally converts Definition 3 to be the approximate DP.

From the gradient function in Eq. (3.8), if $y = 1, D_W < m$ I have

$$||g_r(\cdot)||_2 = |1 - \frac{m}{||W\Delta x||_2}| \cdot ||W_r \Delta x \Delta x^T||_2 \leq 2m. \tag{A.4}$$

Consequently, I have the following Corollary to calculate the sensitivity for the approximate DPP by extending Theorem 2 (also refer to the right of Fig. 3.6).

Corollary 2 *If the objective function in Eq.* (3.7) *is h-Lipschitz (ℓ_2 norm) w.r.t W_r, the ℓ_2 gradient sensitivity Δg_r on any batch $\mathscr{B}$ is at most $\frac{\kappa(g'_r+g''_r)}{|\mathscr{B}|}$, where the batch gradient peak $g'_r = \max(||g_r(p_1)||_2, ..., ||g_r(p_{|\mathscr{B}|})||_2)$, and its possible counterpart $g''_r = \min\{h, \max(4||W_r||_2, 2m)\}$.*

The benefit of concerning approximate DPP is that the total privacy cost can be reduced by slightly increasing the failure of probability of δ. Since the proposed DPP definition is consistent with DP over structure, the advanced composition theory in [Dwork et al., 2014] is naturally inherited in my work. Furthermore, I refer readers to [Jayaraman and Evans, 2019] for a comparison of more DP variants which provide improved composition theories.

Appendix B

Appendix

B.1 Distillation Comparison

For the binary class case, softmax activation is known to degenerate into the logistic function. If $z = (u, 0)$, then $\text{softmax}_1(z) = \sigma(u) := (1 + \exp(-u))^{-1}$. Thus, given $\text{softmax}_1(z) = s$, I have $u = \ln \frac{s}{1-s}$. Meanwhile, according to the solution of sparsemax, for binary class case, it could be expressed as

$$s' := \text{sparsemax}_1(u) = \begin{cases} 1, & u > 1 \\ (u+1)/2, & -1 \leq u \leq 1 \\ 0, & u < -1. \end{cases} \tag{B.1}$$

Now I can formulate sparsemax output s' as a function of s

$$s'(s) = \begin{cases} 1, & s > \frac{e}{e+1} \\ (\ln \frac{s}{1-s} + 1)/2, & \frac{1}{e+1} \leq s \leq \frac{e}{e+1} \\ 0, & s < \frac{1}{e+1}, \end{cases} \tag{B.2}$$

where e is Euler's number. Note that Eq. (B.2) is a binary case for Theorem 3.

Considering the distillation loss with the power $r = 2$, I have the target probability

$$t = \begin{cases} 1, & s > \frac{e}{e+1} \\ t^\star, & \frac{1}{e+1} \leq s \leq \frac{e}{e+1} \\ 0, & s < \frac{1}{e+1}, \end{cases} \tag{B.3}$$

where $t^\star$ is calculated as follows.

$$\begin{aligned} t^\star &= \frac{(\ln\frac{s}{1-s}+1)^2/4}{(\ln\frac{s}{1-s}+1)^2/4+(1-(\ln\frac{s}{1-s}+1)/2)^2} \\ &= \frac{1}{1+(\frac{\ln\frac{e(1-s)}{s}}{\ln\frac{es}{1-s}})^2}. \end{aligned} \tag{B.4}$$

It is easily verified that $t(s)$ is an odd function and $t^\star|_{s=0.5} = 0.5$. Based on Eq. (B.3), it is verified that the distillation loss (Eq. (4.10)) is nonzero iff $\frac{1}{e+1} \leq s \leq \frac{e}{e+1}$,

$$\mathscr{J}_{\mathrm{D}}(s) = t\ln t + (1-t)\ln(1-t)\underbrace{-t\ln s' - (1-t)\ln(1-s')}_{\text{reduced form}}. \tag{B.5}$$

As t will stop the gradients from propagation, let $s^\star := (\ln\frac{s}{1-s}+1)/2$, and only the reduced form has the gradient

$$\begin{aligned} \nabla_s \mathscr{J}_{\mathrm{D}} &= (-\frac{t}{s'}+\frac{1-t}{1-s'})\nabla_s s' \\ &= \frac{1}{2}(-\frac{t^\star}{s^\star}+\frac{1-t^\star}{1-s^\star})(\frac{1}{s}+\frac{1}{1-s}). \end{aligned} \tag{B.6}$$

The above derivatives are used to complete the plot of ADS in Figure 4.3, and other distillation methods are displayed in a similar manner.

B.2 Proof for Corollary 1

Corollary 1. *For a K-way semi-supervised classification problem, the determinate predictions and negligible predictions for* ADS *are masked out by the sample dependent threshold* $\theta_1 \in [\frac{e}{e+K-1}, \frac{e}{e+1}]$ *and* $\theta_2 \in [\frac{e^\rho}{\rho+e^\rho(K-\rho)}, \frac{e^\rho}{\rho+e^\rho}]$ *in the corresponding softmax output space, respectively, where e is Euler number and* ρ *is the population of non-zero predictions.*

Proof: A prediction p is said determinate for ADS if $p = p_{(1)}$ and $p_{(1)} \geq e p_{(2)}$. Obviously, for a K-way classification the maximum θ_1 is picked if $p_{(3)} = p_{(4)} = ... = p_{(K)} = 0$. Combining with the equality $\sum_k p_{(k)} = 1$, we have $\arg\max \theta_1 = \frac{e}{e+1}$. Meanwhile, the minimum θ_1 is picked if $p_{(2)} = p_{(3)} = ... = p_{(K)}$. Thus we have $\arg\min \theta_1 = \frac{e}{e+K-1}$.

