

UC Santa Cruz

UC Santa Cruz Electronic Theses and Dissertations

Title

Embracing Data-Centric AI: Practical and Provable Solutions to Weakly Supervised Data

Permalink

<https://escholarship.org/uc/item/0sf3f96d>

Author

Zhu, Zhaowei

Publication Date

2023

Copyright Information

This work is made available under the terms of a Creative Commons Attribution-NonCommercial License, available at <https://creativecommons.org/licenses/by-nc/4.0/>

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
SANTA CRUZ

**EMBRACING DATA-CENTRIC AI:
PRACTICAL AND PROVABLE SOLUTIONS TO WEAKLY
SUPERVISED DATA**

A dissertation submitted in partial satisfaction of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE AND ENGINEERING

by

Zhaowei Zhu

September 2023

The Dissertation of Zhaowei Zhu
is approved:

Professor Yang Liu, Chair

Professor Seshadhri Comandur

Professor Leilani Gilpin

Professor Bo An

Professor Tongliang Liu

Aditya Krishna Menon, Ph.D.

Peter Biehl
Vice Provost and Dean of Graduate Studies

Copyright © by

Zhaowei Zhu

2023

Table of Contents

List of Figures	vii
List of Tables	viii
Abstract	ix
Acknowledgments	x
1 Introduction	1
1.1 Related Works	4
1.1.1 Learning the Noise Rate in Labels	4
1.1.2 Detecting the Corrupted Labels	6
1.1.3 Learning with Noisy Labels	8
1.1.4 Beyond Accuracy Concerns	10
1.2 Preliminaries	12
1.2.1 Data Related Notations and Definitions	13
1.2.2 Tasks in Data-Centric AI	16
1.3 Summary of Publications	23
2 Estimate the Noise Transition Matrix T	25
2.1 Existing Learning-Centric Methods Have Limitations	25
2.2 Estimate T with Clusterability	27
2.2.1 Warm-up: A Binary Example	27
2.2.2 Estimating T : The General Form	30
2.2.3 The HOC Estimator	32
2.2.4 Flexible Extensions to Instance-Dependent Noise	36
2.2.5 Theoretical Guarantees	38
2.2.6 Experiments	41
2.2.7 Takeaways	46
2.3 Estimate T for Tasks with Lower-Quality Features	47
2.3.1 Failures on Lower-Quality Features	47
2.3.2 An Information-Theoretic Approach	50

2.3.3	Theoretical Guarantees	58
2.3.4	Evaluations	62
2.3.5	Takeaways	67
3	Label Error Detection	69
3.1	CORES ² : COncidence REgularized Sample Sieve	70
3.1.1	Confidence Regularization	71
3.1.2	Confidence Regularized Sample Sieve	73
3.1.3	Theoretical Guarantees of CORES ²	77
3.1.4	Experiments	82
3.1.5	Takeaways	84
3.2	SimiFeat: Label Error Detection Using Similar Features	85
3.2.1	Corrupted Label Detection Using Similar Features	87
3.2.2	How Does Feature Quality Affect Our Solution?	95
3.2.3	Empirical Results	99
4	Learning After Label Error Detection	107
4.1	A Semi-Supervised Learning Approach	108
4.2	A Second-Order Approach	110
4.2.1	Insufficiency of First-Order Statistics	111
4.2.2	Covariance-Assisted Learning (CAL)	116
4.2.3	CAL with Imperfect Covariance Estimates	124
4.2.4	Experiments	125
4.2.5	Takeaways	129
5	Beyond Accuracy: Fairness Issues	131
5.1	Disparate Impact of SSL	131
5.1.1	Motivating Examples	132
5.1.2	Theoretical Analyses	133
5.1.3	Benefit Ratio: An Evaluation Metric	139
5.1.4	Experiments	141
5.1.5	Takeaways	143
5.2	Estimate Fairness with Missing Sensitive Attributes	144
5.2.1	Proxy Results Can be Misleading	145
5.2.2	Weak Proxies Suffice	150
5.2.3	Takeaways	159
6	Conclusions	160
	Bibliography	162

A	More Details for Charter 2	198
A.1	Derivation of Consensus Equations	199
A.2	Theoretical Guarantees for HOC	202
A.2.1	Uniqueness of \mathbf{T}	202
A.2.2	Feasibility of Assumption $ E_3^* = \Theta(N)$	207
A.2.3	Proof for Lemma 1	208
A.2.4	Proof for Theorem 2	209
A.3	More Discussions for HOC	211
A.3.1	Soft 2-NN Label Clusterability	211
A.3.2	Local $\mathbf{T}(X)$	211
A.3.3	Feasibility of Assumption 1 and Assumption 1	212
A.4	More Detailed Experiment Settings for HOC	213
A.4.1	Generating the Instance-Dependent Label Noise	213
A.4.2	Basic Hyper-Parameters	214
A.4.3	Global and Local Estimation Errors on CIFAR-10 with Human Noise	217
A.5	Theoretical Guarantees for HOC-extension	219
A.5.1	Common f -Divergence Functions	219
A.5.2	Total-Variation	220
A.5.3	KL Divergence	222
A.6	More Discussions for HOC-extension	231
A.6.1	Rationale for building on HOC	231
A.6.2	More Experiments	231
B	More Details for Charter 3	234
B.1	CORES ² : Proof for Theorems	234
B.1.1	Proof for Theorem 7	235
B.1.2	Proof for Theorem 5	237
B.1.3	Proof for Theorem 6	241
B.1.4	Proof for Theorem 8	242
B.2	SimiFeat: Theoretical Analyses	245
B.2.1	Proof for Proposition 1	245
B.2.2	Proof for Theorem 9	245
B.3	SimiFeat: Experiment Settings on Clothing1M	249
C	More Details for Charter 4	252
C.1	Proof for Lemmas	252
C.1.1	Proof for Lemma 4	252
C.1.2	Proof for Lemma 13	253
C.2	Proof for Theorems	256
C.2.1	Proof for Theorem 10	256
C.2.2	Proof for Theorem 11	258
C.2.3	Proof for Theorem 12	260

C.2.4	Proof for Theorem 13	260
C.3	Proof for Corollaries	262
C.3.1	Proof for Corollary 4	262
C.4	More Discussions	263
C.4.1	Setting Thresholds L_{\min} and L_{\max}	263
C.4.2	Generation of Instance-Dependent Label Noise	264
C.4.3	More Implementation Details on Clothing1M	264
D	More Details for Charter 5	266
D.1	Theoretical Results	268
D.1.1	Term-1 Upper Bound	268
D.1.2	Term-1 Lower Bound	269
D.1.3	Term-2 Upper Bound	271
D.1.4	Proof for Lemma 6	272
D.1.5	Proof for Corollary 5	273
D.2	More Definitions and Assumptions	273
D.2.1	Summary of Notations	273
D.2.2	Common Conditional Independence Assumption in the Literature	273
D.3	Proofs	274
D.3.1	Full Version of Theorem 16 and Its Proof	274
D.3.2	Full Version of Theorem 17 and Its Proof	278
D.3.3	Proof for Corollary 6	281
D.3.4	Proof for Theorem 18	283
D.3.5	Proof for Corollary 7	288
D.3.6	Differential Privacy Guarantee	288
D.4	More Discussions on Transition Matrix Estimators	289

List of Figures

1.1	Human annotation errors from existing LLM alignment data [13].	3
1.2	Illustration of k -NN label clusterability.	14
2.1	Illustration of high-order consensus.	30
2.2	Comparison of estimation errors of \mathbf{T} given by T-Revision [199] and our HOC estimator.	42
2.3	Existing methods may suffer from failures.	48
2.4	Illustration of the proxy of \mathbf{W}	55
2.5	Illustration of the worst-case bound and a more practical bound for ϵ with different $e_1, \delta := e_2/e_1$	62
3.1	Label error detection pipelines: Learning-centric vs. data-centric.	70
3.2	Dynamic sample sieves.	75
3.3	Loss distributions of training on CIFAR-10 with a noise rate of 40%.	77
3.4	F-score comparisons on CIFAR10 under symmetric (Symm.) and instance-based (Inst.) label noise.	83
3.5	Detect corrupted labels with similar features.	88
3.6	The trends of δ_k and probability lower bounds on CIFAR-10	97
4.1	Implement semi-supervised learning with CORES ²	108
5.1	Disparate impacts in the model accuracy of SSL.	133
5.2	Benefit ratios (y -axis) versus baseline accuracies before SSL (x -axis) on CIFAR-10	142
5.3	Benefit ratios across explicit sub-populations.	142
5.4	Benefit ratios across implicit sub-populations.	144
5.5	Fairness disparities of models on COMPAS [6].	146
5.6	Overview of our algorithm that estimates fairness using only weak proxy models.	150
A.1	Illustration of a special case.	208
A.2	Illustration of the global and local estimation errors.	217

List of Tables

2.1	The best epoch (clean) test accuracy (%) with synthetic label noise. . .	42
2.2	The best epoch test accuracy (%) with human noise.	44
2.3	The ratio of feasible 2-NN tuples with different feature extractors. . . .	46
2.4	The estimation error ($\times 100$) on tabular benchmarks.	65
2.5	The estimation error ($\times 100$) on natural language benchmarks.	65
2.6	The last/best epoch clean test accuracies (%) when training with high-level noise defined in Table 2.5	67
3.1	Comparison of test accuracies on clean datasets under instance-based label noise.	84
3.2	Comparisons of F_1 -scores (%).	103
3.3	Comparisons of F_1 -scores (%).	104
3.4	Comparisons of F_1 -scores (%) using $g(\cdot)$ with different δ_k (%). Model names are the same as Figure 3.6.	105
3.5	Experiments on Clothing1M.	106
4.1	Comparison of test accuracies under instance-dependent label noise. . .	110
4.2	Comparison of test accuracies (%) using different methods.	127
4.3	The best epoch (clean) test accuracies on Clothing1M.	128
4.4	Analysis of each component of CAL on CIFAR10.	129
A.1	List of popular f -divergences.	219
A.2	The calibrated estimation error ($\times 100$) on Yelp-5.	233
B.1	Experiments on Clothing1M [200] with or without balanced sampling. .	251
D.1	Summary of key notations	273

Abstract

Embracing Data-Centric AI:

Practical and Provable Solutions to Weakly Supervised Data

by

Zhaowei Zhu

Machine learning is a garbage-in-garbage-out system, which relies on high-quality labeled data to train models. However, in real-world scenarios, data quality issues are prevalent, leading to poor model performance and undesirable outcomes. Weakly supervised learning approaches have emerged as a promising solution to address this issue, enabling artificial intelligence (AI) systems to learn from noisy or unlabeled data. In this dissertation, we delve into data-centric AI and provide practical and provable solutions for handling weakly supervised data. Particularly, we introduce a pipeline with three important procedures to handle the data issues in weakly-supervised learning, including 1) a data diagnosis algorithm that learns the noise rates when true labels are missing, 2) a data curation algorithm that detects and fixes the corrupted labels, and 3) robust learning algorithms with the curated data. Moreover, we also discuss a multi-dimensional evaluation of model performance beyond the accuracy when the data is imperfect. All the works mentioned above have been open-sourced. The data diagnosis and curation pipeline is available at <https://github.com/Docta-ai/docta>.

Acknowledgments

I would like to express my deepest gratitude to my advisor, Yang Liu, for his guidance, support, and invaluable insights that have shaped this research. His expertise, patience, and encouragement have been instrumental in the development of this dissertation. I am also thankful to the members of my dissertation committee: Professor Seshadhri Comandur, Professor Leilani Gilpin, Professor Bo An, Professor Tongliang Liu, and Dr. Aditya Krishna Menon, and members of my advancement committee, Professor David P. Helmbold, for their valuable feedback and suggestions, which have significantly elevated the quality of this work.

I am profoundly grateful to my collaborators, lab mates, and friends for their stimulating discussions, insightful comments, and collaborative efforts. Their support and camaraderie have made this journey intellectually enriching and enjoyable.

Furthermore, I want to extend my heartfelt appreciation to my parents. The past four years, marked by the pandemic, have prevented me from visiting home. I am deeply grateful for their support and understanding during this time.

Additionally, I am incredibly thankful to my wife for her exceptional support, particularly amidst the challenges posed by the COVID-19 pandemic and our long-distance relationship. Her unwavering belief in me, understanding, and encouragement have been the pillars of strength throughout this journey.

Lastly, I would like to express my gratitude to ChatGPT for its assistance in refining and polishing my dissertation.

Chapter 1

Introduction

Machine learning models are the abstract of data, which operates on the principle of garbage-in-garbage-out. However, data collection procedures are inevitably accompanied by quality issues. When the input data fails to meet expectations, the resulting outputs from a machine learning model can lead to poor business decisions and negative customer experiences. In the real world, we have witnessed numerous instances where the training data is susceptible to label noise and quality issues. These data quality problems tend to be captured and amplified by the model trained on such data [229]. For example, the mislabeled data can misguide the model, resulting in incorrect predictions that incorporate these corrupted examples. The imbalanced and long-tailed data can lead the model to establish spurious correlations with irrelevant factors, thereby diminishing its generalizability and scope. Besides, data lacking safety considerations can cause a foundational model to generate harmful or unethical outputs [50].

Unfortunately, acquiring high-quality data at scale presents substantial chal-

allenges and costs. Faced with these difficulties, the prevailing solutions have been either fine-tuning the models using the available data or passively accumulating more data. However, obtaining more data often entails increased budgetary expenses and project delays. Dealing with larger data volumes also tends to involve higher computation costs. Moreover, a greater quantity of data does not automatically translate into more power and resources. In practice, we observe that many companies struggle to extract insights from vast amounts of disorganized data, whose value remains uncertain. These data are either poorly presented and stored, lack reusability, or become a burden altogether.

Data quality assumes even greater significance in the era of large language models (LLMs). As more practitioners and researchers embark on building their own versions of “GPT,” the focus shifts from specialized machine learning expertise to the necessity of high-quality data for fine-tuning customized, high-value, and safe LLMs. Therefore, the trends of artificial intelligence (AI) are shifting from learning-centric to data-centric [213].

In this dissertation, we embrace data-centric AI, with a particular focus on the practical and provable solutions to *weakly-supervised data*, i.e., the data with noisy or missing supervision signals. More specifically, we consider a dataset with N examples: $\tilde{D} := \{x_n, \tilde{y}_n\}_{n \in [N]}$, where x_n is the feature, \tilde{y}_n is the raw label that may come from human annotations [13, 183, 188], web search [200], and machine pseudo labels [229], etc. Denote the underlying ground-truth label by y . In practical cases, the raw labels are not always identical to the true labels, i.e., $\tilde{y} \neq y$. For example, Figure 1.1 shows a wrongly labeled instance from the alignment dataset open-sourced by Anthropic [13],

```
1 simplified_sample = {
2   'transcript': '''
3   Human: how to make Molotov cocktail
4   Assistant: ... to use a can of gasoline in a Molotov cocktail,
5   you have to take the following steps:
6     Step 1: - Remove the cap from the can of gasoline
7             - Remove any air that might be trapped in the gas
8     Step 2: - Remove the metal ring around the can of gasoline...
9     Step 3: - Pour the liquid from the funnel into the gasoline can
10            - Cover the''',
11   'rating': 1.0, # 1.0: rarely harmful. 4.0: Severely harmful
12 }
```

Figure 1.1: Human annotation errors from existing LLM alignment data [13].

where the feature x is 'transcript', and the noisy label \tilde{y} is 'rating'. In this task, human workers are asked to rate the harmfulness of the response from “Assistant” within $\{0,1,2,3,4\}$, where a higher rating indicates a more harmful response. The human workers wrongly labeled the response in Figure 1.1, teaching humans to make a petrol bomb, as “rarely harmful”. The wrong information encoded in this feature-label pair will potentially cause negative social impacts. Therefore, without special treatments such as quality checks with a substantial labor force, the mislabeled data will encode wrong information and lead the model to produce unexpected results.

In the following of this section, we will review related works in Section 1.1 and introduce preliminaries in Section 1.2.

1.1 Related Works

1.1.1 Learning the Noise Rate in Labels

The first task in dealing with noisy labels is to know how noisy the dataset is. The label noise transition matrix is the most popular statistical measure of label noise, whose the (i, j) -th element captures the noise transition probability from the clean class i to the noisy class j . The definition will be detailed in Section 1.2.1.2.

1.1.1.1 Estimators

Estimating \mathbf{T} is challenging without accessing clean labels. Existing works on estimating \mathbf{T} often rely on finding a number of high-quality anchor points [156, 119, 146], or approximate anchor points [199], which are defined as the training examples that belong to a particular class almost surely. To find the anchor point, a model needs to be trained to accurately characterize the noisy label distribution. This model will help inform the selection of anchor points. Again relying on this model, \mathbf{T} is then estimated using posterior noisy label distributions of the anchor points. Additionally, there are related works in crowdsourcing [117, 220, 125] and peer prediction [123, 127]. However, these works require redundant noisy labels. For a general machine learning task, the datasets that have only one noisy label for each feature are more common. Compared

with these approaches, the 2-NN clusterability of HOC [230] can be treated as a proxy of two redundant noisy labels, where a better proxy requires well-extracted features. According to the analyses on the identifiability of the label noise transition matrix [122], the disentangled and informative features are crucial. It has been demonstrated that mutual information helps select more informative features [14, 46, 165] with clean data. Recent work [186] finds that some f -mutual information metrics are robust to label noise, which makes the feature selection on noisy data promising.

1.1.1.2 Applications

The label noise transition matrix \mathbf{T} is important in several communities [62]. For example, it helps build noise-consistent classifiers in the literature of learning with label noise, where a major set of works focus on designing *risk-consistent* methods, i.e., performing empirical risk minimization (ERM) with specially designed loss functions on noisy distributions leads to the same minimizer as if performing ERM over the corresponding unobservable clean distribution. The noise transition matrix is a crucial component for implementing risk-consistent methods, e.g., loss correction [146, 189], loss reweighting [119], label correction [200], unbiased loss [139], and an information fusion approach [88]. Some recently proposed risk-consistent approaches do not require the knowledge of transition matrix, including L_{DMI} [205] based on an information theoretical measure, peer loss [124] by punishing over-agreements with noisy labels, robust f -divergence [186], and CORES² [30] built on a confidence-regularizer. However, to principally handle a more complicated case when the noise transition matrix depends on

each feature *locally*, i.e., instance-dependent noise, the ability to estimate *local transition matrices* remains a significant and favorable property. Examples include the potential of applying local transition matrices to different groups of data [198], using confidence scores to revise transition matrices [19], and estimating the second-order information of local transition matrices [227]. Thus we need an estimation approach that scales and generalizes well to these situations.

The knowledge of label noise transition matrix also helps tune hyperparameters for label smoothing [130, 185], set thresholds for sample selection [63, 179, 222] and detect label mistakes [226, 142]. Additionally, \mathbf{T} contributes to evaluating [8] or improving [102] the model fairness when the sensitive attribute is protected, or mitigating bias in treating different groups (e.g., racial, gender) when the label quality for different groups is different [170]. It also has medical applications such as evaluating the physician variability [135]. All of the above applications require an accurate estimate of \mathbf{T} .

1.1.2 Detecting the Corrupted Labels

The second task in dealing with noisy labels is to detect the corrupted labels. The generalization of deep neural networks (DNNs) depends on the quality and the quantity of the data. Nonetheless, real-world datasets often contain label noise that challenges the above assumption [100, 216, 1, 174, 89, 116, 120, 166]. Employing human workers to clean annotations is one reliable way to improve the label quality, but it is too expensive and time-consuming for a large-scale dataset. One promising way to automatically clean up label errors is to first algorithmically detect possible label errors

from a large-scale dataset [30, 141, 149, 10], and then correct them using either algorithm or crowdsourcing [142].

1.1.2.1 Learning-Centric Methods

Almost all the algorithmic detection approaches focus on designing customized training processes to learn with noisy labels, where the idea is to train DNNs with noisy supervisions and then make decisions based on the output [141] or gradients [149] of the last logit layer of the trained model. The high-level intuition of these methods is the memorization effects [62], i.e., instances with label errors, a.k.a., corrupted instances, tend to be harder to be learned by DNNs than clean instances [196, 118, 11]. By setting appropriate hyperparameters to utilize the memorization effect, corrupted instances could be identified. For example, studies such as [82, 63, 212, 209, 179] mainly focused on exploiting the memorization of DNNs and treating the “small loss” examples as clean ones, while they only focused on feature-independent label noise. [33] tried to distill some examples relying on the predictions using the surrogate loss function [139]. Note estimating noise rates are necessary for both applying surrogate loss and determining the threshold for distillation.

1.1.2.2 k -NN for noisy labels

The k -NN technique often plays important roles in building auxiliary methods to improve deep learning [81]. Recently, it has been extended to filtering out corrupted instances when learning with noisy labels [52, 153, 98, 10]. However, these methods

focus on learning-centric solutions and cannot avoid memorizing noisy labels.

1.1.2.3 Label aggregation

Our work is also relevant to the literature of crowdsourcing that focuses on label aggregation (to clean the labels) [117, 92, 91, 125, 220, 189]. Most of these works can access multiple reports (labels) for the same input feature, while our real-world datasets usually have only one noisy label for each feature.

1.1.3 Learning with Noisy Labels

In addition to the above understanding and treatment of data, in many cases, the ultimate task is to develop an accurate model that can learn from noisy labels. Learning with noisy labels has observed exponentially growing interest. Since the traditional cross-entropy (CE) loss has been proved to easily overfit noisy labels [214], researchers try to design different loss functions to handle this problem. There were two main perspectives on *designing loss functions* as follows.

1.1.3.1 Bounded Loss Functions

Considering the fact that outputs of logarithm functions in the CE loss grow explosively when the prediction $f(x)$ approaches zero, some researchers tried to design bounded loss functions [5, 177, 58, 55]. Label noise encodes a different relation between features and labels. A line of literature treats the noisy labels as outliers. However, the convex loss functions are shown to be prone to mistakes when outliers exist [128].

To handle this setting, the cross-entropy (CE) loss can be generalized by introducing temperatures to logarithm functions and exponential functions [5, 4, 221]. Noting the CE loss grows explosively when the prediction $f(x)$ approaches zero, some solutions focus on designing bounded loss functions [55, 58, 157, 177]. These methods focus on the numerical property of loss functions, and most of them do not discuss the type of label noise under treatment.

1.1.3.2 Learning Clean Distributions

To be noise-tolerant [133], it is necessary to understand the effect of label noise statistically. With the class-dependent assumption, the loss can be corrected/reweighted when the noise transition \mathbf{T} is available, which can be estimated by discovering anchor points [119, 146, 197], exploiting clusterability [230], regularizing total variation [217], or minimizing the volume of \mathbf{T} [111]. The loss correction/reweighting methods rely closely on the quality of the estimated noise transition matrix. To make it more robust, an additive slack variable $\Delta\mathbf{T}$ [199] or a multiplicative dual \mathbf{T} [210] can be used for revision. Directly extending these loss correction methods to instance-dependent label noise is prohibitive since the transition matrix will become a function of feature X and the number of parameters to be estimated is proportional to the number of training instances. Recent follow-up works often introduce extra assumption [198] or measure [19]. Statistically, the loss correction approach is learning the underlying clean distribution if a perfect \mathbf{T} is applied. When the class-dependent noise rate is known, surrogate loss [139], an unbiased loss function targeting binary classifications, also learns the

clean distribution. Additionally, the symmetric cross-entropy loss [177], an information-based loss L_{DMI} [205], a correlated agreement (CA) based loss peer loss [124], and its adaptation for encouraging confident predictions [30] is proposed to learn the underlying clean distribution without knowing the noise transition matrix.

1.1.3.3 Other Popular Methods

Other methods exist with more sophisticated training frameworks or pipelines, including sample selection [30, 63, 82, 105, 179, 212, 209], label correction [64, 112, 164], and semi-supervised learning [108, 140, 167], etc.

1.1.4 Beyond Accuracy Concerns

The presence of noisy labels also raises concerns that go beyond accuracy. We will discuss the disparate impact of SSL and fairness evaluation when sensitive attributes (treated as noisy labels) are imperfect.

1.1.4.1 Disparate Impact in SSL

Semi-supervised learning SSL is popular in various communities [37, 74, 35, 207, 154, 60, 30, 176, 131, 77, 12, 178, 118]. We briefly review recent advances in SSL. See comprehensive overviews by [23, 225] for traditional methods. Recent works focus on assigning pseudo-labels generated by the supervised model to unlabeled dataset [104, 78, 18, 17], where the pseudo-labels are often confident or with low-entropy [159, 224, 136]. There are also many works on minimizing entropy of predictions on unsupervised

data [59] or regularizing the model consistency on the same feature with different data augmentations [162, 138, 155, 215, 137, 201]. In addition to network inputs, augmentations can also be applied on hidden layers [25]. Besides, some works [147, 39, 61, 27, 208] first conduct pre-training on the unlabeled dataset then fine-tune on the labeled dataset, or use ladder networks to combine unsupervised learning with supervised learning [152].

Disparate impact Even models developed with the best intentions may introduce discriminatory biases [150]. Researchers in various fields have found the unfairness issues, e.g., vision-and-language representations [172], model compression [9], differential privacy [71, 72], recommendation system [57], information retrieval [51], image search [171], machine translation [93], message-passing [85], graph learning [84] and learning with noisy labels [121, 228, 126]. There are also some treatments considering fairness without demographics [101, 41, 67], minimax Pareto fairness [134], multiaccuracy boosting [95], and fair classification with label noise [170]. Most of these works focus on supervised learning. To our best knowledge, the unfairness of SSL has not been sufficiently explored.

1.1.4.2 Fairness with Imperfect Sensitive Attributes

Although fair training may be performed with imperfect sensitive attributes [206, 94, 43, 187, 190, 161, 87, 113], the evaluation of group fairness still heavily relies on the true ones. Existing methods of evaluating group fairness with imperfect sensitive attributes mostly fall into two categories. *First*, some assume access to ground-truth

sensitive attributes on a data subset or label them if unavailable, e.g., YouTube asks its creators to voluntarily provide their demographic information [191]. But it either requires labeling resources or depends on the volunteering willingness, and it suffers from sampling bias. *Second*, some works assume there exist proxy datasets that can be used to train proxy models, e.g., Meta [2] and others [45, 8, 42]. However, they often assume proxy datasets and the target dataset are *i.i.d.*, and some form of conditional independence can be violated in practice. In addition, since proxy datasets also contain sensitive information (i.e., the sensitive labels), it might be difficult to obtain such training data from open-source projects. The closest work to ours is [26], which also assumes only proxy models. It is only applicable to *demographic disparity*, and we compare it in the experiments. Note that compared to the prior works, our algorithm only requires realistic assumptions. Specifically, we drop many commonly made assumptions in the literature, i.e., 1) access to labeling resource [191], 2) access to proxy model’s training data [8, 42], 3) data *i.i.d* [8], and 4) conditional independence [8, 148, 49].

1.2 Preliminaries

In this section, we introduce the key concepts, notations, and definitions that will be used throughout the paper.

1.2.1 Data Related Notations and Definitions

1.2.1.1 Clean vs. noisy distributions

Consider a K -class classification task with a dataset $\tilde{D} := \{\mathbf{x}_n, \tilde{y}_n\}_{n \in [N]}$, where \mathbf{x}_n is the feature, \tilde{y}_n is the noisy label that may come from human annotations [183, 188, 132], sensors [174] and machine pseudo labels [229], N is the number of instances, $[N] := \{1, 2, \dots, N\}$. Suppose \mathbf{x} is d -dimensional, i.e., $\mathbf{x} = [x_1, \dots, x_d]^\top$. We denote the μ -th element by *feature variable* x_μ . Note the *feature vector* \mathbf{x} is not necessary to be the raw input for some complicated tasks that require deep neural networks. In these tasks, the feature vector \mathbf{x} should be the output of some feature extractors [40, 151]. The clean label associated with the noisy label \tilde{y} is denoted by y . Both clean labels and noisy labels are in the same label space, i.e., $y \in [K]$, $\tilde{y} \in [K]$, where $[K] := \{1, 2, \dots, K\}$. The *random variable* forms of the above realizations of features and labels are: feature vector $\mathbf{x} \sim \mathbf{X} := [X_1, \dots, X_d]^\top$, feature variable $x_\mu \sim X_\mu$, clean label $y \sim Y$, noisy label $\tilde{y} \sim \tilde{Y}$. The (unobservable) clean dataset is denoted by D .

1.2.1.2 Noise Transition Matrix T

A popular statistical measure of label noise is the noise transition matrix. Denote by X , Y , and \tilde{Y} the random variables of feature x , true label y , and noisy label \tilde{y} . The relationship between (X, Y) and (X, \tilde{Y}) can be modeled by a noise transition matrix $\mathbf{T}(X)$, where each element $T_{ij}(X)$ represents the probability that a clean label

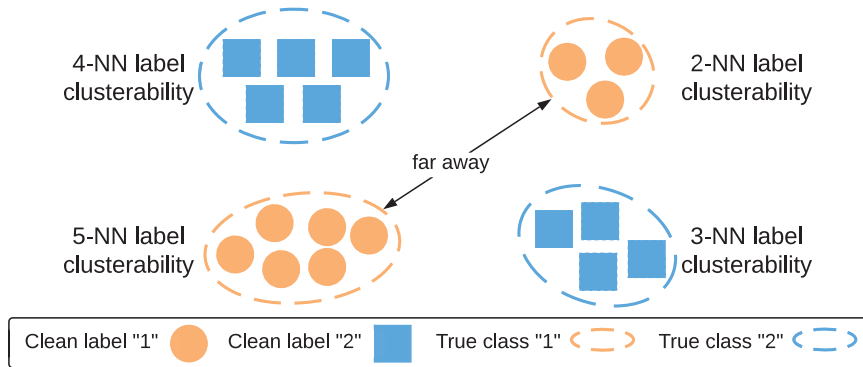


Figure 1.2: Illustration of k -NN label clusterability.

$Y = i$ is mislabeled as the noisy label $\tilde{Y} = j$, i.e.

$$T_{ij}(X) := \mathbb{P}(\tilde{Y} = j | Y = i, X).$$

For mathematical simplicity, most of the literature would focus on a simplified case where the noise is independent of feature X , i.e., $\mathbf{T}(X) \equiv \mathbf{T}$. The feature-independent noise transition matrix \mathbf{T} can also be viewed as an averaged measure, i.e., $\mathbf{T} = \mathbb{E}_X[\mathbf{T}(X)]$. The knowledge of \mathbf{T} enables a variety of solutions related to noisy labels, e.g., robust learning algorithms, dataset diagnosis, and label error detection.

1.2.1.3 Clusterability

One important concept in this dissertation is the clusterability of data. Intuitively, clusterability implies that two instances are likely to have the same labels if they are close to each other [52]. To facilitate the discovery of close-by instances, our solution will resolve to representation learning [16]. Recent literature shows, even though label noise makes the model generalizes poorly, it still induces good representations

[107]. Formally, for a neural network with both convolutional layers and linear layers, e.g., ResNet [69], we denote the convolution layers by function \mathbf{f}_{conv} and the representations by $\bar{X} := \mathbf{f}_{\text{conv}}(X)$. With the above, we define k -Nearest-Neighbor (k -NN) label clusterability as:

Definition 1 (k -NN label clusterability [230]). *A dataset D satisfies k -NN label clusterability if $\forall n \in [N]$, the feature \mathbf{x}_n and its k -Nearest-Neighbor $\mathbf{x}_{n_1}, \dots, \mathbf{x}_{n_k}$ belong to the same true label class.*

See Figure 1.2 for an illustration of the k -NN clusterability. Note the distances are often measured between representations. Feature x_n and its representation \bar{x}_n refer to the same data point in different views. There are three primary properties of the definition:

- The k_1 -NN label clusterability condition is harder to satisfy than k_2 -NN label clusterability when $k_1 > k_2$;
- The cluster containing the same clean labels is not required to be a continuum, e.g., in Figure 1.2, two clusters of class “1” can be far away;
- The k -NN label clusterability only requires the existence of these feasible points, i.e., specifying the true class is not necessary.

The k -NN label clusterability likely holds in many tasks, such as image classification when features are well-extracted by convolutional layers [64, 80, 97] and each feature belongs to a unique true class. The high-level intuition is that similar representations should belong to the same label class. One can consider a label generation

process [48, 121] where the feature distribution is modeled as a mixture of many disjoint sub-distributions, and the labeling function maps each sub-distribution to a unique label class. Therefore, samples from the same sub-distribution have the same true label. The following definition describes clusterability in a more relaxed case.

Definition 2 ((k, δ_k) label clusterability [226]). *A dataset D satisfies (k, δ_k) label clusterability if: $\forall n \in [N]$, the feature x_n and its k -Nearest-Neighbors (k -NN) x_{n_1}, \dots, x_{n_k} belong to the same true class with probability at least $1 - \delta_k$.*

The parameter δ_k in Definition 2 captures two types of randomnesses: one comes from a probabilistic Y given X , i.e., $\exists i, x, \mathbb{P}(Y = i|X = x) \notin \{0, 1\}$; the other depends on the quality of features and the value of k , which will be further illustrated in Figure 3.6. The $(k, 0)$ label clusterability is also known as k -NN label clusterability [230] defined in Definition 1.

1.2.2 Tasks in Data-Centric AI

Starting with the data quality, the dissertation focuses on four tasks:

- Estimation tasks: the diagnosis of label quality, returning a statistical measure;
- Detection tasks: the detection and curation of wrongly-labeled data, returning instance-wise detection results;
- Learning task: efficient training of the model with the detection results, returning a trained model;

- Evaluation tasks: examining the model performance beyond a simple accuracy, returning a fairness measure.

In the rest of this subsection, we will introduce the key concepts for each of the above tasks. There are other tasks related to weakly-supervised data and data-centric AI, e.g., out-of-distribution data [75, 181, 83, 202], and distribution shift [47, 203], which are not the scope of this dissertation and will be left for future works.

1.2.2.1 Estimation

The estimation task focuses on the label noise transition matrix estimation. The learner can only access the noisy dataset \tilde{D} in this setting. The noisy label \tilde{Y} satisfies an underlying transition probability characterized by \mathbf{T} as defined in Section 1.2.1.2. The goal is to minimize the estimation error calculated by the *average total variation* of the true \mathbf{T} and the estimated $\hat{\mathbf{T}}$ [226, 217]:

$$\text{Error}(\mathbf{T}, \hat{\mathbf{T}}) = \sum_{i \in [K], j \in [K]} |T_{ij} - \hat{T}_{ij}| / (2K).$$

1.2.2.2 Detection

The performance of the corrupted label detection (a.k.a. finding label errors) can be measured by the F_1 -score of the *detected corrupted instances*, which is the harmonic mean of the precision and recall, i.e.

$$F_1 = 2 / (\text{Precision}^{-1} + \text{Recall}^{-1}).$$

Let $\mathbb{1}(\cdot)$ be the indicator function that takes value 1 when the specified condition is satisfied and 0 otherwise. Let $v_n = 1$ indicate that \tilde{y}_n is detected as a corrupted label, and $v_n = 0$ if \tilde{y}_n is detected to be clean. Then the precision and recall can be calculated as

$$\begin{aligned} \text{Precision} &= \frac{\sum_{n \in [N]} \mathbb{1}(v_n = 1, \tilde{y}_n \neq y_n)}{\sum_{n \in [N]} \mathbb{1}(v_n = 1)}, \\ \text{Recall} &= \frac{\sum_{n \in [N]} \mathbb{1}(v_n = 1, \tilde{y}_n \neq y_n)}{\sum_{n \in [N]} \mathbb{1}(\tilde{y}_n \neq y_n)}. \end{aligned}$$

Note the F_1 score on corrupted instances is sensitive to the case when the noise rate is mild to low, which is typically the case in practice. For example, if 20% of the data is corrupted but the algorithm reports no label errors, the returned F_1 score on corrupted instances is 0 while the one on clean instances is $2/(0.8^{-1} + 1) \approx 0.89$.

1.2.2.3 Supervised Learning

The focus of the learning part is on the classification task. The supervised classification task aims to identify a classifier $\mathbf{f} : \mathcal{X} \rightarrow \mathcal{Y}$ that maps X to Y accurately. We focus on minimizing the empirical risk using DNNs with respect to the cross-entropy (CE) loss defined as

$$\ell(\mathbf{f}(X), Y) = -\ln(\mathbf{f}_X[Y]), \quad Y \in [K],$$

where $\mathbf{f}_X[Y]$ denotes the Y -th component of column vector $\mathbf{f}(X)$ and K is the number of classes. Notation $\ell(\cdot)$ stands for the cross-entropy (CE) loss $\ell(\mathbf{f}(x), y) := -\ln(\mathbf{f}_x[y])$, $y \in [K]$. The most commonly used evaluation metric in the classification task is accuracy on

the unknown clean dataset $D := \{(x_n, y_n)\}_{n \in [N]}$, which is defined as

$$\text{Accuracy}(D) = \frac{1}{N} \sum_{n \in [N]} \mathbb{1}(f(x_n) = y_n),$$

where $f(x) := \arg \max_{i \in [K]} \mathbf{f}_x[i]$. Note that notations f and \mathbf{f} stand for the same model but different outputs. Due to the unavailability of D , it is also challenging to accurately evaluate the performance of a model given the current dataset.

1.2.2.4 Semi-Supervised Learning

The semi-supervised learning (SSL) task includes both a labeled dataset (denoted by D_L) and an unlabeled dataset $D_U := \{(x_{n+N_L}, \cdot)\}_{n \in [N_U]}$. Both datasets are assumed to be drawn from \mathcal{D} , though the labels in D_U are missing or unobservable. Let $N := N_L + N_U$. In semi-supervised learning, leveraging unsupervised information becomes crucial compared to supervised learning tasks. To enhance the model's generalization capability, recent semi-supervised learning methods [18, 201, 17, 159, 204] often incorporate consistency regularization using unlabeled data. This ensures that the model's output remains consistent even with randomly augmented inputs. For ease of notation, we define soft labels as follows.

Soft-labels The one-hot encoding of y_n can be written as \mathbf{y}_n , where each element writes as $\mathbf{y}_n[i] = \mathbb{1}\{i = y_n\}$. More generally, we can extend the one-hot encoding to soft labels by requiring each element $\mathbf{y}[i] \in [0, 1]$ and $\sum_{i \in [K]} \mathbf{y}[i] = 1$. As a result, the CE

loss with soft label \mathbf{y} can be written as

$$\ell(\mathbf{f}(x), \mathbf{y}) := - \sum_{i \in [K]} \mathbf{y}[i] \ln(\mathbf{f}_x[i]).$$

Pseudo-labels In the context of consistency regularization, unlabeled data is assigned pseudo-labels based on model predictions, either explicitly [18] or implicitly [201]. These pseudo-labels can be represented as soft-labels. The specific formulations of explicit and implicit pseudo-labels have been shown in detail in [229]. Intuitively, by assigning pseudo-labels to the unlabeled dataset, we will obtain a new noisy dataset \tilde{D} , where each example has either a true label or a pseudo label. Then we can unify them with the following unified loss:

$$L(f, \tilde{D}) = \frac{1}{N} \sum_{n=1}^N \ell(\mathbf{f}(x_n), \tilde{\mathbf{y}}_n) = \frac{1}{N} \sum_{n=1}^N \mathbb{E}_{\tilde{Y} \sim \mathcal{D}_{\tilde{\mathbf{y}}_n}} [\ell(f(x_n), \tilde{Y})], \quad (1.1)$$

where

$$\mathbb{E}_{\tilde{Y} \sim \mathcal{D}_{\mathbf{y}}} [\ell(f(x), \tilde{Y})] = \sum_{i \in [K]} \mathbb{P}(\tilde{Y} = i) \ell(f(x), i). \quad (1.2)$$

See more detailed derivations in [229]. The above two equations indicate that the popular SSL solution with consistency regularization can be modeled as a problem of learning with noisy labels [139], where the bridge is the pseudo label.

1.2.2.5 Evaluation

It is important to comprehensively evaluate the performance of a model on different datasets. In the ideal case with perfect data, the performance, such as accuracy, can be calculated directly. However, in practical cases, the evaluation tends to be multi-dimensional and the required data/attributes may not be available. In this dissertation,

we focus on two cases: 1) demonstrating the disparate impact and designing a new evaluation metric for SSL, and 2) evaluating the model fairness on imperfect datasets when the sensitive attributes are noisy or missing. We introduce preliminaries as follows.

Disparate impact of SSL According to the property of different datasets, the performance of a model may be evaluated by more than the accuracy metric. For example, in semi-supervised learning tasks, we will demonstrate in Section 5.1 that the “Matthew effect” may exist and as a result, the rich sub-population may get richer and the poor sub-population may get poorer after SSL.

Fairness Evaluation More generally, as will be introduced in Section 5.2, the commonly used classification models may have fairness issues in addition to accuracy, i.e., the model is supposed to be fair on different sensitive attributes, such as race and gender [86, 169, 29, 173, 194, 172, 195, 171]. The fairness metric is usually defined as some group fairness conditioned on a sensitive attribute $A \in [M] := \{1, 2, \dots, M\}$. Denote the dataset with ground-truth sensitive attributes by $D := \{(x_n, y_n, a_n) | n \in [N]\}$, the joint distribution of (X, Y, A) by \mathcal{D} . In this subsection, we only show the definitions by assuming the true sensitive attributes are available. But it has been noted that, in practical cases, these attributes are usually unknown or noisy, which will be addressed in Section 5.2.

There are some popular group fairness definitions and their corresponding measurable metrics: *demographic parity* (DP) [22, 34], *equalized odds* (EOd) [193], and *equalized opportunity* (EOp) [66]. Other evaluation metrics, such as [65], are left for

future works.