Let ρ denote the population of non-zero outputs given a logits vector z. According to the solution of sparsemax, if $\rho < K$, I have the inequality

$$1 + (\rho + 1)z_{(\rho+1)} \leq \sum_{j=1}^{\rho+1} z_{(j)}. \tag{B.7}$$

Following the derivation of Eq. (4.16), Eq. (B.7) can be rewritten as

$$\begin{aligned} 1 + (\rho + 1)\ln C p_{(\rho+1)} &\leq \sum_{j=1}^{\rho+1} \ln C p_{(j)} \\ \Rightarrow p_{(\rho+1)} &\leq (\frac{1}{e}\prod_{j=1}^{\rho} p_{(j)})^{\frac{1}{\rho}}. \end{aligned} \tag{B.8}$$

Similar to the proof to θ_1, the maximum of θ_2 is derived if $p_{(\rho+2)} = ... = p_{(K)} = 0$, and I have

$$p_{(\rho+1)} = \frac{e^{\rho}(1 - p_{(\rho+1)})}{\rho}, \tag{B.9}$$

which suggests $\arg\max \theta_2 = \frac{e^{\rho}}{\rho + e^{\rho}}$. The minimum of θ_2 is obtained if $p_{(\rho+1)} = p_{(\rho+2)} = ... = p_{(K)}$. That is

$$p_{(\rho+1)} = \frac{e^{\rho}(1 - (K - \rho)p_{(\rho+1)})}{\rho}, \tag{B.10}$$

which suggests $\arg\max \theta_2 = \frac{e^{\rho}}{\rho + e^{\rho}(K-\rho)}$.

The above bounds complete the proof to the range of threshold for θ_1 and θ_2 in ADS in terms of softmax output space. ■

B.3 Example of Calibration Evaluation

Leaving a few of hard unlabelled training samples underfitted may incur a concern about the calibration performance. I use the Expected Calibration Error (ECE) metric [Naeini et al., 2015] to evaluate VAT+ADS on MNIST as an example. In a similar manner to [Guo et al., 2017], I collect the test performance and calculate the samples proportion and accuracy over evenly distributed bins partitioned by prediction confidences (i.e., dominant probabilities).

Figure B.1 shows the experimental results, from which we obtain that: (1) The majority of test samples are predicted with a very high confidence, and some low confidence bins are even empty. This implies that the existence of some underfitted unlabelled samples will not lead the low confidence on test data prediction. (2) The average confidence is slightly greater than the accuracy and the output confidence is only a bit lower than the ideal case.

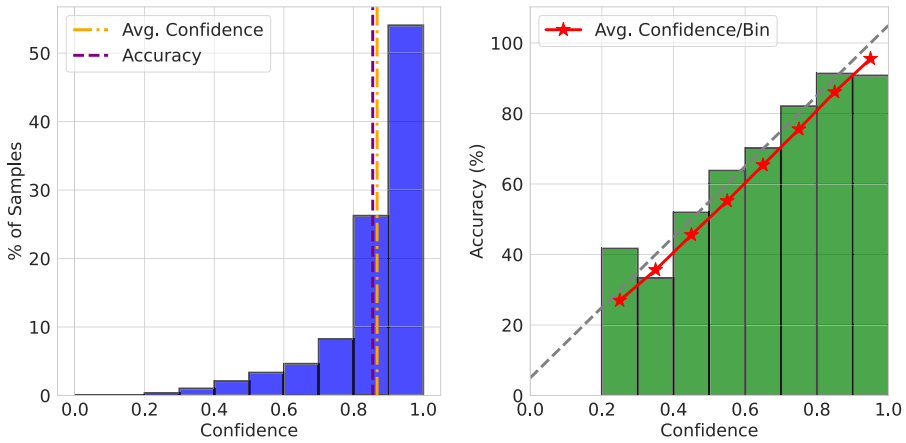


Figure B.1 Calibration performance on the test data of MNIST. Note that the presented confidence on test data is from the softmax output to meet the definition of calibration. To this end, I simply replace sparsemax with softmax during inference, which will not influence the accuracy results. The dashed grey line in the right subfigure denotes the ideal average confidence over bins, and it is shifted leftward by a half of bin-width to visually align with the output confidence, i.e., red stars.

(3) Quantitatively, I calculate ECE=5.2% which is not a big value for calibration error. Hence, it is roughly said the model with ADS is well-calibrated even without any help of other model calibration techniques.

Appendix C

Appendix

C.1 Proof of Theorem 5

Theorem 5. *Given a deep model whose tuned parameters are $\theta = \{\ell_1, \ell_2, ..., \ell_H\}$, for any $\beta > 0$,*

$$G_{\max} - \mathbb{E}[G_{LCPS}] \leq (\beta c(e-2)+1)\, G_{\max} + \frac{c}{\beta} \ln H,$$

holds for any $T > 0$, where $c = \frac{b-Hu}{u}$, (b is the batch size, u is the unit size), and e is Euler's number.

Proof: Let $I^{t+1} \in \mathbb{R}^H$ denote a row vector whose h-th entry is I_h^{t+1}, and similar to p^{t+1} and p^t. Then I have

$$\begin{aligned}
&(I^{t+1})^T (p^{t+1} - p^t) \\
\overset{①}{=}&(I^{t+1})^T \left(\text{softmax}(\alpha^t + \beta I^{t+1}) - \text{softmax}(\alpha^t)\right) \\
\overset{②}{\geq}&\beta (I^{t+1})^T \cdot \nabla_{\alpha^t} \text{softmax}(\alpha^t) \cdot I^{t+1} \\
\overset{③}{\geq}&0,
\end{aligned} \tag{C.1}$$

where ① follows Eq. (5.11), ② keeps only the first-order Taylor expansion, and ③ uses the fact that $\nabla_{\alpha^t} \text{softmax}(\alpha^t)$ is positive semi-definite [Gao and Pavel, 2017]. By rewriting Eq. (C.1) into element-wise multiplication, I have the following inequality,

$$\sum_{h=1}^{H} p_h^t I_h^{t+1} \leq \sum_{h=1}^{H} p_h^{t+1} I_h^{t+1}. \tag{C.2}$$

Suppose normalized average improvement $I_h^t \in [0,1]$ is offered, then I have

$$\sum_{h=1}^{H} p_h^t (I_h^{t+1})^2 \leq \sum_{h=1}^{H} p_h^t I_h^{t+1}. \tag{C.3}$$