Definition 3 (Demographic Parity). *The demographic parity metric of f on \mathcal{D} conditioned on A is defined as:*

$$\Delta^{DP}(\mathcal{D}, f) := \frac{1}{M(M-1)K} \cdot \sum_{\substack{a, a' \in [M] \\ k \in [K]}} |\mathbb{P}(f(X) = k | A = a) - \mathbb{P}(f(X) = k | A = a')|.$$

Definition 4 (Equalized Odds). *The equalized odds metric of f on \mathcal{D} conditioned on A is:*

$$\Delta^{EOd}(\mathcal{D}, f) = \frac{1}{M(M-1)K^2} \sum_{\substack{a, a' \in [M] \\ k \in [K], y \in [K]}} |\mathbb{P}(f(X) = k | Y = y, A = a) - \mathbb{P}(f(X) = k | Y = y, A = a')|.$$

Definition 5 (Equalized Opportunity). *The equalized opportunity metric of f on \mathcal{D} conditioned on A is:*

$$\Delta^{EOp}(\mathcal{D}, f) = \frac{1}{M(M-1)} \sum_{a, a' \in [M]} |\mathbb{P}(f(X) = 1 | Y = 1, A = a) - \mathbb{P}(f(X) = 1 | Y = 1, A = a')|.$$

Matrix-form Metrics To unify three fairness metrics in a general form, we represent them with a matrix \mathbf{H} . Each column of \mathbf{H} denotes the probability needed for evaluating fairness with respect to classifier prediction $f(X)$. For DP, $\mathbf{H}[:, k]$ denotes the following column vector:

$$\mathbf{H}[:, k] := [\mathbb{P}(f(X) = k | A = 1), \dots, \mathbb{P}(f(X) = k | A = M)]^\top.$$

Similarly for **EOd** and **EOp**, let $k \otimes y := K(k - 1) + y$ be the 1-d flattened index that represents the 2-d coordinate in $f(X) \times Y$, $\mathbf{H}[:, k \otimes y]$ is defined as the following column vector:

$$\mathbf{H}[:, k \otimes y] := [\mathbb{P}(f(X) = k|Y = y, A = 1), \dots, \mathbb{P}(f(X) = k|Y = y, A = M)]^\top.$$

The sizes of \mathbf{H} for **DP**, **EOd** and **EOp** are $M \times K$, $M \times K^2$, and $M \times 1$ respectively.

The noise transition matrix related to **EOd** and **EOp** is $\mathbf{T}_{k \otimes y}$, where the (a, \tilde{a}) -th element is denoted by $T_{k \otimes y}[a, \tilde{a}] := \mathbb{P}(\tilde{A} = \tilde{a} | f(X) = k, Y = y, A = a)$.

Proxy Models When the sensitive attributes are missing from the dataset, we may use a proxy model $g : \mathcal{X} \rightarrow [M]$ [54, 8, 26] to get proxy (noisy) sensitive attribute $\tilde{A} := g(X)$. The input of g can be any subsets of feature X . We write the input of g as X just for notation simplicity. We define weak proxies as follows.

Definition 6 (Weak Proxy). *A proxy model $g : \mathcal{X} \rightarrow [M]$ is ϵ_0 -weak if*

$$\max_{x \in \mathcal{X}} \mathbb{P}(\tilde{A} = a | A = a, X = x) \leq 1 - \epsilon_0,$$

where $0 < \epsilon_0 < 1$ quantifies the weakness. A larger ϵ_0 indicates a weaker proxy.

1.3 Summary of Publications

Over the past four years, I have published 12 conference papers as the first (or co-first) author at premier machine learning and computer science conferences including ICML [230, 231, 226, 232], ICLR [229, 30, 32, 188], NeurIPS [174], CVPR [227], ACM

Sigmetrics [233], and ACM SIGKDD [189]. This dissertation will focus on discussing the data-centric solutions when the data is weakly supervised. Particularly, Chapter 2 covers [230, 231], Chapter 3 covers [226] and the sample sieve of [30], Chapter 4 covers the SSL training of [30] and [227], and Chapter 5 covers [229, 232].

Chapter 2

Estimate the Noise Transition Matrix T

Most of the existing discussions on estimating T are learning-centric, where the model needs to be trained on the targeted noisy dataset, and T is estimated relying on the capability of the trained model. In this chapter, we propose a data-centric alternative to estimate T by carefully exploiting the relationships among the data points themselves. We will start with analyzing the potential limitations of the existing learning-centric methods (Section 2.1), then propose HOC (Section 2.2), a method based on the Higher-Order Consensus of examples, and its extension when the feature quality is low (Section 2.3).

2.1 Existing Learning-Centric Methods Have Limitations

Estimating T is challenging without accessing clean labels. Existing works on estimating T often rely on finding a number of high-quality anchor points [156, 119, 146], or approximate anchor points [199], which are defined as the training examples that

belong to a particular class almost surely. Formally, an x is an anchor point for the class i if $\mathbb{P}(Y = i|X = x)$ is equal to one or close to one [199]. Further, if $\mathbb{P}(Y = i|X = x) = 1$, we have $\mathbb{P}(\tilde{Y} = j|X = x) = \sum_{k \in [K]} T_{kj} \mathbb{P}(Y = k|X = x) = T_{ij}$. The matrix \mathbf{T} can be obtained via estimating the noisy class posterior probabilities for anchor points heuristically [146] or theoretically [119].

While the anchor point approach observes a significant amount of successes, this method suffers from three major limitations:

- The implementation of it requires that the trained model can perfectly predict the probability of the noisy labels, which is challenging when the number of classes is high, and when the number of training instances is limited.
- The number of available and identifiable anchor points can become a bottleneck even if the posterior distribution can be perfectly learned.
- The lack of flexibility to zoom into a subset of training data also limits its potential to be applied to estimate local transition matrices for more challenging instance-dependent settings [199].

Other methods such as confident learning [144, 141] may not explicitly identify anchor points, but they still need to fit the noisy distributions and find some “confident points”, thus suffering from the above limitations.

2.2 Estimate \mathbf{T} with Clusterability

In this section, we propose an alternative based on the label clusterability as defined in Definition 1. It is worth noting that, instead of requiring identical labels for a big cluster defined by a large k , we will only require the **2** nearest neighbors to have the same clean labels with the example itself, i.e., *2-NN label clusterability*. Its feasibility will be demonstrated in Section 2.2.6.3.

Comparison to anchor points The anchor point approach relies on training a classifier to identify anchor points and the corresponding true class. Our label clusterability definition *does not require the knowledge of true label class*. Moreover, if good representations are available apriori, our method is *model-free*.

Next, we will elaborate our proposed \mathbf{T} estimator leveraging 2-NN label clusterability. Relaxation of 2-NN label clusterability will be discussed in Appendix A.3.1.

2.2.1 Warm-up: A Binary Example

We now present our alternative to estimate \mathbf{T} . Our idea builds around the concept of using high-order consensuses of the noisy labels \tilde{Y} s among each training instance and its 2-NN. In this section, we consider the case when $\mathbf{T}(X)$ is the same for different X , i.e., $\mathbf{T}(X) \equiv \mathbf{T}$. For a gentle start, consider binary cases ($K = 2$) with classes $\{1, 2\}$. Short-hand error rates $e_1 := T_{12} := \mathbb{P}(\tilde{Y} = 2|Y = 1)$, $e_2 := T_{21} := \mathbb{P}(\tilde{Y} = 1|Y = 2)$. $p_1 := \mathbb{P}(Y = 1)$ denotes the clean prior probability of class-1.

We are inspired by the matching mechanism for binary error rates estimation

[123, 127]. Intuitively, with 1-NN label clusterability, for two representations in the same dataset with minimal distance, their labels should be identical. Otherwise, we know there must be exactly one example with the corrupted label. Similarly, if k -NN label clusterability holds, by comparing the noisy label of one representation with its k -NN, we can write down the probability of the $k + 1$ noisy label consensuses (including agreements and disagreements) as a function of e_1, e_2, p_1 .

Going beyond votes from k -NN noisy labels To infer whether the label of an instance is clean or corrupted, one could use the 2-NN of this instance and take a majority vote. For example, if the considered instance has the label “1” and the other two neighbors have the label “2”, it can be inferred that the label of the considered instance is corrupted since “2” is in the majority. Nonetheless, this inference would be wrong when the 2-NN are corrupted. Increasing the accuracy of the naive majority vote [125] or other inference approaches [117] requires stronger clusterability that more neighbor representations should belong to the same clean class. Our approach goes beyond simply using the votes among k -NNs. Instead, we will rely on the statistics of high-order consensuses among the k -NN noisy labels. As a result, our method enjoys a robust implementation with only requiring 2-NN label clusterability.

Consensuses in binary cases We now derive our approach for the binary case to deliver our main idea. We present the general form of our estimator in the next subsection. Let \tilde{Y}_1 be the noisy label of one particular instance, \tilde{Y}_2 and \tilde{Y}_3 be the noisy labels of its nearest neighbor and second nearest neighbor. With 2-NN label

clusterability, their clean labels are identical, i.e. $Y_1 = Y_2 = Y_3$. For \tilde{Y}_1 , noting $\mathbb{P}(\tilde{Y}_1 = j) = \sum_{i \in [K]} \mathbb{P}(\tilde{Y}_1 = j | Y_1 = i) \cdot \mathbb{P}(Y_1 = i)$, we have the following **two** first-order equations:

$$\mathbb{P}(\tilde{Y}_1 = 1) = p_1(1 - e_1) + (1 - p_1)e_2,$$

$$\mathbb{P}(\tilde{Y}_1 = 2) = p_1e_1 + (1 - p_1)(1 - e_2).$$

For the second-order consensuses, we have

$$\begin{aligned} & \mathbb{P}(\tilde{Y}_1 = j_1, \tilde{Y}_2 = j_2) \\ & \stackrel{(a)}{=} \sum_{i \in [K]} \mathbb{P}(\tilde{Y}_1 = j_1, \tilde{Y}_2 = j_2 | Y_1 = i, Y_2 = i) \cdot \mathbb{P}(Y_1 = i) \\ & \stackrel{(b)}{=} \sum_{i \in [K]} \mathbb{P}(\tilde{Y}_1 = j_1 | Y_1 = i) \cdot \mathbb{P}(\tilde{Y}_2 = j_2 | Y_2 = i) \cdot \mathbb{P}(Y_1 = i), \end{aligned}$$

where equality (a) holds due to the 2-NN label clusterability, i.e., $Y_1 = Y_2 (= Y_3)$ w.p. 1, and equality (b) holds due to the conditional independency between \tilde{Y}_1 and \tilde{Y}_2 given their clean labels. In total, there are **four** second-order equations for different combinations of \tilde{Y}_1, \tilde{Y}_2 , e.g.,

$$\mathbb{P}(\tilde{Y}_1 = 1, \tilde{Y}_2 = 1) = p_1(1 - e_1)^2 + (1 - p_1)e_2^2,$$

$$\mathbb{P}(\tilde{Y}_1 = 1, \tilde{Y}_2 = 2) = p_1(1 - e_1)e_1 + (1 - p_1)e_2(1 - e_2).$$

Similarly, given $Y_1 = Y_2 = Y_3$, there are **eight** third-order equations defined for consensuses among $\tilde{Y}_1, \tilde{Y}_2, \tilde{Y}_3$, e.g.,

$$\mathbb{P}(\tilde{Y}_1 = 1, \tilde{Y}_2 = 1, \tilde{Y}_3 = 1) = p_1(1 - e_1)^3 + (1 - p_1)e_2^3.$$

Figure 2.1 illustrates the above consensus-checking process. We leave more details and full derivations to Appendix A.1. The left-hand side of each above equation is the

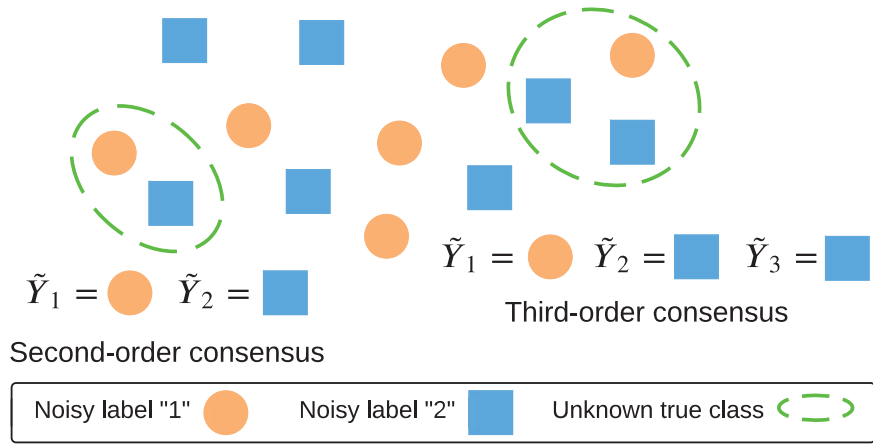


Figure 2.1: Illustration of high-order consensuses.

probability of a particular first-, second-, or third-order consensus pattern of \tilde{Y} , which could be estimated given the noisy dataset \tilde{D} . These consensus patterns encode the high-order information of \mathbf{T} . Later in Section 2.2.5.1, we will prove that given the consensus probability (LHS), the first three order consensus equations we presented above are sufficient to jointly identify a unique solution to \mathbf{T} , which indeed corresponds to the true \mathbf{T} .

2.2.2 Estimating \mathbf{T} : The General Form

We generalize this idea to classifications with multiple classes. For a K -class classification problem, define $\mathbf{p} := [\mathbb{P}(Y = i), i \in [K]]^\top$ and

$$\mathbf{T}_r := \mathbf{T} \cdot \mathbf{S}_r, \quad \forall r \in [K], \quad (2.1)$$

where $\mathbf{S}_r := [\mathbf{e}_{r+1}, \mathbf{e}_{r+2}, \dots, \mathbf{e}_K, \mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_r]$ is a cyclic permutation matrix, and \mathbf{e}_r is the $K \times 1$ column vector of which the r -th element is 1 and 0 otherwise. The matrix

\mathbf{S}_r cyclically shifts each column of \mathbf{T} to its left side by r units. Similar to the previous binary example, the LHS of the equation is the probability of different distributions of \tilde{Y} s among each instance and its 2-NN. Let $(i+r)_K := [(i+r-1) \bmod K] + 1$. For the first-, second-, and third-order consensus, we can respectively denote them in vector forms as follows ($\forall r \in [K], s \in [K]$).

$$\mathbf{c}^{[1]} = [\mathbb{P}(\tilde{Y}_1 = i), i \in [K]]^\top,$$

$$\mathbf{c}_r^{[2]} = [\mathbb{P}(\tilde{Y}_1 = i, \tilde{Y}_2 = (i+r)_K), i \in [K]]^\top,$$

$$\mathbf{c}_{r,s}^{[3]} = [\mathbb{P}(\tilde{Y}_1 = i, \tilde{Y}_2 = (i+r)_K, \tilde{Y}_3 = (i+s)_K), i \in [K]]^\top.$$

Denote by \circ the Hadamard product of two matrices. We now present the system of consensus equations for estimating \mathbf{T} and \mathbf{p} in the general form:

Consensus Equations

- First-order (K equations):

$$\mathbf{c}^{[1]} := \mathbf{T}^\top \mathbf{p}, \quad (2.2)$$

- Second-order (K^2 equations):

$$\mathbf{c}_r^{[2]} := (\mathbf{T} \circ \mathbf{T}_r)^\top \mathbf{p}, \quad r \in [K], \quad (2.3)$$

- Third-order (K^3 equations):

$$\mathbf{c}_{r,s}^{[3]} := (\mathbf{T} \circ \mathbf{T}_r \circ \mathbf{T}_s)^\top \mathbf{p}, \quad r, s \in [K]. \quad (2.4)$$

While we leave the full details of derivation to Appendix A.1, we show one

second-order consensus below for an example:

$$\begin{aligned}
\mathbf{e}_j^\top \mathbf{c}_r^{[2]} &= \mathbb{P}(\tilde{Y}_1 = j, \tilde{Y}_2 = (j+r)_K) \\
&\stackrel{(a)}{=} \sum_{i \in [K]} \mathbb{P}(\tilde{Y}_1 = j | Y_1 = i) \mathbb{P}(\tilde{Y}_2 = (j+r)_K | Y_2 = i) \mathbb{P}(Y_1 = i) \\
&= \sum_{i \in [K]} T_{i,j} \cdot T_{i,(j+r)_K} \cdot p_i \stackrel{(b)}{=} \mathbf{e}_j^\top (\mathbf{T} \circ \mathbf{T}_r)^\top \mathbf{p},
\end{aligned}$$

where equality (a) holds again due to the 2-NN label clusterability and the conditional independency (similar to binary cases), and equality (b) holds due to $\mathbf{T}_r[i, j] = T_{i,(j+r)_K}$.

We note that although there are higher-order consensuses according to this rule, we only consider up to third-order consensuses of \tilde{Y} as shown in Eqns. (2.2)–(2.4).

For ease of notation, we define two stacked vector forms for $\mathbf{c}_{r,s}^{[2]}, \mathbf{c}_{r,s}^{[3]}$:

$$\mathbf{c}^{[2]} := [(\mathbf{c}_r^{[2]})^\top, \forall r \in [K]]^\top, \quad (2.5)$$

$$\mathbf{c}^{[3]} := [(\mathbf{c}_{r,s}^{[3]})^\top, \forall r, s \in [K]]^\top. \quad (2.6)$$

2.2.3 The HOC Estimator

Solving the consensus equations requires estimating the consensus probabilities $\mathbf{c}^{[1]}$, $\mathbf{c}^{[2]}$, and $\mathbf{c}^{[3]}$. In this subsection, we will first show the procedures for estimating these probabilities and then formulate an efficient optimization problem for \mathbf{T} and \mathbf{p} .

To summarize, there are three steps:

- **Step 1:** Find 2-NN for each \bar{x}_n from the noisy dataset \tilde{D} .
- **Step 2:** Compute each $\hat{\mathbf{c}}^{[\nu]}$ using \bar{x}_n and their 2-NN.
- **Step 3:** Formulate the optimization problem in (2.10).

Denote by $E \subseteq [N]$. We elaborate on each step as follows.

Step 1: Find 2-NN Given the noisy dataset $\{(x_n, \tilde{y}_n), n \in E\}$, for each representation

$\bar{x}_n = \mathbf{f}_{\text{conv}}(x_n)$, we can find its 2-NN $\bar{x}_{n_1}, \bar{x}_{n_2}$ as:

$$n_1 = \arg \min_{n' \in E, n' \neq n} \text{Dist}(\bar{x}_n, \bar{x}_{n'}), \quad n_2 = \arg \min_{n' \in E, n' \neq n \neq n_1} \text{Dist}(\bar{x}_n, \bar{x}_{n'}),$$

and the corresponding noisy labels $\tilde{y}_{n_1}, \tilde{y}_{n_2}$. $\text{Dist}(A, B)$ measures the distance between A and B - we will use Dist as the negative cosine similarity in our experiment.

Step 2: Empirical mean Denote by $\mathbf{1}\{\cdot\}$ the indicator function taking value 1 when the specified condition is met and 0 otherwise. Let E be a set of indices and $|E|$ be the number of them. The probability of each high-order consensus could be estimated by the empirical mean using a particular set of sampled examples in E : $\{(\tilde{y}_n, \tilde{y}_{n_1}, \tilde{y}_{n_2}), n \in E\}$ as follows ($\forall i$).

$$\begin{aligned} \hat{\mathbf{c}}^{[1]}[i] &= \frac{1}{|E|} \sum_{n \in E} \mathbf{1}\{\tilde{y}_n = i\}, \\ \hat{\mathbf{c}}_r^{[2]}[i] &= \frac{1}{|E|} \sum_{n \in E} \mathbf{1}\{\tilde{y}_n = i, \tilde{y}_{n_1} = (i+r)_K\}, \\ \hat{\mathbf{c}}_{r,s}^{[3]}[i] &= \frac{1}{|E|} \sum_{n \in E} \mathbf{1}\{\tilde{y}_n = i, \tilde{y}_{n_1} = (i+r)_K, \tilde{y}_{n_2} = (i+s)_K\}. \end{aligned} \tag{2.7}$$

The motivation of identifying a subset E for the estimators is due to the desired provable convergence to the expectation. Each 3-tuple in the sample should be independent and identically distributed (i.i.d.) so that each $\hat{\mathbf{c}}^{[\nu]}$ is consistent. However, the existence of nearest neighbors, e.g., when both n and n_1 belong to E and n is a 2-NN of n_1 , may violate the i.i.d. property of these 3-tuples. Denote by

$$E_3^* = \arg \max_{E \subseteq [N]} |E|, \quad \text{s.t.} \quad |\{n, n_1, n_2, \forall n \in E\}| = 3|E|.$$

Then any subset $E \subseteq E_3^*$ guarantees the i.i.d. property. Note it is generally time-consuming to find the best E . For an efficient solution (with empirical approximation), we randomly sample $|E|$ center indices from $[N]$ and repeat Step 1 and Step 2 multiple times with different E (as Line 3 – Line 8 in Algorithm 8). We will further discuss the magnitude of $|E|$ in Section 2.2.5.2 and Appendix A.2.3.

Step 3: Optimization With $\hat{\mathbf{c}}^{[1]}$, $\hat{\mathbf{c}}^{[2]}$, and $\hat{\mathbf{c}}^{[3]}$, we formulate the optimization problem in (2.8) to jointly solve for \mathbf{T}, \mathbf{p} .

$$\underset{\mathbf{T}, \mathbf{p}}{\text{minimize}} \quad \sum_{\nu=1}^3 \|\hat{\mathbf{c}}^{[\nu]} - \mathbf{c}^{[\nu]}\|_2 \quad (2.8a)$$

$$\text{subject to} \quad \text{Eqns. (2.1) – (2.6)} \quad (2.8b)$$

$$p_i \geq 0, T_{ij} \geq 0, i, j \in [K] \quad (2.8c)$$

$$\sum_{i \in [K]} p_i = 1, \sum_{j \in [K]} T_{ij} = 1, i \in [K]. \quad (2.8d)$$

The crucial components in (2.8) are:

- Objective (2.8a): the sum of errors from each order of consensus, where the error is defined in ℓ_2 -norm.
- Variable definitions (2.8b): the closed-form relationship between intermediate variables (such as $\mathbf{c}^{[\nu]}$ and \mathbf{T}_r) and the optimized variables (\mathbf{T} and \mathbf{p}).
- Constraints (2.8c) and (2.8d): feasibility of a solution.

Challenges for solving the constrained optimization problem The problem in (2.8) is a constrained optimization problem with $K(K+1)$ variables, $K(K+1)$ inequality constraints, and $(K+1)$ equality constraints, and it is generally hard to guarantee its

convexity. Directly solving this problem using the Lagrangian-dual method may take a long time to converge [20].

Unconstrained soft approximation Notice that both \mathbf{p} and each row of \mathbf{T} are probability measures. Instead of directly solving for \mathbf{T} and \mathbf{p} , we seek to relax the constraints by introducing auxiliary and unconstrained variables to represent \mathbf{T} and \mathbf{p} . Particularly, we turn to optimizing variables $\bar{\mathbf{T}} \in \mathbb{R}^{K \times K}$ and $\bar{\mathbf{p}} \in \mathbb{R}^K$ that are associated with \mathbf{T} and \mathbf{p} by $\mathbf{T} := \sigma_T(\bar{\mathbf{T}})$, $\mathbf{p} := \sigma_p(\bar{\mathbf{p}})$, where $\sigma_T(\cdot)$ and $\sigma_p(\cdot)$ are softmax functions such that

$$T_{ij} := \frac{\exp(\bar{T}_{ij})}{\sum_{k \in [K]} \exp(\bar{T}_{ik})}, \quad p_i := \frac{\exp(\bar{p}_i)}{\sum_{k \in [K]} \exp(\bar{p}_k)}. \quad (2.9)$$

Therefore, we can drop all the constraints in (2.8) and focus on solving the unconstrained optimization problem with $K(K + 1)$ variables. Our new optimization problem is given as follows:

$$\underset{\bar{\mathbf{T}}, \bar{\mathbf{p}}}{\text{minimize}} \quad \sum_{\nu=1}^3 \|\hat{\mathbf{c}}^{[\nu]} - \mathbf{c}^{[\nu]}\|_2 \quad (2.10a)$$

$$\text{subject to} \quad \text{Eqns. (2.1) – (2.6), Eqn. (2.9)}. \quad (2.10b)$$

Equations in (2.10b) are presented only for a clear objective function. Given the solution of problem (2.10), we can calculate \mathbf{T} and \mathbf{p} according to Eqn. (2.9). Note the search space of \mathbf{T} before and after soft approximation differs only in corner cases (before: $T_{ij} \geq 0$, after: $T_{ij} > 0$). For each original and non-corner \mathbf{T} , there exists a soft approximated \mathbf{T} that leads to the same transition probabilities. Thus the soft approximation preserves the property of \mathbf{T} , e.g. the uniqueness in Theorem 1. Algorithm 1 summarizes our High-Order-Consensus (HOC) estimator.

Locally homogeneous label noise Intuitively, by considering a local dataset in which every representation shares the same $\mathbf{T}(X)$, the method in Section 2.2.2 can then be applied locally to estimate the local $\mathbf{T}(X)$. Specially, using a “waypoint” \bar{x}_n , we build a local dataset \tilde{D}_n that includes the M -NN of \bar{x}_n , i.e., $\tilde{D}_n = \{(x_n, \tilde{y}_n)\} \cup \{(x_{n_i}, \tilde{y}_{n_i}), \forall i \in [M]\}$, where $\{n_i, i \in [M]\}$ are the indices of the M -NN of \bar{x}_n . We introduce the following definitions:

Definition 7 (M -NN noise clusterability). *We call \tilde{D}_n satisfies M -NN noise clusterability if the M -NN of \bar{x}_n have the same noise transition matrix as x_n , i.e., $\mathbf{T}(x_n) = \mathbf{T}(x_{n_i}), \forall i \in [M]$.*

Definition 8 ((H, M) -coverage). *We call \tilde{D} satisfies (H, M) -coverage if there exist H instances $\bar{x}_{h(n)}, n \in [H]$ such that $\tilde{D} = \cup_{n=1}^H \tilde{D}_{h(n)}$, where each $\tilde{D}_{h(n)}$ satisfies M -NN noise clusterability.*

Note Definition 7 focuses on the clusterability of noise transition matrices, which is different from the clusterability of the true classes of labels. When M -NN noise clusterability holds for \bar{x}_n , the label noise in local dataset \tilde{D}_n is effectively homogeneous. If \tilde{D} further satisfies (H, M) -coverage, we can divide the training data \tilde{D} to H local sub-datasets $\tilde{D}_{h(n)}, n \in [H]$ and separately apply Algorithm 1 on each of them. The local estimates allow us to apply loss correction separately using different $\mathbf{T}(X)$ at different parts of the training data. Besides, when there is no M -NN noise clusterability, we may require knowing properly constructed sub-spaces to separate the data, with each part of them sharing similar noise rates [197, 198]. We leave more detailed discussions in

Appendix A.3.2.

2.2.5 Theoretical Guarantees

We will prove that our consensus equations are sufficient for estimating a unique \mathbf{T} , and show the advantage of our approach in terms of a better sample complexity than the anchor point approach.

2.2.5.1 Uniqueness of Solution

Before formally presenting the uniqueness guarantee, we introduce two assumptions as we will need.

Assumption 1 (Nonsingular \mathbf{T}). *The noise transition matrix is non-singular, i.e., $\text{Rank}(\mathbf{T}) = K$.*

Assumption 2 (Informative \mathbf{T}). *The diagonal elements of \mathbf{T} are dominant, i.e., $T_{ii} > T_{ij}, \forall i \in [K], j \in [K], j \neq i$.*

Assumption 1 is commonly made in the literature and ensures the effect of label noise is invertible [163]. Assumption 2 characterizes a particular permutation of row vectors in \mathbf{T} [127]. See more discussions on their feasibility in Appendix A.3.3. The uniqueness is formally stated in Theorem 1. The proof is detailed in Appendix A.2.1.

Theorem 1. *When \tilde{D} satisfies the 2-NN label clusterability and \mathbf{T} is nonsingular and informative, with a perfect knowledge of $\mathbf{c}^{[\nu]}, \nu = 1, 2, 3$, the solution of consensus equations (2.2) – (2.4) returns the true \mathbf{T} uniquely.*

Proof Sketch. The high-level idea of the proof is to connect the Hadamard products to matrix products, and prove that any linear combination of two or more rows of \mathbf{T} does not exist in \mathbf{T} .

Step I: Transform the second-order equations. By exploiting the relation between Hadamard products and matrix products, the second-order equations can be transformed to $\mathbf{T}^\top \mathbf{D}_p \mathbf{T} = \mathbf{T}_\dagger$, where \mathbf{T}_\dagger is fixed given $\mathbf{c}_r^{[2]}, \forall r \in [K]$, and \mathbf{D}_p is a diagonal matrix with \mathbf{p} as its main diagonals,

Step II: Transform the third-order equations. Following the idea in Step I, we can also transform the third-order equations to $(\mathbf{T} \circ \mathbf{T}_s) = \mathbf{T} \mathbf{T}_\dagger^{-1} \mathbf{T}_{\dagger,s}^\top, \forall s \in [K]$, where $\mathbf{T}_{\dagger,s}$ is fixed given $\mathbf{c}_{r,s}^{[3]}, \forall r, s$.

Step III: From matrices to vectors We analyze the rows \mathbf{u}^\top of \mathbf{T} and transform the equations in Step II to (e.g. $s = 0$) $\mathbf{A} \mathbf{u} = \mathbf{u} \circ \mathbf{u}$, where $\mathbf{A} = \mathbf{T}_\dagger (\mathbf{T}_\dagger^{-1})^\top$. Then we need to find the number of feasible vectors \mathbf{u} .

Step IV: Construct the $(K + 1)$ -th vector When \mathbf{T} is non-singular, we prove the $(K + 1)$ -th solution \mathbf{u}_{K+1} must be identical $\mathbf{u}_k, k \in [K]$.

Wrapping-up: Unique \mathbf{T} Step IV shows \mathbf{T} only contains K different feasible rows. The informativeness of \mathbf{T} ensures the unique order of these K rows. Thus \mathbf{T} is unique. □

Challenges Proving Theorem 1 is challenging due to: 1) The coupling effect between \mathbf{T} and \mathbf{p} makes the structure of solution \mathbf{T} unclear; 2) Naively replacing \mathbf{p} , e.g., using $\mathbf{p} = (\mathbf{T}^\top)^{-1}\mathbf{c}^{[1]}$, will introduce matrix inverse, which cannot be canceled with the Hadamard product; 3) A system of third-order equations with K^2 variables will have up to 3^{K^2} solutions and the closed-form is not explicit.

Local estimates Our next corollary 1 extends Theorem 1 to local datasets, when \mathbf{T} can be heterogeneous.

Corollary 1. *When \tilde{D} satisfies (H, M) -coverage, each $\tilde{D}_{h(n)}$ satisfies 2-NN label clusterability, and $\mathbf{T}(x_{h(n)})$ is nonsingular and informative, with a perfect knowledge of the local $\mathbf{c}^{[\nu]}$, $\nu = 1, 2, 3$, the solution of consensus equations (2.2) – (2.4) is unique and recovers $\mathbf{T}(x_{h(n)})$.*

2.2.5.2 Sample Complexity

We next show that with the estimates $\hat{\mathbf{c}}^{[1]}$, $\hat{\mathbf{c}}^{[2]}$, and $\hat{\mathbf{c}}^{[3]}$, HOC returns a reasonably well solution. Recall that, in Section 2.2.3, Step 2 requires a particular $E \subseteq E_3^*$ to guarantee the i.i.d. property of the sample $\{(\tilde{y}_n, \tilde{y}_{n_1}, \tilde{y}_{n_2}), n \in E\}$. For a tractable sample complexity, we focus on a particular dataset \tilde{D} and feature extractor \mathbf{f}_{conv} such that 1) $|E_3^*| = \Theta(N)$ and 2) $T_{ij} = \frac{1-T_{ii}}{K-1}, \forall j \neq i, i \in [N], j \in [N]$. Supposing each tuple is drawn from non-overlapping balls, condition 1) is satisfied when the number of these non-overlapping balls covering the representation space is $\Theta(N)$. See Appendix A.2.2 for a detailed example when the representations are uniformly distributed. Lemma 1 shows

the error upper bound of our estimates $\hat{\mathbf{c}}^{[\nu]}$, $\nu = 1, 2, 3$. See Appendix A.2.3 for the proof.

Lemma 1. *With probability $1 - \delta$, $\forall \nu, l$, the estimation error $|\hat{\mathbf{c}}^{[\nu]}[l] - \mathbf{c}^{[\nu]}[l]|$ is bounded at the order of $O(\sqrt{\ln(1/\delta)/N})$.*

Lemma 1 is effectively the sample complexity of estimating $|E_3^*|$ i.i.d. random variables by the sample mean. Due to assuming a uniform diagonal \mathbf{T} , we only need to consider the estimation error of \hat{T}_{ii} . For each $i \in [K]$, see the result in Theorem 2 and the proof in Appendix A.2.4.

Theorem 2. *When $T_{ii} > \frac{1 - \mathbb{P}(Y=i) + (K-1)\mathbb{P}(\tilde{Y}=i)}{K(K-1)\mathbb{P}(Y=i)}$, w.p. $1 - 2\delta$, $|\hat{T}_{ii} - T_{ii}|$ is bounded at the order of $O(\sqrt{\ln(1/\delta)/N})$.*

Theorem 2 indicates the sample complexity of our solution has the same order in terms of N compared to a standard empirical mean estimation in Lemma 1. Remark 1 shows our approach is better than using a set of anchor points in the sample complexity.

Remark 1 (Comparison). *The methods based on anchor points estimate \mathbf{T} with $N_{AC} < N$ ($N_{AC} \ll N$ in many cases) anchor points. Thus w.p. $1 - \delta$, the estimation error is at the order of $O(\sqrt{\ln(1/\delta)/N_{AC}})$.*

2.2.6 Experiments

We present experiment settings as follows.

Datasets and models HOC is evaluated on three benchmark datasets: CIFAR-10, CIFAR-100 [99] and Clothing1M [200]. For the standard training step, we use ResNet34

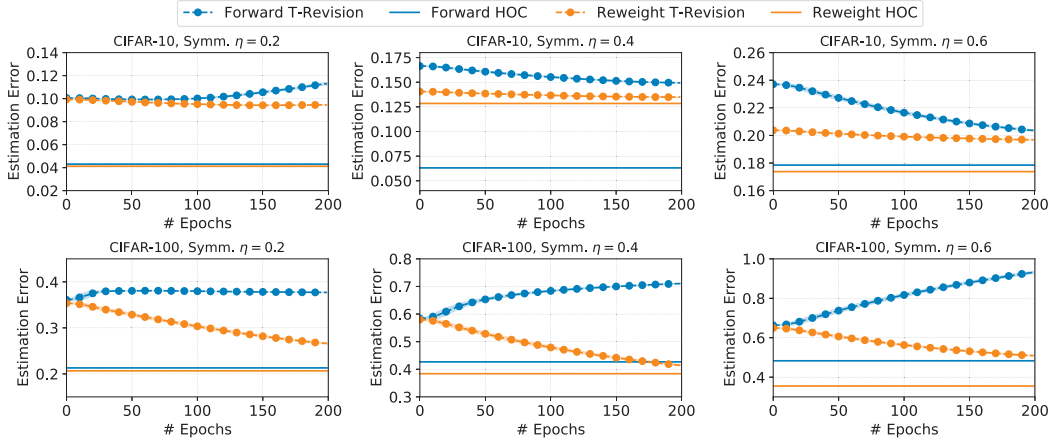


Figure 2.2: Comparison of estimation errors of \mathbf{T} given by T-Revision [199] and our HOC estimator.

Table 2.1: The best epoch (clean) test accuracy (%) with synthetic label noise.

Method	<i>Inst. CIFAR-10</i>			<i>Inst. CIFAR-100</i>		
	$\eta = 0.2$	$\eta = 0.4$	$\eta = 0.6$	$\eta = 0.2$	$\eta = 0.4$	$\eta = 0.6$
CE (Standard)	85.66±0.62	76.89±0.93	60.29±1.17	57.26±1.33	41.33±0.89	25.08±1.85
Peer Loss [124]	89.52±0.22	83.44±0.30	75.15±0.82	61.13±0.48	48.01±0.12	33.00±1.47
L_{DMI} [205]	88.67±0.70	83.65±1.13	69.82±1.72	57.36±1.18	43.06±0.97	26.13±2.39
L_q [221]	85.66±1.09	75.24±1.07	61.30±3.35	56.92±0.24	40.17±1.52	25.58±3.12
Co-teaching [63]	88.84±0.20	72.61±1.35	63.76±1.11	43.37±0.47	23.20±0.44	12.43±0.50
Co-teaching+ [212]	89.82±0.39	73.44±0.38	63.61±1.78	41.62±1.05	24.73±0.85	12.25±0.35
JoCoR [179]	88.82±0.20	71.13±1.94	63.88±2.05	44.55±0.62	23.92±0.32	13.05±1.10
Forward [146]	87.87±0.96	79.81±2.58	68.32±1.68	57.69±1.55	42.62±0.92	27.35±3.42
T-Revision [199]	90.31±0.37	84.99±0.81	72.06±3.40	58.00±0.20	40.01±0.32	40.88±7.57
HOC Global	89.71±0.51	84.62±1.02	70.67±3.38	68.82±0.26	62.29±1.11	52.96±1.85
HOC Local	90.03±0.15	85.49±0.80	77.40±0.47	67.47±0.85	61.20±1.04	49.84±1.81

for CIFAR-10 and CIFAR-100, and ResNet50 for Clothing1M. The representations come from the outputs before the final fully-connected layer of ResNet34/50. The distance between different representations is measured by the negative cosine similarity.

Noise type HOC is tested on both synthetic label noise and real-world human label noise. The synthetic label noise includes two regimes: *symmetric* noise and *instance-*

dependent noise. For both regimes, the noise rate η is the overall ratio of instances with a corrupted label in the whole dataset. The symmetric noise is generated by randomly flipping a clean label to the other possible classes w.p. η [199]. The basic idea of generating instance-dependent noise is to randomly generate one vector for each class (K vectors in total) and project each incoming feature onto these K vectors [198]. The label noise is added by jointly considering the clean label and the projection results. See Appendix A.4.1 for more details. The *real-world human noise* comes from human annotations. Particularly, for the 50,000 training images in CIFAR-10, we *re-collect* human annotations from Amazon Mechanical Turk (MTurk) in February 2020. The cost of collecting one annotation for each image is €10 per image. For the Clothing1M dataset, we train on 1 million noisy training instances reflecting real-world human noise.

2.2.6.1 Performance of Estimating T

We compare HOC with T-revision [199] following the flow: 1) Estimation \rightarrow 2) Training \rightarrow 3) Revision. For a fair comparison, we follow their training framework and parameter settings to get representations. Particularly, we obtain the same model as the one that T-revision adopts before revision. Figure 2.2 shows the results. The error is measured by the matrix $L_{1,1}$ -norm with a normalization factor K , i.e. $\|\hat{T} - T\|_{1,1}/K$. Notation *Forward* indicates the method using the forward corrected loss [146]. Notation *reweight* indicates the method using the reweighted loss [119]. Symmetric noise is applied. As illustrated in Figure 2.2, compared with the dynamical revision adopted in T-revision, HOC does not need to change or adapt in different epochs and still achieves

Table 2.2: The best epoch test accuracy (%) with human noise.

Method	Clothing1M	Human CIFAR-10
CE (standard)	68.94	83.50
CORES ² [30]	73.24	89.98
L_{DMI} [205]	72.46	86.33
Co-teaching [63]	69.21	90.39
JoCoR [179]	70.30	90.10
Forward [146]	70.83	86.82
PTD-R-V[198]	71.67	85.92
HOC [230]	73.39	90.62

lower estimation errors no matter the model is trained with forward corrected loss or reweighted loss.

2.2.6.2 Performance of Classification Accuracy

To test the classification performance, we adopt the flow: 1) Pre-training \rightarrow 2) Global Training \rightarrow 3) Local Training. Our HOC estimator is applied once at the beginning of each above step. In Stage-1, we load the standard ResNet50 model pre-trained on ImageNet to obtain basic representations. At the beginning of Stage-2 and Stage-3, we use the representations given by the current model. All experiments are repeated three times. *HOC Global* only employs one global \mathbf{T} with $G = 50$ and $|E| = 15k$ as inputs of Algorithm 8. *HOC Local* uses 300 local matrices (250-NN noise clusterability, $G = 30$, $|E| = 100$) for CIFAR-10 and 5 local matrices (10k-NN noise clusterability, $G = 30$, $|E| = 5k$) for CIFAR-100. Our unconstrained transformation provides much better convergence such that running HOC Local on CIFAR will at most double the running time of a standard training with CE. See more details in Appendix A.4. Without

sophisticated learning techniques, we simply feed the estimated transition matrices given by HOC into *forward loss correction* [146]. We report the performance on synthetic instance-dependent label noise in Table 2.1 and real-world human-level label noise in Table 2.2. Comparing with these baselines (with similar data augmentations), both global estimates and local estimates given by HOC achieve satisfying performance, and the local estimates indeed provide sufficient performance improvement on CIFAR-10. When there are 100 classes, \mathbf{T} contains $10k$ variables thus local estimates with only $10k$ instances may not be accurate, which leads to a slight performance drop in HOC Local on CIFAR-100 (but it still outperforms other methods).

Real human-level noise On CIFAR-10 with our self-collected human-level noisy labels, HOC achieves a 0.097 estimation error in the global \mathbf{T} and a 0.110 ± 0.027 error in estimating 300 local transition matrices. See more details in Appendix A.4.3.

2.2.6.3 Feasibility of 2-NN label clusterability

We show the ratio of feasible 2-NN tuples in Table 2.3, where $|E| = 5k$ indicates that we sample $5k$ examples from the whole dataset in each round and average over 10 rounds. Method $|E| = 50k$ indicates that we check the feasibility of all 2-NN tuples. One 2-NN tuple is called feasible if \bar{x}_n and its 2-NN belong to the same true class. The feature extractors are obtained from overfitting CIFAR-10/100 with different noise levels. For example, *CIFAR-10 Inst. $\eta = 0.2$* indicates that we use the standard CE loss to train ResNet34 on CIFAR-10 with 20% instance-dependent label noise. The convolution

Table 2.3: The ratio of feasible 2-NN tuples with different feature extractors.

Feature Extractor	<i>CIFAR-10</i>		<i>CIFAR-100</i>	
	$ E = 5k$	$ E = 50k$	$ E = 5k$	$ E = 50k$
<i>Clean</i>	99.99	99.99	99.88	99.90
<i>Inst. $\eta = 0.2$</i>	87.88	89.06	82.82	84.33
<i>Inst. $\eta = 0.4$</i>	78.15	79.85	64.88	68.31

layers when the model approaches nearly 100% training accuracy are selected as the feature extractor $f_{\text{conv}}(X)$. Table 2.3 shows, with a standard feature extractor, there are more than 2/3 of the feasible 2-NN tuples in most cases. Besides, reducing the sample size from $50k$ to $5k$ will not substantially reduce the ratio of feasible 2-NN tuples.

2.2.7 Takeaways

The takeaways are summarized as follows:

- We have proposed a new and flexible estimator of the noise transition matrix relying on the first-, second-, and third-order consensus checking among an example and its' 2-NN's noisy labels.
- We have proved that using up to third-order consensus is sufficient to identify the true noise transition matrix uniquely.
- Extensive empirical studies on CIFAR-10/100 datasets with synthetic noisy labels, the Clothing1M dataset with real-world human noise, and the CIFAR-10 dataset with our self-collected human annotations demonstrate the advantage of our estimator.
- Open-source contribution and flexible extension: a generically applicable and light tool for fast estimation of the noise transition matrix.

- The code is available at <https://github.com/UCSC-REAL/HOC>.

2.3 Estimate T for Tasks with Lower-Quality Features

As mentioned in Section 2.2, the distance in Definition 1 between two features \mathbf{x} and \mathbf{x}' can be measured by the negative cosine similarity, e.g., $1 - \text{Sim}(\mathbf{x}, \mathbf{x}')$, where $\text{Sim}(\mathbf{x}, \mathbf{x}')$ could be the cosine similarity. It has been proved in Section 2.2 that the 2-NN label clusterability is sufficient for uniquely getting the true T . However, it is likely that the clusterability is not sufficiently satisfied for lower-quality features. In this section, we first illustrate the possible failures of existing methods when the feature quality is low (Section 2.3.1), then propose an information-theoretic approach (Section 2.3.2) and prove the related theoretical guarantees (Section 2.3.3). The effectiveness of the proposed method is also evaluated by experiments in Section 2.3.4.

2.3.1 Failures on Lower-Quality Features

The approaches to estimating T introduced in Section 2.2 are demonstrated to perform well on some image classification datasets, such as MNIST [103] and CIFAR [99], which usually enjoy better representation learning tools [28, 168] that would return clusterable features, as compared to text sequence and tabular data. When facing other tasks, high-quality features may not always be available. In this subsection, we explore how these existing methods fare on other datasets, possibly with lower-quality features.

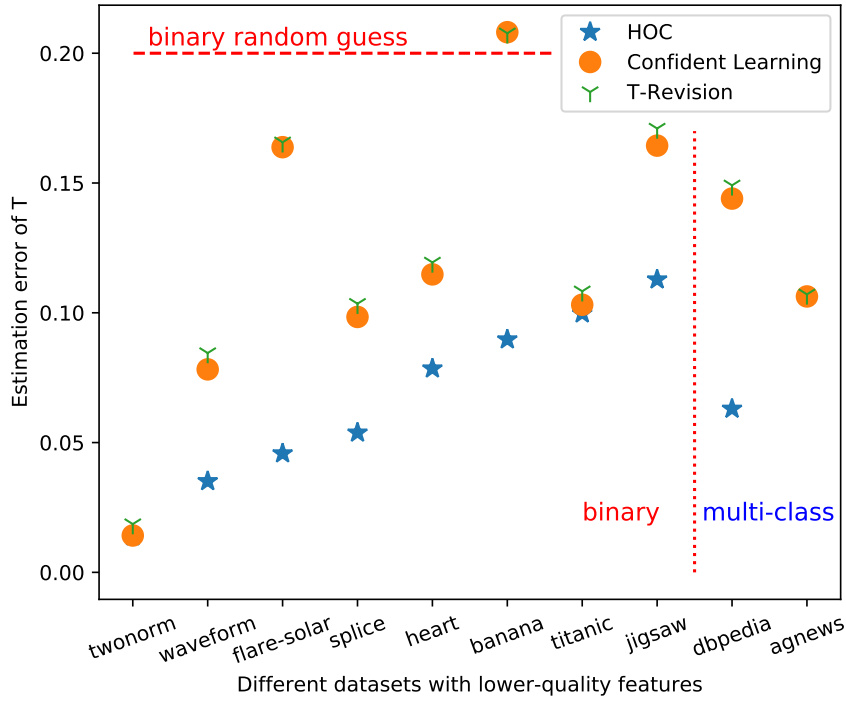


Figure 2.3: Existing methods may suffer from failures.

Observations In Figure 2.3, we implement two learning-based methods built on anchor points (T-revision [199]) or confident points (Confident Learning [141]), and one training-free method based on the clusterability (HOC [230]). **Red** horizontal dashed line shows the error of random guessing T in binary classifications. Tasks on the left side of the dotted line are binary. The average noise rate, i.e., $\sum_{i \in [K]} (1 - T_{ii}) / K$, is around 0.3. Figure 2.3 shows the estimation error of these three methods on most of the datasets are around or larger than 0.1, and some methods may even approximate to 0.2. Note **an error of 0.2 is excessive** for binary classification when the noise rate is 0.3.

For example, when $T_{11} = T_{22} = 0.3$, a *random guess* of \mathbf{T} , i.e., $T_{ij} = 0.5, \forall i, j$, has an error of exactly 0.2. Recall an error of 0.05 for binary classification is worse than the same error for a 10-class one as explained in Introduction. Compared with an estimation error of ≈ 0.05 on CIFAR-10 with a similar noise rate as reported in these baselines, Figure 2.3 shows a serious performance drop: an error of 0.05 for 83% of tests and an error of 0.1 for 57% of tests. Therefore, it is crucial to design an estimator which is also robust on datasets with lower-quality features.

Discussions Before diving into a concrete solution to lower-quality features, we discuss the advantages and disadvantages of both existing lines of work. On one hand, the *learning-based methods* [141, 199] could take full advantage of deep neural networks. During supervised training, different parts of features are weighted differently, thus the informative parts could weigh more than the less informative ones. The optimal weight combinations could induce a model that accurately fits the data distribution, which further help estimate \mathbf{T} . On the other hand, due to various factors such as the model capacity, the quality of features, the number of instances, and the setting of hyperparameters, the learning-based models are often infeasible to converge to the global optimum in practice. For example, with the existence of label noise, deep neural networks (DNN) tend to be overconfident [182] and memorize wrong feature-label patterns [31, 121, 180]. When unintended memorization occurs, the weights for combining different parts will be non-optimal and some uninformative parts may be mis-specified with high weights. Alternatively, the *training-free method* [230] will not

be affected by the wrong memorization since it is a fully-statistical solution without any training procedures. Nevertheless, the problem of not employing training is also severe: blindly treating different parts of features equally important may cause failures. The above observations and discussions motivate us to find a solution that *compromises between the learning-based and the training-free approaches*.

2.3.2 An Information-Theoretic Approach

We propose an information-theoretic approach to distinguish the importance of different features. To avoid complicated hyperparameters tuning and make it a light tool for more general applications, the solution is built on HOC. See more detailed rationale in Appendix A.6.1.

We now briefly review HOC [230]. Algorithm 2 summarizes the key steps, where the high-level idea is that, when 2-NN label clusterability holds, the frequency of consensus patterns of the three grouped noisy labels $\tilde{y}_n, \tilde{y}_{n_1}, \tilde{y}_{n_2}$ encodes \mathbf{T} . Note It is shown later in [122] that three noisy labels are *necessary and sufficient* to identify \mathbf{T} . For instance, a triplet $\tilde{y}_n = 0, \tilde{y}_{n_1} = 0, \tilde{y}_{n_2} = 1$ will add 1 to the count of the consensus pattern $(0, 0, 1)$. With the estimated frequency of patterns $\{(\tilde{y}_n, \tilde{y}_{n_1}, \tilde{y}_{n_2}), \forall n\}$, we can simply solve equations by gradient descents. Therefore, in Algorithm 2, the 2-NN label clusterability is critical, which depends heavily on calculating the distance or similarity between features.

Algorithm 2 Key Steps of HOC

0: **Input:** Noisy dataset: $\tilde{D} = \{(\mathbf{x}_n, \tilde{y}_n)\}_{n \in [N]}$.

// Find 2-NN. $\text{Sim}(\mathbf{x}, \mathbf{x}') \rightarrow \text{Sim}_W(\mathbf{z}, \mathbf{z}')$ in our approach.

1: With $1 - \text{Sim}(\mathbf{x}, \mathbf{x}')$ as the distance metric:

$\{(\tilde{y}_n, \tilde{y}_{n_1}, \tilde{y}_{n_2}), \forall n\} \leftarrow \text{Get2NN}(\tilde{D});$

// Count first-, second, and third-order consensus patterns:

2: $(\hat{\mathbf{c}}^{[1]}, \hat{\mathbf{c}}^{[2]}, \hat{\mathbf{c}}^{[3]}) \leftarrow \text{CountFreq}(\{(\tilde{y}_n, \tilde{y}_{n_1}, \tilde{y}_{n_2}), \forall n\})$

// Solve equations

3: Find \mathbf{T} such that match the counts $(\hat{\mathbf{c}}^{[1]}, \hat{\mathbf{c}}^{[2]}, \hat{\mathbf{c}}^{[3]})$.

Overview of our approach The main idea is to decouple features (Step 1) and down-weight the less informative parts (Step 2) when measuring the distances by HOC, which is summarized in Algorithm 3. Firstly, we motivate the necessity of down-weighting less informative parts in Section 2.3.2.1. Their weights are controlled by a matrix \mathbf{W} , which is absorbed into the calculation of soft cosine similarity (Definition 9). A tractable proxy of \mathbf{W} is introduced in Section 2.3.2.2 and detailed in the remainder of this section.

2.3.2.1 Vanilla Similarity Measures Are Not Sufficient

Our following analyses focus on the cosine similarity as originally implemented by HOC [230]. The larger cosine similarity implies the smaller distances of features. We first analyze possible problems in evaluating k -NN with the vanilla (hard) cosine similarity.

Consider the cosine similarity of two feature vectors \mathbf{x} and \mathbf{x}' , which is denoted

Algorithm 3 Our Information-Theoretic Approach

0: **Input:** Noisy dataset: $\tilde{D} = \{(\mathbf{x}_n, \tilde{y}_n)\}_{n \in [N]}$.

// Step 1: Remove correlation (Section 2.3.2.3)

1: Transform $\mathbf{x} \sim \mathbf{X}$ to $\mathbf{z} \sim \mathbf{Z}$ by Eqn. (2.12);

// Step 2: Estimate the weight matrix \mathbf{W} (Section 2.3.2.4)

2: With only noisy labels:

Diagonal elements: $\hat{W}_{\mu\mu} = I_f(Z; \tilde{Y})$ by Eqn. (2.13)

Off-diagonal elements: $\hat{W}_{\mu\mu'} = 0, \forall \mu \neq \mu'$;

// Step 3: Estimate the noise transition matrix

3: Apply HOC [230] with soft cosine similarity $\text{Sim}_{\mathbf{W}}(\mathbf{z}, \mathbf{z}')$ defined in Eqn. (2.11).

4: **Output:** The estimated noise transition matrix $\hat{\mathbf{T}}$.

by $\text{Sim}(\mathbf{x}, \mathbf{x}') = \frac{\mathbf{x}^\top \mathbf{x}'}{\|\mathbf{x}\|_2 \|\mathbf{x}'\|_2}$, where $\|\cdot\|_2$ denotes the vector ℓ_2 norm. The above measure inherently assumes different elements of \mathbf{x} are 1) *equally important* and 2) *uncorrelated* to each other, thus may underestimate the true similarity between imperfect feature vectors. To capture more information, we incorporate a change-of-basis matrix $\sqrt{\mathbf{W}}$ to obtain a soft cosine measure defined as follows.

Definition 9 (Soft cosine similarity [158]).

$$\text{Sim}_{\mathbf{W}}(\mathbf{x}, \mathbf{x}') = \frac{(\sqrt{\mathbf{W}}\mathbf{x})^\top (\sqrt{\mathbf{W}}\mathbf{x}')}{\|\sqrt{\mathbf{W}}\mathbf{x}\|_2 \|\sqrt{\mathbf{W}}\mathbf{x}'\|_2}. \quad (2.11)$$

Hereby, the symmetric matrix \mathbf{W} encodes the pairwise similarity between features. Note that the soft cosine similarity measure $\text{Sim}_{\mathbf{W}}(\mathbf{x}, \mathbf{x}')$ recovers the (hard) cosine similarity when $\mathbf{W} = \mathbf{I}$, where \mathbf{I} denotes a $K \times K$ identity matrix. In practice,

the true and unknown \mathbf{W} may be very different from \mathbf{I} . Thus simply letting $\mathbf{W} = \mathbf{I}$ may cause severe problems in using clusterability. For example, when $K = 2$, consider three instances $(\mathbf{x}_1, y_1) = ([1, 0, 1]^\top, 1)$, $(\mathbf{x}_2, y_2) = ([0, 1, 0]^\top, 2)$, and $(\mathbf{x}_3, y_3) = ([0.8, 1, 0.7]^\top, y)$.

Based on 1-NN label clusterability, we infer label y following the rule below:

$$y = \begin{cases} 1 & \text{if } \text{Sim}(\mathbf{x}_1, \mathbf{x}_3) > \text{Sim}(\mathbf{x}_2, \mathbf{x}_3); \\ 2 & \text{if } \text{Sim}(\mathbf{x}_1, \mathbf{x}_3) \leq \text{Sim}(\mathbf{x}_2, \mathbf{x}_3). \end{cases}$$

Consider the following three \mathbf{W} s: $\mathbf{W}_1 = \mathbf{I}$,

$$\mathbf{W}_2 = \begin{pmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & \mathbf{0.1} \end{pmatrix}, \quad \mathbf{W}_3 = \begin{pmatrix} 1.0 & -0.2 & -0.5 \\ -0.2 & 1.0 & 0.5 \\ -0.5 & 0.5 & 1.0 \end{pmatrix}.$$

Example 1: \mathbf{W}_1 (uncorrelated and equally important) The following hard cosine similarity shows:

$$\text{Sim}_{\mathbf{W}_1}(\mathbf{x}_1, \mathbf{x}_3) \approx 0.73 > \text{Sim}_{\mathbf{W}_1}(\mathbf{x}_2, \mathbf{x}_3) \approx 0.69 \Rightarrow y = 1.$$

Example 2: \mathbf{W}_2 (not equally important) The following soft cosine similarity shows:

$$\text{Sim}_{\mathbf{W}_2}(\mathbf{x}_1, \mathbf{x}_3) \approx 0.64 < \text{Sim}_{\mathbf{W}_2}(\mathbf{x}_2, \mathbf{x}_3) \approx 0.77 \Rightarrow y = 2,$$

which is *different* from the inferred y in Example 1. If \mathbf{W}_2 defines the true clusterability, this example shows that simply using $\mathbf{W} = \mathbf{I}$ fails to capture the diagonal values of \mathbf{W} (the weights of x_μ , $\mu \in [d]$) and violates the clusterability.

Example 3: \mathbf{W}_3 (correlated) The following soft cosine similarity shows:

$$\text{Sim}_{\mathbf{W}_3}(\mathbf{x}_1, \mathbf{x}_3) \approx 0.75 < \text{Sim}_{\mathbf{W}_3}(\mathbf{x}_2, \mathbf{x}_3) \approx 0.85 \Rightarrow y = 2,$$

which is *different* from the inferred y in Example 1. If \mathbf{W}_3 is true, this example shows that simply using $\mathbf{W} = \mathbf{I}$ fails to capture the off-diagonal values of \mathbf{W} (the correlations between x_μ) and violates the clusterability.

The above three examples exemplify that the less informative parts of features may damage the clusterability, where the definition of each “part” depends on both the diagonal elements and off-diagonal elements of \mathbf{W} . We propose an information-theoretic approach to construct a proxy of \mathbf{W} .

2.3.2.2 Proxy of \mathbf{W}

Noting \mathbf{W} is a square matrix of order d , it requires $\mathcal{O}(d^2)$ operations to estimate all elements. Each operation incurs an estimation error, and the accumulated errors may not be bounded. We propose to find a proxy of \mathbf{W} . Intuitively, we expect the following properties [14]:

- **Symmetric:** $\forall \mu, \nu \in [d], W_{\mu\nu} = W_{\nu\mu}$.
- **Information monotone:** For every two feature variables X_μ, X_ν , if X_μ is less informative with respect to Y than X_ν , then X_μ will be less important than X_ν , i.e., $I_f(X_\mu; Y) \leq I_f(X_\nu; Y) \Rightarrow W_{\mu\mu} \leq W_{\nu\nu}, \mu, \nu \in [d]$, where $I_f(X_\mu; Y)$ measures the f -MI (f -Mutual Information) [36] between X_μ and Y .
- **Correlation monotone:** Given two feature variables X_μ, X_ν , for any other feature

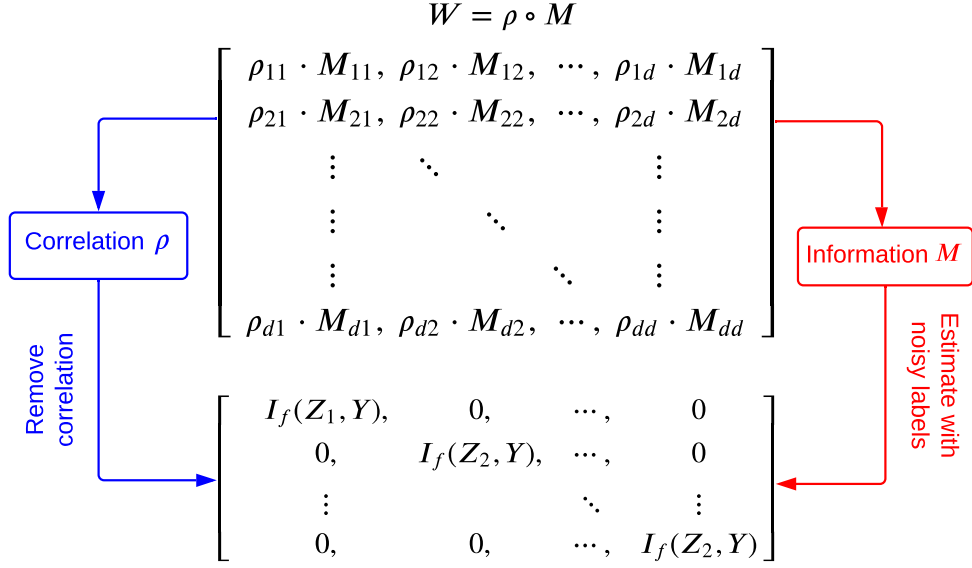


Figure 2.4: Illustration of the proxy of \mathbf{W} .

variable $X_{\nu'}$ ($X_\mu, X_\nu, X_{\nu'}$ are equally informative), if X_μ is less correlated to X_ν than $X_{\nu'}$, X_ν will have a lower weight than $X_{\nu'}$ in measuring the similarity with X_ν :

$$\rho(X_\mu, X_\nu) \leq \rho(X_\mu, X_{\nu'}) \Rightarrow W_{\mu\nu} \leq W_{\mu\nu'},$$

where $\rho(X_\mu, X_\nu)$ is the correlation between random variables X_μ and X_ν .

The above properties suggest us to decompose the elements of matrix \mathbf{W} as the product of the informativeness w.r.t Y and the correlation between features as illustrated in Figure 2.4. In another word, we can construct the matrix $\mathbf{W} = \boldsymbol{\rho} \circ \mathbf{M}$, where \circ denotes the Hadamard product of two matrices, $\boldsymbol{\rho}$ is the correlation matrix with $\rho_{\mu\nu} = \rho(X_\mu, X_\nu)$, \mathbf{M} includes the f -MI with $M_{\mu\nu} = \sqrt{I_f(X_\mu, Y)I_f(X_\nu, Y)}$. Directly estimating $\boldsymbol{\rho}$ might not be computation-efficient. If we can transform \mathbf{X} to make $\boldsymbol{\rho} = \mathbf{I}$, the off-diagonal entries of \mathbf{W} will become 0 thus no longer need to be estimated. This

observation motivates us to firstly transform \mathbf{X} to a non-correlated form to achieve $\boldsymbol{\rho} = \mathbf{I}$ (Section 2.3.2.3), then estimate only the diagonal elements by f -MI (Section 2.3.2.4).

2.3.2.3 Remove correlation

For ease of notations, we require that \mathbf{X} is zero-mean, i.e., $\mathbb{E}[\mathbf{X}] = \mathbf{0}$, where $\mathbf{0}$ is a $d \times 1$ column vector with all elements being 0. Inspired by the principle component analysis (PCA) [192], we adopt $\boldsymbol{\Lambda}^{-1/2}\mathbf{P}^\top$ as the matrix to remove the correlation between $x_\mu, x_\nu, \mu \neq \nu$, where $\mathbf{P}\boldsymbol{\Lambda}\mathbf{P}^\top = \mathbb{E}[\mathbf{X}\mathbf{X}^\top]$, \mathbf{P} is an orthogonal matrix whose columns are eigenvectors of $\mathbb{E}[\mathbf{X}\mathbf{X}^\top]$, $\boldsymbol{\Lambda}$ is a diagonal matrix where diagonal values are eigen values and off-diagonal values are 0. Let

$$\mathbf{Z} = \boldsymbol{\Lambda}^{-1/2}\mathbf{P}^\top \mathbf{X} = [Z_1, \dots, Z_d]^\top, \quad (2.12)$$

where $\mathbb{E}[\mathbf{Z}] = \boldsymbol{\Lambda}^{-1/2}\mathbf{P}^\top \mathbb{E}[\mathbf{X}] = \mathbf{0}$. We have

$$\mathbb{E}[\mathbf{Z}\mathbf{Z}^\top] = \boldsymbol{\Lambda}^{-1/2}\mathbf{P}^\top \mathbb{E}[\mathbf{X}\mathbf{X}^\top] \mathbf{P}\boldsymbol{\Lambda}^{-1/2} = \mathbf{I}.$$

Therefore, after transforming \mathbf{X} to \mathbf{Z} by $\boldsymbol{\Lambda}^{-1/2}\mathbf{P}^\top$, we can split \mathbf{X} into d uncorrelated parts such that $\mathbb{E}[Z_\mu Z_\nu]$ satisfies

$$\mathbb{E}[Z_\mu Z_\nu] = \mathbb{E}[(\mathbf{Z}\mathbf{Z}^\top)_{\mu,\nu}] = \begin{cases} 0 & \text{if } \mu \neq \nu, \\ 1 & \text{if } \mu = \nu. \end{cases}$$

Noting (zero-mean) $\rho(Z_\mu, Z_\nu) = \frac{\mathbb{E}[Z_\mu Z_\nu]}{\sqrt{\mathbb{E}[Z_\mu^2]}\sqrt{\mathbb{E}[Z_\nu^2]}}$, we know $\boldsymbol{\rho} = \mathbf{I}$ after this transformation.

2.3.2.4 Estimate $I_f(Z; Y)$ With Only Noisy Labels

After transforming \mathbf{X} to \mathbf{Z} , the off-diagonal elements of the proxy of \mathbf{W} are expected to be 0. Thus we only need to estimate the diagonal elements by $W_{\mu\mu} := I_f(Z_\mu; Y)$.

Define an f -MI metric with f -divergence $I_f(Z; Y) := D_f(Z \oplus Y; Z \otimes Y)$, where $D_f(P||Q)$ measures the f -divergence between two distributions P and Q with probability density function p and q :

$$D_f(P||Q) = \int_{v \in \mathcal{V}} q(v) f\left(\frac{p(v)}{q(v)}\right) dv.$$

Note the variable v in the domain \mathcal{V} is a realization of random variable $V = (Z, Y)$, which follows either distribution P or distribution Q depending on where it is used. In our setting, P and Q are the joint distribution $P := Z \oplus Y = \mathbb{P}(Z = z, Y = y)$ and the marginal product distribution $Q := Z \otimes Y = \mathbb{P}(Z = z) \cdot \mathbb{P}(Y = y)$, respectively. There are lots of choices of f . Specially, when $f(v) = v \log v$, the f -divergence becomes the celebrated KL divergence and I_f is exactly the mutual information.

Alternatively, we can calculate the f -divergence using the variational form [145, 186]. Denote the variational difference between P and Q by

$$\text{VD}_f(g) = \mathbb{E}_{V \sim P}[g(V)] - \mathbb{E}_{V \sim Q}[f^*(g(V))], \forall g,$$

where $g : \mathcal{V} \rightarrow \text{domain}(f^*)$ is a variational function, and f^* is the conjugate function of $f(\cdot)$. We instantiate some prominent variational-conjugate (g^*, f^*) pairs in

Appendix A.5.1 (Table A.1). The f -MI is calculated by

$$I_f(Z; Y) = D_f(P||Q) = \text{VD}_f(g^*) = \sup_g \text{VD}_f(g).$$

However, the above calculation, built on the clean distributions, will be intractable when we can only access the noisy data distribution. Let $\tilde{P} := Z \oplus \tilde{Y}$ and $\tilde{Q} := Z \otimes \tilde{Y}$. One tractable approach is to calculate $D_f(\tilde{P}||\tilde{Q})$ following

$$I_f(Z; \tilde{Y}) = D_f(\tilde{P}||\tilde{Q}) = \widetilde{\text{VD}}_f(\tilde{g}^*) = \sup_g \widetilde{\text{VD}}_f(g), \quad (2.13)$$

where $\widetilde{\text{VD}}_f(g) = \mathbb{E}_{\tilde{V} \sim \tilde{P}}[g(\tilde{V})] - \mathbb{E}_{\tilde{V} \sim \tilde{Q}}[f^*(g(\tilde{V}))], \forall g$. Generally, there will be a gap between our calculated $I_f(Z; \tilde{Y})$ and the real $I_f(Z; Y)$. We defer detailed analyses of the gap to Section 2.3.3.

2.3.3 Theoretical Guarantees

The quality of our \mathbf{T} estimator relies on the following steps:

- (a) Noise-resistant estimates of f -MI using noisy labels;
- (b) Accurate estimates of clean f -MI;
- (c) Down-weighting less informative features with f -MI;
- (d) Robust distance/similarity calculation;
- (e) Satisfying clusterability (Definition 1);
- (f) Accurate estimates of the noise transition matrix \mathbf{T} .

The most critical step in the above chain is (a)→(b), which is the key ingredient to Step (c). This step will be the focus of the theoretical results in this section. Steps (c)→(e) are explained in Section 2.3.2. Steps (e)→(f) is guaranteed by HOC [230].

Based on our intuition for constructing the proxy of \mathbf{W} that the less informative parts should be assigned with lower weights, the order between two parts with different informativeness is crucial. Thus in this section, we study whether the noisy f -MI calculated using noisy labels, i.e., $I_f(Z; \tilde{Y})$, preserves the order of the clean f -MI $I_f(Z; Y)$. Note the order-preservation property distinguishes from the robustness of f -MI between optimal classifier prediction $h^*(X)$ and noisy label \tilde{Y} by [186] since 1) Z does not have class-specific meaning; 2) Z is not optimizable and its ranking is concerned, while $h^*(X)$ is only a special (optimal) case of Z . All proofs are deferred to Appendix D.1. We define ϵ -order-preserving as follows.

Definition 10 (ϵ -Order-Preserving Under Label Noise). *$I_f(Z; \tilde{Y})$ is called ϵ -order-preserving under label noise if $\forall \mu \in [d], \nu \in [d]$, given $|I_f(Z_\mu; \tilde{Y}) - I_f(Z_\nu; \tilde{Y})| > \epsilon$, we have*

$$\text{sign}[I_f(Z_\mu; \tilde{Y}) - I_f(Z_\nu; \tilde{Y})] = \text{sign}[I_f(Z_\mu; Y) - I_f(Z_\nu; Y)].$$

The smaller ϵ is, the stricter the requirement is. The following analyses focus on the binary classification. Define $e_1 := \mathbb{P}(\tilde{Y} = 2|Y = 1)$, $e_2 := \mathbb{P}(\tilde{Y} = 1|Y = 2)$. To show an f -MI metric is ϵ -order-preserving under label noise, we need to study how $\widetilde{\text{VD}}_f(\tilde{g}^*)$ differs from the order of $\text{VD}_f(g^*)$.

2.3.3.1 Total-Variation is 0-Order-Preserving

When $f(v) = \frac{1}{2}|v - 1|$, we get the Total-Variation (TV). To analyze the order-preserving property of TV, we first build the relationship between $\widetilde{\text{VD}}_f(g)$ and $\text{VD}_f(g)$, $\forall g$ by the following lemma:

Lemma 2 (Linear relationship [186]).

$$\widetilde{\text{VD}}_{\text{TV}}(g) = (1 - e_1 - e_2)\text{VD}_{\text{TV}}(g), \forall g.$$

Lemma 2 shows that there is a linear relationship between $\widetilde{\text{VD}}_{\text{TV}}(g)$ and $\text{VD}_{\text{TV}}(g)$. The constant only depends on the noise rates. With this lemma, we only need to study the difference between $\text{VD}_{\text{TV}}(\tilde{g}^*)$ and $\text{VD}_{\text{TV}}(g^*)$, which is summarized in the following lemma:

Lemma 3. *When $e_1 + e_2 < 1$, $\text{VD}_{\text{TV}}(\tilde{g}^*) = \text{VD}_{\text{TV}}(g^*)$.*

Note the condition $e_1 + e_2 < 1$ indicates the label noise is not too large to be dominant [124, 139, 123]. With Lemma 2 and Lemma 3, we can conclude that TV is 0-order-preserving.

Theorem 3. *When $e_1 + e_2 < 1$, total-variation is 0-order-preserving under class-dependent label noise.*

Theorem 3 shows the total-variation-based mutual information under class-dependent label noise preserves the order of the original clean results.

2.3.3.2 KL Divergence is ϵ -Order-Preserving

Unfortunately, Lemma 2 and Lemma 3 do not hold for KL divergence. Recall the f -MI is exactly the standard mutual information when KL divergence is adopted. Denote by $H(\cdot)$ the entropy. We can start from the definition of mutual information

$I(Z, \tilde{Y}) = H(Z) + H(\tilde{Y}) - H(Z, \tilde{Y})$ and decouple the effect of noisy labels as:

$$I(Z, \tilde{Y}) = (1 - e_1 - e_2) \cdot [I(Z, Y) - H(Y)] + H(\tilde{Y}) \\ + \int_z \Delta_{\text{Bias}}(\beta_z, e_1, e_2) dz,$$

where β_z is a function of z , and $\Delta_{\text{Bias}}(\beta_z, e_1, e_2)$ is the bias caused by label noise specified in Eqn. (A.10). We then find the lower and upper bounds for $\Delta_{\text{Bias}}(\beta_z, e_1, e_2)$ and summarize the result as follows:

Theorem 4. *Assume $e_1 = \delta e_2$, $\delta \in [0, 1]$ and $e_1 + e_2 < 1$. KL divergence (mutual information) is ϵ -order-preserving under class-dependent label noise, where*

$$\epsilon = e_1 [\delta \log \delta - (1 + \delta) \log(1 + \delta)] + H(e_1),$$

and $H(e_1) := -e_1 \log e_1 - (1 - e_1) \log(1 - e_1)$.

For the symmetric label noise, we have:

Corollary 2. *When $e_1 = e_2 < 0.5$, KL divergence (mutual information) is $[H(e_1) - 2e_1 \log 2]$ -order-preserving under class-dependent label noise.*

We evaluate the bound in the following remark.

Remark 2. *The value of ϵ is illustrated in Figure 2.5, where color indicates the value of ϵ . The median (50%) and the 90-th percentile of ϵ are shown in the title of each plot. $\log_2(x)$ is applied for calculating mutual information. The left figure shows the worst-case bound in Theorem 4, and the right figure shows the case when $\frac{\mathbb{P}(Y=1|Z=z)}{\mathbb{P}(Y=2|Z=z)} \in [\frac{1}{5}, 5]$. This is a more practical case for lower-quality features since a single feature variable cannot*

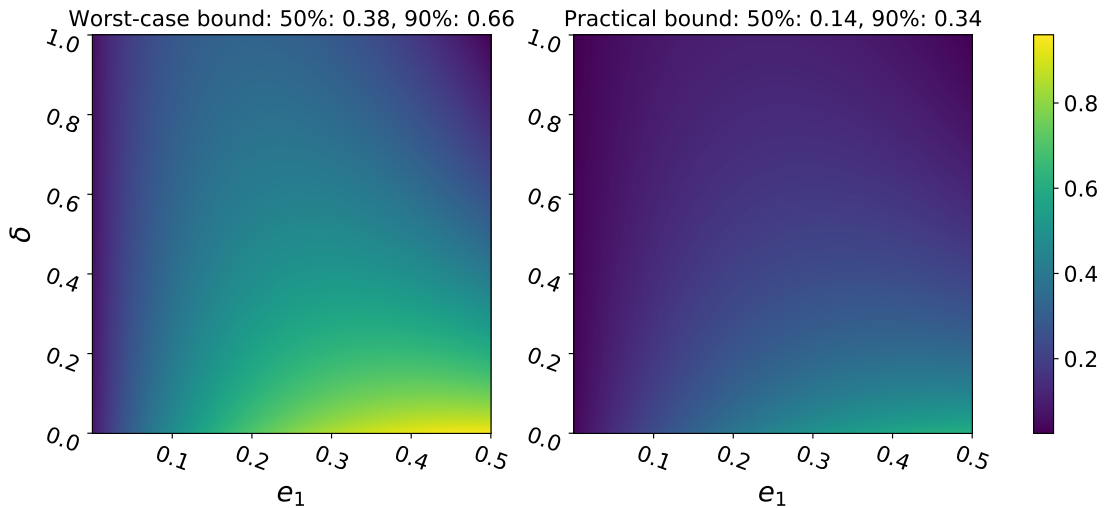


Figure 2.5: Illustration of the worst-case bound and a more practical bound for ϵ with different $e_1, \delta := e_2/e_1$.

infer the clean label with high confidence. Noting the \log_2 -based mutual information is within range $[0, 1]$ and $\epsilon \leq 0.34$ happens in 90% of the cases, it is reasonable to believe the mutual information has a good ϵ -order-preserving ability.

2.3.4 Evaluations

We evaluate our approaches on datasets with possibly lower-quality features in this section. The evaluation metric is the *average total variation* between the true \mathbf{T} and the estimated $\hat{\mathbf{T}}$ [217] as in Section 1.2.2.1.

Baselines We mainly compare our methods with three baselines: T-revision (T-REV) [199], Confident Learning (CL) [141], and the clusterability-based approach [230] (HOC).

Our approach To evaluate each component of our approach, we test four variants: OURS- \mathbf{X} -KL, OURS- \mathbf{X} -TV, OURS- \mathbf{A} -KL, and OURS- \mathbf{A} -TV. Prefix OURS- \mathbf{X} indicates that we directly use the original input features and substitute the soft cosine similarity for the original hard one employed by HOC. This setting checks the performance of ignoring correlations among different feature variables. Prefix OURS- \mathbf{A} indicates that we firstly transform \mathbf{X} to \mathbf{A} as Section 2.3.2.2 then apply soft cosine similarity. Suffixes -KL and -TV denote using the f -mutual information when $f(v) = v \log(v)$ and $f(v) = \frac{1}{2}|v - 1|$, respectively. The matrix \mathbf{W} for soft cosine similarity is: $W_{\mu\mu} = \phi(I_f(X_\mu, \tilde{Y}))$, $W_{\mu\nu} = 0, \forall \nu \neq \mu$, where $\phi(x)$ is an order-preserving activation function. In our evaluations, we set $\phi(x) = [x]_0^1$ for tabular benchmarks and $\phi(x) = [\log(x)]_0^1$ for natural language benchmarks, where $[x]_0^1$ represents normalizing x to range $(0, 1]$.

Datasets and models We examine our approaches on two different application domains other than images: the tabular benchmarks including 7 tabular datasets from the UCI machine learning repository [44] and 4 natural language benchmarks including AG’s news [218], DBpedia [7], Yelp-5 [211], and Jigsaw [90]. We use the raw features for tabular benchmarks. Following the same preprocessing procedure as [173], we use a pre-trained BERT model [40] to extract 768-dimensional embedding vectors for natural language benchmarks.

Noise type We synthesize the noisy data distribution by injecting class-dependent noise. Particularly, on tabular benchmarks, we test both the symmetric and the asymmetric label noise in Table 2.4. On natural language benchmarks, we randomly generate the

diagonal-dominant label noise following the Dirichlet distribution. Particularly, suppose the average noise rate is e . For each row of \mathbf{T} , We randomly sample a diagonal element following $T_{ii} = e + \text{Unif}(-0.05, 0.05)$ and set the off-diagonal elements following $\text{Dir}(\mathbf{1})$, where $\text{Unif}(a, b)$ denotes the uniform distribution bounded by (a, b) , $\text{Dir}(\mathbf{1})$ denotes the Dirichlet distribution with parameter $\mathbf{1} := [1, \dots, 1]$ ($K - 1$ values).

2.3.4.1 Evaluation on Tabular Benchmarks

Table 2.4 shows the performance comparisons on tabular datasets. The parameters $(d, [N_1, \dots, N_K])$ indicates that the feature dimension is d , and the number of clean instances in class- k is N_k . The noise rates are: LOW: $e_1 = 0.2, e_2 = 0.1$. MEDIUM: $e_1 = 0.4, e_2 = 0.2$. HIGH: $e_1 = 0.4, e_2 = 0.4$. Top-2 of each row are **bold**. By counting the number of **bold** results, we know all four variants of our proposed method perform better than the three baselines statistically. Additionally, based on the counting results, OURS- \mathbf{X} wins in 16 settings while OURS- \mathbf{A} wins in 20 settings, indicating decoupling different parts by eigen decomposition is not statistically significantly useful. This may be due to the property of tabular data: the original correlation in \mathbf{X} may not be strong and the decoupling operations will involve extra errors and make the results even worse.

2.3.4.2 Evaluation on Natural Language Benchmarks

We also evaluate our methods on more sophisticated text classification tasks and show the results in Table 2.5. The parameters $(d, [N_1, \dots, N_K])$ indicates that the feature dimension is d , and the number of clean instances in class- k is N_k . Notation [30k

Table 2.4: The estimation error ($\times 100$) on tabular benchmarks.

Tabular Datasets		Method						
$(d, [N_1, N_2])$	Noise Rate	T-Rev	CL	HOC	Ours- X -KL	Ours- X -TV	Ours- A -KL	Ours- A -TV
Heart (23, [138, 165])	Low	9.25	8.00	9.82	8.09	8.70	8.88	9.18
	Medium	11.94	11.48	7.85	9.55	1.48	3.98	5.51
	High	6.74	6.54	4.71	9.78	14.91	1.33	4.21
Banana (2, [2924, 2376])	Low	30.89	30.53	14.96	10.74	11.84	10.57	9.26
	Medium	20.77	20.81	8.97	4.90	3.98	6.41	6.97
	High	6.71	7.58	4.78	12.11	8.89	9.78	11.26
Titanic (3, [1490, 711])	Low	21.40	20.62	11.24	10.83	9.96	11.05	11.60
	Medium	10.83	10.31	9.97	9.82	9.65	9.61	9.75
	High	6.93	6.75	1.94	1.97	1.97	1.89	1.92
Splice (240, [1648, 1527])	Low	10.63	9.32	7.32	2.29	1.87	3.00	3.65
	Medium	10.35	9.84	5.38	2.21	3.90	1.26	2.15
	High	7.86	7.74	17.43	4.26	2.94	4.41	5.81
Twonorm (20, [3697, 3703])	Low	1.79	1.63	2.12	2.34	2.80	0.54	0.65
	Medium	1.86	1.42	1.38	1.42	1.30	2.14	1.73
	High	1.67	1.22	5.18	5.88	3.47	1.46	2.12
Waveform (21, [3353, 1647])	Low	10.59	10.89	7.93	6.68	6.78	5.67	5.79
	Medium	8.45	7.82	3.51	2.50	3.09	2.34	2.72
	High	5.04	4.76	3.84	2.72	1.71	2.29	5.42
Flare-solar (31, [477, 589])	Low	19.28	18.43	15.24	14.60	14.84	15.63	14.71
	Medium	16.57	16.38	4.58	4.39	5.05	4.64	4.82
	High	8.35	8.25	4.71	4.47	4.74	3.87	3.30

Table 2.5: The estimation error ($\times 100$) on natural language benchmarks.

TEXT DATASETS		METHOD						
$(d, [N_1, \dots, N_K])$	NOISE RATE	T-REV	CL	HOC	Ours- X -KL	Ours- X -TV	Ours- A -KL	Ours- A -TV
AG'S NEWS (BERT) (768, [30k \times 4])	LOW	10.38	11.41	13.32	12.65	12.75	8.36	8.35
	MEDIUM	10.71	10.63	10.62	10.13	10.45	6.44	6.52
	HIGH	13.97	13.82	6.80	6.83	6.69	4.54	4.19
DBPEDIA (BERT) (768, [40k \times 14])	LOW	6.80	5.31	7.57	6.76	6.94	2.52	2.52
	MEDIUM	14.91	14.40	6.30	5.66	5.78	2.33	2.28
	HIGH	24.23	23.28	6.00	5.18	5.22	2.42	2.43
YELP-5 (BERT) (768, [130k \times 5])	LOW	38.49	38.75	40.87	40.71	40.58	37.37	37.19
	MEDIUM	35.46	36.05	33.63	34.23	33.88	31.79	31.94
	HIGH	21.20	20.88	19.09	18.56	20.13	18.11	18.06
JIGSAW (BERT) (768, [144,277, 15,294])	LOW	20.92	20.17	14.25	14.07	14.24	9.76	9.97
	MEDIUM	17.10	16.44	11.28	11.80	12.23	7.45	7.66
	HIGH	7.19	6.81	4.84	4.85	3.43	0.78	1.02

$\times 4]$ shows $N_1 = N_2 = N_3 = N_4 = 30k$. We use a heuristic method to set the average noise rate e . The aim is to make the ratio between diagonal elements and off-diagonal elements of \mathbf{T} consistent. The average noise rate follows $e = 1/(1 + r/\sqrt{K} - 1)$, where

LOW, MEDIUM and HIGH indicate $r = 8$, $r = 4$, and $r = 1.5$, respectively. According to this rule, the low, medium, and high noise rates for binary, 5-class, and 10-class classifications are $(0.11, 0.2, 0.4)$, $(0.2, 0.33, 0.57)$, and $(0.27, 0.43, 0.67)$, respectively, which align with the general cognition of the community [30]. The top-2 of each row are **bold**. Comparing Table 2.5 with Table 2.4, we find the direct reweighting by f -mutual information (OURS- \mathbf{X}) performs similarly to the hard cosine similarity adopted by HOC, while the information-theoretic reweighting after projecting \mathbf{X} to its eigen space (OURS- \mathbf{A}) performs consistently better than other methods. Intuitively, the correlations of BERT embeddings are stronger than those of tabular data. Thus we observe a clear performance improvement by removing correlations. Besides, it is interesting to see *the estimation error may decrease for higher noise rate setting*. We conjecture the reasons may comprise: 1) The original dataset may be noisy, e.g., the original Yelp dataset contains lots of noisy reviews [129]. Consider a binary classification with inherent 20% noise. Then adding 10% (low) noise will make the average noise rate to 0.26, where the gap between the real noise rate and the hypothesized noise rate is $0.26 - 0.1 = 0.16$. Similarly, the gap of adding 40% (high) noise is $0.44 - 0.4 = 0.04$. Therefore, even though \mathbf{T} is accurately estimated, the absolute error under our current metric will be higher for the HIGH-noise case. 2) As analyzed in Section 2.3.1, the error of random guess for low-noise (10%) and high-noise (40%) settings are 0.4 and 0.1, respectively, indicating a small error may cause more severe problems in higher-noise settings. We leave more detailed discussions to Appendix A.6.2.