Let $W^t = \exp(\alpha_1^t) + ... + \exp(\alpha_H^t)$. By conducting LCPS, I have

$$\begin{aligned}
\frac{W^{t+1}}{W^t} &= \sum_{h=1}^{H} \frac{\exp(\alpha_h^{t+1})}{W^t} \\
&\overset{④}{=} \sum_{h=1}^{H} \frac{\exp(\alpha_h^t) \cdot \exp(\beta I_h^{t+1})}{W^t} \\
&\overset{⑤}{=} \sum_{h=1}^{H} p_h^t \cdot \exp(\beta I_h^{t+1}) \\
&\overset{⑥}{\leq} \sum_{h=1}^{H} p_h^t [1 + \beta I_h^{t+1} + (e-2)(\beta I_h^{t+1})^2] \\
&\leq 1 + \beta \sum_{h=1}^{H} p_h^t I_h^{t+1} + (e-2)\beta^2 \sum_{h=1}^{H} p_h^t (I_h^{t+1})^2 \\
&\overset{⑦}{\leq} 1 + \beta \sum_{h=1}^{H} p_h^{t+1} I_h^{t+1} + (e-2)\beta^2 \sum_{h=1}^{H} p_h^t I_h^{t+1}
\end{aligned} \tag{C.4}$$

where ④ follows Eq. (5.12), ⑤ uses the definition of p_h from Eq. (5.11), ⑥ is derived from the inequality of $e^x \leq 1 + x + (e-2)x^2$, and ⑦ uses the Eqs. (C.2) and (C.3). Taking logarithms and using $1 + x \leq e^x$ comes

$$\ln \frac{W^{t+1}}{W^t} \leq \beta \sum_{h=1}^{H} p_h^{t+1} I_h^{t+1} + (e-2)\beta^2 \sum_{h=1}^{H} p_h^t I_h^{t+1}. \tag{C.5}$$

Summing over t of Eq. (C.5), I have

$$\ln \frac{W^T}{W^0} \leq \beta \sum_{t=1}^{T} \sum_{h=1}^{H} p_h^t I_h^t + (e-2)\beta^2 \sum_{t=1}^{T} \sum_{h=1}^{H} p_h^{t-1} I_h^t. \tag{C.6}$$

Let $c = \frac{b-Hu}{u}$. By assuming that the average improvement I_h^t stays constant in each batch optimization, I can obtain

$$\mathbb{E}[G_{\text{LCPS}}] = \sum_{t=1}^{T} \left(\sum_{h=1}^{H} I_h^t + c \sum_{h=1}^{H} p_h^t I_h^t \right). \tag{C.7}$$

Taking Eq. (C.7) into Eq. (C.6), I can get

$$\ln \frac{W^T}{W^0} \leq \frac{\beta}{c} \left(\mathbb{E}[G_{\text{LCPS}}] - \sum_{t=1}^{T} \sum_{h=1}^{H} I_h^t \right) + (e-2)\beta^2 \sum_{t=1}^{T} \sum_{h=1}^{H} p_h^{t-1} I_h^t. \tag{C.8}$$

For any layer j is selected,

$$\ln \frac{W^T}{W^0} \geq \ln \frac{\exp(\alpha_j^T)}{W^0} = \beta \sum_{t=1}^{T} I_j^t - \ln H. \tag{C.9}$$

Combining Eqs. (C.8) and (C.9) obtains

$$\mathbb{E}[G_{\text{LCPS}}] \geq \sum_{t=1}^{T} \sum_{h=1}^{H} I_h^t + c \sum_{t=1}^{T} I_j^t - \frac{c}{\beta} \ln H - \beta c(e-2) \sum_{t=1}^{T} \sum_{h=1}^{H} p_h^{t-1} I_h^t. \tag{C.10}$$

In addition, it is verified that

$$\sum_{t=1}^{T} \sum_{h=1}^{H} p_h^{t-1} I_h^t \leq \sum_{t=1}^{T} \max_h I_h^t \leq G_{\max}. \tag{C.11}$$

Combining Eqs. (C.11) and (C.10) lets me obtain the inequality of the Theorem 5. ■

C.2 Algorithm for Fairness Learning

In the fair classification task, an initially provided model is tuned by relying on both the model accuracy and fairness. Demographic parity is used as the fairness metric in this experiment. By employing an extra weight factor ρ to balance two measurements, I update the model by Alg. 4. $\rho = 0.4$ is used for the reported experimental results in Fig. 5.7.

Algorithm 4 Performance-guided Parameter Search (PPS) for Fair Classification

Require: Initially provided model F_{θ_0}, query budget Q, learning rate η, batch size b, variance σ^2, weight factor ρ

for $t = 0, ..., \lfloor Q/b \rfloor$ **do**

Sample $\{\varepsilon_j\}_{j=1}^{b/2} \sim \mathcal{N}(0, I)$, and for each j get $\varepsilon_{b-j+1} = -\varepsilon_j$.

Generate candidate models $\{\theta_i\}_i^b$ as queries where $\theta_i = \theta^t + \sigma\varepsilon_i$.

Collect and normalize $\{E(\mathscr{D}; \theta_i), \Gamma(\mathscr{D}; \theta_i)\}_{i=1}^b$.

$\theta^{t+1} \leftarrow \theta^t - \frac{\eta}{\sigma b} \sum_{i=1}^b \varepsilon_i [\rho E(\mathscr{D}; \theta^t + \sigma\delta_i) - \Gamma(\mathscr{D}; \theta^t + \sigma\delta_i)]$

end for

Ensure: $\theta^{\lfloor Q/b \rfloor + 1}$

C.3 Discussion of Private Tuning Application

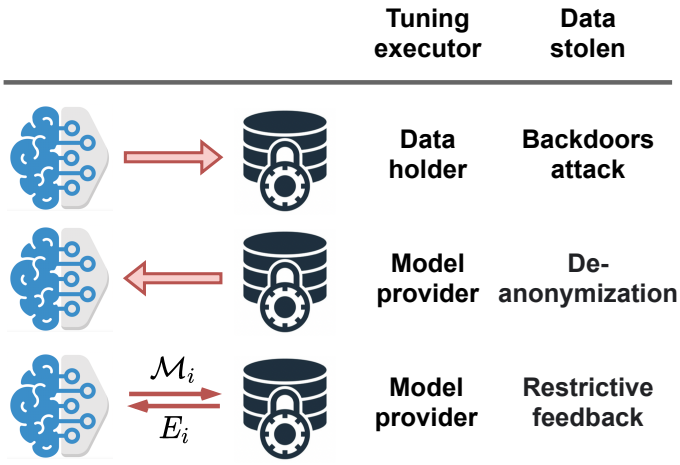


Figure C.1 Comparison among three forms of model tuning.