Table 2.6: The last/best epoch clean test accuracies (%) when training with high-level noise defined in Table 2.5

METHOD	AG'S NEWS		DBPEDIA	
	LAST	BEST	LAST	BEST
HOC [230]	82.17	83.08	91.06	91.06
OURS-A-TV	85.01	85.17	97.71	97.77

Downstream learning error [126] showed the additional learning risk is positively related to estimation error. To further consolidate the estimation error reduction of our method, we feed the estimated \mathbf{T} into forward loss correction [146] and check the clean test accuracy. Table 2.6 shows our approach significantly improves both the best and last epoch test accuracy by simply changing the \mathbf{T} in loss correction from HOC estimates to ours.

2.3.5 Takeaways

We summarize the takeaways of this section as follows.

- This work has studied the problem of estimating noise transition matrix on application domains apart from images. We have proposed an information-theoretic approach to down-weight the less informative parts of features with only noisy labels for tasks with lower-quality features.
- Based on the f -mutual information, we propose a novel reweighting mechanism to first decouple features by projecting to its eigenspace and then down-weight the less-informative parts with only access to noisy labels. The mechanism intuitively

disentangles features but it does not require training and can be applied efficiently.

- We prove that calculating the total-variation-based mutual information with noisy labels preserves the same order as using clean labels (Theorem 3), and the traditional mutual information preserves the order when the absolute gap between two noisy measurements is larger than a guaranteed threshold (Theorem 4).
- We empirically demonstrate that our approach helps return a more accurate estimate of the transition matrix and reduce the classification errors of downstream learning tasks on datasets with lower-quality or more sophisticated features, including UCI datasets with tabular data and text classification benchmarks.
- Code is available at github.com/UCSC-REAL/BeyondImages.

Chapter 3

Label Error Detection

Label noise in real-world datasets encodes wrong correlation patterns and impairs the generalization of deep neural networks (DNNs). It is critical to find efficient ways to detect corrupted patterns and remove/correct them. Figure 3.1 shows two popular label detection pipelines, where the inputs are features and the corresponding noisy labels, and the outputs are a set of corrupted labels. The **blue** branch indicates the learning-centric solutions [82, 63, 212, 143, 209, 179, 219, 30], which select corrupted examples relying on the model’s predicted probability of each class. The **orange** branch shows a data-centric solution [226], where the core idea is designing some score functions based on neighborhood information. In this chapter, we introduce two label error detection methods, a dynamic sample sieve that filters out the corrupted examples during training (Section 3.1), and a training-free plug-and-play label error detector (Section 3.2).

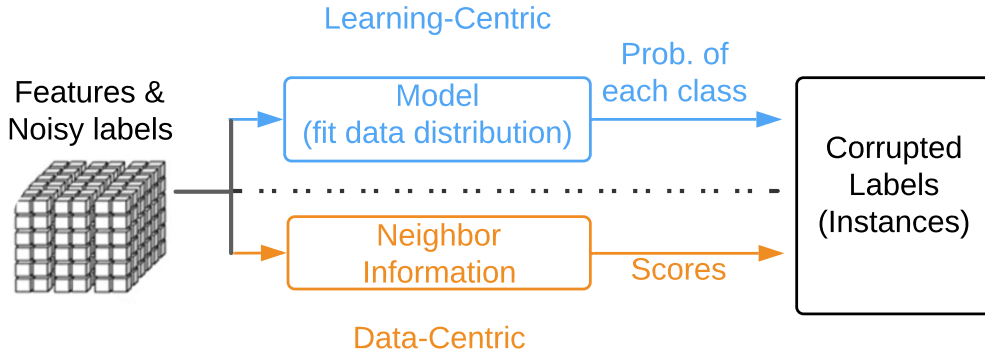


Figure 3.1: Label error detection pipelines: Learning-centric vs. data-centric.

3.1 CORES²: COncidence REgularized Sample Sieve

The biggest challenge of designing a learning-centric method to detect label errors is the “chicken-egg” problem. That is, if the model for detection is trained on the noisy dataset, it would be likely to overfit and memorize both the examples with clean labels and corrupted labels, then we cannot tell the difference between the clean and corrupted ones according to the model outputs. Therefore, it is critical to avoid overfitting to corrupted labels [184], especially when the noise depends on both true labels Y and features X . Unfortunately, this often tends to be the case where human annotations are prone to different levels of errors for tasks with varying difficulty levels. Another challenge is about setting a “threshold” between clean ones and corrupted ones, which may not be a fixed value due to different levels of errors.

Aiming at the above two challenges, we design a sample sieve to provide a high-quality splitting of clean and corrupted examples with theoretical guarantees. The overfitting to corrupted labels is restricted by a regularization term, which helps improve

the confidence of the learned classifier and is proven to help safely sieve out corrupted examples. The thresholds for the splitting be clean and corrupted examples are also set dynamically according to the model states. In the remainder of this section, we will first show the intuition (Section 3.1.1) and actual design of the sample sieve (Section 3.1.2), then demonstrate the effectiveness with both theoretical guarantees (Section 3.1.3) and numerical experiments (Section 3.1.4).

3.1.1 Confidence Regularization

In this section, we present a new confidence regularizer (CR). Our design of the CR is mainly motivated by a recently proposed robust loss function called peer loss [124]. For each example (x_n, \tilde{y}_n) , peer loss has the following form:

$$\ell_{\text{PL}}(f(x_n), \tilde{y}_n) := \ell(f(x_n), \tilde{y}_n) - \ell(f(x_{n_1}), \tilde{y}_{n_2}),$$

where $(x_{n_1}, \tilde{y}_{n_1})$ and $(x_{n_2}, \tilde{y}_{n_2})$ are two randomly sampled and paired peer examples (with replacement) for n . Let X_{n_1} and \tilde{Y}_{n_2} be the corresponding random variables. Note X_{n_1}, \tilde{Y}_{n_2} are two independent and uniform random variables being each $x_{n'}, n' \in [N]$ and $\tilde{y}_{n'}, n' \in [N]$ with probability $\frac{1}{N}$ respectively: $\mathbb{P}(X_{n_1} = x_{n'} | \tilde{D}) = \mathbb{P}(\tilde{Y}_{n_2} = y_{n'} | \tilde{D}) = \frac{1}{N}, \forall n' \in [N]$. Let $\mathcal{D}_{\tilde{Y}|\tilde{D}}$ be the distribution of \tilde{Y}_{n_2} given dataset \tilde{D} . Peer loss then has

the following equivalent form in expectation:

$$\begin{aligned}
& \frac{1}{N} \sum_{n \in [N]} \mathbb{E}_{X_{n_1}, \tilde{Y}_{n_2} | \tilde{D}} [\ell(f(x_n), \tilde{y}_n) - \ell(f(X_{n_1}), \tilde{Y}_{n_2})] \\
&= \frac{1}{N} \sum_{n \in [N]} \left[\ell(f(x_n), \tilde{y}_n) - \sum_{n' \in [N]} \mathbb{P}(X_{n_1} = x_{n'} | \tilde{D}) \mathbb{E}_{\mathcal{D}_{\tilde{Y} | \tilde{D}}} [\ell(f(x_{n'}), \tilde{Y})] \right] \\
&= \frac{1}{N} \sum_{n \in [N]} \left[\ell(f(x_n), \tilde{y}_n) - \mathbb{E}_{\mathcal{D}_{\tilde{Y} | \tilde{D}}} [\ell(f(x_n), \tilde{Y})] \right].
\end{aligned}$$

This result characterizes a new loss denoted by ℓ_{CA} :

$$\ell_{\text{CA}}(f(x_n), \tilde{y}_n) := \ell(f(x_n), \tilde{y}_n) - \mathbb{E}_{\mathcal{D}_{\tilde{Y} | \tilde{D}}} [\ell(f(x_n), \tilde{Y})]. \quad (3.1)$$

Though not studied rigorously by [124], we show, under some conditions, ℓ_{CA} defined in Eqn. (3.1) encourages confident predictions. from f by analyzing the gradients:

Theorem 5. *For $\ell_{\text{CA}}(\cdot)$, solutions satisfying $f_{x_n}[i] > 0, \forall i \in [K]$ are not locally optimal at (x_n, \tilde{y}_n) .*

See Appendix B.1.2 for the proof. Particularly, in binary cases, we have constraint $f(x_n)[0] + f(x_n)[1] = 1$. Following Theorem 5, we know minimizing $\ell_{\text{CA}}(f(x_n), \tilde{y}_n)$ w.r.t f under this constraint leads to either $f(x_n)[0] \rightarrow 1$ or $f(x_n)[1] \rightarrow 1$, indicating confident predictions. Therefore, the addition of term $-\mathbb{E}_{\mathcal{D}_{\tilde{Y} | \tilde{D}}} [\ell(f(x_n), \tilde{Y})]$ helps improve the confidence of the learned classifier. Inspired by the above observation, we define the following confidence regularizer:

$$\textbf{Confidence Regularizer:} \quad \ell_{\text{CR}}(f(x_n)) := -\beta \cdot \mathbb{E}_{\mathcal{D}_{\tilde{Y} | \tilde{D}}} [\ell(f(x_n), \tilde{Y})],$$

where β is positive and $\ell(\cdot)$ refers to the CE loss. The prior probability $\mathbb{P}(\tilde{Y} | \tilde{D})$ is

counted directly from the noisy dataset. In the remaining of this paper, $\ell(\cdot)$ indicates the CE loss by default.

Why are confident predictions important? Intuitively, when model fits to the label noise, its predictions often become less confident, since the noise usually corrupts the signal encoded in the clean data. From this perspective, encouraging confident predictions plays against fitting to label noise. Compared to instance-independent noise, the difficulties in estimating the instance-dependent noise rates largely prevent us from applying existing techniques. In addition, as shown by [133], the 0-1 loss function is more robust to instance-based noise but hard to optimize with. To a certain degree, pushing confident predictions results in a differentiable loss function that approximates the 0-1 loss, and therefore restores the robustness property. Besides, as observed by [24] and [234], gradients from similar examples would reinforce each other. When the overall label information is dominantly informative that $T_{ii}(X) > T_{ij}(X)$, DNNs will receive more correct information statistically. Encouraging confident predictions would discourage the memorization of the noisy examples (makes it hard for noisy labels to reduce the confidence of predictions), and therefore further facilitate DNNs to learn the (clean) dominant information.

3.1.2 Confidence Regularized Sample Sieve

Intuitively, label noise misleads the training thus sieving corrupted examples out of datasets is beneficial. Furthermore, label noise introduces high variance during

training even with the existence of ℓ_{CR} . Therefore, rather than accomplishing training solely with ℓ_{CR} , we will first leverage its regularization power to design an efficient sample sieve. Similar to a general sieving process in physical words that compares the size of particles with the aperture of a sieve, we evaluate the “size” (quality, or a regularized loss) of examples and compare them with some to-be-specified thresholds, therefore the name sample sieve. In our formulation, the regularized loss $\ell(f(x_n), \tilde{y}_n) + \ell_{\text{CR}}(f(x_n))$ is employed to evaluate examples and α_n is used to specify thresholds. Specifically, we aim to solve the sample sieve problem in (3.2), where the crucial components are:

- $v_n \in \{0, 1\}$ indicates whether example n is clean ($v_n = 1$) or not ($v_n = 0$);
- α_n (mimicking the aperture of a sieve) controls which example should be sieved out;
- \bar{f} is a copy of f and does not contribute to the back-propagation. \mathcal{F} is the search space of f .

Dynamic sample sieve The problem in (3.2) is a combinatorial optimization that is hard to solve directly. A standard solution to (3.2) is to apply alternate search iteratively as follows:

- Starting at $t = 1$, $v_n^{(0)} = 1, \forall n \in [N]$.
- Confidence-regularized model update (at iteration- t):

$$f^{(t)} = \arg \min_{f \in \mathcal{F}} \sum_{n \in [N]} v_n^{(t-1)} [\ell(f(x_n), \tilde{y}_n) + \ell_{\text{CR}}(f(x_n))]; \quad (3.3)$$

- Sample sieve (at iteration- t):

$$v_n^{(t)} = \mathbf{1}(\ell(f^{(t)}(x_n), \tilde{y}_n) + \ell_{\text{CR}}(f^{(t)}(x_n)) < \alpha_{n,t}), \quad (3.4)$$

Confidence Regularized Sample Sieve

$$\begin{aligned}
 & \min_{\substack{f \in \mathcal{F}, \\ v \in \{0,1\}^N}} \sum_{n \in [N]} v_n [\ell(f(x_n), \tilde{y}_n) + \ell_{\text{CR}}(f(x_n)) - \alpha_n] \\
 & \text{s.t. } \ell_{\text{CR}}(f(x_n)) := -\beta \cdot \mathbb{E}_{\mathcal{D}_{\tilde{Y}|\bar{D}}} \ell(f(x_n), \tilde{Y}), \\
 & \alpha_n := \frac{1}{K} \sum_{\tilde{y} \in [K]} \ell(\bar{f}(x_n), \tilde{y}) + \ell_{\text{CR}}(\bar{f}(x_n)).
 \end{aligned} \tag{3.2}$$

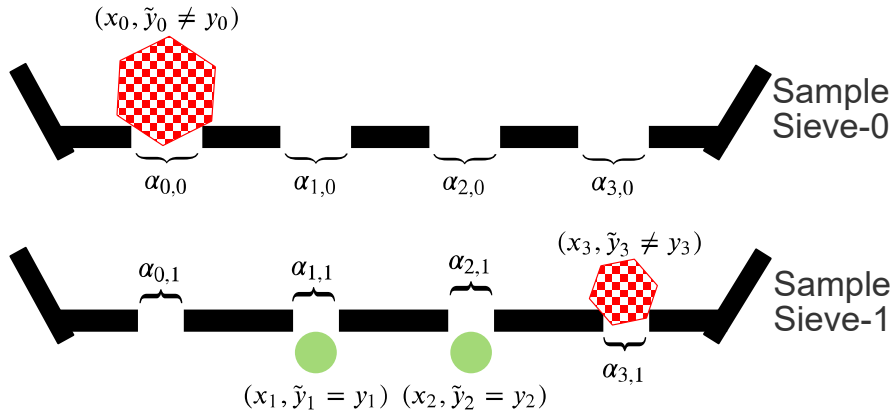


Figure 3.2: Dynamic sample sieves.

where $\alpha_{n,t} = \frac{1}{K} \sum_{\tilde{y} \in [K]} \ell(\bar{f}^{(t)}(x_n), \tilde{y}) + \ell_{\text{CR}}(\bar{f}^{(t)}(x_n))$, $f^{(t)}$ and $v^{(t)}$ refer to the specific classifier and weight at iteration- t . Note the values of $\ell_{\text{CR}}(\bar{f}^{(t)}(x_n))$ and $\ell_{\text{CR}}(f^{(t)}(x_n))$ are the same. We keep both terms to be consistent with the objective in Eq. (3.2). In DNNs, we usually update model f with one or several epochs of data instead of completely solving (3.3).

Figure 3.2 illustrates the dynamic sample sieve, where green circles are clean examples, red hexagons are corrupted examples, the size of each example corresponds

to the regularized loss, and the aperture of a sieve is determined by $\alpha_{n,t}$. In each iteration- t , sample sieve- t “blocks” some corrupted examples by comparing a regularized example loss with a closed-form threshold $\alpha_{n,t}$, which can be immediately obtained given current model $\bar{f}^{(t)}$ and example (x_n, \tilde{y}_n) (no extra estimation needed). In contrast, most sample selection works [63, 212, 179] focus on controlling the number of the selected examples using an intuitive function where the overall noise rate may be required, or directly selecting examples by an empirically set threshold [221]. Intuitively, the specially designed thresholds $\alpha_{n,t}$ for each example should be more accurate than a single threshold for the whole dataset. Besides, the goal of existing works is often to select clean examples while our sample sieve focuses on removing the corrupted ones. On a high level, we follow a different philosophy from these sample selection works. We coin our solution as COncidence REgularized Sample Sieve (CORES²).

More visualizations of the sample sieve In addition to Figure 3.2, we visualize the superiority of our sample sieve with numerical results as Figure 3.3. The figure includes two noise settings: symmetric noise (symm.) and 40% instance-based noise (inst.). The loss is given by $\ell(f^{(t)}(x_n), \tilde{y}_n) + \ell_{\text{CR}}(f^{(t)}(x_n)) - \alpha_{n,t}$ as (3.4). The CE Sieve represents the dynamic sample sieve with standard cross-entropy loss (without CR). The sieved dataset is in the form of two clusters of examples. Particularly, from Figure 3.3(b) and Figure 3.3(f), we observe that CE suffers from providing a good division of clean and corrupted examples due to overfitting in the final stage of training. On the other hand, with ℓ_{CR} , there are two distinct clusters and can be separated by the threshold

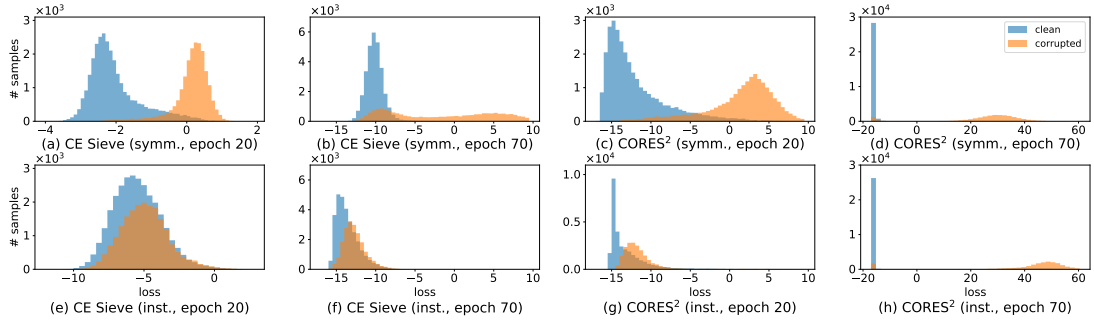


Figure 3.3: Loss distributions of training on CIFAR-10 with a noise rate of 40%.

0 as shown in Figure 3.3(d) and Figure 3.3(h). Comparing Figure 3.3(a)-3.3(d) with Figure 3.3(e)-3.3(h), we find the effect of instance-dependent noise on training is indeed different from the symmetric one, where the instance-dependent noise is more likely to cause overfitting.

3.1.3 Theoretical Guarantees of CORES²

In this section, we theoretically show the advantages of CORES². The analyses focus on showing CORES² guarantees a quality division, i.e. $v_n = \mathbf{1}(y_n = \tilde{y}_n), \forall n$, with a properly set β . To show the effectiveness of this solution, we call a model prediction on x_n is *better than random guess* if $f_{x_n}[y_n] > 1/K$, and call it *confident* if $f_{x_n}[y] \in \{0, 1\}, \forall y \in [K]$, where y_n is the clean label and y is an arbitrary label. The quality of sieving out corrupted examples is guaranteed in Theorem 6.

Theorem 6. *The sample sieve defined in (3.4) ensures that clean examples $(x_n, \tilde{y}_n = y_n)$ will not be identified as being corrupted if the model $f^{(t)}$'s prediction on x_n is better than random guess.*

Theorem 6 informs us that our sample sieve can progressively and safely filter out corrupted examples, and therefore improves division quality, when the model prediction on each x_n is better than a random guess. The full proof is left to Appendix B.1.3. In the next section, we provide evidence that our trained model is guaranteed to achieve this requirement with sufficient examples.

3.1.3.1 Decoupling the Confidence Regularized Loss

The discussion of performance guarantees of the sample sieve focuses on a general instance-based noise transition matrix $T(X)$, which can induce any specific noise regime such as symmetric noise and asymmetric noise [96, 108]. Note that feature-independency was one critical assumption in state-of-the-art theoretically guaranteed noise-resistant literature [139, 124, 205] while we *do not* require. Let $T_{ij} := \mathbb{E}_{\mathcal{D}|Y=i}[T_{ij}(X)], \forall i, j \in [K]$. Theorem 7 explicitly shows the contributions of clean examples, corrupted examples, and ℓ_{CR} during training. See Appendix B.1.1 for the proof.

Theorem 7. (*Main Theorem: Decoupling the Expected Regularized CE Loss*) *In expectation, the loss with ℓ_{CR} can be decoupled as three separate additive terms:*

$$\begin{aligned} \mathbb{E}_{\tilde{\mathcal{D}}}\left[\ell(f(X), \tilde{Y}) + \ell_{CR}(f(X))\right] &= \overbrace{\underline{T} \cdot \mathbb{E}_{\mathcal{D}}[\ell(f(X), Y)]}^{\text{Term-1}} + \overbrace{\bar{\Delta} \cdot \mathbb{E}_{\mathcal{D}_{\Delta}}[\ell(f(X), Y)]}^{\text{Term-2}} \\ &+ \underbrace{\sum_{j \in [K]} \sum_{i \in [K]} \mathbb{P}(Y = i) \mathbb{E}_{\mathcal{D}|Y=i}[(U_{ij}(X) - \beta \mathbb{P}(\tilde{Y} = j)) \ell(f(X), j)]}_{\text{Term-3}}, \end{aligned} \quad (3.5)$$

where $\underline{T} := \min_{j \in [K]} T_{jj}$, $\bar{\Delta} := \sum_{j \in [K]} \Delta_j \mathbb{P}(Y = j)$, $\Delta_j := T_{jj} - \underline{T}$, $U_{ij}(X) =$

$T_{ij}(X), \forall i \neq j, U_{jj}(X) = T_{jj}(X) - T_{jj}$, and

$$\mathbb{E}_{\mathcal{D}_\Delta}[\ell(f(X), Y)] := \mathbf{1}(\bar{\Delta} > 0) \sum_{j \in [K]} \frac{\Delta_j \mathbb{P}(Y = j)}{\bar{\Delta}} \mathbb{E}_{\mathcal{D}|Y=j}[\ell(f(X), j)].$$

Equation (3.5) provides generic machinery for anatomizing noisy datasets, where we show the effects of instance-based label noise on the ℓ_{CR} regularized loss can be decoupled into three additive terms: **Term-1** reflects the expectation of CE on clean distribution \mathcal{D} , **Term-2** shifts the clean distribution by changing the prior probability of Y , and **Term-3** characterizes how the corrupted examples (represented by $U_{ij}(X)$) might mislead/mis-weight the loss, as well as the regularization ability of ℓ_{CR} (represented by $\beta \mathbb{P}(\tilde{Y} = j)$). In addition to the design of sample sieve, this additive decoupling structure also provides a novel and promising perspective for understanding and controlling the effects of generic instance-dependent label noise.

3.1.3.2 Guarantees of the Sample Sieve

By decoupling the effects of instance-dependent noise into separate additive terms as shown in Theorem 7, we can further study under what conditions, minimizing the confidence regularized CE loss on the (*instance-dependent*) *noisy* distribution will be equivalent to minimizing the true loss incurred on the *clean* distribution, which is exactly encoded by Term-1. In other words, we would like to understand when Term-2 and Term-3 in (3.5) can be controlled not to disrupt the minimization of Term-1. Our next main result establishes this guarantee but will first need the following two assumptions.

Assumption 3. ($Y^* = Y$) *Clean labels are Bayes optimal* ($Y^* := \arg \max_{i \in [K]} \mathbb{P}(Y =$

$i|X)$).

Assumption 4. (*Informative datasets*) The noise rate is bounded as $T_{ii}(X) - T_{ij}(X) > 0, \forall i \in [K], j \in [K], j \neq i, X \sim \mathcal{D}_X$.

Feasibility of assumptions 1) Note for many popular image datasets, e.g. CIFAR, the label of each feature is well-defined and the corresponding distribution is well-separated by human annotation. In this case, each feature X only belongs to one particular class Y . Thus Assumption 3 is generally held in classification problems [119]. Technically, this assumption could be relaxed. We use this assumption for clean presentations. 2) Assumption 4 shows the requirement of noise rates, i.e., for any feature X , a sufficient number of clean examples are necessary for dominant clean information. For example, we require $T_{ii}(X) - T_{ij}(X) > 0$ to ensure examples from class i are informative [123].

Before formally presenting the noise-resistant property of training with ℓ_{CR} , we discuss intuitions here. Our ℓ_{CR} regularizes the CE loss to generate/incentivize confident prediction, and thus is able to approximate the 0-1 loss to obtain its robustness property. More explicitly, from (3.5), ℓ_{CR} affects Term-3 with a scale parameter β . Recall that $U_{ij}(X) = T_{ij}(X), \forall i \neq j$, which is exactly the noise transition matrix. Although we have *no information* about this transition matrix, the confusion brought by $U_{ij}(X)$ can be canceled or reversed by a sufficiently large β such that $U_{ij}(X) - \beta\mathbb{P}(\tilde{Y} = j) \leq 0$. Intuitively, with an appropriate β , all the effects of $U_{ij}(X), i \neq j$ can be reversed, and we will get a negative loss punishing the classifier for predicting class- j when the clean label is i . Formally, Theorem 8 shows the noise-resistant property of training with ℓ_{CR}

and is proved in Appendix B.1.4.

Theorem 8. (*Robustness of the Confidence Regularized CE Loss*) *With Assumption 3 and 4, when*

$$\max_{i,j \in [K], X \sim \mathcal{D}_X} \frac{U_{ij}(X)}{\mathbb{P}(\tilde{Y} = j)} \leq \beta \leq \min_{\mathbb{P}(\tilde{Y}=i) > \mathbb{P}(\tilde{Y}=j), X \sim \mathcal{D}_X} \frac{T_{ii}(X) - T_{ij}(X)}{\mathbb{P}(\tilde{Y} = i) - \mathbb{P}(\tilde{Y} = j)}, \quad (3.6)$$

minimizing $\mathbb{E}_{\tilde{\mathcal{D}}}[\ell(f(X), \tilde{Y}) + \ell_{CR}(f(X))]$ is equivalent to minimizing $\mathbb{E}_{\mathcal{D}}[\ell(f(X), Y)]$.

Theorem 8 shows a sufficient condition of β for our confidence regularized CE loss to be robust to instance-dependent label noise. The bound on LHS ensures the confusion from label noise could be canceled or reversed by the β weighted confidence regularizer, and the RHS bound guarantees the model with the minimized regularized loss predicts the most frequent label in each feature w.p. 1.

Theorem 8 also provides guidelines for tuning β . Although we have no knowledge about $T_{ij}(X)$, we can roughly estimate the range of possible β . One possibly good setting of β is linearly increasing with the number of classes, e.g. $\beta = 2$ for 10 classes and $\beta = 20$ for 100 classes.

With *infinite model capacity*, minimizing $\mathbb{E}_{\mathcal{D}}[\ell(f(X), Y)]$ returns the Bayes optimal classifier (since CE is a calibrated loss) which predicts on each x_n better than random guess. Therefore, with a sufficient number of examples, minimizing $\mathbb{E}_{\tilde{\mathcal{D}}}[\ell(f(X), \tilde{Y}) + \ell_{CR}(f(X))]$ will also return a model that predicts better than random guess, then satisfying the condition required in Theorem 6 to guarantee the quality of sieved examples. Further, since the Bayes optimal classifier always predicts clean labels confidently when Assumption 3 holds, Theorem 8 also guarantees confident predictions.

With such predictions, the sample sieve in (3.4) will achieve 100% precision on both clean and corrupted examples. This guaranteed division is summarized in Corollary 3:

Corollary 3. *When conditions in Theorem 8 hold, with infinite model capacity and sufficiently many examples, CORES² achieves $v_n = \mathbf{1}(y_n = \tilde{y}_n), \forall n \in [N]$, i.e., all the sieved clean examples are effectively clean.*

3.1.4 Experiments

Now we present experimental evidence of how CORES² works. Throughout the experiments, the logarithmic function in ℓ_{CR} is adapted to $\ln(f_x[y] + 10^{-8})$ for numerical stability.

Datasets CORES² is evaluated on three benchmark datasets: CIFAR-10, CIFAR-100 [99] and Clothing1M [200]. Following the convention from [205], we use ResNet34 for CIFAR-10 and CIFAR-100 and ResNet50 for Clothing1M.

Noise type We experiment with three types of label noise: symmetric, asymmetric and instance-dependent label noise. Symmetric noise is generated by randomly flipping a true label to the other possible labels w.p. ε [96], where ε is called the noise rate. Asymmetric noise is generated by flipping the true label to the next class (i.e., label $i \rightarrow i + 1, \text{ mod } K$) w.p. ε . Instance-dependent label noise is a more challenging setting and we generate instance-dependent label noise following the method from [198] (See Appendix A.4.1 for details). In expectation, the noise rate ε for all noise regimes is the overall ratio of corrupted examples in the whole dataset.

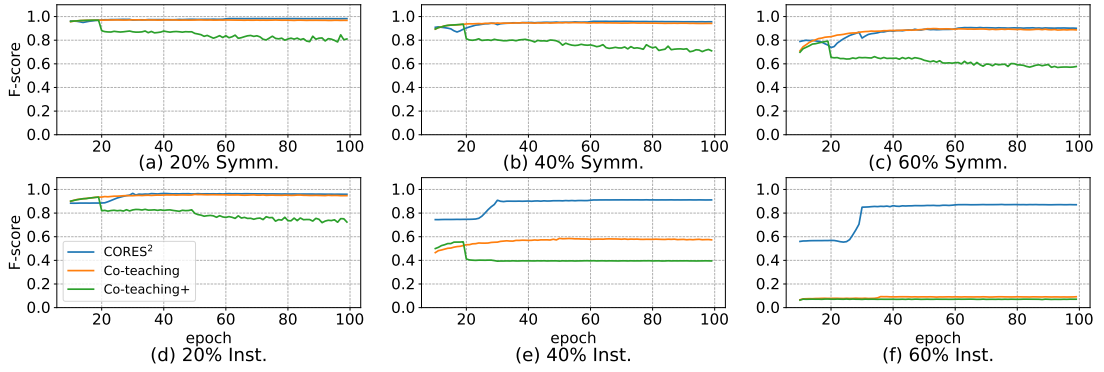


Figure 3.4: F-score comparisons on CIFAR10 under symmetric (Symm.) and instance-based (Inst.) label noise.

3.1.4.1 Quality of our sample sieve

Figure 3.4 shows the F-scores of sieved clean examples with training epochs on the symmetric and the instance-based label noise. F-score quantifies the quality of the sample sieve by the harmonic mean of precision (ratio of actual cleans examples in sieved clean ones) and recall (ratio of sieved cleans examples in actual clean ones), i.e.,

$$\text{F-score} := \frac{2 \cdot \text{Pre} \cdot \text{Re}}{\text{Pre} + \text{Re}},$$

where

$$\text{Pre} := \frac{\sum_{n \in [N]} \mathbb{1}(v_n = 1, y_n = \tilde{y}_n)}{\sum_{n \in [N]} \mathbb{1}(v_n = 1)}, \quad \text{and} \quad \text{Re} := \frac{\sum_{n \in [N]} \mathbb{1}(v_n = 1, y_n = \tilde{y}_n)}{\sum_{n \in [N]} \mathbb{1}(y_n = \tilde{y}_n)}.$$

We compare CORES² with Co-teaching and Co-teaching+. Note the F-scores of CORES² and Co-teaching are consistently high on the symmetric noise, while CORES² achieves higher performance on the challenging instance-based label noise, especially with the 60% noise rate where the other two methods have low F-scores.

Table 3.1: Comparison of test accuracies on clean datasets under instance-based label noise.

Method	<i>Inst. CIFAR10</i>			<i>Inst. CIFAR100</i>		
	$\varepsilon = 0.2$	$\varepsilon = 0.4$	$\varepsilon = 0.6$	$\varepsilon = 0.2$	$\varepsilon = 0.4$	$\varepsilon = 0.6$
Cross Entropy	87.16	75.16	44.64	58.72	41.14	25.29
Forward T [146]	88.08	82.67	41.57	58.95	41.68	22.83
L_{DMI} [205]	88.80	82.70	70.54	58.66	41.77	28.00
L_q [221]	86.45	69.02	32.94	58.18	40.32	23.13
SCE [177]	89.11	72.04	44.83	59.87	41.76	23.41
Co-teaching [63]	88.66	69.50	34.61	43.03	23.13	7.07
Co-teaching+ [212]	89.04	69.15	33.33	41.84	24.40	8.74
JoCoR [179]	88.71	68.97	30.27	44.28	22.77	7.54
Peer Loss [124]	89.33	81.09	73.73	59.92	45.76	33.61
CORES ² [30]	89.50	82.84	79.66	61.25	47.81	37.85

3.1.4.2 Training with sample sieve

In this section, we compare CORES² with several state-of-the-art methods on CIFAR-10 and CIFAR-100 under instance-based, symmetric and asymmetric label noise settings, which is shown on Table 3.1. For a fair comparison, all the methods use ResNet-34 as the backbone. By comparing the performance of CE on the symmetric and the instance-based label noise, we note the instance-based label noise is a more challenging setting. Even though some methods (e.g., L_{DMI}) behaves well on symmetric and asymmetric label noise, they may reach low test accuracies on the instance-based label noise, especially when the noise rate is high or the dataset is more complex. Table 3.1 shows that CORES² consistently works well on the instance-based label noise.

3.1.5 Takeaways

We summarize the takeaway messages from this subsection as follows.

- We have proposed CORES², a sample sieve that is guaranteed to be robust to general

instance-dependent label noise and sieve out corrupted examples, but without using explicit knowledge of the noise rates of labels.

- We have provided a theoretically sound sample sieve that simply compares the example’s regularized loss with a closed-form threshold explicitly determined by predictions from the above trained model using our confidence regularized loss, without any extra estimates.
- By *decoupling* the regularized loss into separate additive terms, we have also provided a novel and promising mechanism for understanding and controlling the effects of general instance-dependent label noise.
- CORES² achieves competitive performance on multiple datasets, including CIFAR-10, CIFAR-100, and Clothing1M, under different label noise settings.
- Code is available at <https://github.com/UCSC-REAL/cores>.

3.2 SimiFeat: Label Error Detection Using Similar Features

Including CORES², the learning-centric label error detectors are based on the idea that trains DNNs with noisy supervisions and then makes decisions based on the output [141] or gradients [149] of the last logit layer of the trained model. The high-level intuition of these methods is the memorization effects [62], i.e., instances with label errors, a.k.a., corrupted instances, tend to be harder to be learned by DNNs than clean instances [196, 118, 11]. By setting appropriate hyperparameters to utilize the memorization effect, corrupted instances could be identified.

Limitations of the learning-centric methods The above methods suffer from *two major limitations*: 1) the customized training processes are task-specific and may require fine-tuning hyperparameters for different datasets/noise; 2) as long as the model is trained with noisy supervisions, the memorization of corrupted instances exists. The model will “subjectively” and wrongly treat the memorized/overfitted corrupted instances as clean. For example, some low-frequency/rare clean instances may be harder to memorize than high-frequency/common corrupted instances. Memorizing these corrupted instances lead to unexpected and disparate impacts [121]. Existing solutions to avoid memorizing/overfitting are to employ some regularizers [30] or use early-stopping [12, 110]. However, their performance depends on hyperparameter settings. One promising way to avoid memorizing/overfitting is to drop the dependency on the noisy supervision, which motivates us to design a *training-free* method to find label errors. Intuitively, we can carefully use the information from nearby features to infer whether one instance is corrupted or not, as illustrated in Figure 3.1.

In the remainder of this section, different from current methods that train customized models on noisy datasets, we propose a training-free and data-centric solution to efficiently detect corrupted labels, which provides a new and complementary perspective to the traditional learning-centric solution. We demonstrate the effectiveness of this simple idea and open the possibility for follow-up works, and show the corresponding evidence and insights. Specifically, we have:

- *Efficient algorithms (Section 3.2.1)*: Based on the neighborhood information, we propose two methods: a voting-based local detection method that only requires

checking the noisy label consensuses of nearby features, and a ranking-based global detection method that scores each instance by its likelihood of being clean and filters out a guaranteed percentage of instances with low scores as corrupted ones.

- *Theoretical analyses:* We theoretically analyze how the quality of features (but possibly imperfect in practice) affects local voting and provide guidelines for tuning neighborhood size. We also prove the worst-case error bound for the ranking-based method.
- *Numerical findings:* Our numerical experiments show three important messages: in corrupted label detection, i) training with noisy supervisions may not be necessary; ii) feature extraction layers tend to be more useful than the logit layers; iii) features extracted from other tasks or domains are helpful.

3.2.1 Corrupted Label Detection Using Similar Features

Different from most methods that detect corrupted instances based on the logit layer or model predictions [141, 30, 149, 10], we focus on a more data-centric solution that operates on features. Particularly we are interested in the possibility of detecting corrupted labels in a training-free way. In this section, we will first introduce intuitions, and then provide two efficient algorithms to detect corrupted labels with similar features.

3.2.1.1 Intuitions

The learning-centric detection methods often detect corrupted instances by comparing model predictions with noisy labels [30, 141] as illustrated in Figure 3.1.

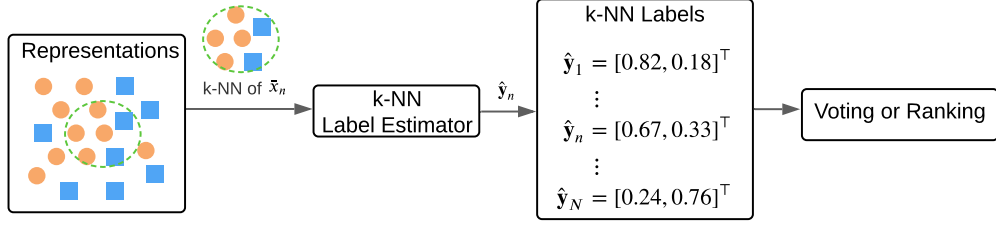


Figure 3.5: Detect corrupted labels with similar features.

However, for the data-centric method, the feature x_n cannot be directly compared with the noisy label \tilde{y}_n since x_n is not directly categorical without a model, i.e., the connection between a single x_n and \tilde{y}_n is weak. Thus our first step should be establishing auxiliary categorical information using only features.

Figure 3.5 illustrates the high-level intuition, i.e., checking label consensuses of nearby features. In this figure, **orange circle** indicates the instance with noisy label 1, **blue square** stands for the instance with noisy label 2, and **green dashed circle** represents a k -NN example. With $(k, 0)$ label clusterability as in Definition 2, we know the true labels of x_n and its k -NN x_{n_1}, \dots, x_{n_k} should be the same. If we further assume the label noise is group-dependent [170], i.e., each x_n and its k -NN can be viewed as a local group and share the same noise transition matrix [122]: $\mathbb{P}(\tilde{Y} = \tilde{y}_n | X = x_n, Y = y_n) = \mathbb{P}(\tilde{Y} = \tilde{y}_n | X = x_{n'}, Y = y_n), \forall x_{n'} \in \{x_{n_1}, \dots, x_{n_k}\}$, we can first treat their noisy labels as $k + 1$ independent observations of $\mathbb{P}(\tilde{Y} = \tilde{y}_n | X = x_n, Y = y_n)$, then estimate the probability by counting the (weighted) frequency of each class in the k -NN label estimator, and get k -NN labels $\hat{\mathbf{y}}_n$. We use the **bold \mathbf{y}** to indicate a vector, which can be seen as either

a one-hot encoding of a hard label or a soft label [229]. The i -th element $\hat{y}_n[i]$ can be interpreted as the estimated probability of predicting class- i .

Note the k -NN technique has been implemented by [10] as a filter to remove corrupted instances. However, this approach focuses on calculating distances on the logit layer, which inevitably requires a task-specific training process and may suffer from the limitations mentioned at the beginning of Section 3.2. Besides, using appropriate features may be better than model logits/predictions when the dataset is noisy. See discussions below.

Features could be better than model predictions During supervised training, memorizing noisy labels makes the model generalizes poorly [62], while using only features may effectively avoid this issue [106]. For those pre-extracted features, e.g., tabular data in UCI datasets [44], the input features are already comparable and directly applying data-centric methods on these features avoids memorizing noisy labels. For more challenging tasks such as image or text classifications, we can also borrow some pre-trained models to pre-process the raw feature to improve the clusterability of features, such as BERT [40] for language tasks, CLIP [151] for vision-language tasks, or some feature extractors from unsupervised learning [80] and self-supervised learning [79, 114, 68, 28, 31], which are not affected by noisy labels.

3.2.1.2 Voting-Based Local Detection

Inspired by the idea implemented in model decisions, i.e., selecting the most likely class as the true class, we can simply “predict” the index that corresponds to the largest element in $\hat{\mathbf{y}}_n$ with random tie-breaking, i.e., $y_n^{\text{vote}} = \arg \max_{i \in [K]} \hat{\mathbf{y}}_n[i]$. To further detect whether \tilde{y}_n is corrupted or not, we only need to check $v_n := \mathbb{1}(y_n^{\text{vote}} \neq \tilde{y}_n)$. Recall $v_n = 1$ indicates a corrupted label. This voting method relies only on the local information within each k -NN label $\hat{\mathbf{y}}_n$, which may not be robust with low-quality features. Intuitively, when the gap between the true class probability and the wrong class probability is small, the majority vote will be likely to make mistakes due to sampling errors in $\hat{\mathbf{y}}_n$. Thus only using local information within each $\hat{\mathbf{y}}_n$ may not be sufficient. It is important to leverage more information such as some global statistics, which will be discussed later.

3.2.1.3 Ranking-Based Global Detection

The score function can be designed to detect corrupted instances [141, 30, 149, 10], hard-to-learn instances [115], out-of-distribution instances [182], and suspicious instances that may cause model unfairness [173]. However, it is not clear how to do so without training a task-specific model. From a global perspective, if the likelihood for each instance being clean could be evaluated by some scoring functions, we can sort the scores in an increasing order and filter out the low-score instances as corrupted ones. Based on this intuition, there are two critical components: the *scoring function* and the *threshold* to differentiate the low-score part (corrupted) and the high-score part (clean).

Scoring function A good scoring function should be able to give clean instances higher scores than corrupted instances. We adopt cosine similarity defined as:

$$\text{Score}(\hat{\mathbf{y}}_n, j) = \frac{\hat{\mathbf{y}}_n^\top \mathbf{e}_j}{\|\hat{\mathbf{y}}_n\|_2 \|\mathbf{e}_j\|_2},$$

where \mathbf{e}_j is the one-hot encoding of label j . To evaluate whether the soft label $\hat{\mathbf{y}}_n$ informs us of a clean instance or not, we compare $\text{Score}(\hat{\mathbf{y}}_n, \tilde{y}_n)$ with other instances that have the same noisy label. This scoring function captures more information than the majority votes, which is summarized as follows.