Preventing the adversarial inference about tuning data during the model tuning is of significance when target data involves sensitive information. I present two instances shown as the first two rows in Fig. C.1 which illustrate how existing works conduct model tuning with the data privacy concern.

One-way data holder tuning. Given a pre-trained model (e.g. downloaded from the internet under some agreed licenses), a data holder can execute model tuning on the local private data in an unobserved manner. Once the model is accessible in a white-box or black-box way after tuning, this one-way data holder tuning is at the risk of data leakage if the model provider provide a back-door model to data holder [Song et al., 2017a]. For example, private data could be encoded in the least significant (lower) bits of the deep model's parameters (white-box) or the label vector of augmented data (black-box) to intentionally extract private data. Please also note in this case, the original source model should be stored on local device and the data holder is assumed to be capable to model tuning.

One-way model provider tuning. Alternatively, a data holder can pay experienced model providers to do tuning on the model providers' side. In this case, to preserve data privacy, the main attention of the data holder is on how to "reedit" private information before sending them to a model provider. Anonymization [Zhou et al., 2008] seems a workaround to this problem, but it is quite limited to tabular data, and it has been demonstrated weak by de-anonimization [Porter, 2008]. Other techniques like local differential privacy [Cormode et al., 2018] by randomizing raw feature is of low utility for real world applications.

Essentially, the above two instances do not change the tuning process itself; just like training, both of them straightforwardly feed the (original/edited) target data to the model

for update. I realize the common root of data leakage for two instances is that they allow the model and data to stay on the same side, which serves as the base of the standard model tuning. As the goal of model tuning is to deliver a satisfactory model for data holders, the introduced EXPECTED is a solution for this challenge.

Two-way EXPECTED. EXPECTED keeps data and model staying on their sides. Without any demands to data holders' ability on model tuning or data edition like previous two instances, EXPECTED only requires data holders to do the model evaluation on private data and return the performance to the model provider. Within limited queries, the model provider is expected to craft a satisfactory model for data holder. The key here is that model providers only receive restrictive feedbacks which will not expose much information about the private data. For example, in case data information is encoded via feedback scores, we have shown in Section 5.4.5 that it will not be very risky as 2 decimals precision might be sufficient to use in EXPECTED.

References

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318. ACM, 2016.

Abbas Acar, Z Berkay Celik, Hidayet Aksu, A Selcuk Uluagac, and Patrick McDaniel. Achieving secure and differentially private computations in multiparty settings. In *2017 IEEE Symposium on Privacy-Aware Computing (PAC)*, pages 49–59. IEEE, 2017.

Tameem Adel, Isabel Valera, Zoubin Ghahramani, and Adrian Weller. One-network adversarial fairness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 2412–2420, 2019.

Eric Arazo, Diego Ortego, Paul Albert, Noel E O'Connor, and Kevin McGuinness. Pseudo-labeling and confirmation bias in deep semi-supervised learning. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

Raef Bassily, Adam Smith, and Abhradeep Thakurta. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 464–473. IEEE, 2014.

Eugene Belilovsky, Michael Eickenberg, and Edouard Oyallon. Greedy layerwise learning can scale to imagenet. In *International conference on machine learning*, pages 583–593. PMLR, 2019.

Aurélien Bellet, Amaury Habrard, and Marc Sebban. A survey on metric learning for feature vectors and structured data. *arXiv preprint arXiv:1306.6709*, 2013.

James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2), 2012.

David Berthelot, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. *arXiv preprint arXiv:1911.09785*, 2019a.

David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems*, pages 5049–5059, 2019b.

Mathieu Blondel, André F. T. Martins, and Vlad Niculae. Learning with fenchel-young losses. *J. Mach. Learn. Res.*, 21:35:1–35:69, 2020.

Philip Buczak and Daniel Horn. Using sequential statistical tests to improve the performance of random search in hyperparameter tuning. *arXiv preprint arXiv:2112.12438*, 2021.

Rich Caruana, Steve Lawrence, and C Giles. Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. *Advances in neural information processing systems*, 13, 2000.

Paola Cascante-Bonilla, Fuwen Tan, Yanjun Qi, and Vicente Ordonez. Curriculum labeling: Self-paced pseudo-labeling for semi-supervised learning. *arXiv preprint arXiv:2001.06001*, 2020.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*, 2017.

Olivier Chapelle and Bernhard Scholkopf. Semi-supervised learning. *MIT Press*, 2006.

Kamalika Chaudhuri and Claire Monteleoni. Privacy-preserving logistic regression. In *Advances in neural information processing systems*, pages 289–296, 2009.

Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(Mar):1069–1109, 2011.

Can Chen, Shuhao Zheng, Xi Chen, Erqun Dong, Xue Steve Liu, Hao Liu, and Dejing Dou. Generalized dataweighting via class-level gradient manipulation. *Advances in Neural Information Processing Systems*, 34:14097–14109, 2021.

Cen Chen, Bingzhe Wu, Minghui Qiu, Li Wang, and Jun Zhou. A comprehensive analysis of information leakage in deep transfer learning. *arXiv preprint arXiv:2009.01989*, 2020a.

John Chen, Vatsal Shah, and Anastasios Kyrillidis. Negative sampling in semi-supervised learning. In *International Conference on Machine Learning*, pages 1704–1714. PMLR, 2020b.

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big self-supervised models are strong semi-supervised learners. *arXiv preprint arXiv:2006.10029*, 2020c.

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E. Hinton. Big self-supervised models are strong semi-supervised learners. In *Annual Conference on Neural Information Processing Systems*, 2020d.

Xi Chen, Paul N Bennett, Kevyn Collins-Thompson, and Eric Horvitz. Pairwise ranking aggregation in a crowdsourced setting. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 193–202, 2013.

Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 539–546. IEEE, 2005.

Alexandra Chouldechova. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data*, 5(2):153–163, 2017.

Evgenii Chzhen, Christophe Denis, Mohamed Hebiri, and Titouan Lorieul. Set-valued classification–overview via a unified framework. *arXiv preprint arXiv:2102.12318*, 2021.

Graham Cormode, Somesh Jha, Tejas Kulkarni, Ninghui Li, Divesh Srivastava, and Tianhao Wang. Privacy at scale: Local differential privacy in practice. In *Proceedings of the 2018 International Conference on Management of Data*, pages 1655–1658, 2018.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 113–123, 2019.

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*, pages 248–255. Ieee, 2009.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019.

Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Uwe Dick, Peter Haider, and Tobias Scheffer. Learning from incomplete data with infinite imputations. In *Proceedings of the 25th international conference on Machine learning*, pages 232–239, 2008.

Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655. PMLR, 2014.

Dheeru Dua and Efi Karra Taniskidou. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

John Duchi, Shai Shalev-Shwartz, Yoram Singer, and Tushar Chandra. Efficient projections onto the l 1-ball for learning in high dimensions. In *Proceedings of the 25th international conference on Machine learning*, pages 272–279, 2008.

John C Duchi, Michael I Jordan, and Martin J Wainwright. Local privacy, data processing inequalities, and statistical minimax rates. *arXiv preprint arXiv:1302.3203*, 2013.

John C Duchi, Michael I Jordan, and Martin J Wainwright. Minimax optimal procedures for locally private estimation. *Journal of the American Statistical Association*, 113(521): 182–201, 2018.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

Michael TM Emmerich and André H Deutz. A tutorial on multiobjective optimization: fundamentals and evolutionary methods. *Natural computing*, 17(3):585–609, 2018.

Li Fei-Fei, Robert Fergus, and Pietro Perona. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):594–611, 2006.

Martínez-Plumed Fernando, Ferri Cèsar, Nieves David, and Hernández-Orallo José. Missing the missing values: The ugly duckling of fairness in machine learning. *International Journal of Intelligent Systems*, 36(7):3217–3258, 2021.

Kazuto Fukuchi, Quang Khai Tran, and Jun Sakuma. Differentially private empirical risk minimization with input perturbation. In *International Conference on Discovery Science*, pages 82–90. Springer, 2017.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR, 2015.

Bolin Gao and Lacra Pavel. On the properties of the softmax function with application in game theory and reinforcement learning. *arXiv preprint arXiv:1704.00805*, 2017.

John Geweke. Antithetic acceleration of monte carlo integration in bayesian inference. *Journal of Econometrics*, 38(1-2):73–89, 1988.

Zoubin Ghahramani and Michael Jordan. Supervised learning from incomplete data via an em approach. *Advances in neural information processing systems*, 6, 1993.

Alexander N Gorban, Ivan Yu Tyukin, Danil V Prokhorov, and Konstantin I Sofeikov. Approximation with random bases: Pro et contra. *Information Sciences*, 364:129–145, 2016.

Frank Göring. Short proof of menger's theorem. *Discrete Mathematics*, 219(1-3):295–296, 2000.

Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *Advances in neural information processing systems*, pages 529–536, 2005.

Matthieu Guillaumin, Jakob Verbeek, and Cordelia Schmid. Is that you? metric learning approaches for face identification. In *2009 IEEE 12th international conference on computer vision*, pages 498–505. IEEE, 2009.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR, 2017.

Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

Bo Han, Yuangang Pan, and Ivor W Tsang. Robust plackett–luce model for k-ary crowdsourced preferences. *Machine Learning*, 107(4):675–702, 2018.

Tao Han, Junyu Gao, Yuan Yuan, and Qi Wang. Unsupervised semantic aggregation and deformable template matching for semi-supervised learning. *Advances in Neural Information Processing Systems*, 33, 2020.

Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29, 2016.

Tatsunori Hashimoto, Megha Srivastava, Hongseok Namkoong, and Percy Liang. Fairness without demographics in repeated loss minimization. In *International Conference on Machine Learning*, pages 1929–1938. PMLR, 2018.

Michael Hay, Chao Li, Gerome Miklau, and David Jensen. Accurate estimation of the degree distribution of private networks. In *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*, pages 169–178. IEEE, 2009.

Michael Hay, Liudmila Elagina, and Gerome Miklau. Differentially private rank aggregation. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 669–677. SIAM, 2017.

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.

Jeff Heaton. Ian goodfellow, yoshua bengio, and aaron courville: Deep learning, 2018.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. Pretrained transformers improve out-of-distribution robustness. *arXiv preprint arXiv:2004.06100*, 2020.

Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.

Junlin Hu, Jiwen Lu, and Yap-Peng Tan. Discriminative deep metric learning for face verification in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1875–1882, 2014.

Mengdi Huai, Chenglin Miao, Qiuling Suo, Yaliang Li, Jing Gao, and Aidong Zhang. Uncorrelated patient similarity learning. In *Proceedings of the 2018 SIAM International Conference on Data Mining*, pages 270–278. SIAM, 2018.

Mengdi Huai, Di Wang, Chenglin Miao, Jinhui Xu, and Aidong Zhang. Pairwise learning with differential privacy guarantees. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 694–701, 2020.

Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In *International Conference on Machine Learning*, pages 2137–2146. PMLR, 2018.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

Niels Ipsen, Pierre-Alexandre Mattei, and Jes Frellsen. How to deal with missing data in supervised deep learning? In *Artemiss-ICML Workshop on the Art of Learning with Missing Values*, 2020.

Janus Christian Jakobsen, Christian Gluud, Jørn Wetterslev, and Per Winkel. When and how should multiple imputation be used for handling missing data in randomised clinical trials–a practical guide with flowcharts. *BMC medical research methodology*, 17(1):1–10, 2017.

Bargav Jayaraman and David Evans. Evaluating differentially private machine learning in practice. In *28th {USENIX} Security Symposium ({USENIX} Security 19)*, pages 1895–1912, 2019.