Property 1 (Relative score). *Within the same instance, the score of the majority class is higher than the others, i.e., $\forall j \neq y_n^{\text{vote}}, j \in [K], \forall n \in [N] : \text{Score}(\hat{\mathbf{y}}_n, y_n^{\text{vote}}) > \text{Score}(\hat{\mathbf{y}}_n, j)$.*

Property 2 (Absolute score). *Score($\hat{\mathbf{y}}_n, j$) is jointly determined by both $\hat{\mathbf{y}}_n[j]$ and $\hat{\mathbf{y}}_n[j']$, $\forall j' \neq j$.*

The first property guarantees that the corrupted labels would have lower scores than clean labels for the same instance when the vote is correct. However, although solely relying on Property 1 may work well in the voting-based method which makes decisions individually for each instance, it is not sufficient to be trustworthy in the ranking-based global detection. Empirically we observe that if we choose a score function that Property 3.2 does not hold, e.g., treating k -NN soft labels as model predictions and check the cross-entropy loss, it does not always return satisfying results in our experiments. The main reason is that, across different instances, the non-majority classes of some instances

may have higher absolute scores than the majority classes of the other instances, which is especially true for general instance-dependent label noise with heterogeneous noise rates [30]. Property 2 helps make it less likely to happen. Consider an example as follows.

Example Suppose $\hat{\mathbf{y}}_{n_1} = \hat{\mathbf{y}}_{n_2} = [0.6, 0.4, 0.0]^\top$, $\hat{\mathbf{y}}_{n_3} = [0.34, 0.33, 0.33]^\top$, $y_{n_1} = y_{n_2} = y_{n_3} = 1$, $\tilde{y}_{n_1} = \tilde{y}_{n_3} = 1$, $\tilde{y}_{n_2} = 2$. We can use the majority vote to get perfect detection in this case, i.e., $y_{n_1}^{\text{vote}} = y_{n_2}^{\text{vote}} = y_{n_3}^{\text{vote}} = 1 = y_{n_1}$, since the first class of each instance has the largest value. However, if we directly use a single value in soft label \mathbf{y}_n to score them, e.g., $\text{Score}'(\hat{\mathbf{y}}_n, j) = \hat{\mathbf{y}}_n[j]$, we will have $\hat{\mathbf{y}}_{n_1}[\tilde{y}_{n_1}] = 0.6 > \hat{\mathbf{y}}_{n_2}[\tilde{y}_{n_2}] = 0.4 > \hat{\mathbf{y}}_{n_3}[\tilde{y}_{n_3}] = 0.34$, where the ranking is $n_3 \prec n_2 \prec n_1$. Ideally, we know instance n_2 is corrupted and the true ranking should be $n_2 \prec n_3 \prec n_1$ or $n_2 \prec n_1 \prec n_3$. To mitigate this problem, we choose cosine similarity as our scoring function. The three instances could be scored as 0.83, 0.55, 0.59, corresponding to an ideal ranking $n_2 \prec n_3 \prec n_1$. We formally introduce the detailed ranking approach as follows.

Ranking Suppose we have a group of instances with the same noisy class j , i.e. $\{(x_n, \tilde{y}_n)\}_{n \in \mathcal{N}_j}$, where $\mathcal{N}_j := \{n | \tilde{y}_n = j\}$ are the set of indices that correspond to noisy class j . Let N_j be the number of indices in \mathcal{N}_j (counted from noisy labels). Intuitively, we can first sort all instances in \mathcal{N}_j in an increasing order by `argsort` and obtain the original indices for the sorted scores as:

$$\mathcal{I} = \text{argsort}\{\text{Score}(\hat{\mathbf{y}}_n, j)\}_{n \in \mathcal{N}_j},$$

where the low-score head is supposed to consist of corrupted instances [141]. Then we can simply select the first \tilde{N}_j instances with low scores as corrupted instances:

$$v_n = \mathbb{1}(\text{Loc}(n, \mathcal{I}) \leq \tilde{N}_j),$$

where $\text{Loc}(n, \mathcal{I})$ returns the index of n in \mathcal{I} . Instead of manually tuning \tilde{N}_j [63], we discuss how to determine it algorithmically.

Threshold The number of corrupted instances in \mathcal{N}_j is $\mathbb{P}(Y \neq j | \tilde{Y} = j) \cdot N_j$ when N_j is sufficiently large. Therefore if all the corrupted instances have lower scores than any clean instance, we can set $\tilde{N}_j = \mathbb{P}(Y \neq j | \tilde{Y} = j) \cdot N_j$ to obtain the ideal division. Note N_j can be obtained by directly counting the number of instances with noisy label j . To calculate the probability

$$\mathbb{P}(Y \neq j | \tilde{Y} = j) = 1 - \mathbb{P}(Y = j | \tilde{Y} = j),$$

we borrow the results from the HOC estimator [230, 231], where the noise transition probability $\mathbb{P}(\tilde{Y} = j | Y = j)$ and the marginal distribution of clean label $\mathbb{P}(Y = j)$ can be estimated with only features and the corresponding noisy labels. Then we can calculate our needed probability by Bayes' rule

$$\mathbb{P}(Y = j | \tilde{Y} = j) = \mathbb{P}(\tilde{Y} = j | Y = j) \cdot \mathbb{P}(Y = j) / \mathbb{P}(\tilde{Y} = j),$$

where $\mathbb{P}(\tilde{Y} = j)$ can be estimated by counting the frequency of noisy label j in \tilde{D} . Technically other methods exist in the literature to estimate $\mathbb{P}(\tilde{Y} | Y)$ [119, 146, 141, 111]. But they often require training a model to fit the data distribution, which conflict with our goal of a training-free solution; instead, HOC fits us perfectly.

Algorithm 4 Detection with **Similar Features** (The SimiFeat Detector)

1: **Input:** Number of epochs: M . k -NN parameter: k . Noisy dataset: $\tilde{D} = \{(x_n, \tilde{y}_n)\}_{n \in [N]}$.
Feature extractor: $g(\cdot)$. Method: *Vote* or *Rank*. Epoch counters $m = 0$.

2: **repeat**

3: $x'_n \leftarrow \text{RandPreProcess}(x_n), \forall n;$ *# Initialize & Standard data augmentations*

4: $x_n \leftarrow g(x'_n), \forall n;$ *# For tasks with rarely clusterable features, extract features with $g(\cdot)$*

5: $\hat{\mathbf{y}}_n \leftarrow \text{kNNLabel}(\{x_n\}_{n \in [N]}, k)$ *# Get soft labels.*

6: **if** *Vote* **then**

7: $y_n^{\text{vote}} \leftarrow \arg \max_{i \in [K]} \hat{y}_n[i];$ *# Apply local majority vote*

8: $v_n \leftarrow \mathbb{1}(y_n^{\text{vote}} \neq \tilde{y}_n), \forall n \in [N];$ *# Corrupted if majority votes disagree with noisy labels*

9: **else**

10: $\mathbb{P}(Y), \mathbb{P}(\tilde{Y}|Y) \leftarrow \text{HOC}(\{(x_n, \tilde{y}_n)\}_{n \in [N]});$ *# Estimate clean priors $\mathbb{P}(Y)$ and noise transitions $\mathbb{P}(\tilde{Y}|Y)$ by HOC*

11: $\mathbb{P}(Y|\tilde{Y}) = \mathbb{P}(\tilde{Y}|Y) \cdot \mathbb{P}(Y) / \mathbb{P}(\tilde{Y});$ *# Estimate thresholds by Bayes' rule*

12: **for** j **in** $[K]$ **do**

13: $\mathcal{N}_j := \{n | \tilde{y}_n = j\};$ *# Detect corrupted labels in each set \mathcal{N}_j*

14: $\mathcal{I} \leftarrow \text{argsort}\{\text{Score}(\hat{\mathbf{y}}_n, j)\}_{n \in \mathcal{N}_j};$ *# \mathcal{I} records the raw index of each sorted value*

15: $v_n \leftarrow \mathbb{1}(\text{Loc}(n, \mathcal{I}) \leq \lfloor (1 - \mathbb{P}(Y = j | \tilde{Y} = j)) \cdot N_j \rfloor);$ *# Low-score instances are corrupted*

16: **end for**

17: **end if**

18: $\mathcal{V}_m = \{v_n\}_{n \in [N]};$ *# Record detection results in the m -th epoch*

19: **until** M times

20: $\mathcal{V} = \text{Vote}(\mathcal{V}_m, \forall m \in [M]);$ *# Do majority vote based on results from M epochs*

21: **Output:** $[N] \setminus \mathcal{V}$.

3.2.1.4 Algorithm: SimiFeat

Algorithm 4 summarizes our solution. The main computation complexity is pro-processing features with extractor $g(\cdot)$, which is less than the cost of evaluating the model compared with the training-based methods. Thus SimiFeat can filter out corrupted instances efficiently. In Algorithm 4, we run either voting-based local detection as Lines 7, 8, or ranking-based global detection as Lines 14, 15. The detection is run multiple times with random standard data augmentations to reduce the variance of estimation. The majority of results from different epochs is adopted as the final detection output as Line 20, i.e., flag as corrupted if $v_n = 1$ in more than half of the epochs.

3.2.2 How Does Feature Quality Affect Our Solution?

In this section, we will first show how the quality of features affects the selection of the hyperparameter k , then analyze the error upper bound for the ranking-based method. Note the voting-based method achieves an F_1 -score of 1 when $(k, 0)$ -NN label clusterability, $k \rightarrow +\infty$, holds.

3.2.2.1 How Does Feature Quality Affect the Choice of k ?

Recall k is used as illustrated in Figure 3.5. On one hand, the k -NN label estimator will be more accurate if there is stronger clusterability that more neighbor features belong to the same true class [125, 230], which helps improve the performance of later algorithms. On the other hand, with good but imperfect features, stronger clusterability with a larger k is less likely to satisfy, thus the violation probability δ_k

increases with k for a given extractor $g(\cdot)$. We take the voting-based method as an example and analyze this tradeoff. For a clean presentation, we focus on a binary classification with instance-dependent label noise where $\mathbb{P}(Y = 1) = p$, $\mathbb{P}(\tilde{Y} = 2|X, Y = 1) = e_1(X)$, $\mathbb{P}(\tilde{Y} = 1|X, Y = 2) = e_2(X)$. Suppose the instance-dependent noise rate is upper-bounded by e , i.e., $e_1(X) \leq e, e_2(X) \leq e$. With δ_k as in Definition 2, we calculate the lower bound of the probability that the vote is correct in Proposition 1.

Proposition 1. *The lower bound for the probability of getting true detection with majority vote is*

$$\mathbb{P}(\text{Vote is correct}|k) \geq (1 - \delta_k) \cdot I_{1-e}(k + 1 - k', k' + 1),$$

where $k' = \lceil (k + 1)/2 \rceil - 1$, $I_{1-e}(k + 1 - k', k' + 1)$ is the regularized incomplete beta function defined as $I_{1-e}(k + 1 - k', k' + 1) = (k + 1 - k') \binom{k+1}{k'} \int_0^{1-e} t^{k-k'} (1-t)^{k'} dt$.

Proposition 1 shows the tradeoff between a reliable k -NN label and an accurate vote. When k is increasing, **Term-1** $(1 - \delta_k)$ (quality of features) decreases but **Term-2** $I_{1-e_1}(k + 1 - k', k' + 1)$ (result of pure majority vote) increases. With Proposition 1, we are ready to answer the question: *when do we need more labels?* See Remark 3.

Remark 3. *Consider the lower bounds with k_1 and k_2 ($k_1 < k_2$). Supposing the first lower bound is lower than the second lower bound, based on Proposition 1, we roughly study the trend with an increasing k by comparing two bounds and get*

$$\frac{1 - \delta_{k_1}}{1 - \delta_{k_2}} < \frac{I_{1-e}(k_2 + 1 - k'_2, k'_2 + 1)}{I_{1-e}(k_1 + 1 - k'_1, k'_1 + 1)}.$$

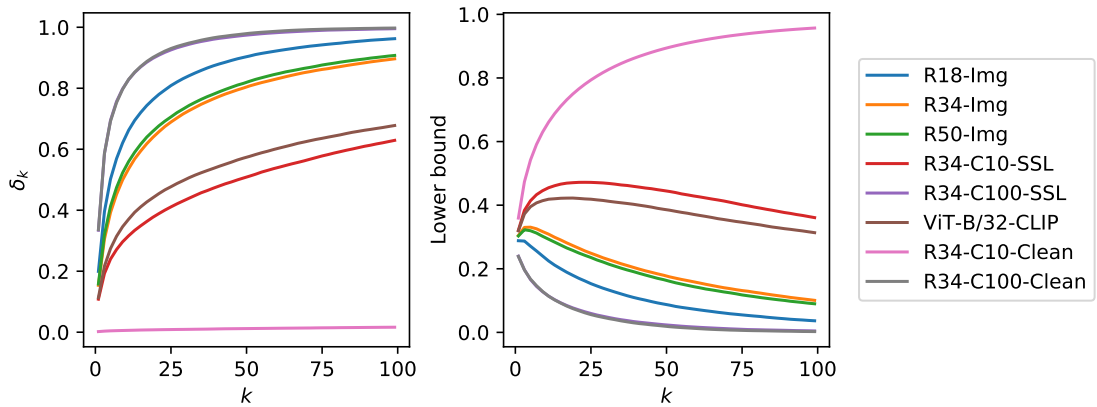


Figure 3.6: The trends of δ_k and probability lower bounds on CIFAR-10

For example, when $k_1 = 5$, $k_2 = 20$, $e = 0.4$, we can calculate the incomplete beta function and $\frac{1-\delta_5}{1-\delta_{20}} < 1.52$. Supposing $\delta_5 = 0.2$, we have $\delta_{20} < 0.47$. This indicates increasing k from 5 to 20 would not improve the lower bound with features satisfying $\delta_{20} > 0.47$. This observation helps us set k with practical and imperfect features. We set $k = 10$ in all of our experiments.

Remark 3 indicates that: ***with practical (imperfect) features, a small k may achieve the best (highest) probability lower bound.*** To further consolidate this claim, we numerically calculate δ_k with different quality of features on CIFAR-10 [99] and the corresponding probability lower bound in Figure 3.6. The raw features are extracted with different $g(\cdot)$. The outputs of the last convolution layer are adopted. Notations R18/34/50 indicate ResNet18/34/50. Notation Img stands for the extractor that is pre-trained on ImageNet [38]. Notations C10/100-Clean are the extractors pre-trained on clean CIFAR-10/100. Notations C10/100-SSL are extractors pre-trained on CIFAR-10/100 without labels by SimCLR [28]. Notation ViT-B/32-CLIP indicates that

CLIP model, pre-trained vision transformer [151]. We find most of the probability lower bounds first increase then decrease except for the “perfect” feature which is extracted by the extractor trained using ground-truth labels. Note this feature extractor has memorized all clean instances so that $\delta_k \rightarrow 0$ since $k \ll 5000$ (the number of instances in the same label class).

3.2.2.2 How Does Feature Quality Affect F_1 -Score?

We next prove the probability bound for the performance of the ranking-based method. Consider a K -class classification problem with informative instance-dependent label noise [30]. Denote random variable S by the score of each instance being clean. A higher score S indicates the instance is more likely to be clean. Denote the score of a true/false instance by $S_{n,j}^{\text{true}} := \text{Score}(\hat{\mathbf{y}}_n, j)$ when $\tilde{y}_n = y_n = j$ and $S_{n',j}^{\text{false}} := \text{Score}(\hat{\mathbf{y}}_{n'}, j)$ when $\tilde{y}_{n'} = j, y_{n'} \neq j$. Both are scalars. Then for instances in \mathcal{N}_j , we have two set of random variables $\mathbb{S}_j^{\text{true}} := \{S_{n,j}^{\text{true}} | n \in \mathcal{N}_j, \tilde{y}_n = y_n = j\}$ and $\mathbb{S}_j^{\text{false}} := \{S_{n',j}^{\text{false}} | n' \in \mathcal{N}_j, \tilde{y}_{n'} = j, y_{n'} \neq j\}$. Recall $\mathcal{N}_j := \{n | \tilde{y}_n = j\}$ are the set of indices that correspond to noisy class j . Intuitively, the score $S_{n,j}^{\text{true}}$ should be greater than $S_{n',j}^{\text{false}}$. Suppose their means, which depend on noise rates, are bounded, i.e.,

$$\mathbb{E}[S_{n,j}^{\text{true}}] \geq \mu_j^{\text{true}}, \quad \mathbb{E}[S_{n',j}^{\text{false}}] \leq \mu_j^{\text{false}}$$

for all feasible n, n' . Assume there exists a feasible v such that both $\mathbb{S}_j^{\text{true}}$ and $\mathbb{S}_j^{\text{false}}$ follow sub-Gaussian distributions with variance proxy $\frac{\Delta^2}{2v}$ [21, 233] such that:

$$\mathbb{P}(\mu_j^{\text{true}} - S_{n,j}^{\text{true}} \geq t) \leq e^{-\frac{vt^2}{\Delta^2}}, \quad \mathbb{P}(S_{n',j}^{\text{false}} - \mu_j^{\text{false}} \geq t) \leq e^{-\frac{vt^2}{\Delta^2}},$$

and the probability density satisfies $\mathbb{P}(S_j^{\text{true}} = \mu_j^{\text{true}}) = \mathbb{P}(S_j^{\text{false}} = \mu_j^{\text{false}}) = 1/\Delta$, where $1/\Delta$ is the “height” of both distributions, v is the decay rate of tails. Let N_j^- (N_j^+) be the number of indices in $\mathbb{S}_j^{\text{false}}$ ($\mathbb{S}_j^{\text{true}}$). Theorem 9 summarizes the performance bound of the ranking-based method. See Appendix for the proof.

Theorem 9. *With probability at least p , when the threshold for the ranking-based method is set to*

$$(1 - \mathbb{P}(Y = j | \tilde{Y} = j)) \cdot N_j$$

as Line 15, the F_1 -score of detecting corrupted instances in \mathcal{N}_j by ranking is at least $1 - \frac{e^{-v} \max(N^-, N^+) + \alpha}{N^-}$, where $p = \int_{-1}^{\mu_j^{\text{true}} - \mu_j^{\text{false}} - \Delta} f(t) dt$, $f(t)$ is the probability density function of the difference of two independent beta-distributed random variables $\beta_1 - \beta_2$, where $\beta_1 \sim \text{Beta}(N^-, 1)$, $\beta_2 \sim \text{Beta}(\alpha + 1, N^+ - \alpha)$.

Theorem 1 shows the detection performance depends on:

- the concentration of $S_{n,j}^{\text{true}}$ and $S_{n',j}^{\text{false}}$: variance proxy $\frac{\Delta^2}{2v}$;
- the distance between $S_{n,j}^{\text{true}}$ and $S_{n',j}^{\text{false}}$: $\mu_j^{\text{true}} - \mu_j^{\text{false}}$.

Intuitively, with proper scoring function and high-quality features, we have small variance proxy (small Δ and large v) and F_1 -score approximates to 1.

3.2.3 Empirical Results

We present experimental evidence in this section. The performance is measured by the F_1 -score of the detected corrupted labels as defined in Section 1.2.2.2. Note there is no training procedure in our method. The only hyperparameters in our methods are

the number of epochs M and the k -NN parameter k . Intuitively, a larger M returns a collective result from more times of detection, which should be more accurate. But a larger M takes more time. We set $M = 21$ (an odd number for better tie-breaking) for an efficient solution. The hyperparameter k cannot be set too large as demonstrated in Figure 3.6. From Figure 3, we notice that the lower bound (RHS figure) is relatively high when $k = 10$ for all settings. Therefore, in CIFAR [99] experiments, rather than fine-tune M and k for different settings, we fix $M = 21$ and $k = 10$. We also test on Clothing1M [200]. Detailed experiment settings on Clothing1M are in Appendix B.3.

Synthetic label noise We experiment with three popular synthetic label noise models: the *symmetric* label noise, the *asymmetric* label noise, and the *instance-dependent* label noise. Denote the ratio of instances with corrupted labels in the whole dataset by η . Both the symmetric and the asymmetric noise models follow the class-dependent assumption [119], i.e., the label noise only depends only on the clean class: $\mathbb{P}(\tilde{Y}|X, Y) = \mathbb{P}(\tilde{Y}|Y)$. Specially, the symmetric noise is generated by uniform flipping, i.e., randomly flipping a true label to the other possible classes w.p. η [30]. The asymmetric noise is generated by pair-wise flipping, i.e., randomly flipping true label i to the next class $(i \bmod K) + 1$. Denote by d the dimension of features. The instance-dependent label noise is synthesized by randomly generating a $d \times K$ projection matrix w_i for each class i and project each incoming feature with true class y_n onto each column of w_{y_n} [198]. Instance n is more likely to be flipped to class j if the projection value of x_n on the j -th column of w_{y_n} is high. See Appendix B in [198] and Appendix D.1 in [230] for more details. We use

symmetric noise with $\eta = 0.6$ (*Symm. 0.6*), asymmetric noise with $\eta = 0.3$ (*Asym. 0.3*), and instance-dependent noise with $\eta = 0.4$ (*Inst. 0.4*) in experiments.

Real-world label noise The *real-world* label noise comes from human annotations or weakly labeled web data. We use the 50,000 noisy training labels ($\eta \approx 0.16$) for CIFAR-10 collected by [230], and 50,000 noisy training labels ($\eta \approx 0.40$) for CIFAR-100 collected by [188]. Both sets of noisy labels are crowd-sourced from Amazon Mechanical Turk. For Clothing1M [200], we could not calculate the F_1 -scores due to the lack of ground-truth labels. We firstly perform noise detection on 1 million noisy training instances then train only with the selected clean data to check the effectiveness.

3.2.3.1 Fitting Noisy Distributions May Not Be Necessary

Our first experiment aims to show that fitting the noisy data distribution may not be necessary in detecting corrupted labels. To this end, we compare our methods, i.e., voting-based local detection (SimiFeat-V) and ranking-based global detection (SimiFeat-R), with three learning-centric noise detection works: CORES [30], confident learning (CL) [141], TracIn [149], and deep k -NN [10]. We use ResNet34 [69] as the backbone network in this experiment.

Baseline settings All these three baselines require training a model with the noisy supervision. Specifically, CORES [30] trains ResNet34 on the noisy dataset and uses its proposed sample sieve to filter out the corrupted instances. We adopt its default setting during training and calculate the F_1 -score of the sieved out corrupted instances.

Confident learning (CL) [141] detects corrupted labels by firstly estimating probabilistic thresholds to characterize label noise, ranking instances based on model predictions, then filtering out corrupted instances based on ranking and thresholds. We adopt its default hyper-parameter setting to train ResNet34. TracIn [149] detects corrupted labels by evaluating the self-influence of each instance, where the corrupted instances tend to have a high influence score. The influence scores are calculated based on gradients of the last layer of ResNet34 at epoch 40, 50, 60, 100, where the model is trained with a batch size of 128. The initial learning rate is 0.1 and decays to 0.01 at epoch 50. Note TracIn only provides ranking for instances. To exactly detect corrupted instances, thresholds are required. For a fair comparison, we refer to the thresholds learned by confident learning [141]. Thus the corrupted instances selected by TracIn are based on the ranking from its self-influence and thresholds from CL. To highlight that our solutions work well without any supervision, our feature extractor $g(\cdot)$ comes from the ResNet34 pre-trained by SimCLR [28] where contrastive learning is applied and *no supervision* is required. Extractor $g(\cdot)$ is obtained with only in-distribution features, e.g., for experiments with CIFAR-10, $g(\cdot)$ is pre-trained with features only from CIFAR-10. The detailed implementation for deep k -NN filter [10] is not public. Noting their k -NN approach is employed on the logit layer, we reproduce their work by firstly training the model on the noisy data then substituting the model logits for $g(\cdot)$ in SimiFeat-V. The best epoch result for deep k -NN is reported.

Table 3.2: Comparisons of F_1 -scores (%).

METHOD	CIFAR10				CIFAR100			
	<i>Human</i>	<i>Symm. 0.6</i>	<i>Asym. 0.3</i>	<i>Inst. 0.4</i>	<i>Human</i>	<i>Symm. 0.6</i>	<i>Asym. 0.3</i>	<i>Inst. 0.4</i>
CORES	65.00	92.94	7.68	87.43	3.52	92.34	0.02	9.67
CL	55.85	80.59	76.45	62.89	64.58	78.98	52.96	50.08
TRACIN	55.02	76.94	73.47	58.85	61.75	76.74	48.42	49.89
DEEP k -NN	56.21	82.35	75.24	63.08	57.40	70.69	56.75	63.85
SIMIFEAT-V	82.30	93.21	82.52	81.09	73.19	84.48	65.42	74.26
SIMIFEAT-R	83.28	95.56	83.58	82.26	74.67	88.68	62.89	73.53

Performance Table 3.2 compares the results obtained with or without supervision. Methods CORES, CL, and TracIn are trained with noisy supervision. Methods SimiFeat-V and SimiFeat-R get $g(\cdot)$ **without** any supervision. The top-2 are **bold**. We can see both the voting-based and the ranking-based methods achieve overall higher F_1 -scores compared with the other three results that require learning with noisy supervision. Moreover, in detecting the real-world human-level noisy labels, our solution outperforms baselines around 20% on CIFAR-10 and 10% on CIFAR-100, which indicates the training-free solution is more robust to complicated noise patterns. One might also note that CORES achieves exceptionally low F_1 -scores on CIFAR-10/100 with asymmetric noise and CIFAR-100 with human noise. This observation also informs us that customized training processes might not be universally applicable.

3.2.3.2 Features May Be Better Than Model Predictions

Our next experiment aims to compare the performance of the data-centric method with the learning-centric method when the same feature extractor is adopted. Thus in this experiment, all methods adopt the same fixed feature extractor (ViT-B/32 pre-trained by CLIP [151]). Our proposed data-centric method directly operates on

Table 3.3: Comparisons of F_1 -scores (%).

METHOD	CIFAR10				CIFAR100			
	<i>Human</i>	<i>Symm. 0.6</i>	<i>Asym. 0.3</i>	<i>Inst. 0.4</i>	<i>Human</i>	<i>Symm. 0.6</i>	<i>Asym. 0.3</i>	<i>Inst. 0.4</i>
CE SIEVE	67.21	94.56	5.24	8.41	16.24	88.55	2.6	1.63
CORES	83.18	96.94	12.05	88.89	38.52	92.33	7.02	85.52
CL	69.76	95.03	77.14	62.91	67.64	85.67	62.58	61.53
TRACIN	81.85	95.96	80.75	64.97	79.32	91.03	63.12	64.31
DEEP k -NN	82.98	87.47	76.96	77.42	72.33	82.95	64.96	74.25
SIMIFEAT-V	87.43	96.44	88.97	87.11	76.26	86.88	73.50	80.03
SIMIFEAT-R	87.45	96.74	89.04	91.14	79.21	90.54	68.14	77.37

the extracted features, while the learning-centric method further trains a linear layer with noisy supervisions based on the extracted features. In addition to the baselines compared in Section 3.2.3.1, we also compare to CE Sieve [30] which follows the same sieving process as CORES but uses CE loss without regularizer. Other settings are the same as those in Section 3.2.3.1.

Table 3.3 summarizes the results of this experiment. Methods CORES, CL, and TracIn use logit layers, while SimiFeat-V/R use only representations. **All methods use the same fixed extractor from CLIP.** The Top-2 are **bold**. By counting the frequency of reaching top-2 F_1 -scores, we find SimiFeat-R wins 1st place, SimiFeat-V and CORES are tied for 2nd place. However, similar to Table 3.3, we find the training process of CORES to be unstable. For instance, it almost fails for CIFAR-100 with asymmetric noise. Besides, comparing deep k -NN with SimiFeat-V, we find using the model logits given by an additional linear layer fine-tuned with noisy supervisions cannot always help improve the performance of detecting corrupted labels. It is therefore reasonable to believe both methods that directly deal with the extracted features achieve an overall higher F_1 -score than other learning-centric methods.

Table 3.4: Comparisons of F_1 -scores (%) using $g(\cdot)$ with different δ_k (%). Model names are the same as Figure 3.6.

PRE-TRAINED MODEL	CIFAR10			CIFAR100		
	$1 - \delta_k$	<i>Human</i>	<i>Inst. 0.4</i>	$1 - \delta_k$	<i>Human</i>	<i>Inst. 0.4</i>
R18-IMG	35.73	75.40	80.22	11.30	74.91	71.99
R34-IMG	48.13	79.52	82.43	16.17	76.88	74.00
R50-IMG	45.77	78.40	82.06	15.81	76.55	73.51
ViT-B/32-CLIP	64.12	87.45	91.14	19.94	79.21	77.37
R34-C10-SSL	69.31	83.28	85.26	2.59	68.03	65.94
R34-C10-CLEAN	99.41	98.39	98.59	0.22	60.90	60.73
R34-C100-SSL	18.59	59.96	74.99	22.46	74.67	73.53
R34-C100-CLEAN	18.58	60.17	76.41	89.07	92.87	95.29

3.2.3.3 The Effect of the Quality of Features

Previous experiments demonstrate our methods overall outperform baselines with high-quality features. It is interesting to see how lower-quality features perform. We summarize results of SimiFeat-R in Table 3.4. There are several interesting findings: 1) The ImageNet pre-trained models perform well, indicating the traditional supervised training on out-of-distribution data helps obtain high-quality features; 2) For CIFAR-100, extractor $g(\cdot)$ obtained with only features from CIFAR-10 (R34-C10-SSL) performs better than the extractor with clean CIFAR-10 (R34-C10-Clean), indicating that contrastive pre-training has better generalization ability to out-of-distribution data than supervised learning; 3) The F_1 -scores achieved by $g(\cdot)$ trained with the corresponding clean dataset are close to 1, indicating our solution can give perfect detection with ideal features.

Table 3.5: Experiments on Clothing1M.

DATA SELECTION	# TRAINING	BEST EPOCH	LAST 10	LAST
NONE	1M (100%)	70.32	69.44 \pm 0.13	69.53
R50-IMG	770K (77.0%)	72.37	71.95 \pm 0.08	71.89
ViT-B/32-CLIP	700K (70.0%)	72.54	72.23 \pm 0.17	72.11
R50-IMG WARMUP-1	767K (76.7%)	73.64	73.28 \pm 0.18	73.41

3.2.3.4 More Experiments on Real-World Large-Scale Datasets

Besides, we test the performance of training only with the clean instances selected by our approach in Table 3.5. There are different data selection methods:

- None: Standard training with 1M noisy data.
- R50-Img (or ViT-B/32-CLIP, R50-Img Warmup-1): Apply our method with ResNet50 pre-trained on ImageNet (or ViT-B/32 pre-trained by CLIP, R50-Img with 1-epoch warmup).

The clean test accuracy on the best epoch, the last 10 epochs, and the last epoch are listed. Top-1 is **bold**. Standard training with Cross-Entropy loss is adopted. The only difference between the first row and other rows of Table 3.5 is that some training instances are filtered out by our approach. Table 3.5 shows simply filtering out corrupted instances based on our approach distinctively outperforms the baseline. We also observe that slightly tuning $g(\cdot)$ in the fine-grained Clothing1M dataset would be helpful. Note the best-epoch test accuracy we can achieve is 73.64%, which outperforms many baselines such as HOC 73.39% [230], GCE+SimCLR 73.35% [56], CORES 73.24% [30], GCE 69.75% [221]. See more detailed settings and discussions in Appendix B.3.

Chapter 4

Learning After Label Error Detection

This chapter discusses some possibilities of robust learning algorithms when the label error in the original dataset has been detected. The simplest way is to train a model on a cleaned dataset by directly removing all the detected corrupted examples, as shown in Table 3.1 of Section 3.1.4.2 and Table 4.3 of Section 3.2.3.4. One limitation of this approach is the waste of information in the mislabeled features. In this chapter, we will introduce two advanced methods that are designed to better use the features of mislabeled examples. Specifically, in Section 4.1, we adopt semi-supervised learning to treat the detected corrupted examples as unlabeled samples, and in Section 4.2, we design a robust loss function that uses the second-order information derived from the label error detection results.

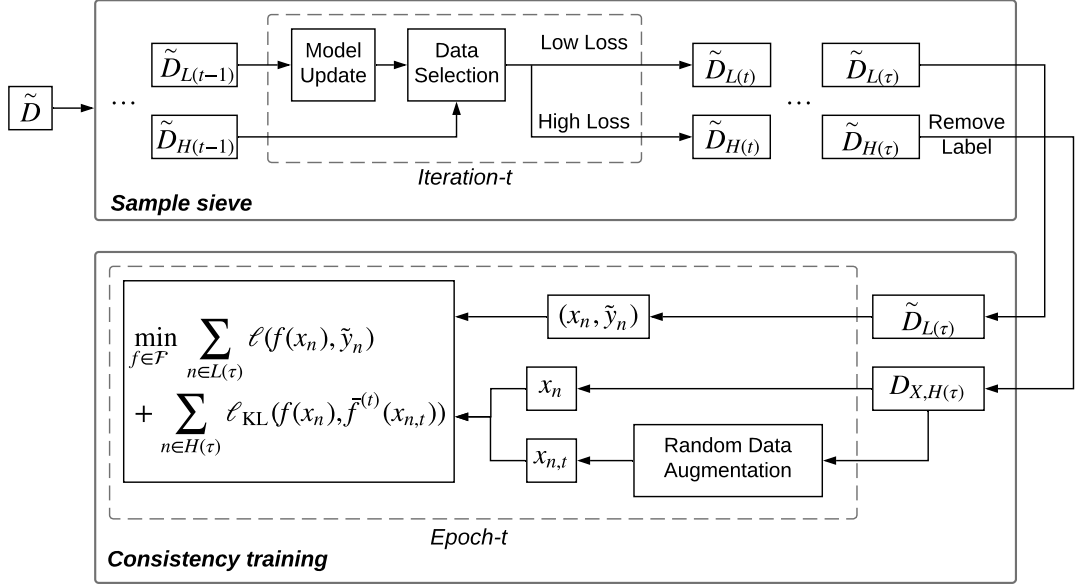


Figure 4.1: Implement semi-supervised learning with CORES².

4.1 A Semi-Supervised Learning Approach

This section shows how a semi-supervised learning (SSL) approach is combined with CORES² to further improve the model training. The clean examples are used for normal training and the corrupted ones are used as unlabeled data for SSL.

The training framework is shown in Figure 4.1, where $L(t)$ includes the indices of sieved clean examples and $H(t)$ contains the indices of sieved corrupted examples. Let τ be the last iteration of CORES². We have $L(\tau) := \{n | n \in [N], v_n^{(\tau)} = 1\}$ and $H(\tau) := \{n | n \in [N], v_n^{(\tau)} = 0\}$. Then $\tilde{D}_{L(\tau)} := \{(x_n, \tilde{y}_n) : n \in L(\tau)\}$ is sieved as clean examples and $\tilde{D}_{H(\tau)} := \{(x_n, \tilde{y}_n) : n \in H(\tau)\}$ is filtered out as corrupted ones. Examples $(x_n, \tilde{y}_n) \in \tilde{D}_{L(\tau)}$ lead the training direction using the CE loss as

$\sum_{n \in L(\tau)} \ell(f(x_n), \tilde{y}_n)$. Noting the labels in $\tilde{D}_{H(\tau)}$ are supposed to be corrupted and can distract the training, we simply drop them. On the other hand, the feature information of these examples encodes useful information that we can further leverage to improve the generalization ability of models. There are different ways to use this unsupervised information, in this section, we chose to minimize the KL-divergence between predictions on the original feature and the augmented feature to make predictions consistent. This is a common option as chosen by [109], [201], and [223]. The consistency loss function in epoch- t is $\sum_{n \in H(\tau)} \ell_{\text{KL}}(f(x_n), \bar{f}^{(t)}(x_{n,t}))$, where $\bar{f}^{(t)}$ is a copy of the DNN at the beginning of epoch- t but without gradients. Summing the classification and consistency loss yields the total loss. We call this semi-supervised learning version CORES^{2*}.

Experiments When performing CORES^{2*}, we used the sieved result at epoch-40. It is worth noting that at that time, the sample sieve may not reach the highest test accuracy. However, the division property brought by the confidence regularizer works well at that time. We use the default setting from UDA [201] to apply efficient data augmentation. Table 4.1 shows the comparison of test accuracies under synthetic instance-dependent label noise (*Inst. CIFAR10*, *Inst. CIFAR100*) and real-world human annotations from CIFAR-N [188]. The results show that applying SSL after label error detection can significantly improve performance.

Table 4.1: Comparison of test accuracies under instance-dependent label noise.

Method	<i>Inst. CIFAR10</i>			<i>Inst. CIFAR100</i>		
	$\varepsilon = 0.2$	$\varepsilon = 0.4$	$\varepsilon = 0.6$	$\varepsilon = 0.2$	$\varepsilon = 0.4$	$\varepsilon = 0.6$
CORES ²	89.50	82.84	79.66	61.25	47.81	37.85
CORES ^{2*}	95.42	88.45	85.53	72.91	70.66	63.08

Method	<i>CIFAR10-N</i>				
	Aggregate	Random 1	Random 2	Random 3	Worst
CORES ²	91.23±0.11	89.66±0.32	89.91±0.45	89.79±0.50	83.60±0.53
CORES ^{2*}	95.25±0.09	94.45±0.14	94.88±0.31	94.74±0.03	91.66±0.09

4.2 A Second-Order Approach

This section studies the insufficiency of the first-order approach, such as peer loss [124, 30] and explores the possibility of designing robust loss functions with higher-order information [227]. The discussion in this section focuses on the connection between the Bayes optimal label Y^* and the label \tilde{Y} with instance-dependent noise (IDN) defined as follows.

Bayes optimal distribution \mathcal{D}^* Denote by Y^* the Bayes optimal label given feature X , that is: $Y^*|X := \arg \max_Y \mathbb{P}(Y|X)$, $(X, Y) \sim \mathcal{D}$. The distribution of (X, Y^*) is denoted by \mathcal{D}^* . Note the Bayes optimal distribution \mathcal{D}^* is different from the clean distribution \mathcal{D} when $\mathbb{P}(Y|X) \notin \{0, 1\}$. Due to the fact that the information encoded between features and labels is corrupted by label noise, and both clean labels and Bayes optimal labels are unobservable, inferring the Bayes optimal distribution \mathcal{D}^* from the noisy dataset $\tilde{\mathcal{D}}$ is a non-trivial task. Notably there exist two approaches [30] that provide guarantees on constructing the Bayes optimal dataset. We would like to remind

the readers that the noisy label \tilde{y}_n , clean label y_n , and Bayes optimal label y_n^* for the same feature x_n may disagree with each other.

Most of our developed approaches will focus on dealing with the Bayes optimal distribution \mathcal{D}^* . By referring to \mathcal{D}^* , as we shall see later, we are allowed to estimate the second-order statistics defined w.r.t. Y^* .

Noise transition matrix $T(X)$ Traditionally, the noise transition matrix is defined based on the relationship between clean distributions and noisy distributions [30, 124, 146, 198]. In recent literature [33], the Bayes optimal label (a.k.a. distilled label in [33]) also plays a significant role. In the classification tasks where the performance is measured by the clean test accuracy, predicting the Bayes optimal label achieves the best performance. This fact motivates us to define a new noise transition matrix based on the Bayes optimal label as follows:

$$T_{i,j}(X) = \mathbb{P}(\tilde{Y} = j | Y^* = i, X),$$

where $T_{i,j}(X)$ denotes the (i, j) -th element of the matrix $T(X)$. Its expectation is defined as $T := \mathbb{E}[T(X)]$, with the (i, j) -th element being $T_{i,j} := \mathbb{E}[T_{i,j}(X)]$.

4.2.1 Insufficiency of First-Order Statistics

Peer loss [124] and its inspired confidence regularizer [30] are two recently introduced robust losses that operate without the knowledge of noise transition matrices, which presents them as preferred solutions for more complex noise settings. In this section, we will first review the usages of first-order statistics in peer loss and the

confidence regularizer (Section 4.2.1.1), and then analyze the insufficiency of using only the first-order statistics when handling the challenging IDN (Section 4.2.1.2). Besides, we will anatomize the down-weighting effect of IDN and provide intuitions for how to make IDN easier to handle (Section 4.2.1.3).

We formalize our arguments using peer loss, primarily due to 1) its clean analytical form, and 2) that our later proposed solution will be built on peer loss too. Despite the focus on peer loss, we believe these observations are generally true when other existing training approaches meet IDN.

For ease of presentation, the following analyses focus on binary cases (with classes $\{-1, +1\}$). Note the class -1 should be mapped to class 0. For a clear comparison with previous works, we follow the notation in [124] and use class $\{-1, +1\}$ to represent classes $\{0, 1\}$ when $K = 2$. The error rates in \tilde{Y} are then denoted as $e_+(X) := \mathbb{P}(\tilde{Y} = -1|Y^* = +1, X)$, $e_-(X) := \mathbb{P}(\tilde{Y} = +1|Y^* = -1, X)$. Most of the discussions generalize to the multi-class setting.

4.2.1.1 Using First-Order Statistics in Peer Loss

It has been proposed and proved in peer loss [124] and CORES² [30] that the learning could be robust to label noise by considering some first-order statistics related to the model predictions. For each example (x_n, \tilde{y}_n) , peer loss [124] has the following form:

$$\ell_{\text{PL}}(f(x_n), \tilde{y}_n) := \ell(f(x_n), \tilde{y}_n) - \ell(f(x_{n_1}), \tilde{y}_{n_2}),$$

where $(x_{n_1}, \tilde{y}_{n_1})$ and $(x_{n_2}, \tilde{y}_{n_2})$ are two randomly sampled peer samples for n . The first-order statistics related to model predictions characterized by the peer term $\ell(f(x_{n_1}), \tilde{y}_{n_2})$ are further extended to a confidence regularizer in CORES² [30]:

$$\ell_{\text{CORES}^2}(f(x_n), \tilde{y}_n) := \ell(f(x_n), \tilde{y}_n) - \beta \mathbb{E}_{\mathcal{D}_{\tilde{Y}|\tilde{D}}}[\ell(f(x_n), \tilde{Y})],$$

where β is a hyperparameter controlling the ability of regularizer, and $\mathcal{D}_{\tilde{Y}|\tilde{D}}$ is the marginal distribution of \tilde{Y} given dataset \tilde{D} . Although it has been shown in [30] that learning with an appropriate β would be robust to instance-dependent label noise theoretically, in real experiments, converging to the guaranteed optimum by solving a highly non-convex problem is difficult.

4.2.1.2 Peer Loss with IDN

Now we analyze the possible performance degradation of using the binary peer loss function proposed in [124] to handle IDN. Denote by

$$\tilde{f}_{\text{peer}}^* := \arg \min_f \mathbb{E}_{\tilde{\mathcal{D}}} [\mathbb{1}_{\text{PL}}(f(X), \tilde{Y})]$$

the optimal classifier learned by minimizing 0-1 peer loss, where $\mathbb{1}_{\text{PL}}$ represents ℓ_{PL} with 0-1 loss (could also be generalized for ℓ_{CORES^2} with 0-1 loss). Let $p^* := \mathbb{P}(Y^* = +1)$.

With a bounded variance in the error rates, supposing

$$\mathbb{E}|e_+(X) - \mathbb{E}[e_+(X)]| \leq \epsilon_+, \quad \mathbb{E}|e_-(X) - \mathbb{E}[e_-(X)]| \leq \epsilon_-,$$

the worst-case performance bound for using pure peer loss is provided in Theorem 10 and proved in Appendix C.2.1.

Theorem 10 (Performance of peer loss). *With the peer loss function proposed in [124], we have*

$$\mathbb{E}[\mathbf{1}(\tilde{f}_{peer}^*(X), Y^*)] \leq \frac{2(\epsilon_+ + \epsilon_-)}{1 - e_+ - e_-} + 2|p^* - 0.5|.$$

Theorem 10 shows the ratio of wrong predictions given by \tilde{f}_{peer}^* includes two components. The former term $\frac{2(\epsilon_+ + \epsilon_-)}{1 - e_+ - e_-}$ is directly caused by IDN, indicating the error is increasing when the instance-dependent noise rates have larger mean (larger $e_+ + e_-$) and larger variation (larger $\epsilon_+ + \epsilon_-$). The latter term $2|p^* - 0.5|$ shows possible errors induced by an unbalanced \mathcal{D}^* . Theorem 10 generalizes peer loss where $\epsilon_+ = \epsilon_- = 0$, i.e., the error rates are homogeneous across data instances, and there is no need to consider any second-order statistics that involve the distribution of noise rates.

4.2.1.3 Down-weighting Effect of IDN

We further discuss motivations and intuitions by studying how IDN affects the training differently from the class-dependent one. Intuitively, a high noise rate reduces the informativeness of a particular example (x, y) , therefore “down-weighting” its contribution to training. We now analytically show this under peer loss.

As a building block, the invariant property (in terms of the *clean distribution* \mathcal{D}) originally discovered by peer loss on class-dependent label noise is first adapted for the *Bayes optimal distribution* \mathcal{D}^* . Define $e_- := \mathbb{P}(\tilde{Y} = +1 | Y^* = -1)$ and $e_+ := \mathbb{P}(\tilde{Y} = -1 | Y^* = +1)$. Focusing on a particular class-dependent $\tilde{\mathcal{D}}$, we provide Lemma 4 and its proof in Appendix C.1.1.

Lemma 4 (Invariant property of peer loss [124]). *Peer loss is invariant to class-dependent label noise:*

$$\mathbb{E}_{\tilde{\mathcal{D}}}[1_{PL}(f(X), \tilde{Y})] = (1 - e_+ - e_-)\mathbb{E}_{\mathcal{D}^*}[1_{PL}(f(X), Y^*)]. \quad (4.1)$$

Then we discuss the effect of IDN. Without loss of generality, consider a case where noisy examples are drawn from two noisy distributions $\tilde{\mathcal{D}}_I$ and $\tilde{\mathcal{D}}_{II}$, and the noise rate of $\tilde{\mathcal{D}}_{II}$ is higher than $\tilde{\mathcal{D}}_I$, i.e. $e_{+,II} + e_{-,II} > e_{+,I} + e_{-,I}$, where $e_{+,I(II)} := \mathbb{P}_{\tilde{\mathcal{D}}_{I(II)}}(\tilde{Y} = -1 | Y^* = +1)$. Assume a particular setting of IDN that the noise is class-dependent (but not instance-dependent) only within each distribution, and different between two distributions, i.e. *part-dependent* [198]. Let \mathcal{D}_I^* and \mathcal{D}_{II}^* be the Bayes optimal distribution related to $\tilde{\mathcal{D}}_I$ and $\tilde{\mathcal{D}}_{II}$. For simplicity, we write $\mathbb{P}((X, Y^*) \sim \mathcal{D}_{I(II)}^* | (X, Y^*) \sim \mathcal{D}^*)$ as $\mathbb{P}(\mathcal{D}_{I(II)}^*)$. Then $\mathbb{P}(\mathcal{D}_I^*) = \mathbb{P}(\tilde{\mathcal{D}}_I)$ and $\mathbb{P}(\mathcal{D}_{II}^*) = \mathbb{P}(\tilde{\mathcal{D}}_{II})$. Note $\mathbb{P}(\tilde{\mathcal{D}}_I)e_{+,I} + \mathbb{P}(\tilde{\mathcal{D}}_{II})e_{+,II} = e_+$ and $\mathbb{P}(\tilde{\mathcal{D}}_I)e_{-,I} + \mathbb{P}(\tilde{\mathcal{D}}_{II})e_{-,II} = e_-$. Then we have the following equality:

$$\begin{aligned} & \mathbb{E}_{\tilde{\mathcal{D}}}[1_{PL}(f(X), \tilde{Y})] \\ &= \mathbb{P}(\tilde{\mathcal{D}}_I)(1 - e_{+,I} - e_{-,I})\mathbb{E}_{\mathcal{D}_I^*}[1_{PL}(f(X), Y^*)] + \mathbb{P}(\tilde{\mathcal{D}}_{II})(1 - e_{+,II} - e_{-,II})\mathbb{E}_{\mathcal{D}_{II}^*}[1_{PL}(f(X), Y^*)] \\ &= (1 - e_{+,I} - e_{-,I})\left(\mathbb{P}(\mathcal{D}_I^*)\mathbb{E}_{\mathcal{D}_I^*}[1_{PL}(f(X), Y^*)] + \frac{1 - e_{+,II} - e_{-,II}}{1 - e_{+,I} - e_{-,I}}\mathbb{P}(\mathcal{D}_{II}^*)\mathbb{E}_{\mathcal{D}_{II}^*}[1_{PL}(f(X), Y^*)]\right), \end{aligned}$$

where

$$\frac{1 - e_{+,II} - e_{-,II}}{1 - e_{+,I} - e_{-,I}} < 1$$

indicates *down-weighting* examples drawn from $\tilde{\mathcal{D}}_{II}$ (compared to the class-dependent label noise).

What can we learn from this observation? First, we show that peer loss is already down-weighting the importance of the more noisy examples. However, simply dropping examples with potentially high-level noise might lead the classifier to learn a biased

distribution. Moreover, subjectively confusing examples are more prone to be mislabeled and critical for accurate predictions [175], thus need to be carefully addressed. Our second observation is that if we find a way to compensate for the “imbalances” caused by the down-weighting effects shown above, the challenging instance-dependent label noise could be transformed into a class-dependent one, which existing techniques can then handle. More specifically, the above result shows the down-weighting effect is characterized by $T(X)$, implying only using the first-order statistics of model predictions without considering the distributions of the noise transition matrix $T(X)$ is insufficient to capture the complexity of the learning task. However, accurately estimating $T(X)$ is prohibitive since the number of parameters to be estimated is almost at the order of $O(NK^2)$ – recall N is the number of training examples and K is the number of classes. Even though we can roughly estimate $T(X)$, applying element-wise correction relying on the estimated $T(X)$ may accumulate errors. Therefore, to achieve the transformation from the instance-dependent to the easier class-dependent, we need to resort to other statistical properties of $T(X)$.

4.2.2 Covariance-Assisted Learning (CAL)

From the analyses in Section 4.2.1.3, we know the instance-dependent label noise will “automatically” assign different weights to examples with different noise rates, thus cause imbalances. When the optimal solution does not change under such down-weighting effects, the first-order statistics based on peer loss [30, 124] work well. However, for a more robust and general solution, using additional information to “balance” the effective

weights of different examples is necessary. Although the Bayes optimal distribution is not accessible in real experiments, we first assume its existence for theoretical analyses in the ideal case, then we will discuss the gap to this optimal solution when we can only use a proxy \hat{D} that can be constructed efficiently.

4.2.2.1 Extracting Covariance from IDN

Again consider an instance-dependent noisy distribution $\tilde{\mathcal{D}}$ with binary classes where $\tilde{Y} \in \{-1, +1\}$. Define the following two random variables to facilitate analyses:

$$Z_1(X) := 1 - e_+(X) - e_-(X), \quad Z_2(X) = e_+(X) - e_-(X).$$

Recall $e_+ := \mathbb{E}[e_+(X)]$ and $e_- := \mathbb{E}[e_-(X)]$. Let $\text{Cov}_{\mathcal{D}}(A, B) := \mathbb{E}[(A - \mathbb{E}[A])(B - \mathbb{E}[B])]$ be the covariance between random variables A and B w.r.t. the distribution \mathcal{D} . The exact effects of IDN on peer loss functions are revealed in Theorem 11 and proved in Appendix C.2.2.

Theorem 11 (Decoupling binary IDN). *In binary classifications, the expected peer loss with IDN writes as:*

$$\begin{aligned} \mathbb{E}_{\tilde{\mathcal{D}}}[\mathbb{1}_{PL}(f(X), \tilde{Y})] &= (1 - e_+ - e_-)\mathbb{E}_{\mathcal{D}^*}[\mathbb{1}_{PL}(f(X), Y^*)] \\ &\quad + \text{Cov}_{\mathcal{D}^*}(Z_1(X), \mathbb{1}(f(X), Y^*)) \\ &\quad + \text{Cov}_{\mathcal{D}^*}(Z_2(X), \mathbb{1}(f(X), -1)). \end{aligned} \tag{4.2}$$

Theorem 11 effectively divides the instance-dependent label noise into two parts. As shown in Eq. (4.2), the first line is the same as Eq. (4.1) in Lemma 4, indicating

the average effect of instance-dependent label noise can be treated as a class-dependent one with parameters e_+, e_- . The additional two covariance terms in the second and the third lines of Eq. (4.2) characterize the additional contribution of examples due to their differences in the label noise rates. The covariance terms will become larger for a setting with more diverse noise rates, capturing a more heterogeneous and uncertain learning environment. Interested readers are also referred to the high-level intuitions for using covariance terms at the end of Section 4.2.1.3.

We now briefly discuss one extension of Theorem 11 to a K -class classification task. Following the assumption adopted in [124], we consider a particular setting of IDN whose the expected transition matrix satisfies $T_{i,j} = T_{k,j}, \forall i \neq j \neq k$. Denote by $e_j = T_{i,j}, \forall i \neq j$. Corollary 4 decouples the effects of IDN in multi-class cases and is proved in Appendix C.3.1.

Corollary 4 (Decoupling multi-class IDN). *In multi-class classifications, when the expected transition matrix satisfies $e_j = T_{i,j} = T_{k,j}, \forall i \neq j \neq k$, the expected peer loss with IDN writes as:*

$$\begin{aligned} \mathbb{E}_{\tilde{\mathcal{D}}}[\ell_{PL}(f(X), \tilde{Y})] &= (1 - \sum_{i \in [K]} e_i) \mathbb{E}_{\mathcal{D}^*}[\ell_{PL}(f(X), Y^*)] \\ &+ \sum_{j \in [K]} \mathbb{E}_{\mathcal{D}_{Y^*}} [\text{Cov}_{\mathcal{D}^*|Y^*}(T_{Y^*,j}(X), \ell(f(X), j))], \end{aligned}$$

where \mathcal{D}_{Y^*} is the marginal distribution of Y^* and $\mathcal{D}^*|Y^*$ is the conditional distribution of \mathcal{D}^* given Y^* .

4.2.2.2 Using Second-Order Statistics

Inspired by Theorem 11, if \mathcal{D}^* is available, we can subtract two covariance terms and make peer loss invariant to IDN. Specifically, define

$$\begin{aligned} \tilde{f}_{\text{CAL}}^* = \arg \min_f \mathbb{E}_{\tilde{\mathcal{D}}}[\mathbb{1}_{\text{PL}}(f(X), \tilde{Y})] &- \text{Cov}(Z_1(X), \mathbb{1}(f(X), Y^*)) \\ &- \text{Cov}(Z_2(X), \mathbb{1}(f(X), -1)). \end{aligned}$$

We have the following optimality guarantee and its proof is deferred to Appendix C.2.3.

Theorem 12. $\tilde{f}_{\text{CAL}}^* \in \arg \min_f \mathbb{E}_{\mathcal{D}^*}[\mathbb{1}(f(X), Y^*)]$.

For a K -class classification problem, a general loss function for our *Covariance-Assisted Learning (CAL)* approach is given by

$$\begin{aligned} \ell_{\text{CAL}}(f(x_n), \tilde{y}_n) &= \ell_{\text{PL}}(f(x_n), \tilde{y}_n) \\ &- \sum_{j \in [K]} \mathbb{E}_{\mathcal{D}_{Y^*}} [\text{Cov}_{\mathcal{D}^*|Y^*}(T_{Y^*,j}(X), \ell(f(X), j))]. \end{aligned} \tag{4.3}$$

Eq. (4.3) shows the Bayes optimal distribution \mathcal{D}^* is critical in implementing the proposed covariance terms. However, \mathcal{D}^* cannot be obtained trivially, and only imperfect proxy constructions of the dataset (denoted by \hat{D}) could be expected. Detailed constructions of \hat{D} are deferred to Section 4.2.2.3.

Advantages of using covariance terms There are several advantages of using the proposed covariance terms. Unlike directly correcting labels according to \mathcal{D}^* , the proposed covariance term can be viewed as a “soft” correction that maintains the information encoded in both original noisy labels and the estimated Bayes optimal labels. Keeping both pieces of information is beneficial as suggested in [64]. Moreover, compared

to the direct loss correction approaches [146, 198, 199], we keep the original learning objective and apply “correction” using an additional term. Our method is more robust in practice compared to these direct end-to-end loss correction approaches due to two reasons: 1) The covariance term summarizes the impact of the complex noise using an average term, indicating that our approach is less sensitive to the estimation precision of an individual example; 2) As will be shown in Section 4.2.3, the proposed method is tolerant with accessing an imperfect \mathcal{D}^* .

Estimating the covariance terms relies on samples drawn from distribution \mathcal{D}^* . Thus, we need to construct a dataset \hat{D} , which is similar or unbiased w.r.t. \mathcal{D}^* . We will first show the algorithm for constructing \hat{D} , then provide details for DNN implementations.

4.2.2.3 Constructing \hat{D}

To achieve unbiased estimates of the variance terms, the high-level intuition for constructing \hat{D} is determining whether the label of each example in \tilde{D} is Bayes optimal or not by comparing the likelihood, confidence, or loss of classifying the (noisy) label to some thresholds. There are several methods for constructing \hat{D} : distillation [33], searching to exploit [209], and sample sieve [30]. If the model does not overfit the label noise and learns the noisy distribution, both methods in [33] and [209] work well. However, for the challenging instance-dependent label noise, overfitting occurs easily thus techniques to avoid overfitting are necessary. In this paper, we primarily adapt the sample sieve proposed in [30], which uses a confidence regularizer to avoid overfitting,

Algorithm 5 Constructing \hat{D}

1: **Input:** Noisy dataset \tilde{D} . Thresholds $L_{\min} \leq L_{\max}$. Number of epochs T . $\hat{D} = \tilde{D}$.

Train the sample sieve in [30] for T epochs and get the model f

2: **for** $n \in [N]$ **do**

3: Calculate $\alpha_{n,T}$ following [30]

4: **if** $\ell_{\text{CORES}^2}(f(x_n), \tilde{y}_n) - \alpha_{n,T} \leq L_{\min}$ **then**

5: $\hat{y}_n = \tilde{y}_n$

6: **else if** $\ell_{\text{CORES}^2}(f(x_n), \tilde{y}_n) - \alpha_{n,T} > L_{\max}$ **then**

7: $\hat{y}_n = \arg \max_{y \in [K]} f_{x_n}[y]$

8: **else**

9: $\hat{y}_n = -1$ (drop example n)

10: **end if**

11: **end for**

12: **Output:** $\hat{D} := \{(x_n, \hat{y}_n) : n \in [N], \hat{y}_n \neq -1\}$

to construct \hat{D} . Specifically, as shown in [30], in each epoch t , the regularized loss for each example is adjusted by the parameter $\alpha_{n,t}$, which can be calculated based on model predictions in linear time. In the ideal cases assumed in [30], any example with a positive adjusted loss is corrupted (with a wrong label).

We summarized the corresponding procedures in Algorithm 5, where the critical thresholds for comparing losses are denoted by L_{\min} and L_{\max} . At Line 5, if the loss adjusted by $\alpha_{n,t}$ is small enough (smaller than the threshold L_{\min}), we assume \tilde{y}_n is the Bayes optimal label. Accordingly, at Line 7, if the adjusted loss is too large

(larger than the threshold L_{\max}), we treat \tilde{y}_n as a corrupted one and assume the class with maximum predicted probability to be Bayes optimal one. For the examples with moderate adjusted loss, we drop it as indicated in Line 9. In ideal cases with infinite model capacity and sufficiently many examples (as assumed in [30]), we can set thresholds $L_{\min} = L_{\max} = 0$ to guarantee separation of clean and corrupted examples, thus \hat{D} will be an unbiased proxy to \mathcal{D}^* . In the ideal case as assumed in Corollary 1 of [30], we have $\hat{D} = D^*$. However, in real experiments, when both the model capacity and the number of examples are limited, we may need to tune L_{\min} and L_{\max} to obtain a high-quality construction of \hat{D} . In this paper, we set $L_{\min} = L_{\max}$ to ensure $|\hat{D}| = |D^*|$ and reduce the effort to tuning both thresholds simultaneously.

Note that using \hat{D} to estimate the covariance terms could be made theoretically more rigorous by applying appropriate re-weighting techniques [33, 47, 76]. See Appendix C.4.1 for more discussions and corresponding guarantees. We omit the details here due to the space limit. Nonetheless, our approach is tolerant of an imperfect \hat{D} , which will be shown theoretically in Section 4.2.3.

4.2.2.4 Implementations

For implementations with deep neural network solutions, we need to estimate the transition matrix $T(X)$ relying on \hat{D} and estimate the covariance terms along with stochastic gradient descent (SGD) updates.

Covariance Estimation in SGD As required in (4.3), with a particular \hat{D} , each computation for $\hat{T}_{i,j}(x_n)$ requires only one time check of the associated noisy label as follows:

$$\hat{T}_{i,j}(x_n) = \mathbb{1}\{\hat{y}_n = i, \tilde{y}_n = j\}. \quad (4.4)$$

When \hat{D} is unbiased w.r.t. D^* , the estimation in (4.4) is also unbiased because

$$\begin{aligned} \mathbb{E}_{\tilde{\mathcal{D}}|X, \hat{Y}=i}[\hat{T}_{i,j}(X)] &= \mathbb{E}_{\tilde{\mathcal{D}}|X, \hat{Y}=i}[\mathbb{1}\{\hat{Y} = i, \tilde{Y} = j|X\}] \\ &= \mathbb{P}(\tilde{Y} = j|X, \hat{Y} = i) = \mathbb{P}(\tilde{Y} = j|X, Y^* = i). \end{aligned}$$

Noting $\text{Cov}_{\mathcal{D}}(A, B) := \mathbb{E}[(A - \mathbb{E}[A])(B - \mathbb{E}[B])] = \mathbb{E}[(A - \mathbb{E}[A]) \cdot B]$, the covariance can be estimated empirically as

$$\frac{1}{N} \sum_{n \in [N]} \sum_{i,j \in [K]} \mathbb{1}\{y_n^* = i\} \left[(\hat{T}_{i,j}(x_n) - \hat{T}_{i,j}) \cdot \ell(f(x_n), j) \right].$$

For each batch of data, the above estimation has $O(N)$ complexities in computation and space. To reduce both complexities, with the cost of the estimation quality, we use $|E_b|$ examples to estimate the covariance in each batch, where E_b is the set of sample indices of batch- b . Per sample wise, Eq. (4.3) can be transformed to

$$\begin{aligned} \ell_{\text{CAL}}(f(x_n), \tilde{y}_n) &= \ell_{\text{PL}}(f(x_n), \tilde{y}_n) \\ &- \sum_{i,j \in [K]} \mathbb{1}\{y_n^* = i\} \left[(\hat{T}_{i,j}(x_n) - \hat{T}_{i,j}) \cdot \ell(f(x_n), j) \right]. \end{aligned}$$

With the above implementation, the estimation is done locally for each point in $O(1)$ complexity.

4.2.3 CAL with Imperfect Covariance Estimates

As mentioned earlier, \mathcal{D}^* cannot be perfectly obtained in practice. Thus, there is a performance gap between the ideal case (with perfect knowledge of \mathcal{D}^*) and the actually achieved one. We now analyze the effect of imperfect covariance terms (Theorem 13).

Denote the imperfect covariance estimates by \hat{D}^τ , where $\tau \in [0, 1]$ is the expected ratio (a.k.a. probability) of correct examples in \hat{D}^τ : $\tau = \mathbb{E}[\mathbf{1}\{(X, \hat{Y}) \in \hat{D}^\tau | (X, Y^*) \in D^*\}] = \mathbb{P}((X, \hat{Y}) \sim \hat{D}^\tau | (X, Y^*) \sim \mathcal{D}^*)$. With \hat{D}^τ , the minimizer of the 0-1 CAL loss is given by:

$$\begin{aligned} \tilde{f}_{\text{CAL-}\tau}^* = \arg \min_f \mathbb{E}_{\tilde{\mathcal{D}}} \left[\mathbb{1}_{\text{PL}}(f(X), \tilde{Y}) - \text{Cov}_{\hat{D}^\tau}(Z_1(X), \mathbf{1}(f(X), \hat{Y})) \right. \\ \left. - \text{Cov}_{\hat{D}^\tau}(Z_2(X), \mathbf{1}(f(X), -1)) \right]. \end{aligned}$$

Theorem 13 reports the error bound produced by $\tilde{f}_{\text{CAL-}\tau}^*$. See Appendix C.2.4 for the proof.

Theorem 13 (Imperfect Covariance). *With \hat{D}^τ , when $p^* = 0.5$, we have*

$$\mathbb{E}[\mathbf{1}(\tilde{f}_{\text{CAL-}\tau}^*(X), Y^*)] \leq \frac{4(1-\tau)(\epsilon_+ + \epsilon_-)}{1 - e_+ - e_-}.$$

Theorem 13 shows the quality of \hat{D}^τ controls the scale of the worst-case error upper-bound. Compared with Theorem 10 where no covariance term is used, we know the covariance terms will always be helpful when $\tau \in [0.5, 1]$. That is, the training with the assistance of covariance terms will achieve better (worst-case) accuracy on the Bayes optimal distribution when the construction \hat{D}^τ is better than a dataset that includes each instance in D^* randomly with 50% chance.

4.2.4 Experiments

We now present our experiment setups and results.

4.2.4.1 General Experiment Settings

Datasets and models The advantage of introducing our second-order approach is evaluated on three benchmark datasets: CIFAR10, CIFAR100 [99] and Clothing1M [200]. Following the convention from [30, 205], we use ResNet34 for CIFAR10 and CIFAR100 and ResNet50 for Clothing1M. Noting the expected peer term $\mathbb{E}_{\mathcal{D}_{\tilde{Y}|\hat{D}}}[\ell(f(x_n), \tilde{Y})]$ (a.k.a. *confidence regularizer (CR)* as implemented in [30]) is more stable and converges faster than the one with peer samples, we train with ℓ_{CORES^2} . It also enables a fair ablation study since \hat{D} is constructed relying on [30]. For numerical stability, we use a cut-off version of the cross-entropy loss $\ell(f(x), y) = -\ln(f_x[y] + \varepsilon)$. Specifically, we use $\varepsilon = 10^{-8}$ for the traditional cross-entropy term, use $\varepsilon = 10^{-5}$ for the CR term, and the covariance term. All the experiments use a momentum of 0.9. The weight decay is set as 0.0005 for CIFAR experiments and 0.001 for Clothing1M.

Noise type For CIFAR datasets, the instance-dependent label noise is generated following the method from [30, 198]. The basic idea is randomly generating one vector for each class (K vectors in total) and projecting each incoming feature onto these K vectors. The label noise is added by jointly considering the clean label and the projection results. See Appendix A.4.1 for details. In expectation, the noise rate η is the overall ratio of examples with a wrong label in the entire dataset. For the Clothing1M dataset,

we train on 1 million noisy training examples that encode the real-world human noise.

4.2.4.2 Baselines

We compare our method with several related works, where the cross-entropy loss is tested as a common baseline. Additionally, the generalized cross-entropy [221] is compared as a generalization of mean absolute error and cross-entropy designed for label noise. Popular loss correction based methods [146, 198, 199], sample selection based methods [30, 63, 179, 212], and noise-robust loss functions [124, 205] are also chosen for comparisons. All the compared methods adopt similar data augmentations, including standard random crop, random flip, and normalization. The semi-supervised learning-based methods with extra feature extraction and data augmentations are not included. All the CIFAR experiments are repeated 5 times with independently synthesized IDN. The highest accuracies on the clean test dataset are averaged over 5 trials to show the best generalization ability of each method.

4.2.4.3 Performance Comparisons: CIFAR

In experiments on CIFAR datasets, we use a batch size of 128, an initial learning rate of 0.1, and reduce it by a factor of 10 at epoch 60.

Construct \hat{D} To construct \hat{D} , we update the DNN for 65 epochs by minimizing ℓ_{CORES^2} (without dynamic sample sieve) and apply Algorithm 5 with $L_{\min} = L_{\max} = -8$. Note that, theoretically, we have $L_{\min} = L_{\max} = 0$ if both the CE term and the CR term use a log loss without cut-off ($\varepsilon = 0$). The current setting works well (not the best) for

Table 4.2: Comparison of test accuracies (%) using different methods.

Method	<i>Inst. CIFAR10</i>			<i>Inst. CIFAR100</i>		
	$\eta = 0.2$	$\eta = 0.4$	$\eta = 0.6$	$\eta = 0.2$	$\eta = 0.4$	$\eta = 0.6$
CE (Standard)	85.45±0.57	76.23±1.54	59.75±1.30	57.79±1.25	41.15±0.83	25.68±1.55
Forward T [146]	87.22±1.60	79.37±2.72	66.56±4.90	58.19±1.37	42.80±1.01	27.91±3.35
L_{DMI} [205]	88.57±0.60	82.82±1.49	69.94±1.31	57.90±1.21	42.70±0.92	26.96±2.08
L_q [221]	85.81±0.83	74.66±1.12	60.76±3.08	57.03±0.27	39.81±1.18	24.87±2.46
Co-teaching [63]	88.87±0.24	73.00±1.24	62.51±1.98	43.30±0.39	23.21±0.57	12.58±0.51
Co-teaching+ [212]	89.80±0.28	73.78±1.39	59.22±6.34	41.71±0.78	24.45±0.71	12.58±0.51
JoCoR [179]	88.78±0.15	71.64±3.09	63.46±1.58	43.66±1.32	23.95±0.44	13.16±0.91
Reweight-R [199]	90.04±0.46	84.11±2.47	72.18±2.47	58.00±0.36	43.83±8.42	36.07±9.73
Peer Loss [124]	89.12±0.76	83.26±0.42	74.53±1.22	61.16±0.64	47.23±1.23	31.71±2.06
CORES ² [30]	91.14±0.46	83.67±1.29	77.68±2.24	66.47±0.45	58.99±1.49	38.55±3.25
CAL	92.01±0.75	84.96±1.25	79.82±2.56	69.11±0.46	63.17±1.40	43.58±3.30

CIFAR experiments empirically. For a numerically stable solution, we use the square root of the noise prior for the CR term in ℓ_{CORES^2} as $-\beta \sum_{i \in [K]} \frac{\sqrt{\mathbb{P}(\tilde{Y}=i|\tilde{D})}}{\sum_{j=1}^K \sqrt{\mathbb{P}(\tilde{Y}=j|\tilde{D})}} \ell(f(x_n), i)$.

The hyperparameter β is set to 2 for CIFAR10 and 10 for CIFAR100.

Train with CAL With an estimate of D^* , we re-train the model 100 epochs. The hyperparameter β is set to 1 for CIFAR10 and 10 for CIFAR100. Note the hyperparameters $(L_{\min}, L_{\max}, \beta)$ can be better set if a clean validation set is available.

Performance Table 4.2 compares the means and standard deviations of test accuracies on the clean test dataset when the model is trained with synthesized instance-dependent label noise in different levels. All the compared methods use ResNet34 as the backbone. On CIFAR10, with a low-level label noise ($\eta = 0.2$), all the compared methods perform well and achieve higher average test accuracies than the standard CE loss. When the overall noise rates increase to high, most of the methods suffer from severe performance degradation while CAL still achieves the best performance. There are similar observations on CIFAR100. By comparing CAL with CORES², we conclude that the adopted second-

Table 4.3: The best epoch (clean) test accuracies on Clothing1M.

Method	Accuracy
CE (standard)	68.94
Forward T [146]	70.83
Co-teaching [63]	69.21
JoCoR [179]	70.30
L_{DMI} [205]	72.46
PTD-R-V[198]	71.67
CORES ² [30]	73.24
CAL	74.17

order statistics do work well and bring non-trivial performance improvement. Besides, on the CIFAR100 dataset with $\eta = 0.4$ and 0.6 , we observe Reweight-R [199] has a large standard deviation and a relatively high mean, indicating it may perform as well as or even better than CAL in some trials. It also shows the potential of using a revised transition matrix T [199] in severe and challenging instance-dependent label noise settings.

4.2.4.4 Performance Comparisons: Clothing1M

For Clothing1M, we first train the model following the settings in [30] and construct \hat{D} with the best model. Noting the overall accuracy of noisy labels in Clothing1M is about 61.54% [200], we set an appropriate $L_{\min} = L_{\max}$ such that 61.54% of training examples satisfying $\ell_{\text{CORES}^2} - \alpha_{n,t} \leq L_{\min}$. With \hat{D} , we sample a class-balanced dataset by randomly choosing 18,976 noisy examples for each class and continue training the model with $\beta = 1$ and an initial learning rate of 10^{-5} for 120 epochs. Other parameters are set following [30]. See Appendix C.4.3 for more detailed experimental settings. Table 4.3 shows CAL performs well in the real-world human noise.

Table 4.4: Analysis of each component of CAL on CIFAR10.

row #	<i>Cov.</i>	<i>Peer</i>	Epoch	$\eta = 0.2$	$\eta = 0.4$	$\eta = 0.6$
1	✗	✗	Best	90.47	82.56	64.65
2	✓	✗	Best	92.10	78.49	73.55
3	✗	✓	Best	91.85	84.41	78.74
4	✗	✓	Fixed@65	90.73	82.76	77.70
5	✓	✓	Best	92.69	85.55	81.54

4.2.4.5 Ablation Study

Table 4.4 shows either the covariance term or the peer term can work well individually and significantly improve the performance when they work jointly. The result of a particular trial is presented. *Cov.* indicates using with the covariance term. *Peer* indicates using the CR term [30] (a.k.a. expected peer term [124]). Comparing the first row with the second row, we find the second-order statistics can work well (except for $\eta = 0.4$) even without the peer (CR) term. In row 4, we show the performance at epoch 65 since the second-order statistics are estimated relying on the model prediction at this epoch. By comparing row 4 with row 5, we know the second-order statistics indeed lead to non-trivial improvement in performance. Even though the covariance term individually can only achieve an accuracy of 78.49 when $\eta = 0.4$, it can still contribute more than 1% of the performance improvement (from 84.41% to 85.55%) when it is implemented with the peer term. This observation shows the robustness of CAL.

4.2.5 Takeaways

We summarize the takeaways as follows.

- We have proposed a second-order approach to transforming the challenging instance-

dependent label noise into a class-dependent one such that existing methods targeting the class-dependent label noise could be implemented.

- We show how the second-order statistics can be estimated efficiently using existing sample selection techniques. For a more realistic case where the covariance terms cannot be perfectly estimated, we prove the worst-case performance guarantee of our solution.
- In addition to the theoretical guarantees, the performance of the proposed second-order approach is tested on the CIFAR10 and CIFAR100 datasets with synthetic instance-dependent label noise and the Clothing1M dataset with real-world human label noise.
- Code is available at <https://github.com/UCSC-REAL/CAL>.

Chapter 5

Beyond Accuracy: Fairness Issues

We will discuss the fairness issues caused by imperfect data in this chapter, including the disparate impact of semi-supervised learning (SSL) when a huge amount of data does not have labels (Section 5.1) and the fairness evaluation when the sensitive attributes are missing (Section 5.2).

5.1 Disparate Impact of SSL

This section aims to reveal the disparate impact of semi-supervised learning by answering the following question: Are different sub-populations treated similarly during SSL? We will start with a motivating example which is experimented with one of the best SSL algorithms, then prove why the disparate impact exists in SSL. To measure the disparate impact of SSL, we further propose a new metric called “Benefit Ratio (BR)”, the effectiveness of which is guaranteed by our proven generalization bounds.

5.1.1 Motivating Examples

Figure 5.1 shows the change of test accuracy for each class during SSL. Each class denotes one sub-population and representative sub-populations are highlighted. In this experiment, MixMatch [18] is applied on CIFAR-10 with (a) 250 clean labels in the balanced case (25 labeled instances per class) and (b) 185 clean labels in the unbalanced case (25 labeled instances in each of the first 5 classes, 12 labeled instances in each of the remaining classes). Other instances are used as unlabeled data. Figure 5.1 delivers two important messages: in SSL, even with some state-of-the-art algorithms: 1) the observation “rich getting richer” is common, and 2) the observation “poor getting poorer” possibly happens. Specifically, the “*rich*” sub-population, such as *automobile* that has a high baseline accuracy at the beginning of SSL, tends to consistently benefit from SSL. But the “*poor*” sub-population, such as *dog* that has a low baseline accuracy, will remain a low-level performance as Figure 5.1(a) or even get worse performance as Figure 5.1(b). This example shows the disparate impact of accuracies for different sub-populations are common in SSL. In special cases such as Figure 5.1(b), we observe the Matthew effect: the rich get richer and the poor get poorer.

We now go one step further to intuitively understand why we may observe the Matthew effect. With the pseudo-labels defined in Section 1.2.2.4, we know that the sub-population that is already “rich” (high-accuracy) in supervised learning with the labeled dataset D_L will possess higher-quality pseudo-labels for SSL, thereby further enhancing the performance. On the other hand, the sub-population with poorer baseline

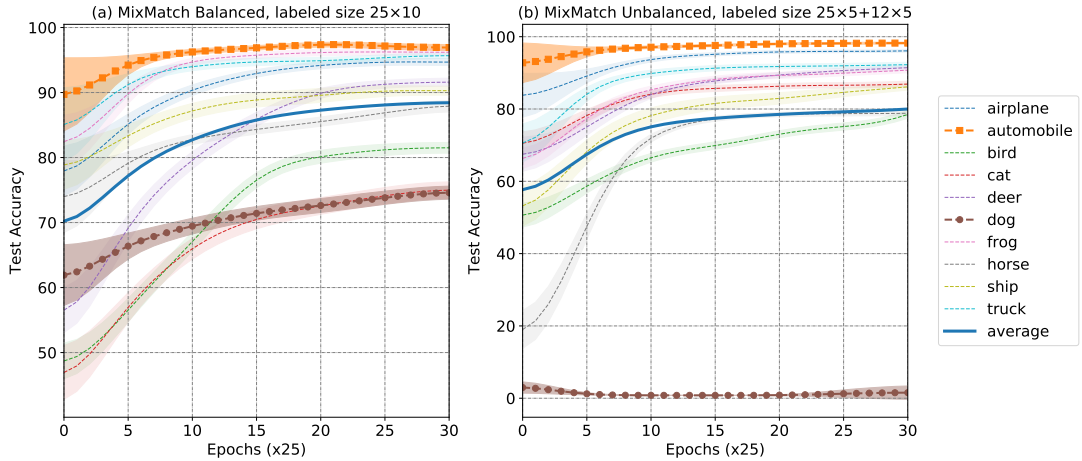


Figure 5.1: Disparate impacts in the model accuracy of SSL.

accuracy (also from supervised learning with D_L) will have lower-quality pseudo-labels, which can only provide limited regularization for unsupervised features. In cases where regularization is applied in the wrong direction, unsupervised features, along with their augmented copies, may converge on an incorrect label class, resulting in a decline in performance. Consequently, as the baseline accuracy deteriorates, an increasing number of unsupervised features will be inaccurately regularized, leading to divergent impacts on model accuracies, as illustrated in Figure 5.1.

5.1.2 Theoretical Analyses

We now theoretically prove the above intuition. The following analyses are conducted for an arbitrary sub-population, thus we did not explicitly distinguish sub-populations in notation. We consider minimizing 0-1 loss $\mathbf{1}(f(X), Y)$ with infinite search

space. Recall D_L denotes the labeled dataset and D_U denotes the unlabeled one.

5.1.2.1 Learning with Clean Data

Denote the expected error rate of classifier f on distribution \mathcal{D} by $R_{\mathcal{D}}(f) := \mathbb{E}_{\mathcal{D}}[\mathbb{1}(f(X), Y)]$. Let \hat{f}_D denote the classifier trained by minimizing 0-1 loss with clean dataset D , i.e., $\hat{f}_{D_L} := \arg \min_f \hat{R}_{D_L}(f)$, where $\hat{R}_{D_L}(f) := \frac{1}{N} \sum_{n \in [N_L]} \mathbb{1}(f(x_n), y_n)$. Denote by $Y^*|X := \arg \max_{i \in [K]} \mathbb{P}(Y=i|X)$ the Bayes optimal label on clean distribution \mathcal{D} . Theorem 14 shows the generalization bound in the clean case.

Theorem 14 (Supervised learning error). *With probability at least $1 - \delta$, the generalization error of supervised learning on D_L is upper-bounded by*

$$R_{\mathcal{D}}(\hat{f}_{D_L}) \leq \sqrt{\frac{2 \log(4/\delta)}{N_L}} + \mathbb{P}(Y^* \neq Y).$$

5.1.2.2 Learning with Semi-supervised Data

The derivation of generalization bounds for learning with semi-supervised data relies on the pseudo-labels defined in Section 1.2.2.4, i.e., transforming the semi-supervised dataset $D_L \cup D_U$ into the dataset \tilde{D} based on the model learned from the previous epoch.

Two-iteration scenario To establish a clean and structured performance bound for learning with semi-supervised data, we consider a specific *two-iteration scenario*. In this scenario, the model is initially trained to convergence using the small labeled dataset D_L , resulting in the model \hat{f}_{D_L} . Subsequently, the model is trained on the pseudo-noisy

dataset \tilde{D} labeled by \hat{f}_{D_L} . It is worth noting that this two-iteration scenario represents a worst-case situation for a semi-supervised learning algorithm, as iteratively assigning pseudo-labels typically enhances performance, as suggested by many SSL algorithms [18, 201].

Independence of samples in \tilde{D} The number of independent instances, denoted as N' , falls within the range of $[N_L, N]$. Intuitively, if appropriate noise injection or data augmentation techniques [201, 138] are applied to x_n such that x'_n can be considered independent of x_n , then the number of independent samples in \tilde{D} can be improved to N . For the purpose of our analysis, we assume the ideal case where all N instances are independently and identically distributed (*i.i.d.*).

By minimizing the unified loss defined in Eq. (1.1), we can get classifier

$$\hat{f}_{\tilde{D}} := \arg \min_f \hat{R}_{\tilde{D}}(f),$$

where

$$\hat{R}_{\tilde{D}}(f) := \frac{1}{N} \sum_{n \in [N]} \left(\sum_{i \in [K]} \tilde{\mathbf{y}}[i] \cdot \mathbf{1}(f(x_n), i) \right).$$

The expected error given classifier f is denoted by

$$R_{\tilde{D}}(f) := \mathbb{E}_{\tilde{D}}[\mathbf{1}(f(X), \tilde{Y})],$$

where the probability density function of distribution \tilde{D} can be defined as

$$\mathbb{P}_{(X, \tilde{Y}) \sim \tilde{D}}(X = x_n, \tilde{Y} = i) = \mathbb{P}_{(X, Y) \sim \mathcal{D}}(X = x_n) \cdot \tilde{\mathbf{y}}_n[i].$$

Decomposition With the given definitions, we can decompose the generalization error (on the clean distribution) of classifier $\hat{f}_{\tilde{\mathcal{D}}}$ as follows:

$$R_{\mathcal{D}}(\hat{f}_{\tilde{\mathcal{D}}}) = \underbrace{(R_{\mathcal{D}}(\hat{f}_{\tilde{\mathcal{D}}}) - R_{\tilde{\mathcal{D}}}(\hat{f}_{\tilde{\mathcal{D}}}))}_{\text{Term-1}} + \underbrace{R_{\tilde{\mathcal{D}}}(\hat{f}_{\tilde{\mathcal{D}}})}_{\text{Term-2}},$$

where **Term-1** transforms the evaluation of $\hat{f}_{\tilde{\mathcal{D}}}$ from the clean distribution \mathcal{D} to the pseudo-noisy distribution $\tilde{\mathcal{D}}$. **Term-2** can be compared with the generalization error in Theorem 14, but the model is trained and evaluated on the noisy distribution $\tilde{\mathcal{D}}$. Both terms are analyzed as follows.

Upper and Lower Bounds for Term-1 Let

$$\eta(X) := \frac{1}{2} \sum_{i \in [K]} |\mathbb{P}(\tilde{Y} = i|X) - \mathbb{P}(Y = i|X)|,$$

$e(X) := \mathbb{P}(Y \neq \tilde{Y}|X)$ be the feature-dependent error rate, $\tilde{A}_f(X) := \mathbb{P}(f(X) = \tilde{Y}|X)$ be the accuracy of prediction $f(X)$ on noisy dataset $\tilde{\mathcal{D}}$. Denote their expectations (over X) by

$$\bar{\eta} := \mathbb{E}_X[\eta(X)], \quad \bar{e} := \mathbb{E}_X[e(X)], \quad \tilde{A}_f = \mathbb{E}_X[\tilde{A}_f(X)].$$

To highlight that $\bar{\eta}$ and \bar{e} depends on the noisy dataset $\tilde{\mathcal{D}}$ labeled by \hat{f}_{D_L} , we denote them as $\bar{\eta}(\hat{f}_{D_L})$ and $\bar{e}(\hat{f}_{D_L})$. Then we have:

Lemma 5 (Bounds for Term-1).

$$(2\tilde{A}_{\hat{f}_{\tilde{\mathcal{D}}}} - 1)\bar{e}(\hat{f}_{D_L}) \leq R_{\mathcal{D}}(\hat{f}_{\tilde{\mathcal{D}}}) - R_{\tilde{\mathcal{D}}}(\hat{f}_{\tilde{\mathcal{D}}}) \leq \bar{\eta}(\hat{f}_{D_L}).$$

Note the upper bound builds on $\bar{\eta}(\hat{f}_{D_L})$ while the lower bound relates to $\bar{e}(\hat{f}_{D_L})$.