Zhanglong Ji, Zachary C Lipton, and Charles Elkan. Differential privacy and machine learning: a survey and review. *arXiv preprint arXiv:1412.7584*, 2014.

Rong Jin, Shijun Wang, and Yang Zhou. Regularized distance metric learning: Theory and algorithm. In *Advances in neural information processing systems*, pages 862–870, 2009.

Josh Joy and Mario Gerla. Differential privacy by sampling. *arXiv preprint arXiv:1708.01884*, 2017.

Yilin Kang, Yong Liu, Jian Li, and Weiping Wang. Differential privacy for pairwise learning: Non-convex analysis. *arXiv preprint arXiv:2105.03033*, 2021a.

Yilin Kang, Yong Liu, Jian Li, and Weiping Wang. Towards sharper utility bounds for differentially private pairwise learning. *arXiv preprint arXiv:2105.03033*, 2021b.

Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *arXiv preprint arXiv:2006.06676*, 2020.

Vishesh Karwa, Sofya Raskhodnikova, Adam Smith, and Grigory Yaroslavtsev. Private analysis of graph structure. *Proceedings of the VLDB Endowment*, 4(11):1146–1157, 2011.

Shiva Prasad Kasiviswanathan, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Analyzing graphs with node differential privacy. In *Theory of Cryptography Conference*, pages 457–476. Springer, 2013.

Daniel Kifer and Ashwin Machanavajjhala. No free lunch in data privacy. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, pages 193–204. ACM, 2011.

Daniel Kifer and Ashwin Machanavajjhala. A rigorous and customizable framework for privacy. In *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGAI symposium on Principles of Database Systems*, pages 77–88. ACM, 2012.

Daniel Kifer, Adam Smith, and Abhradeep Thakurta. Private convex empirical risk minimization and high-dimensional regression. In *Conference on Learning Theory*, pages 25–1, 2012.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Bart P Knijnenburg, Martijn C Willemsen, Zeno Gantner, Hakan Soncu, and Chris Newell. Explaining the user experience of recommender systems. *User modeling and user-adapted interaction*, 22(4):441–504, 2012.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR, 2017.

Ron Kohavi. Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In *Kdd*, volume 96, pages 202–207, 1996.

Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

Agustinus Kristiadi, Matthias Hein, and Philipp Hennig. Being bayesian, even just a bit, fixes overconfidence in relu networks. In *International Conference on Machine Learning*, pages 5436–5446. PMLR, 2020.

James Kwok and Ivor W. Tsang. Learning with idealized kernels. In *International Conference on Machine Learning*, pages 400–407, 2003.

Condat Laurent. Fast projection onto the simplex and the l1 ball. *Math. Prog*, 158:575–585, 2016.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896, 2013.

Jaewoo Lee and Daniel Kifer. Concentrated differentially private gradient descent with adaptive per-iteration privacy budget. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1656–1665. ACM, 2018.

Kangwook Lee, Hoon Kim, Kyungmin Lee, Changho Suh, and Kannan Ramchandran. Synthesizing differentially private datasets using random mixing. In *2019 IEEE International Symposium on Information Theory (ISIT)*, pages 542–546. IEEE, 2019.

Rui Li, Qianfen Jiao, Wenming Cao, Hau-San Wong, and Si Wu. Model adaptation: Unsupervised domain adaptation without source data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9641–9650, 2020a.

Wenbin Li, Jing Huo, Yinghuan Shi, Yang Gao, Lei Wang, and Jiebo Luo. Online deep metric learning. *arXiv preprint arXiv:1805.05510*, 2018.

Yandong Li, Lijun Li, Liqiang Wang, Tong Zhang, and Boqing Gong. Nattack: Learning the distributions of adversarial examples for an improved black-box attack on deep neural networks. In *International Conference on Machine Learning*, pages 3866–3876. PMLR, 2019.

Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779*, 2016.

Yijun Li, Richard Zhang, Jingwan Lu, and Eli Shechtman. Few-shot image generation with elastic weight consolidation. *arXiv preprint arXiv:2012.02780*, 2020b.

Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *International Conference on Machine Learning*, pages 6028–6039. PMLR, 2020.

Changchang Liu, Supriyo Chakraborty, and Prateek Mittal. Dependence makes you vulnberable: Differential privacy under dependent tuples. In *NDSS*, volume 16, pages 21–24, 2016.

Guangcan Liu, Zhouchen Lin, Shuicheng Yan, Ju Sun, Yong Yu, and Yi Ma. Robust recovery of subspace structures by low-rank representation. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):171–184, 2012.

Suyun Liu and Luis Nunes Vicente. Accuracy and fairness trade-offs in machine learning: A stochastic multi-objective approach. *Computational Management Science*, pages 1–25, 2022.

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.

Zhengdong Lu. Semi-supervised clustering with pairwise constraints: A discriminative approach. In *Artificial Intelligence and Statistics*, pages 299–306. PMLR, 2007.

Andre Martins and Ramon Astudillo. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International Conference on Machine Learning*, pages 1614–1623, 2016.

Jirı Matoušek. Lecture notes on metric embeddings. Technical report, Technical report, ETH Zürich, 2013.

Pratik Mazumder, Pravendra Singh, and Mohammed Asad Karim. Restricted category removal from model representations using limited data. 2021.

Julian McAuley and Jure Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 165–172, 2013.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

Frank D McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, pages 19–30. ACM, 2009.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, Ken Nakae, and Shin Ishii. Distributional smoothing with virtual adversarial training. *arXiv preprint arXiv:1507.00677*, 2015.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993, 2018.

Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. Monte carlo gradient estimation in machine learning. *J. Mach. Learn. Res.*, 21(132):1–62, 2020.

Daniel Moyer, Shuyang Gao, Rob Brekelmans, Aram Galstyan, and Greg Ver Steeg. Invariant representations without adversarial training. *Advances in Neural Information Processing Systems*, 31, 2018.

Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large datasets (how to break anonymity of the netflix prize dataset). *University of Texas at Austin*, 2008.

Bac Nguyen, Carlos Morell, and Bernard De Baets. Supervised distance metric learning through maximization of the jeffrey divergence. *Pattern Recognition*, 64:215–225, 2017.

Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 75–84. ACM, 2007.