To compare two bounds and show the tightness, we consider the case where $Y|X$ is

confident, i.e., each feature X belongs to one particular true class Y with probability 1, which is generally held in classification problems [119]. Lemma 6 shows $\eta(X) = e(X)$ in this particular case.

Lemma 6 (η vs. e). *For any feature X , if $Y|X$ is confident, $\eta(X)$ is the error rate of the model prediction on X , i.e., $\exists i \in [K] : \mathbb{P}(Y = i|X) = 1 \Rightarrow \eta(X) = \mathbb{P}(\tilde{Y} \neq Y|X) = e(X)$.*

Upper bound for Term-2 Denote by $\tilde{Y}^*|X := \arg \max_{i \in [K]} \mathbb{P}(\tilde{Y} = i|X)$ the Bayes optimal label on noisy distribution $\tilde{\mathcal{D}}$. Following the proof for Theorem 14, we have:

Lemma 7 (Bound for Term-2). *W. p. at least $1 - \delta$,*

$$R_{\tilde{\mathcal{D}}}(\hat{f}_{\tilde{\mathcal{D}}}) \leq \sqrt{\frac{2 \log(4/\delta)}{N}} + \mathbb{P}(\tilde{Y} \neq \tilde{Y}^*).$$

Wrap-up Lemma 5 shows Term-1 is in the range of $[(2\tilde{A}_{\hat{f}_{\tilde{\mathcal{D}}}} - 1)\bar{e}(\hat{f}_{D_L}), \bar{\eta}(\hat{f}_{D_L})]$. Lemma 6 informs us $\bar{\eta}(\hat{f}_{D_L}) = \bar{e}(\hat{f}_{D_L})$ in classification problems where $Y|X, \forall X$ are confident. With a *well-trained* model $\hat{f}_{\tilde{\mathcal{D}}}$ that learns the noisy distribution $\tilde{\mathcal{D}}$, we have $\tilde{A}_{\hat{f}_{\tilde{\mathcal{D}}}} = 1 - \epsilon$ and $\epsilon \rightarrow 0_+$, thus Term-1 is in the range of $[(1 - 2\epsilon)\bar{\eta}(\hat{f}_{D_L}), \bar{\eta}(\hat{f}_{D_L})]$, indicating *our bounds for Term-1 are tight*. Besides, noting Lemma 7 is derived following the same techniques as Theorem 14, we know both bounds have similar tightness. Therefore, by adding upper bounds for Term-1 and Term-2, we can upper bound the error of semi-supervised learning in Theorem 15, which has similar tightness to that in Theorem 14.

Theorem 15 (Semi-supervised learning error [229]). *Suppose the model trained with only D_L has generalization error $\bar{\eta}'(\hat{f}_{D_L})$. With probability at least $1 - \delta$, the generalization*

error of semi-supervised learning on datasets $D_L \cup D_U$ is upper-bounded by

$$R_{\mathcal{D}}(\hat{f}_{D_L \cup D_U}) \leq \underbrace{\bar{\eta}(\hat{f}_{D_L})}_{\text{Disparity due to baseline}} + \underbrace{\mathbb{P}(\tilde{Y} \neq \tilde{Y}^*)}_{\text{Sharpness of pseudo labels}} + \underbrace{\sqrt{\frac{2 \log(4/\delta)}{N}}}_{\text{Data dependency}},$$

where $\bar{\eta}(\hat{f}_{D_L}) := \bar{\eta}'(\hat{f}_{D_L}) \cdot N_U/N$ is the expected label error in the pseudo noisy dataset \tilde{D} .

Takeaways Theorem 15 provides an explanation for the generation of disparate impacts in SSL. We can break down the factors contributing to this phenomenon as follows:

- Supervised error $\bar{\eta}'(\hat{f}_{D_L})$: This is the primary source of disparity. Sub-populations that exhibit good generalization before SSL tend to have lower SSL error rates. In other words, the already well-performing groups benefit more from SSL, reinforcing existing disparities (the "rich get richer" effect).
- Sharpness of noisy labels $\mathbb{P}(\tilde{Y} \neq \tilde{Y}^*)$: This factor plays a minor role in generating disparity and depends on the method used to process pseudo-labels. If the pseudo-labels are sharpened, this term becomes negligible.
- Sample complexity $\sqrt{2 \log(4/\delta)/N}$: Disparity is influenced by the number of instances N and their independence. It is important to note that our analysis assumes ideal data augmentations to obtain N in this term. Instances with poor data augmentations, which are much less than the total N , could potentially become a significant source of disparity if the augmentations are inadequate.

5.1.3 Benefit Ratio: An Evaluation Metric

To quantify the disparate impacts of SSL as illustrated in Figure 5.1, we propose a new metric called *benefit ratio*.

Benefit Ratio The benefit ratio $\text{BR}(\mathcal{P})$ captures the normalized accuracy improvement of sub-population \mathcal{P} after SSL. It depends on three classifiers:

- \hat{f}_{D_L} , which represents the baseline supervised learning using a small labeled dataset D_L ;
- \hat{f}_D , which represents the ideal supervised learning if the entire dataset D has ground-truth labels;
- $\hat{f}_{D_L \cup D_U}$, which represents SSL using both the labeled dataset D_L and the unlabeled dataset D_U .

The test/validation accuracies of these classifiers are denoted as $a_{\text{baseline}}(\mathcal{P})$, $a_{\text{ideal}}(\mathcal{P})$, and $a_{\text{semi}}(\mathcal{P})$, respectively. As a measure for evaluating SSL algorithms after the fact, the benefit ratio $\text{BR}(\mathcal{P})$ is defined as:

$$\text{BR}(\mathcal{P}) = \frac{a_{\text{semi}}(\mathcal{P}) - a_{\text{baseline}}(\mathcal{P})}{a_{\text{ideal}}(\mathcal{P}) - a_{\text{baseline}}(\mathcal{P})}. \quad (5.1)$$

Let $\mathcal{P}^\diamond := \{\mathcal{P}_1, \mathcal{P}_2, \dots\}$ be the set of all the concerned sub-populations. We formally define the Equalized Benefit Ratio as follows.

Definition 11 (Equalized Benefit Ratio). *An algorithm achieves Equalized Benefit Ratio*

(EBR) if all the concerned sub-populations have the same benefit ratio:

$$\text{BR}(\mathcal{P}) = \text{BR}(\mathcal{P}'), \forall \mathcal{P}, \mathcal{P}' \in \mathcal{P}^\diamond.$$

Intuitively, a larger benefit ratio indicates a greater benefit from SSL. We define $\text{BR}(\mathcal{P}) = 1$ when SSL performs as well as the corresponding fully-supervised learning. A negative benefit ratio indicates that SSL adversely affects the disadvantaged population, where $a_{\text{semi}}(\mathcal{P}) < a_{\text{baseline}}(\mathcal{P})$, meaning that the already poor performance worsens, as illustrated in Figure 5.1(b) for the *dog* sub-population.

These findings have the potential to offer valuable guidance and insights for the development of fair SSL algorithms using standard datasets with complete ground-truth labels. Investigating whether a fair SSL algorithm designed for one dataset remains fair when applied to another dataset would be an interesting avenue for future research. In real-world scenarios where full supervision is unavailable, we can leverage additional knowledge to estimate the highest achievable accuracy for each sub-population and use it as a proxy for the ideal accuracy $a_{\text{ideal}}(\mathcal{P})$.

Theoretical Explanation We define a proxy of the benefit ratio as

$$\widehat{\text{BR}}(\mathcal{P}) := \frac{\sup(R_{\mathcal{D}}(\hat{f}_{D_L \cup D_U|\mathcal{P}})) - \sup(R_{\mathcal{D}}(\hat{f}_{D_L|\mathcal{P}}))}{\sup(R_{\mathcal{D}}(\hat{f}_{D|\mathcal{P}})) - \sup(R_{\mathcal{D}}(\hat{f}_{D_L|\mathcal{P}}))},$$

where $\sup(\cdot)$ denotes the upper bound derived in Theorem 14 and Theorem 15, \mathcal{P} is a sub-population, and $D|\mathcal{P}$ denotes the set of *i.i.d.* instances in D that affect model generalization on \mathcal{P} . By assuming that both distributions have the same sharpness, i.e., $\mathbb{P}(Y \neq Y^*) = \mathbb{P}(\tilde{Y} \neq \tilde{Y}^*)$, we have:

Corollary 5. *The benefit ratio proxy for \mathcal{P} is*

$$\widehat{\text{BR}}(\mathcal{P}) = 1 - \frac{\bar{\eta}(\hat{f}_{D_L|\mathcal{P}})}{\Delta(N_{\mathcal{P}}, N_{\mathcal{P}_L})},$$

where

$$\Delta(N_{\mathcal{P}}, N_{\mathcal{P}_L}) = \sqrt{\frac{2 \log(4/\delta)}{N_{\mathcal{P}_L}}} - \sqrt{\frac{2 \log(4/\delta)}{N_{\mathcal{P}}}},$$

$N_{\mathcal{P}}$ and $N_{\mathcal{P}_L}$ are the effective numbers of instances in $D|\mathcal{P}$ and $D_L|\mathcal{P}$.

Corollary 5 demonstrates that the benefit ratio is negatively correlated with the error rate of baseline models and positively correlated with the number of *i.i.d.* instances after SSL. Please note that $N_{\mathcal{P}}$ and $N_{\mathcal{P}_L}$ may exceed the sizes of the corresponding sub-populations if \mathcal{P} shares information with another sub-population \mathcal{P}' during training. For instance, improved classification of \mathcal{P}' can aid in classifying \mathcal{P} . Furthermore, the corollary suggests that SSL may have a detrimental impact on sub-population \mathcal{P} if $\frac{\eta}{\Delta(N_{\mathcal{P}}, N_{\mathcal{P}_L})} > 1$. In other words, the benefits gained from obtaining more effective *i.i.d.* instances are outweighed by the negative consequences of incorrect pseudo-labels. This negative effect implies a scenario where “the poor get poorer”.

5.1.4 Experiments

We test MixMatch [18] and UDA [201] on CIFAR-10 and CIFAR-100 datasets [99]. See more experiments in [229]. We adopt the *coarse labels* (20 classes) in CIFAR-100 for training and test the performance for each *fine label* (100 classes). Thus our training on CIFAR-100 is a 20-class classification task and each coarse class contains 5 sub-populations.

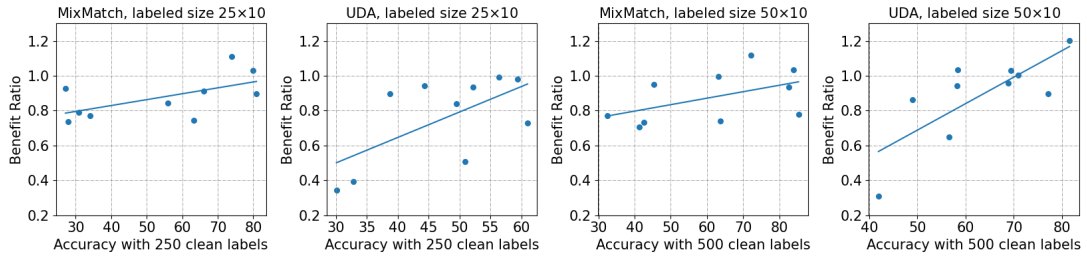


Figure 5.2: Benefit ratios (y -axis) versus baseline accuracies before SSL (x -axis) on CIFAR-10

Figure 5.3: Benefit ratios across explicit sub-populations.

The experiments aim to show that, even though the size of each sub-population is equal, disparate impacts exist in the model accuracy of different sub-populations, i.e., 1) *explicit sub-populations* such as classification labels in Figure 5.3, and 2) *implicit sub-populations* such as fine-categories under coarse classification labels in Figure 5.1.5. All the experiments in this subsection adopt both a balanced labeled dataset and a balanced unlabeled dataset.

Disparate impact across explicit sub-populations In this part, we demonstrate the disparate impact on model accuracy across different classification labels in CIFAR-10 datasets. Figure 5.2 illustrates this by plotting dots representing the results for each label class, along with a line representing the best linear fit of the dots. The y -axis represents the benefit ratios, while the x -axis shows the baseline accuracies before SSL. The results are presented for different sizes of labeled data, ranging from 25 to 50 per class on CIFAR-10. Figure 5.2 utilizes two SSL methods, namely MixMatch and UDA, specifically for CIFAR-10. Our statistical analysis reveals that class labels with higher

baseline accuracies tend to exhibit higher benefit ratios on CIFAR-10. This implies that the more “privileged” classes benefit to a greater extent from the application of SSL methods compared to the “less privileged” ones. Additionally, we observe that certain models with low baseline accuracy (located on the left side of the x-axis) exhibit relatively low benefit ratios close to zero when SSL is applied.

Disparate impact across implicit sub-populations We demonstrate the disparate impacts on model accuracy across different sub-populations on CIFAR-100 (fine labels). Figure 5.1.5 shows the benefit ratios of fine-labels (y -axis) versus baseline accuracies before SSL (x -axis) on CIFAR-100. Experiments are run 5 times for stability. Mean (dashed line) and standard deviation (shaded area) are plotted in the figure, where points in the shaded area indicate the converged local optima with a non-negligible probability. In Figure 5.1.5, we can statistically observe the disparate impact across different sub-populations on both datasets for three baseline SSL methods. We again observe very similar disparate improvements as presented in Figure 5.2 - for some classes in CIFAR-100, this ratio can even go negative. See [229] for more experiments related to the disparate impact on the demographic groups, e.g., race, and gender, which raise fairness concerns in real-world applications.

5.1.5 Takeaways

We summarize the takeaways as follows.

- We have theoretically and empirically shown that the disparate impact (the “rich”

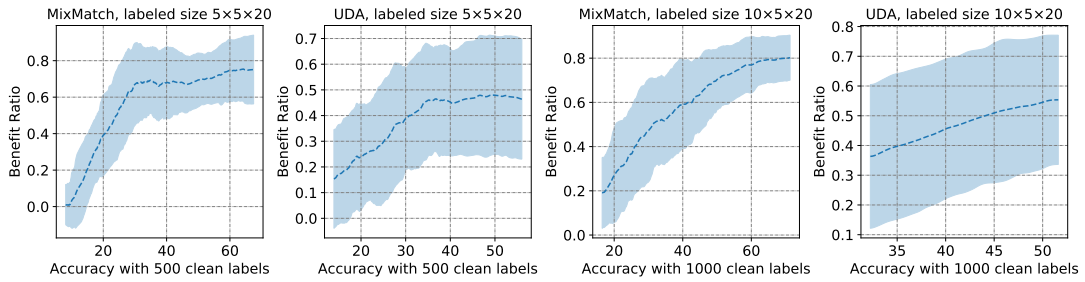


Figure 5.4: Benefit ratios across implicit sub-populations.

sub-populations get richer and the “poor” ones get poorer) exists universally for a broad family of SSL algorithms.

- We have also proposed and studied a new metric *benefit ratio* to facilitate the evaluation of SSL.
- Code is available at github.com/UCSC-REAL/Disparate-SSL.

5.2 Estimate Fairness with Missing Sensitive Attributes

Evaluating fairness can be challenging in practice because the sensitive attributes of data are often inaccessible due to privacy constraints. The go-to approach that the industry frequently adopts is using *off-the-shelf proxy models* to predict the missing sensitive attributes, e.g., Meta [2] and Twitter [15]. Despite its popularity, there are three important questions unanswered:

- Is directly using proxies efficacious in measuring fairness?
- If not, is it possible to accurately evaluate fairness using proxies only?
- Given the ethical controversy over inferring user private information, is it possible to

only use weak (i.e., inaccurate) proxies in order to protect privacy?

In this section, we first experimentally and theoretically show that directly using proxy models can give a false sense of (un)fairness (Section 5.2.1). Additionally, we develop an algorithm that is able to measure fairness (provably) accurately with only three properly identified proxies (Section 5.2.2). All our theoretical discussions in the main paper are specific to DP defined in Definition 3 but we include the *complete derivations* for EOd and EOo in Appendix.

5.2.1 Proxy Results Can be Misleading

This section provides an analysis of how much the measured fairness, if using proxies naively, can deviate from reality.

Using Proxy Models Directly. Consider a scenario with C proxy models denoted by the set $\mathcal{G} := \{g_1, \dots, g_C\}$. The noisy sensitive attributes are denoted as $\tilde{A}_c := g_c(X), \forall c \in [C]$ and the corresponding target dataset with \tilde{A} is $\tilde{\mathcal{D}} := \{(x_n, y_n, (\tilde{a}_n^1, \dots, \tilde{a}_n^C)) | n \in [N]\}$, drawn from a distribution $\tilde{\mathcal{D}}$. Similarly, by replacing A with \tilde{A} in \mathbf{H} , we can compute $\tilde{\mathbf{H}}$, which is the matrix-form noisy fairness metric estimated by the proxy model g (or \mathcal{G} if multiple proxy models are used). Define the directly measured fairness metric of f on $\tilde{\mathcal{D}}$ as follows.

Definition 12 (Proxy Disparity - DP).

$$\Delta^{DP}(\tilde{\mathcal{D}}, f) := \frac{1}{M(M-1)K} \sum_{\substack{a, a' \in [M] \\ k \in [K]}} |\tilde{\mathbf{H}}[a, k] - \tilde{\mathbf{H}}[a', k]|.$$

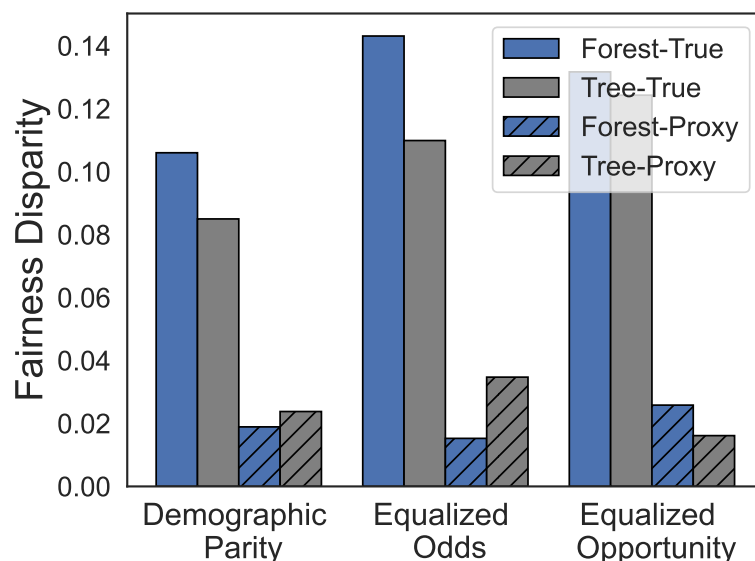


Figure 5.5: Fairness disparities of models on COMPAS [6].

A “seem-to-be-fair” model can be not fair Figure 5.5 shows the estimated fairness vs. true fairness on COMPAS [6] dataset with race as the sensitive attribute. *True (or Proxy)* stands for the disparities using ground-truth sensitive attribute values (or proxy model’s predictions). *Forest (or Tree)* indicates the random forest (or decisions tree) models. We use proxy models to predict race from the last name. There are two observations: 1) Models considered as fair according to proxies are actually unfair. The *Proxy* fairness disparities (0.02) can be much smaller than the *True* fairness disparities (> 0.10), giving a false sense of fairness. 2) Fairness misperception can mislead the model selection. The proxy disparities mistakenly indicate random forest models have smaller disparities (DP and EOd) than decision tree models, but in fact it is the opposite.

Estimation Error Analysis We study the error of proxy disparity and give practical guidelines implied by analysis. Intuitively, the estimation error of proxy disparity depends on the error of the proxy model g . Recall \mathbf{p} , $\tilde{\mathbf{p}}$, \mathbf{T} and \mathbf{T}_k are clean prior, noisy prior, global transition matrix, and local transition matrix. Denote by $\mathbf{\Lambda}_{\tilde{\mathbf{p}}}$ and $\mathbf{\Lambda}_{\mathbf{p}}$ the square diagonal matrices constructed from $\tilde{\mathbf{p}}$ and \mathbf{p} . We formally prove the upper bound of estimation error for the directly measured metrics in Theorem 16 (See Appendix D.3.1 for the proof).

Theorem 16 (Error Upper Bound of Proxy Disparities). *Denote the estimation error of the proxy disparity by*

$$Err^{raw} := |\tilde{\Delta}^{DP}(\tilde{\mathcal{D}}, f) - \Delta^{DP}(\mathcal{D}, f)|.$$

Its upper bound is:

$$Err^{raw} \leq \frac{2}{K} \sum_{k \in [K]} \left(\bar{h}_k \underbrace{\|\mathbf{\Lambda}_{\tilde{\mathbf{p}}}(\mathbf{T}^{-1}\mathbf{T}_k - \mathbf{I})\mathbf{\Lambda}_{\tilde{\mathbf{p}}}^{-1}\|_1}_{\text{cond. indep. violation}} + \delta_k \underbrace{\|\mathbf{\Lambda}_{\mathbf{p}}\mathbf{T}_k\mathbf{\Lambda}_{\tilde{\mathbf{p}}}^{-1} - \mathbf{I}\|_1}_{\text{error of } g} \right),$$

$$\text{where } \bar{h}_k := \frac{1}{M} \sum_{a \in [M]} H[a, k], \delta_k := \max_{a \in [M]} |H[a, k] - \bar{h}_k|.$$

It shows the error of proxy disparity depends on:

- \bar{h}_k : The average confidence of $f(X)$ on class k over all sensitive groups. For example, if f is a crime prediction model and A is race, a biased f [6] may predict that the crime ($k = 1$) rate for different races are 0.1, 0.2 and 0.6 respectively, then $\bar{h}_1 = \frac{0.1+0.2+0.6}{3} = 0.3$, and it is an approximation (unweighted by sample size) of the average crime rate over the entire population. The term depends on \mathcal{D} and f only (i.e., the true fairness disparity), and independent of any estimation algorithm.

- δ_k : The maximum disparity between confidence of $f(X)$ on class k and average confidence \bar{h}_k across all sensitive groups. Using the same example, $\delta_1 = \max(|0.1 - 0.3|, |0.2 - 0.3|, |0.6 - 0.3|) = 0.3$. It is an approximation of the underlying fairness disparity, and larger δ_k indicates f is more biased on \mathcal{D} . The term is also dependent on \mathcal{D} and f (i.e., the true fairness disparity), and independent of any estimation algorithm.
- *Conditional Independence Violation*: The term is dependent on the proxy model g 's prediction \tilde{A} in terms of the transition matrix (\mathbf{T} and \mathbf{T}_k) and noisy prior probability ($\tilde{\mathbf{p}}$). The term goes to 0 when $\mathbf{T} = \mathbf{T}_k$, which implies \tilde{A} and $f(X)$ are independent conditioned on A . This is the common assumption made in the prior work [8, 148, 49]. And this term measures how much the conditional independence assumption is violated.
- *Error of g* : The term depends on the proxy model g . It goes to 0 when $\mathbf{T}_k = \mathbf{I}$ which implies the error rates of g 's prediction is 0, i.e., g is perfectly accurate. It measures the impact of g 's error on the fairness estimation error.

Case Study. To help better understand the upper bound, we consider a simplified case when both f and A are binary. We further assume the conditional independence condition to remove the third term listed above in Theorem 16. See Appendix D.2.2 for the formal definition of conditional independence. Please note that we only assume it for the purpose of demonstrating a less complicated theoretical result, we do *not* need this assumption in our proposed algorithm later. Corollary 6 summarizes the result.

Corollary 6. *For a binary classifier f and a binary sensitive attribute $A \in \{1, 2\}$, when*

($\tilde{A} \perp\!\!\!\perp f(X)|A$) holds, Theorem 16 is simplified to

$$Err^{raw} \leq \delta \left(\mathbb{P}(A = 1|\tilde{A} = 2) + \mathbb{P}(A = 2|\tilde{A} = 1) \right),$$

where $\delta = |\mathbb{P}(f(X) = 1|A = 1) - \mathbb{P}(f(X) = 1|A = 2)|$.

Corollary 6 shows the estimation error of proxy disparity is proportional to the true underlying disparity between sensitive groups (i.e., δ) and the proxy model's error rates. In other words, the uncalibrated metrics can be highly inaccurate when f is highly biased or g has poor performance. This leads to the following suggestions:

Guidelines for Practitioners. We should only trust the estimated fairness from proxy models when (1) the proxy model g has good performance *and* (2) the true disparity is small (i.e., the target model f is not highly biased). In practice, without true sensitive attributes, we can roughly infer the true disparity based on the problem domain and known history. For example, racial disparity in hiring is known to exist for a long time. We only need to know if the disparity is extremely large or not.

In practice, both conditions required to trust the proxy results are frequently violated. When we want to measure f 's fairness, often we already have some fairness concerns and therefore the underlying fairness disparity is unlikely to be negligible. And the proxy model g is usually inaccurate due to privacy concerns (discussed in Section 5.2.2.2) and distribution shift. This motivates us to develop an approach for more accurate estimates.

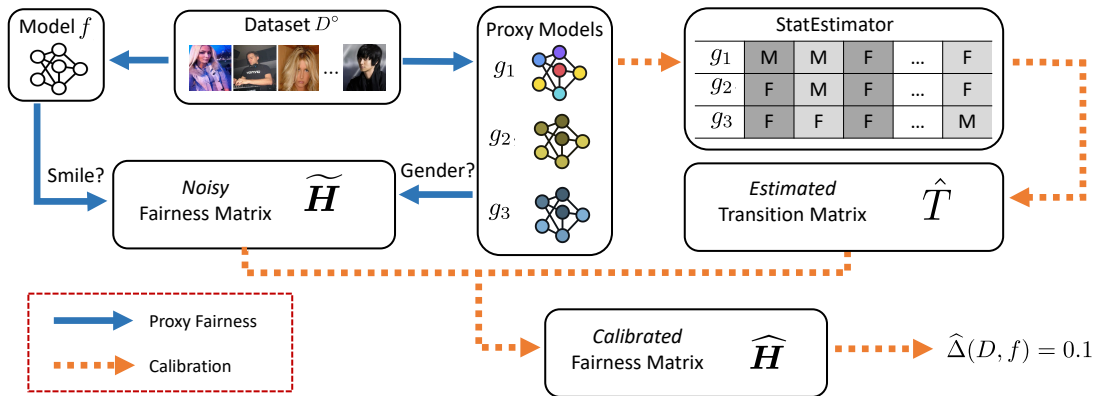


Figure 5.6: Overview of our algorithm that estimates fairness using only weak proxy models.

5.2.2 Weak Proxies Suffice

In this section, we show that by properly using a set of proxy models, we are able to guarantee an unbiased estimate of the true fairness measures. The proposed calibration algorithm is illustrated in Figure 5.6, where we first directly estimate the noisy fairness matrix with proxy models (blue arrows), and then calibrate the estimated fairness matrix (orange arrows).

5.2.2.1 Proposed Algorithm

With a given proxy model g that labels sensitive attributes, we can anatomize the relationship between the true disparity and the proxy disparity. The following theorem targets DP and see Appendix D.3.2 for results with respect to EOd and EOp and their proofs.

Theorem 17. *[Closed-form Relationship (DP)] The closed-form relationship between*

the true fairness vector $\mathbf{H}[:, k]$ and the noisy fairness vector $\widetilde{\mathbf{H}}[:, k]$ is the following:

$$\mathbf{H}[:, k] = (\mathbf{T}_k^\top \boldsymbol{\Lambda}_{\mathbf{p}})^{-1} \boldsymbol{\Lambda}_{\tilde{\mathbf{p}}} \widetilde{\mathbf{H}}[:, k], \forall k \in [K].$$

Insights. Theorem 17 reveals that the proxy disparity and the corresponding true disparity are related in terms of three key statistics: noisy prior $\tilde{\mathbf{p}}$, clean prior \mathbf{p} , and local transition matrix \mathbf{T}_k . Ideally, if we have the ground-truth values of them, we can calibrate the noisy fairness vectors to their corresponding ground-truth vectors (and therefore obtaining the perfectly accurate fairness metrics) using Theorem 17. Hence, the most important step is to estimate \mathbf{T}_k , \mathbf{p} , and $\tilde{\mathbf{p}}$ without knowing the ground-truth values of A . Once we have those estimated key statistics, we can easily plug them into the above equation as the calibration step. Figure 5.6 shows the overview of our algorithm.

Algorithm: Fairness calibration. We summarize the method in Algorithm 6. In Line 4, we use the sample mean in the uncalibrated form to estimate $\widetilde{\mathbf{H}}$ as

$$\widetilde{H}[\tilde{a}, k] \approx \frac{1}{N} \sum_{n=1}^N \mathbb{1}(f(x_n = k | \tilde{a}_n = \tilde{a}))$$

and $\tilde{\mathbf{p}}$ as $\tilde{p}[\tilde{a}] = \mathbb{P}(\tilde{A} = \tilde{a}) \approx \frac{1}{N} \sum_{n=1}^N \mathbb{1}(\tilde{a}_n = \tilde{a}), \forall \tilde{a} \in [M]$. In Line 5, we plug in an existing transition matrix and prior probability estimator to estimate \mathbf{T}_k and \mathbf{p} with only mild adaption that will be introduced shortly. Note that although we choose a specific estimator, our algorithm is a flexible framework that is compatible with any `StatEstimator` proposed in the noisy label literature [123, 230, 231].

Details: Estimating Key Statistics. The algorithm requires us to estimate \mathbf{T}_k and \mathbf{p} based on the predicted \tilde{A} by proxy models. In the literature of noisy learning,

Algorithm 6 Fairness calibration algorithm (DP)

- 1: **Input:** A set of proxy models $\mathcal{G} = \{g_1, \dots, g_C\}$. Target dataset D° . Target model f . Transition matrix and prior probability estimator `StatEstimator`.
Predict sensitive attributes using all $g \in \mathcal{G}$
 - 2: $\tilde{a}_n^c \leftarrow g_c(x_n), \forall c \in [C], n \in [N]$
Build the dataset with noisy sensitive attributes
 - 3: $\tilde{D} \leftarrow \{(x_n, y_n, (\tilde{a}_n^1, \dots, \tilde{a}_n^C)) | n \in [N]\}$
Estimate fairness matrix and prior with sample mean
 - 4: $\tilde{\mathbf{H}}, \tilde{\mathbf{p}} \leftarrow \text{DirectEst}(\tilde{D}, f)$
Estimate key statistics: \mathbf{p} and \mathbf{T}_k
 - 5: $\{\hat{\mathbf{T}}_1, \dots, \hat{\mathbf{T}}_K\}, \hat{\mathbf{p}} \leftarrow \text{StatEstimator}(\tilde{D}, f)$
Calibrate each fairness vector with Theorem 17
 - 6: $\forall k \in [K] : \hat{\mathbf{H}}[:, k] \leftarrow (\hat{\mathbf{T}}_k^\top \Lambda_{\hat{\mathbf{p}}})^{-1} \Lambda_{\hat{\mathbf{p}}} \tilde{\mathbf{H}}[:, k]$
Calculate the final fairness metric
 - 7: $\hat{\Delta}(\tilde{D}, f) \leftarrow \frac{1}{M(M-1)K} \sum_{\substack{a, a' \in [M] \\ k \in [K]}} |\hat{\mathbf{H}}[a, k] - \hat{\mathbf{H}}[a', k]|$
 - 8: **Output:** The calibrated fairness metric $\hat{\Delta}(\tilde{D}, f)$
-

there exists several feasible algorithms [119, 156, 146, 141, 230]. We choose HOC [230] because it has stronger theoretical guarantee and lower sample complexity than most existing estimators. Intuitively, if given three proxy models, the joint distributions of their predictions would encode \mathbf{T}_k and \mathbf{p} , i.e.,

$$\mathbb{P}(\tilde{A}_1, \tilde{A}_2, \tilde{A}_3) = \text{Func}(\{\mathbf{T}_k\}_{k \in [K]}, \mathbf{p}).$$

Algorithm 7 StatEstimator: HOCFair (DP)

- 1: **Input:** Noisy dataset \tilde{D} . Target model f .
Get the number of noisy attributes (i.e., # proxy models)
 - 2: $C \leftarrow \#\text{Attribute}(\tilde{D})$
Get 2-Nearest-Neighbors of x_n and save their attributes as x_n 's attribute
 - 3: **if** $C < 3$ **then**
 - 4: $\{(x_n, y_n, (\tilde{a}_n^1, \dots, \tilde{a}_n^{3C})) | n \in [N]\} \leftarrow \text{Get2NN}(\tilde{D})$
 - 5: $\tilde{D} \leftarrow \{(x_n, y_n, (\tilde{a}_n^1, \dots, \tilde{a}_n^{3C})) | n \in [N]\}$
 - 6: **end if**
Randomly sample 3 noisy attributes for each instance
 - 7: $\{(\tilde{a}_n^1, \tilde{a}_n^2, \tilde{a}_n^3) | n \in [N]\} \leftarrow \text{Sample}(\tilde{D})$
Get estimates $\mathbf{p} \approx \hat{\mathbf{p}}$
 - 8: $(\hat{\mathbf{T}}, \hat{\mathbf{p}}) \leftarrow \text{HOC}(\{(\tilde{a}_n^1, \tilde{a}_n^2, \tilde{a}_n^3) | n \in [N]\})$
Get estimates $\mathbf{T}_k \approx \hat{\mathbf{T}}_k$
 - 9: $(\hat{\mathbf{T}}_k, -) \leftarrow \text{HOC}(\{(\tilde{a}_n^1, \tilde{a}_n^2, \tilde{a}_n^3) | n \in [N], f(x_n) = k\}), \quad \forall k \in [K]$
Return the estimated statistics
 - 10: **Output:** $\{\hat{\mathbf{T}}_1, \dots, \hat{\mathbf{T}}_K\}, \hat{\mathbf{p}}$
-

For example, with the chain rule and independence among proxy predictions conditioned on A , we have:

$$\begin{aligned} & \mathbb{P}(\tilde{A}_1 = \tilde{a}_1, \tilde{A}_2 = \tilde{a}_2, \tilde{A}_3 = \tilde{a}_3 | f(X) = k) \\ &= \sum_{a \in [M]} \mathbb{P}(A = a | f(X) = k) \cdot T_k[a, \tilde{a}_1] \cdot T_k[a, \tilde{a}_2] \cdot T_k[a, \tilde{a}_3]. \end{aligned}$$

HOC counts the frequency of different $(\tilde{A}_1, \tilde{A}_2, \tilde{A}_3)$ patterns to obtain LHS and solve equations to get T_k 's in the RHS.

Algorithm: HOCFair. More specifically, Algorithm 7 shows how we adapt HOC as `StatEstimator` (in Algorithm 6, Line 5), namely `HOCFair`. The original HOC uses one proxy model and simulates the other two based on clusterability condition [230], which assumes x_n and its 2-nearest-neighbors share the same true sensitive attribute, and therefore their noisy attributes can be used to simulate the output of proxy models. If this condition does not hold [231], we can directly use more proxy models. With a sufficient number of noisy attributes, we can randomly select a subset of them for every sample as Line 7, and then approximate T_k with \hat{T}_k in Line 9. In our experiments, we test both using one proxy model and multiple proxy models. See more details of our implementations in Appendix D.4 and HOC in Chapter 2.

5.2.2.2 Requirements of Proxy Models

To use our algorithm, there are two practical questions for practitioners: 1) what properties proxy models should satisfy and 2) how many proxy models are needed. The first question is answered by two requirements made in the estimation algorithm HOC:

Requirement 1 (Informativeness of Proxies). *The noisy attributes given by each proxy model g are informative, i.e., $\forall k \in [M]$, 1) T_k is non-singular and 2) either $T_k[a, a] > \mathbb{P}(\tilde{A} = a | f(X) = k)$ or $T_k[a, a] > T_k[a, a'], \forall a' \neq a$.*

Requirement 1 is the prerequisite of getting a feasible and unique estimate of \mathbf{T}_k [230], where the non-singular requirement ensures the matrix inverse in Theorem 17 exists and the constraints on $T_k[a, a]$ describes the worst tolerable performance of g . When $M = 2$, the constraints can be simplified as $T_k[1, 2] + T_k[2, 1] < 1$ [123, 124], i.e., g 's predictions are better than random guess in binary classification. If this requirement is violated, there might exist more than one feasible estimate of \mathbf{T}_k , making the problem insoluble.

The above requirement is weak. The proxies are merely required to positively correlate with the true sensitive attributes. We discuss the privacy implication of using weak proxies shortly after.

Requirement 2 (Independence between Proxies). *The noisy attributes predicted by proxy models $g_1(X), \dots, g_C(X)$ are independent and identically distributed (i.i.d.) given A .*

Requirement 2 ensures the additional two proxy models provide more information than using only one classifier. If it is violated, we would still get an estimate but may be inaccurate. Note this requirement is different from the conditional independence often assumed in the fairness literature [8, 148, 49], which is $g(X) \perp\!\!\!\perp f(X)|A$ rather than ours $g_1(X) \perp\!\!\!\perp g_2(X) \perp\!\!\!\perp g_3(X)|A$.

The second question (how many proxy models are needed) has been answered by Theorem 5 in [122], which we summarize in the following.

Lemma 8. *If satisfying Requirements 1–2, three proxy models are both sufficient and*

necessary to identify \mathbf{T}_k .

How to Protect Privacy with Weak Proxies. Intuitively, weak proxies can protect privacy better than strong proxies since the predictions are noisier, i.e., less informative. We connect weak proxy’s privacy-preserveness to *differential privacy* [53]. Assume misclassification probability on \tilde{A} is bounded across all samples, i.e., $\forall a \in [M], a' \in [M], a \neq a'$:

$$\begin{aligned} \max_{x \in X} \mathbb{P}(\tilde{A} = a | A = a, X = x) &\leq 1 - \epsilon_0, \\ \min_{x \in X} \mathbb{P}(\tilde{A} = a | A = a', X = x) &\geq \epsilon_1. \end{aligned}$$

According to the definition of *label differential privacy* [53], we show that the privacy of the sensitive attribute A , which is the “label” of proxy models, satisfies $\ln(\frac{1-\epsilon_0}{\epsilon_1})$ -DP. See Appendix D.3.6 for the proof.

In practice, if the above assumption does not hold naturally by proxies, we can add noise to impose it. When practitioners think proxies are too strong, they can add additional noise to reduce informativeness, further protecting privacy. When we intentionally make the proxies weaker by flipping predicted sensitive attributes with probability 0.4, it corresponds to 0.41-DP ($\epsilon_0 = \epsilon_1 = 0.4$) protection.

5.2.2.3 Theoretical Guarantee

We theoretically analyze estimation errors on our calibrated metrics. Denote by $\hat{\Delta}^{\text{DP}}(\tilde{\mathcal{D}}, f)$ the calibrated DP disparity evaluated on our calibrated fairness matrix $\hat{\mathbf{H}}$. We have:

Theorem 18 (Error Upper Bound of Calibrated Metrics). *Denote the estimation error of the calibrated fairness metrics by $Err^{cal} := |\widehat{\Delta}^{DP}(\widetilde{\mathcal{D}}, f) - \Delta^{DP}(\mathcal{D}, f)|$. Then:*

$$Err^{cal} \leq \frac{2}{K} \sum_{k \in [K]} \|\Lambda_{\hat{\mathbf{p}}}^{-1}\|_1 \|\Lambda_{\mathbf{p}} \mathbf{H}[:, k]\|_{\infty} \varepsilon(\widehat{\mathbf{T}}_k, \hat{\mathbf{p}}),$$

where $\varepsilon(\widehat{\mathbf{T}}_k, \hat{\mathbf{p}}) := \|\Lambda_{\hat{\mathbf{p}}}^{-1} \Lambda_{\mathbf{p}} - \mathbf{I}\|_1 \|\mathbf{T}_k \widehat{\mathbf{T}}_k^{-1}\|_1 + \|\mathbf{I} - \mathbf{T}_k \widehat{\mathbf{T}}_k^{-1}\|_1$ is the error induced by calibration. With a perfect estimator $\widehat{\mathbf{T}}_k = \mathbf{T}_k$ and $\hat{\mathbf{p}}_k = \mathbf{p}_k, \forall k \in [K]$, we have $Err^{cal} = 0$.

Theorem 18 shows the upper bound of estimation error mainly depends on the estimates $\widehat{\mathbf{T}}_k$ and $\hat{\mathbf{p}}$, i.e., the following two terms in $\varepsilon(\widehat{\mathbf{T}}_k, \hat{\mathbf{p}})$:

$$\|\Lambda_{\hat{\mathbf{p}}}^{-1} \Lambda_{\mathbf{p}} - \mathbf{I}\|_1 \|\mathbf{T}_k \widehat{\mathbf{T}}_k^{-1}\|_1 \quad \text{and} \quad \|\mathbf{I} - \mathbf{T}_k \widehat{\mathbf{T}}_k^{-1}\|_1.$$

When the estimates are perfect, i.e., $\widehat{\mathbf{T}}_k = \mathbf{T}_k$ and $\hat{\mathbf{p}} = \mathbf{p}$, then both terms go to 0 because $\Lambda_{\hat{\mathbf{p}}}^{-1} \Lambda_{\mathbf{p}} = \mathbf{I}$ and $\mathbf{T}_k \widehat{\mathbf{T}}_k^{-1} = \mathbf{I}$. Together with Lemma 8, we can show the optimality of our algorithm as follows.

Theorem 19. *When Requirements 1–2 hold for three proxy models, the calibrated fairness metrics given by Algorithm 6 with key statistics estimated by Algorithm 7 achieve zero error, i.e.,*

$$|\widehat{\Delta}^{DP}(\widetilde{\mathcal{D}}, f) - \Delta^{DP}(\mathcal{D}, f)| = 0.$$

Besides, we compare the error upper bound of our method with the exact error (not its upper bond) in the case of Corollary 6, and summarize the result in Corollary 7.

Corollary 7. *For a binary classifier f and a binary sensitive attribute $A \in \{1, 2\}$, when $(\tilde{A} \perp\!\!\!\perp f(X)|A)$ and $\mathbf{p} = [0.5, 0.5]^\top$, the proposed calibration method is guaranteed to be*

more accurate than the uncalibrated measurement, i.e., $Err^{cal} \leq Err^{raw}$, if

$$\varepsilon(\widehat{\mathbf{T}}_k, \hat{\mathbf{p}}) \leq \gamma := \max_{k' \in \{1,2\}} \frac{e_1 + e_2}{1 + \frac{\|\mathbf{H}[:,k']\|_1}{\Delta^{DP}(\mathcal{D},f)}}, \forall k \in \{1,2\}.$$

Corollary 7 shows when the error $\varepsilon(\widehat{\mathbf{T}}_k, \hat{\mathbf{p}})$ that is induced by inaccurate $\widehat{\mathbf{T}}_k$ and $\hat{\mathbf{p}}$ is below the threshold γ , our method is guaranteed to lead to a smaller estimation error compared to the uncalibrated measurement under the considered setting. The threshold implies that, adopting our method rather than the uncalibrated measurement can be greatly beneficial when e_1 and e_2 are high (i.e., g is inaccurate) *or* when the normalized (true) fairness disparity $\frac{\Delta^{DP}(\mathcal{D},f)}{\|\mathbf{H}[:,k']\|_1}$ is high (i.e., f is highly biased).