Gang Niu, Bo Dai, Makoto Yamada, and Masashi Sugiyama. Information-theoretic semi-supervised metric learning via entropy regularization. *Neural computation*, 26(8):1717–1762, 2014.

Avital Oliver, Augustus Odena, Colin A Raffel, Ekin Dogus Cubuk, and Ian Goodfellow. Realistic evaluation of deep semi-supervised learning algorithms. In *Advances in neural information processing systems*, pages 3235–3246, 2018.

Karol R Opara and Jarosław Arabas. Differential evolution: A survey of theoretical analyses. *Swarm and evolutionary computation*, 44:546–558, 2019.

Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*, 2017.

Vadim Popov, Mikhail Kudinov, Irina Piontkovskaya, Petr Vytovtov, and Alex Nevidomsky. Distributed fine-tuning of language models on private data. In *International Conference on Learning Representations*, 2018.

C Christine Porter. De-identified data and third party data mining: the risk of re-identification of personal information. *Shidler JL Com. & Tech.*, 5:1, 2008.

Zhan Qin, Ting Yu, Yin Yang, Issa Khalil, Xiaokui Xiao, and Kui Ren. Generating synthetic decentralized social graphs with local differential privacy. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 425–438, 2017.

Joaquin Quiñonero-Candela, Masashi Sugiyama, Neil D Lawrence, and Anton Schwaighofer. *Dataset shift in machine learning*. Mit Press, 2009.

Ilija Radosavovic, Justin Johnson, Saining Xie, Wan-Yen Lo, and Piotr Dollár. On network design spaces for visual recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1882–1890, 2019.

Parisa Rashidi and Diane J Cook. Keeping the resident in the loop: Adapting the smart home to the user. *IEEE Transactions on systems, man, and cybernetics-part A: systems and humans*, 39(5):949–959, 2009.

Vibhor Rastogi, Michael Hay, Gerome Miklau, and Dan Suciu. Relationship privacy: output perturbation for queries with joins. In *Proceedings of the twenty-eighth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 107–116, 2009.

Payam Refaeilzadeh, Lei Tang, and Huan Liu. Cross-validation. *Encyclopedia of database systems*, 5:532–538, 2009.

Zhongzheng Ren, Raymond Yeh, and Alexander Schwing. Not all unlabeled data are equal: learning to weight data in semi-supervised learning. *Advances in Neural Information Processing Systems*, 33, 2020.

Mamshad Nayeem Rizve, Kevin Duarte, Yogesh S Rawat, and Mubarak Shah. In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning. *arXiv preprint arXiv:2101.06329*, 2021.

Steven Ruggles, Katie Genadek, Ronald Goeken, Josiah Grover, and Matthew Sobek. Integrated public use microdata series, minnesota population center, 2018. URL http://international.ipums.org.

Roshni Sahoo, Divya Shanmugam, and John Guttag. Unsupervised domain adaptation in the absence of source data. *arXiv preprint arXiv:2007.10233*, 2020.

Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. Improving robustness against common corruptions by covariate shift adaptation. *Advances in Neural Information Processing Systems*, 33:11539–11551, 2020.

Yevgeny Seldin, Csaba Szepesvári, Peter Auer, and Yasin Abbasi-Yadkori. Evaluation and analysis of the performance of the exp3 algorithm in stochastic environments. In *European Workshop on Reinforcement Learning*, pages 103–116. PMLR, 2013.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE, 2017.

Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in Neural Information Processing Systems*, pages 1857–1865, 2016.

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. *arXiv preprint arXiv:2001.07685*, 2020.

Congzheng Song, Thomas Ristenpart, and Vitaly Shmatikov. Machine learning models that remember too much. In *Proceedings of the 2017 ACM SIGSAC Conference on computer and communications security*, pages 587–601, 2017a.

Shuang Song, Kamalika Chaudhuri, and Anand D Sarwate. Stochastic gradient descent with differentially private updates. In *2013 IEEE Global Conference on Signal and Information Processing*, pages 245–248. IEEE, 2013.

Shuang Song, Yizhen Wang, and Kamalika Chaudhuri. Pufferfish privacy mechanisms for correlated data. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 1291–1306. ACM, 2017b.

Jordi Soria-Comas and Josep Domingo-Ferrer. Optimal data-independent noise for differential privacy. *Information Sciences*, 250:200–214, 2013.

Haipei Sun, Boxiang Dong, Hui Wendy Wang, Ting Yu, and Zhan Qin. Truth inference on sparse crowdsourcing data with local differential privacy. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 488–497. IEEE, 2018.

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International Conference on Machine Learning*, pages 9229–9248. PMLR, 2020.

Qiuling Suo, Weida Zhong, Fenglong Ma, Ye Yuan, Jing Gao, and Aidong Zhang. Metric learning on healthcare data with incomplete modalities. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 3534–3540. AAAI Press, 2019.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

Latanya Sweeney. Matching known patients to health records in washington state data. 2013.

Nima Tajbakhsh, Jae Y Shin, Suryakanth R Gurudu, R Todd Hurst, Christopher B Kendall, Michael B Gotway, and Jianming Liang. Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE transactions on medical imaging*, 35(5): 1299–1312, 2016.

Kunal Talwar, Abhradeep Thakurta, and Li Zhang. Private empirical risk minimization beyond the worst case: The effect of the constraint set geometry. *arXiv preprint arXiv:1411.5417*, 2014.

Kunal Talwar, Abhradeep Guha Thakurta, and Li Zhang. Nearly optimal private lasso. In *Advances in Neural Information Processing Systems*, pages 3025–3033, 2015.

Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in neural information processing systems*, pages 1195–1204, 2017.

Ivor W. Tsang and James Kwok. Distance metric learning with kernels. In *International Conference on Artificial Neural Networks*, pages 126–129, 2003.

Michael Carl Tschantz, Shayak Sen, and Anupam Datta. Sok: Differential privacy as a causal property. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 354–371. IEEE, 2020.

Ryan Turner, David Eriksson, Michael McCourt, Juha Kiili, Eero Laaksonen, Zhen Xu, and Isabelle Guyon. Bayesian optimization is superior to random search for machine learning hyperparameter tuning: Analysis of the black-box optimization challenge 2020. *arXiv preprint arXiv:2104.10201*, 2021.

Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. Simultaneous deep transfer across domains and tasks. In *Proceedings of the IEEE international conference on computer vision*, pages 4068–4076, 2015.

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

Vladimir Vapnik. Principles of risk minimization for learning theory. In *Advances in neural information processing systems*, pages 831–838, 1992.

Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 1999.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Fully test-time adaptation by entropy minimization. *arXiv preprint arXiv:2006.10726*, 2020a.

Di Wang, Minwei Ye, and Jinhui Xu. Differentially private empirical risk minimization revisited: Faster and more general. In *Advances in Neural Information Processing Systems*, pages 2722–2731, 2017.

Fei Wang, Jimeng Sun, and Shahram Ebadollahi. Integrating distance metrics learned from multiple experts and its application in patient similarity assessment. In *Proceedings of the 2011 SIAM International Conference on Data Mining*, pages 59–70. SIAM, 2011.

Li Wang, Raymond Chan, and Tieyong Zeng. Probabilistic semi-supervised learning via sparse graph structure learning. *IEEE transactions on neural networks and learning systems*, 2020b.

Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys (csur)*, 53(3):1–34, 2020c.

Stanley L Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.

Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(Feb):207–244, 2009.

Daan Wierstra, Tom Schaul, Tobias Glasmachers, Yi Sun, Jan Peters, and Jürgen Schmidhuber. Natural evolution strategies. *The Journal of Machine Learning Research*, 15(1): 949–980, 2014.

Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. Semi-fairvae: Semi-supervised fair representation learning with adversarial variational autoencoder. *arXiv preprint arXiv:2204.00536*, 2022.

Pengtao Xie, Hongbao Zhang, Yichen Zhu, and Eric P Xing. Nonoverlap-promoting variable selection. In *International Conference on Machine Learning*, pages 5409–5418, 2018.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*, 2019.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised data augmentation for consistency training. *Advances in Neural Information Processing Systems*, 33:6256–6268, 2020.

Eric P Xing, Michael I Jordan, Stuart J Russell, and Andrew Y Ng. Distance metric learning with application to clustering with side-information. In *Advances in neural information processing systems*, pages 521–528, 2003.

Zhiyu Xue, Shaoyang Yang, Mengdi Huai, and Di Wang 0015. Differentially private pairwise learning revisited. In *IJCAI*, pages 3242–3248, 2021.

Zhenhuan Yang, Yunwen Lei, Siwei Lyu, and Yiming Ying. Stability and differential privacy of stochastic gradient descent for pairwise learning with non-smooth loss. In *International Conference on Artificial Intelligence and Statistics*, pages 2026–2034. PMLR, 2021.

David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd annual meeting of the association for computational linguistics*, pages 189–196, 1995.

Hongxu Yin, Arun Mallya, Arash Vahdat, Jose M Alvarez, Jan Kautz, and Pavlo Molchanov. See through gradients: Image batch recovery via gradinversion. *arXiv preprint arXiv:2104.07586*, 2021.

Yiming Ying and Peng Li. Distance metric learning with eigenvalue optimization. *Journal of Machine Learning Research*, 13(Jan):1–26, 2012.

Seunghyun Yoon, Hyeongu Yun, Yuna Kim, Gyu-tae Park, and Kyomin Jung. Efficient transfer learning schemes for personalized language modeling using recurrent neural network. *arXiv preprint arXiv:1701.03578*, 2017.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *arXiv preprint arXiv:1411.1792*, 2014.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Yan Zhai, Lichao Yin, Jeffrey S. Chase, Thomas Ristenpart, and Michael M. Swift. CQSTR: securing cross-tenant applications with cloud containers. In Marcos K. Aguilera, Brian Cooper, and Yanlei Diao, editors, *Proceedings of the Seventh ACM Symposium on Cloud Computing, Santa Clara, CA, USA, October 5-7, 2016*, pages 223–236. ACM, 2016.

Cha Zhang and Yunqian Ma. *Ensemble machine learning: methods and applications*. Springer, 2012.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017a.

Jiaqi Zhang, Kai Zheng, Wenlong Mou, and Liwei Wang. Efficient private erm for smooth objectives. *arXiv preprint arXiv:1703.09947*, 2017b.

Jingzhao Zhang, Tianxing He, Suvrit Sra, and Ali Jadbabaie. Why gradient clipping accelerates training: A theoretical justification for adaptivity. *arXiv preprint arXiv:1905.11881*, 2019.

Jun Zhang, Graham Cormode, Cecilia M Procopiuc, Divesh Srivastava, and Xiaokui Xiao. Private release of graph statistics using ladder functions. In *Proceedings of the 2015 ACM SIGMOD international conference on management of data*, pages 731–745, 2015.

Sheng Zhang, Weihong Wang, James Ford, and Fillia Makedon. Learning from incomplete ratings using non-negative matrix factorization. In *Proceedings of the 2006 SIAM international conference on data mining*, pages 549–553. SIAM, 2006.

Jun Zhao, Junshan Zhang, and H Vincent Poor. Dependent differential privacy for correlated data. In *2017 IEEE Globecom Workshops (GC Wkshps)*, pages 1–7. IEEE, 2017.

Yushan Zhao, Gaoming Yang, Xianjin Fang, and Bin Ge. Preventing privacy disclosure from hostility attack base on associated attributes. In *International Conference on Applications and Techniques in Cyber Security and Intelligence*, pages 1315–1325. Springer, 2019.

Bin Zhou, Jian Pei, and WoShun Luk. A brief survey on anonymization techniques for privacy preserving publishing of social network data. *ACM Sigkdd Explorations Newsletter*, 10(2): 12–22, 2008.

Bing Zhu, Changzheng He, and Panos Liatsis. A robust missing value imputation method for noisy data. *Applied Intelligence*, 36(1):61–74, 2012.

Tianqing Zhu, Ping Xiong, Gang Li, and Wanlei Zhou. Correlated differential privacy: Hiding information in non-iid data set. *IEEE Transactions on Information Forensics and Security*, 10(2):229–242, 2014.

Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*, pages 912–919, 2003.