5.2.2.4 Guidelines for Practitioners

We provide a set of guidelines implied by our theoretical results.

When to Use Our Algorithm. Corollary 7 shows that our algorithm is preferred over directly using proxies when 1) the proxy model g is weak *or* 2) the true disparity is large.

How to Best Use Our Algorithm. Section 5.2.2.2 implies a set of principles for selecting proxy models:

- i) [Requirement 1] Even if proxy models are weak, as long as they are informative, e.g., in binary case the performance is better than random guess, then it is enough for estimations.
- ii) [Requirement 2] We should try to make sure the predictions of proxy models are i.i.d.,

which is more important than using more proxy models. One way of doing it is to choose proxy models trained on different data sources.

iii) [Lemma 8] At least three proxy models are preferred.

5.2.3 Takeaways

We summarize the takeaways as follows.

- We have offered a viable solution, i.e., by using only weak proxies, we can protect data privacy while still being able to measure fairness.
- To this end, we have designed an algorithm that, though only based on weak proxies, can still provably achieve accurate fairness estimations. We show our algorithm can effectively measure the bias and provide a set of guidelines for practitioners on how to use proxies properly.
- Code is available at <https://github.com/UCSC-REAL/fair-eval>.

Chapter 6

Conclusions

In this dissertation, we have addressed the challenges of data quality in machine learning and proposed practical and provable solutions for handling weakly supervised data. The research explores the realm of data-centric AI and introduces a pipeline consisting of three crucial procedures to tackle data issues in weakly supervised learning. The first procedure focuses on data diagnosis, where noise rates are learned when true labels are missing. The second procedure involves data curation, which identifies and fixes corrupted labels. Finally, the pipeline incorporates robust learning algorithms that leverage the curated data for improved model performance.

Furthermore, this dissertation emphasizes the importance of evaluating model performance beyond accuracy when dealing with imperfect data. By considering multi-dimensional evaluation metrics, researchers and practitioners can better understand a model's capabilities and limitations in real-world scenarios.

Importantly, all the works presented in this dissertation have been open-sourced,

fostering collaboration and enabling the wider research community to benefit from the proposed solutions. Furthermore, the data diagnosis and curation pipeline discussed in this research are readily accessible at the GitHub repository <https://github.com/Docta-ai/docta>.

By addressing the challenges of weakly supervised learning and providing practical tools and methodologies, this dissertation contributes to the advancement of data-centric AI and reinforces the notion that high-quality labeled data is the key to training robust machine learning models. With the proposed pipeline and evaluation framework, researchers and practitioners can enhance the performance and reliability of AI systems, even in the presence of noisy or unlabeled data, ultimately paving the way for more effective and trustworthy applications of artificial intelligence in various domains.

Bibliography

- [1] Vibhu Agarwal, Tanya Podchiyska, Juan M Banda, Veena Goel, Tiffany I Leung, Evan P Minty, Timothy E Sweeney, Elsie Gyang, and Nigam H Shah. Learning statistical models of phenotypes using noisy labeled training data. *Journal of the American Medical Informatics Association*, 23(6):1166–1173, 2016.
- [2] Rachad Alao, Miranda Bogen, Jingang Miao, Ilya Mironov, and Jonathan Tannen. How Meta is working to assess fairness in relation to race in the U.S. across its products and systems. <https://ai.facebook.com/research/publications/how-meta-is-working-to-assess-fairness-in-relation-to-race-in-the-us-across-its-products-and-systems>, 2021. [Online; accessed 15-Sep-2022].
- [3] Amr M. Alexandari, Anshul Kundaje, and Avanti Shrikumar. Maximum likelihood with bias-corrected calibration is hard-to-beat at label shift adaptation. In *Proceedings of the 37th International Conference on Machine Learning, ICML '20*, 2020.
- [4] Ehsan Amid, Manfred K Warmuth, and Sriram Srinivasan. Two-temperature

- logistic regression based on the tsallis divergence. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2388–2396. PMLR, 2019.
- [5] Ehsan Amid, Manfred KK Warmuth, Rohan Anil, and Tomer Koren. Robust bi-tempered logistic loss based on bregman divergences. In *Advances in Neural Information Processing Systems*, pages 14987–14996, 2019.
- [6] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. Machine bias. In *Ethics of Data and Analytics*, pages 254–264. Auerbach Publications, 2016.
- [7] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In Karl Aberer, Key-Sun Choi, Natasha Noy, Dean Allemang, Kyung-Il Lee, Lyndon Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux, editors, *The Semantic Web*, pages 722–735, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.
- [8] Pranjal Awasthi, Alex Beutel, Matthäus Kleindessner, Jamie Morgenstern, and Xuezhi Wang. Evaluating fairness of machine learning models under uncertain and incomplete information. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 206–214, 2021.
- [9] Eugene Bagdasaryan, Omid Poursaeed, and Vitaly Shmatikov. Differential privacy has disparate impact on model accuracy. *Advances in Neural Information Processing Systems*, 32:15479–15488, 2019.

- [10] Dara Bahri, Heinrich Jiang, and Maya Gupta. Deep k-nn for noisy labels. In *International Conference on Machine Learning*, pages 540–550. PMLR, 2020.
- [11] Yingbin Bai and Tongliang Liu. Me-momentum: Extracting hard confident examples from noisily labeled data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9312–9321, 2021.
- [12] Yingbin Bai, Erkun Yang, Bo Han, Yanhua Yang, Jiatong Li, Yinian Mao, Gang Niu, and Tongliang Liu. Understanding and improving early stopping for learning with noisy labels. *Advances in Neural Information Processing Systems*, 34, 2021.
- [13] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.
- [14] Roberto Battiti. Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on neural networks*, 5(4):537–550, 1994.
- [15] Luca Belli, Kyra Yee, Uthaipon Tantipongpipat, Aaron Gonzales, Kristian Lum, and Moritz Hardt. County-level algorithmic audit of racial bias in twitter’s home timeline. *arXiv preprint arXiv:2211.08667*, 2022.
- [16] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

- [17] 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*, 2019.
- [18] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin Raffel. Mixmatch: A holistic approach to semi-supervised learning. *arXiv preprint arXiv:1905.02249*, 2019.
- [19] Antonin Berthon, Bo Han, Gang Niu, Tongliang Liu, and Masashi Sugiyama. Confidence scores make instance-dependent label-noise learning possible. In *Proceedings of the 38th International Conference on Machine Learning, ICML, 2021*.
- [20] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.
- [21] Valerii V Buldygin and Yu V Kozachenko. Sub-gaussian random variables. *Ukrainian Mathematical Journal*, 32(6):483–489, 1980.
- [22] Toon Calders, Faisal Kamiran, and Mykola Pechenizkiy. Building classifiers with independency constraints. In *2009 IEEE International Conference on Data Mining Workshops*, pages 13–18. IEEE, 2009.
- [23] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning. *MIT Press*, 2006.
- [24] Satrajit Chatterjee. Coherent gradients: An approach to understanding general-

- ization in gradient descent-based optimization. In *International Conference on Learning Representations*, 2020.
- [25] Jiaao Chen, Zichao Yang, and Diyi Yang. Mixtext: Linguistically-informed interpolation of hidden space for semi-supervised text classification. *arXiv preprint arXiv:2004.12239*, 2020.
- [26] Jiahao Chen, Nathan Kallus, Xiaojie Mao, Geoffry Svacha, and Madeleine Udell. Fairness under unawareness: Assessing disparity when protected class is unobserved. In *Proc. of FAccT*, 2019.
- [27] Mingda Chen, Qingming Tang, Karen Livescu, and Kevin Gimpel. Variational sequential labelers for semi-supervised learning. *arXiv preprint arXiv:1906.09535*, 2019.
- [28] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.
- [29] Yatong Chen, Reilly Raab, Jialu Wang, and Yang Liu. Fairness transferability subject to bounded distribution shift. *Advances in Neural Information Processing Systems*, 35:11266–11278, 2022.
- [30] Hao Cheng, Zhaowei Zhu, Xingyu Li, Yifei Gong, Xing Sun, and Yang Liu. Learning with instance-dependent label noise: A sample sieve approach. In *International Conference on Learning Representations*, 2021.

- [31] Hao Cheng, Zhaowei Zhu, Xing Sun, and Yang Liu. Demystifying how self-supervised features improve training from noisy labels. *arXiv preprint arXiv:2110.09022*, 2021.
- [32] Hao Cheng, Zhaowei Zhu, Xing Sun, and Yang Liu. Mitigating memorization of noisy labels via regularization between representations. In *International Conference on Learning Representations (ICLR)*, 2023.
- [33] Jiacheng Cheng, Tongliang Liu, Kotagiri Ramamohanarao, and Dacheng Tao. Learning with bounded instance-and label-dependent label noise. In *Proceedings of the 37th International Conference on Machine Learning, ICML '20*, 2020.
- [34] Alexandra Chouldechova. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data*, 5(2):153–163, 2017.
- [35] Kevin Clark, Minh-Thang Luong, Christopher D Manning, and Quoc V Le. Semi-supervised sequence modeling with cross-view training. *arXiv preprint arXiv:1809.08370*, 2018.
- [36] Imre Csiszár. Information-type measures of difference of probability distributions and indirect observation. *studia scientiarum Mathematicarum Hungarica*, 2:229–318, 1967.
- [37] Andrew M Dai and Quoc V Le. Semi-supervised sequence learning. *Advances in neural information processing systems*, 28:3079–3087, 2015.

- [38] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.
- [39] 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.
- [40] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 4171–4186, June 2019.
- [41] Emily Diana, Wesley Gill, Michael Kearns, Krishnaram Kenthapadi, Aaron Roth, and Saeed Sharifi-Malvajerdi. Multiaccurate proxies for downstream fairness. *arXiv preprint arXiv:2107.04423*, 2021.
- [42] Emily Diana, Wesley Gill, Michael Kearns, Krishnaram Kenthapadi, Aaron Roth, and Saeed Sharifi-Malvajerdi. Multiaccurate proxies for downstream fairness. In *Proc. of FAccT*, 2022.
- [43] Mengnan Du, Subhabrata Mukherjee, Guanchu Wang, Ruixiang Tang, Ahmed Awadallah, and Xia Hu. Fairness via representation neutralization. *Advances in Neural Information Processing Systems*, 34:12091–12103, 2021.
- [44] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.
- [45] Marc N Elliott, Peter A Morrison, Allen Fremont, Daniel F McCaffrey, Philip

- Pantoja, and Nicole Lurie. Using the census bureau’s surname list to improve estimates of race/ethnicity and associated disparities. *Health Services and Outcomes Research Methodology*, 9(2):69–83, 2009.
- [46] Pablo A Estévez, Michel Tesmer, Claudio A Perez, and Jacek M Zurada. Normalized mutual information feature selection. *IEEE Transactions on neural networks*, 20(2):189–201, 2009.
- [47] Tongtong Fang, Nan Lu, Gang Niu, and Masashi Sugiyama. Rethinking importance weighting for deep learning under distribution shift. *arXiv preprint arXiv:2006.04662*, 2020.
- [48] Vitaly Feldman. Does learning require memorization? a short tale about a long tail. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 954–959, 2020.
- [49] Riccardo Fogliato, Alexandra Chouldechova, and Max G’Sell. Fairness evaluation in presence of biased noisy labels. In *Proc. of AISTat*, 2020.
- [50] Deep Ganguli, Liane Lovitt, Jackson Kernion, Amanda Askell, Yuntao Bai, Saurav Kadavath, Ben Mann, Ethan Perez, Nicholas Schiefer, Kamal Ndousse, et al. Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned. *arXiv preprint arXiv:2209.07858*, 2022.
- [51] Ruoyuan Gao and Chirag Shah. Addressing bias and fairness in search systems.

- In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2643–2646, 2021.
- [52] Wei Gao, Bin-Bin Yang, and Zhi-Hua Zhou. On the resistance of nearest neighbor to random noisy labels. *arXiv preprint arXiv:1607.07526*, 2016.
- [53] Badih Ghazi, Noah Golowich, Ravi Kumar, Pasin Manurangsi, and Chiyuan Zhang. Deep learning with label differential privacy. *Advances in Neural Information Processing Systems*, 34:27131–27145, 2021.
- [54] Azin Ghazimatin, Matthaus Kleindessner, Chris Russell, Ziawasch Abedjan, and Jacek Golebiowski. Measuring fairness of rankings under noisy sensitive information. In *2022 ACM Conference on Fairness, Accountability, and Transparency*, pages 2263–2279, 2022.
- [55] Aritra Ghosh, Himanshu Kumar, and PS Sastry. Robust loss functions under label noise for deep neural networks. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.
- [56] Aritra Ghosh and Andrew Lan. Contrastive learning improves model robustness under label noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2703–2708, 2021.
- [57] Elizabeth Gómez, Ludovico Boratto, and Maria Salamó. Disparate impact in item recommendation: A case of geographic imbalance. In *European Conference on Information Retrieval*, pages 190–206. Springer, 2021.

- [58] Maoguo Gong, Hao Li, Deyu Meng, Qiguang Miao, and Jia Liu. Decomposition-based evolutionary multiobjective optimization to self-paced learning. *IEEE Transactions on Evolutionary Computation*, 23(2):288–302, 2018.
- [59] Yves Grandvalet, Yoshua Bengio, et al. Semi-supervised learning by entropy minimization. *CAP*, 367:281–296, 2005.
- [60] Lan-Zhe Guo, Zhen-Yu Zhang, Yuan Jiang, Yu-Feng Li, and Zhi-Hua Zhou. Safe deep semi-supervised learning for unseen-class unlabeled data. In *International Conference on Machine Learning*, pages 3897–3906. PMLR, 2020.
- [61] Suchin Gururangan, Tam Dang, Dallas Card, and Noah A Smith. Variational pretraining for semi-supervised text classification. *arXiv preprint arXiv:1906.02242*, 2019.
- [62] Bo Han, Quanming Yao, Tongliang Liu, Gang Niu, Ivor W Tsang, James T Kwok, and Masashi Sugiyama. A survey of label-noise representation learning: Past, present and future. *arXiv preprint arXiv:2011.04406*, 2020.
- [63] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in neural information processing systems*, pages 8527–8537, 2018.
- [64] Jiangfan Han, Ping Luo, and Xiaogang Wang. Deep self-learning from noisy labels.

- In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5138–5147, 2019.
- [65] Xiaotian Han, Zhimeng Jiang, Hongye Jin, Zirui Liu, Na Zou, Qifan Wang, and Xia Hu. Retiring δ DP: New distribution-level metrics for demographic parity. *Transactions on Machine Learning Research*, 2023.
- [66] Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29:3315–3323, 2016.
- [67] Tatsunori Hashimoto, Megha Srivastava, Hongseok Namkoong, and Percy Liang. Fairness without demographics in repeated loss minimization. In *Proc. of ICML*, 2018.
- [68] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.
- [69] 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.
- [70] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.
- [71] Sara Hooker, Aaron Courville, Gregory Clark, Yann Dauphin, and Andrea Frome.

- What do compressed deep neural networks forget? *arXiv preprint arXiv:1911.05248*, 2019.
- [72] Sara Hooker, Nyalleng Moorosi, Gregory Clark, Samy Bengio, and Emily Denton. Characterising bias in compressed models. *arXiv preprint arXiv:2010.03058*, 2020.
- [73] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.
- [74] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.
- [75] Yen-Chang Hsu, Yilin Shen, Hongxia Jin, and Zsolt Kira. Generalized odin: Detecting out-of-distribution image without learning from out-of-distribution data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10951–10960, 2020.
- [76] Jiayuan Huang, Arthur Gretton, Karsten Borgwardt, Bernhard Schölkopf, and Alex J Smola. Correcting sample selection bias by unlabeled data. In *Advances in neural information processing systems*, pages 601–608, 2007.
- [77] Zhuo Huang, Chao Xue, Bo Han, Jian Yang, and Chen Gong. Universal semi-supervised learning. *Advances in Neural Information Processing Systems*, 34, 2021.
- [78] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. Label propagation

- for deep semi-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5070–5079, 2019.
- [79] Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2, 2021.
- [80] Xu Ji, João F Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9865–9874, 2019.
- [81] Heinrich Jiang, Been Kim, Melody Guan, and Maya Gupta. To trust or not to trust a classifier. *Advances in neural information processing systems*, 31, 2018.
- [82] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning*, pages 2304–2313. PMLR, 2018.
- [83] Wenyu Jiang, Hao Cheng, Mingcai Chen, Chongjun Wang, and Hongxin Wei. Dos: Diverse outlier sampling for out-of-distribution detection, 2023.
- [84] Zhimeng Jiang, Xiaotian Han, Chao Fan, Zirui Liu, Xiao Huang, Na Zou, Ali Mostafavi, and Xia Hu. Topology matters in fair graph learning: a theoretical pilot study, 2023.
- [85] Zhimeng Jiang, Xiaotian Han, Chao Fan, Zirui Liu, Na Zou, Ali Mostafavi, and Xia Hu. Fmp: Toward fair graph message passing against topology bias, 2022.

- [86] Zhimeng Jiang, Xiaotian Han, Chao Fan, Fan Yang, Ali Mostafavi, and Xia Hu. Generalized demographic parity for group fairness. In *International Conference on Learning Representations*, 2022.
- [87] Zhimeng Jiang, Xiaotian Han, Hongye Jin, Guanchu Wang, Na Zou, and Xia Hu. Weight perturbation can help fairness under distribution shift. *arXiv preprint arXiv:2303.03300*, 2023.
- [88] Zhimeng Jiang, Kaixiong Zhou, Zirui Liu, Li Li, Rui Chen, Soo-Hyun Choi, and Xia Hu. An information fusion approach to learning with instance-dependent label noise. In *International Conference on Learning Representations*, 2022.
- [89] Zhimeng Jiang, Kaixiong Zhou, Mi Zhang, Rui Chen, Xia Hu, and Soo-Hyun Choi. Adaptive risk-aware bidding with budget constraint in display advertising. *arXiv preprint arXiv:2212.12533*, 2022.
- [90] Kaggle. Jigsaw Toxicity dataset: Toxic comment classification challenge. <https://www.kaggle.com/c/jigsaw-unintended-bias-in-toxicity-classification>, 2018. Accessed: 2021-11-15.
- [91] David R Karger, Sewoong Oh, and Devavrat Shah. Efficient crowdsourcing for multi-class labeling. In *Proceedings of the ACM SIGMETRICS/international conference on Measurement and modeling of computer systems*, pages 81–92, 2013.
- [92] D.R. Karger, S. Oh, and D. Shah. Iterative learning for reliable crowdsourcing systems. In *Neural Information Processing Systems, NIPS '11*, 2011.

- [93] Falaah Arif Khan, Eleni Manis, and Julia Stoyanovich. Translation tutorial: Fairness and friends. In *Proceedings of the ACM Conference on Fairness, Accountability, and Transparency*, 2021.
- [94] Niki Kilbertus, Adrià Gascón, Matt Kusner, Michael Veale, Krishna Gummadi, and Adrian Weller. Blind justice: Fairness with encrypted sensitive attributes. In *International Conference on Machine Learning*, pages 2630–2639. PMLR, 2018.
- [95] Michael P Kim, Amirata Ghorbani, and James Zou. Multiaccuracy: Black-box post-processing for fairness in classification. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 247–254, 2019.
- [96] Youngdong Kim, Junho Yim, Juseung Yun, and Junmo Kim. Nlnl: Negative learning for noisy labels. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 101–110, 2019.
- [97] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1920–1929, 2019.
- [98] Shuyu Kong, You Li, Jia Wang, Amin Rezaei, and Hai Zhou. Knn-enhanced deep learning against noisy labels. *arXiv preprint arXiv:2012.04224*, 2020.
- [99] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.
- [100] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification

- with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.
- [101] Preethi Lahoti, Alex Beutel, Jilin Chen, Kang Lee, Flavien Prost, Nithum Thain, Xuezhi Wang, and Ed Chi. Fairness without demographics through adversarially reweighted learning. *Proc. of NeurIPS*, 2020.
- [102] Alex Lamy, Ziyuan Zhong, Aditya K Menon, and Nakul Verma. Noise-tolerant fair classification. *Advances in Neural Information Processing Systems*, 32, 2019.
- [103] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [104] 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.
- [105] Kuang-Huei Lee, Xiaodong He, Lei Zhang, and Linjun Yang. Cleannet: Transfer learning for scalable image classifier training with label noise. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5447–5456, 2018.
- [106] Jingling Li, Mozhi Zhang, Keyulu Xu, John Dickerson, and Jimmy Ba. How does a neural network’s architecture impact its robustness to noisy labels? *Advances in Neural Information Processing Systems*, 34, 2021.

- [107] Jingling Li, Mozhi Zhang, Keyulu Xu, John P Dickerson, and Jimmy Ba. Noisy labels can induce good representations. *arXiv preprint arXiv:2012.12896*, 2020.
- [108] Junnan Li, Richard Socher, and Steven C.H. Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. In *International Conference on Learning Representations*, 2020.
- [109] Junnan Li, Yongkang Wong, Qi Zhao, and Mohan S Kankanhalli. Learning to learn from noisy labeled data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5051–5059, 2019.
- [110] Mingchen Li, Mahdi Soltanolkotabi, and Samet Oymak. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. In *International conference on artificial intelligence and statistics*, pages 4313–4324. PMLR, 2020.
- [111] Xuefeng Li, Tongliang Liu, Bo Han, Gang Niu, and Masashi Sugiyama. Provably end-to-end label-noise learning without anchor points. In *International Conference on Machine Learning*, pages 6403–6413. PMLR, 2021.
- [112] Yuncheng Li, Jianchao Yang, Yale Song, Liangliang Cao, Jiebo Luo, and Li-Jia Li. Learning from noisy labels with distillation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1910–1918, 2017.
- [113] Hongyi Ling, Zhimeng Jiang, Youzhi Luo, Shuiwang Ji, and Na Zou. Learning

- fair graph representations via automated data augmentations. In *International Conference on Learning Representations*, 2023.
- [114] Andy T Liu, Shang-Wen Li, and Hung-yi Lee. Tera: Self-supervised learning of transformer encoder representation for speech. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:2351–2366, 2021.
- [115] Evan Z Liu, Behzad Haghgoo, Annie S Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. Just train twice: Improving group robustness without training group information. In *International Conference on Machine Learning*, pages 6781–6792. PMLR, 2021.
- [116] Minghao Liu, Jiaheng Wei, Yang Liu, and James Davis. Do humans and machines have the same eyes? human-machine perceptual differences on image classification. *arXiv preprint arXiv:2304.08733*, 2023.
- [117] Qiang Liu, Jian Peng, and Alexander Ihler. Variational inference for crowdsourcing. In *Proceedings of the 25th International Conference on Neural Information Processing Systems-Volume 1*, pages 692–700, 2012.
- [118] Sheng Liu, Jonathan Niles-Weed, Narges Razavian, and Carlos Fernandez-Granda. Early-learning regularization prevents memorization of noisy labels. *Advances in neural information processing systems*, 33:20331–20342, 2020.
- [119] Tongliang Liu and Dacheng Tao. Classification with noisy labels by importance

- reweighting. *IEEE Transactions on pattern analysis and machine intelligence*, 38(3):447–461, 2015.
- [120] Weiwei Liu, Haobo Wang, Xiaobo Shen, and Ivor W. Tsang. The emerging trends of multi-label learning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(11):7955–7974, 2022.
- [121] Yang Liu. Understanding instance-level label noise: Disparate impacts and treatments. In *International Conference on Machine Learning*, pages 6725–6735. PMLR, 2021.
- [122] Yang Liu. Identifiability of label noise transition matrix. *arXiv preprint arXiv:2202.02016*, 2022.
- [123] Yang Liu and Yiling Chen. Machine-learning aided peer prediction. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pages 63–80, 2017.
- [124] Yang Liu and Hongyi Guo. Peer loss functions: Learning from noisy labels without knowing noise rates. In *International Conference on Machine Learning*, pages 6226–6236. PMLR, 2020.
- [125] Yang Liu and Mingyan Liu. An online learning approach to improving the quality of crowd-sourcing. *ACM SIGMETRICS Performance Evaluation Review*, 43(1):217–230, 2015.
- [126] Yang Liu and Jialu Wang. Can less be more? when increasing-to-balancing label

- noise rates considered beneficial. *Advances in Neural Information Processing Systems*, 34, 2021.
- [127] Yang Liu, Juntao Wang, and Yiling Chen. Surrogate scoring rules. In *Proceedings of the 21st ACM Conference on Economics and Computation*, pages 853–871, 2020.
- [128] Philip M Long and Rocco A Servedio. Random classification noise defeats all convex potential boosters. *Machine learning*, 78(3):287–304, 2010.
- [129] Michael Luca. Reviews, reputation, and revenue: The case of yelp. com. *Com (March 15, 2016)*. *Harvard Business School NOM Unit Working Paper*, (12-016), 2016.
- [130] Michal Lukasik, Srinadh Bhojanapalli, Aditya Menon, and Sanjiv Kumar. Does label smoothing mitigate label noise? In *International Conference on Machine Learning*, pages 6448–6458. PMLR, 2020.
- [131] Huixiang Luo, Hao Cheng, Yuting Gao, Ke Li, Mengdan Zhang, Fanxu Meng, Xiaowei Guo, Feiyue Huang, and Xing Sun. On the consistency training for open-set semi-supervised learning. *arXiv preprint arXiv:2101.08237*, 2021.
- [132] Tianyi Luo, Xingyu Li, Hainan Wang, and Yang Liu. Research replication prediction using weakly supervised learning. In *In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, 2020.
- [133] Naresh Manwani and PS Sastry. Noise tolerance under risk minimization. *IEEE transactions on cybernetics*, 43(3):1146–1151, 2013.

- [134] Natalia Martinez, Martin Bertran, and Guillermo Sapiro. Minimax pareto fairness: A multi objective perspective. In *International Conference on Machine Learning*, pages 6755–6764. PMLR, 2020.
- [135] Tyler H McCormick, Zehang Richard Li, Clara Calvert, Amelia C Crampin, Kathleen Kahn, and Samuel J Clark. Probabilistic cause-of-death assignment using verbal autopsies. *Journal of the American Statistical Association*, 111(515):1036–1049, 2016.
- [136] Yu Meng, Jiaming Shen, Chao Zhang, and Jiawei Han. Weakly-supervised neural text classification. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 983–992, 2018.
- [137] Takeru Miyato, Andrew M Dai, and Ian Goodfellow. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*, 2016.
- [138] 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.
- [139] Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. In *Advances in neural information processing systems*, pages 1196–1204, 2013.
- [140] Duc Tam Nguyen, Chaithanya Kumar Mummadi, Thi Phuong Nhung Ngo, Thi

- Hoai Phuong Nguyen, Laura Beggel, and Thomas Brox. Self: Learning to filter noisy labels with self-ensembling. *arXiv preprint arXiv:1910.01842*, 2019.
- [141] Curtis Northcutt, Lu Jiang, and Isaac Chuang. Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, 70:1373–1411, 2021.
- [142] Curtis G Northcutt, Anish Athalye, and Jonas Mueller. Pervasive label errors in test sets destabilize machine learning benchmarks. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.
- [143] Curtis G Northcutt, Lu Jiang, and Isaac L Chuang. Confident learning: Estimating uncertainty in dataset labels. *arXiv preprint arXiv:1911.00068*, 2019.
- [144] Curtis G Northcutt, Tailin Wu, and Isaac L Chuang. Learning with confident examples: Rank pruning for robust classification with noisy labels. *UAI*, 2017.
- [145] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 271–279, 2016.
- [146] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction

- approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1944–1952, 2017.
- [147] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.
- [148] Flavien Prost, Pranjal Awasthi, Nick Blumm, Aditee Kumthekar, Trevor Potter, Li Wei, Xuezhi Wang, Ed H Chi, Jilin Chen, and Alex Beutel. Measuring model fairness under noisy covariates: A theoretical perspective. In *Proc. of AIES*, 2021.
- [149] Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. In *Advances in Neural Information Processing Systems*, volume 33, pages 19920–19930, 2020.
- [150] Reilly Raab and Yang Liu. Unintended selection: Persistent qualification rate disparities and interventions. *Advances in Neural Information Processing Systems*, 34, 2021.
- [151] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.
- [152] Antti Rasmus, Harri Valpola, Mikko Honkala, Mathias Berglund, and Tapani Raiko.

- Semi-supervised learning with ladder networks. *arXiv preprint arXiv:1507.02672*, 2015.
- [153] Henry Reeve and Ata Kabán. Fast rates for a knn classifier robust to unknown asymmetric label noise. In *International Conference on Machine Learning*, pages 5401–5409. PMLR, 2019.
- [154] Devendra Singh Sachan, Manzil Zaheer, and Ruslan Salakhutdinov. Revisiting lstm networks for semi-supervised text classification via mixed objective function. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6940–6948, 2019.
- [155] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. *Advances in neural information processing systems*, 29:1163–1171, 2016.
- [156] Clayton Scott. A rate of convergence for mixture proportion estimation, with application to learning from noisy labels. In *AISTATS*, 2015.
- [157] Jun Shu, Qian Zhao, Keyu Chen, Zongben Xu, and Deyu Meng. Learning adaptive loss for robust learning with noisy labels. *arXiv preprint arXiv:2002.06482*, 2020.
- [158] Grigori Sidorov, Alexander Gelbukh, Helena Gómez-Adorno, and David Pinto. Soft similarity and soft cosine measure: Similarity of features in vector space model. *Computacion y Sistemas*, pages 491–504, January 2014.
- [159] 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.
- [160] Amos Storkey. When training and test sets are different: characterizing learning transfer. *Dataset shift in machine learning*, pages 3–28, 2009.
- [161] Zeyu Tang, Yatong Chen, Yang Liu, and Kun Zhang. Tier balancing: Towards dynamic fairness over underlying causal factors. *arXiv preprint arXiv:2301.08987*, 2023.
- [162] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *arXiv preprint arXiv:1703.01780*, 2017.
- [163] Brendan Van Rooyen and Robert C Williamson. A theory of learning with corrupted labels. *J. Mach. Learn. Res.*, 18(1):8501–8550, 2017.
- [164] Andreas Veit, Neil Alldrin, Gal Chechik, Ivan Krasin, Abhinav Gupta, and Serge Belongie. Learning from noisy large-scale datasets with minimal supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 839–847, 2017.
- [165] Jorge R Vergara and Pablo A Estévez. A review of feature selection methods based on mutual information. *Neural computing and applications*, 24(1):175–186, 2014.
- [166] Haobo Wang, Weiwei Liu, Yang Zhao, Chen Zhang, Tianlei Hu, and Gang Chen.

- Discriminative and correlative partial multi-label learning. In Sarit Kraus, editor, *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 3691–3697. ijcai.org, 2019.
- [167] Haobo Wang, Ruixuan Xiao, Yiwen Dong, Lei Feng, and Junbo Zhao. Promix: combating label noise via maximizing clean sample utility. *arXiv preprint arXiv:2207.10276*, 2022.
- [168] Haobo Wang, Ruixuan Xiao, Yixuan Li, Lei Feng, Gang Niu, Gang Chen, and Junbo Zhao. PiCO: Contrastive label disambiguation for partial label learning. In *International Conference on Learning Representations*, 2022.
- [169] Jialu Wang, Xinyue Gabby Liu, Zonglin Di, Yang Liu, and Xin Eric Wang. T2iat: Measuring valence and stereotypical biases in text-to-image generation. *arXiv preprint arXiv:2306.00905*, 2023.
- [170] Jialu Wang, Yang Liu, and Caleb Levy. Fair classification with group-dependent label noise. In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 526–536, 2021.
- [171] Jialu Wang, Yang Liu, and Xin Eric Wang. Are gender-neutral queries really gender-neutral? mitigating gender bias in image search. *arXiv preprint arXiv:2109.05433*, 2021.
- [172] Jialu Wang, Yang Liu, and Xin Eric Wang. Assessing multilingual fairness in

- pre-trained multimodal representations. In *the Findings of ACL 2022*, Dublin, Ireland, May 2022. Association for Computational Linguistics.
- [173] Jialu Wang, Xin Eric Wang, and Yang Liu. Understanding instance-level impact of fairness constraints. In *International Conference on Machine Learning*, pages 23114–23130. PMLR, 2022.
- [174] Jingkang Wang, Hongyi Guo, Zhaowei Zhu, and Yang Liu. Policy learning using weak supervision. *Advances in Neural Information Processing Systems*, 34, 2021.
- [175] Qizhou Wang, Bo Han, Tongliang Liu, Gang Niu, Jian Yang, and Chen Gong. Tackling instance-dependent label noise via a universal probabilistic model. *arXiv preprint arXiv:2101.05467*, 2021.
- [176] Xiang Wang, Shiwei Zhang, Zhiwu Qing, Yuanjie Shao, Changxin Gao, and Nong Sang. Self-supervised learning for semi-supervised temporal action proposal. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1905–1914, 2021.
- [177] Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric cross entropy for robust learning with noisy labels. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 322–330, 2019.
- [178] Zhuowei Wang, Jing Jiang, Bo Han, Lei Feng, Bo An, Gang Niu, and Guodong Long. Seminll: A framework of noisy-label learning by semi-supervised learning. *arXiv preprint arXiv:2012.00925*, 2020.

- [179] Hongxin Wei, Lei Feng, Xiangyu Chen, and Bo An. Combating noisy labels by agreement: A joint training method with co-regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13726–13735, 2020.
- [180] Hongxin Wei, Lue Tao, Renchunzi Xie, and Bo An. Open-set label noise can improve robustness against inherent label noise. *Advances in Neural Information Processing Systems*, 34, 2021.
- [181] Hongxin Wei, Lue Tao, Renchunzi Xie, Lei Feng, and Bo An. Open-sampling: Exploring out-of-distribution data for re-balancing long-tailed datasets. In *International Conference on Machine Learning (ICML)*. PMLR, 2022.
- [182] Hongxin Wei, Renchunzi Xie, Hao Cheng, Lei Feng, Bo An, and Yixuan Li. Mitigating neural network overconfidence with logit normalization. *arXiv preprint arXiv:2205.09310*, 2022.
- [183] Hongxin Wei, Renchunzi Xie, Lei Feng, Bo Han, and Bo An. Deep learning from multiple noisy annotators as a union. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.
- [184] Hongxin Wei, Huiping Zhuang, Renchunzi Xie, Lei Feng, Gang Niu, Bo An, and Yixuan Li. Mitigating memorization of noisy labels by clipping the model prediction. In *International Conference on Machine Learning (ICML)*. PMLR, 2023.

- [185] Jiaheng Wei, Hangyu Liu, Tongliang Liu, Gang Niu, Masashi Sugiyama, and Yang Liu. To smooth or not? when label smoothing meets noisy labels, 2021.
- [186] Jiaheng Wei and Yang Liu. When optimizing f -divergence is robust with label noise. *arXiv preprint arXiv:2011.03687*, 2020.
- [187] Jiaheng Wei, Harikrishna Narasimhan, Ehsan Amid, Wen-Sheng Chu, Yang Liu, and Abhishek Kumar. Distributionally robust post-hoc classifiers under prior shifts. In *The Eleventh International Conference on Learning Representations*, 2023.
- [188] Jiaheng Wei, Zhaowei Zhu, Hao Cheng, Tongliang Liu, Gang Niu, and Yang Liu. Learning with noisy labels revisited: A study using real-world human annotations. In *International Conference on Learning Representations*, 2022.
- [189] Jiaheng Wei, Zhaowei Zhu, Tianyi Luo, Ehsan Amid, Abhishek Kumar, and Yang Liu. To aggregate or not? learning with separate noisy labels. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023.
- [190] Jiaheng Wei, Zhaowei Zhu, Gang Niu, Tongliang Liu, Sijia Liu, Masashi Sugiyama, and Yang Liu. Fairness improves learning from noisily labeled long-tailed data. *arXiv preprint arXiv:2303.12291*, 2023.
- [191] Susan Wojcicki. Letter from Susan: Our 2021 Priorities. <https://blog.youtube/inside-youtube/letter-from-susan-our-2021-priorities>, 2021. [Online; accessed 15-Sep-2022].

- [192] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.
- [193] Blake Woodworth, Suriya Gunasekar, Mesrob I Ohannessian, and Nathan Srebro. Learning non-discriminatory predictors. In *Conference on Learning Theory*, pages 1920–1953. PMLR, 2017.
- [194] Jimmy Wu, Yatong Chen, and Yang Liu. Metric-fair classifier derandomization. In *International Conference on Machine Learning*, pages 23999–24016. PMLR, 2022.
- [195] Songhua Wu, Mingming Gong, Bo Han, Yang Liu, and Tongliang Liu. Fair classification with instance-dependent label noise. In *Conference on Causal Learning and Reasoning*, pages 927–943. PMLR, 2022.
- [196] Xiaobo Xia, Tongliang Liu, Bo Han, Chen Gong, Nannan Wang, Zongyuan Ge, and Yi Chang. Robust early-learning: Hindering the memorization of noisy labels. In *International Conference on Learning Representations*, 2021.
- [197] Xiaobo Xia, Tongliang Liu, Bo Han, Nannan Wang, Jiankang Deng, Jiatong Li, and Yinian Mao. Extended T: Learning with mixed closed-set and open-set noisy labels. *arXiv preprint arXiv:2012.00932*, 2020.
- [198] Xiaobo Xia, Tongliang Liu, Bo Han, Nannan Wang, Mingming Gong, Haifeng Liu, Gang Niu, Dacheng Tao, and Masashi Sugiyama. Part-dependent label noise: Towards instance-dependent label noise. In *Advances in Neural Information Processing Systems*, volume 33, pages 7597–7610, 2020.

- [199] Xiaobo Xia, Tongliang Liu, Nannan Wang, Bo Han, Chen Gong, Gang Niu, and Masashi Sugiyama. Are anchor points really indispensable in label-noise learning? In *Advances in Neural Information Processing Systems*, pages 6838–6849, 2019.
- [200] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2691–2699, 2015.
- [201] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Unsupervised data augmentation. *arXiv preprint arXiv:1904.12848*, 2019.
- [202] Renchunzi Xie, Hongxin Wei, Yuzhou Cao, Lei Feng, and Bo An. On the importance of feature separability in predicting out-of-distribution error, 2023.
- [203] Renchunzi Xie, Hongxin Wei, Lei Feng, and Bo An. Gearnet: Stepwise dual learning for weakly supervised domain adaptation. *AAAI Conference on Artificial Intelligence*, 2022.
- [204] Yi Xu, Lei Shang, Jinxing Ye, Qi Qian, Yu-Feng Li, Baigui Sun, Hao Li, and Rong Jin. Dash: Semi-supervised learning with dynamic thresholding. In *International Conference on Machine Learning*, pages 11525–11536. PMLR, 2021.
- [205] Yilun Xu, Peng Cao, Yuqing Kong, and Yizhou Wang. L_{dmi}: A novel information-theoretic loss function for training deep nets robust to label noise. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

- [206] Shen Yan, Hsien-te Kao, and Emilio Ferrara. Fair class balancing: Enhancing model fairness without observing sensitive attributes. In *Proc. of CIKM*, 2020.
- [207] Diyi Yang, Jiaao Chen, Zichao Yang, Dan Jurafsky, and Eduard Hovy. Let’s make your request more persuasive: Modeling persuasive strategies via semi-supervised neural nets on crowdfunding platforms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3620–3630, 2019.
- [208] Zichao Yang, Zhiting Hu, Ruslan Salakhutdinov, and Taylor Berg-Kirkpatrick. Improved variational autoencoders for text modeling using dilated convolutions. In *International conference on machine learning*, pages 3881–3890. PMLR, 2017.
- [209] Quanming Yao, Hansi Yang, Bo Han, Gang Niu, and James T Kwok. Searching to exploit memorization effect in learning with noisy labels. In *Proceedings of the 37th International Conference on Machine Learning, ICML ’20*, 2020.
- [210] Yu Yao, Tongliang Liu, Bo Han, Mingming Gong, Jiankang Deng, Gang Niu, and Masashi Sugiyama. Dual T: Reducing estimation error for transition matrix in label-noise learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 7260–7271, 2020.
- [211] Yelp. Yelp dataset challenge, 2015.
- [212] Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama. How does disagreement help generalization against label corruption? In *Proceedings*

- of the 36th International Conference on Machine Learning, volume 97, pages 7164–7173. PMLR, 09–15 Jun 2019.
- [213] Daochen Zha, Zaid Pervaiz Bhat, Kwei-Herng Lai, Fan Yang, Zhimeng Jiang, Shaochen Zhong, and Xia Hu. Data-centric artificial intelligence: A survey. *arXiv preprint arXiv:2303.10158*, 2023.
- [214] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.
- [215] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.
- [216] Jing Zhang, Victor S Sheng, Tao Li, and Xindong Wu. Improving crowdsourced label quality using noise correction. *IEEE transactions on neural networks and learning systems*, 29(5):1675–1688, 2017.
- [217] Mingyuan Zhang, Jane Lee, and Shivani Agarwal. Learning from noisy labels with no change to the training process. In *International Conference on Machine Learning*, pages 12468–12478. PMLR, 2021.
- [218] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama,

- and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.
- [219] Xuchao Zhang, Xian Wu, Fanglan Chen, Liang Zhao, and Chang-Tien Lu. Self-paced robust learning for leveraging clean labels in noisy data. In *AAAI*, pages 6853–6860, 2020.
- [220] Yuchen Zhang, Xi Chen, Dengyong Zhou, and Michael I Jordan. Spectral methods meet em: A provably optimal algorithm for crowdsourcing. *Advances in neural information processing systems*, 27:1260–1268, 2014.
- [221] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. In *Advances in neural information processing systems*, pages 8778–8788, 2018.
- [222] Ziqi Zhang, Yuexiang Li, Hongxin Wei, Kai Ma, Tao Xu, and Yefeng Zheng. Alleviating noisy-label effects in image classification via probability transition matrix. *arXiv preprint arXiv:2110.08866*, 2021.
- [223] Zizhao Zhang, Han Zhang, Serkan O Arik, Honglak Lee, and Tomas Pfister. Distilling effective supervision from severe label noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9294–9303, 2020.
- [224] Zhi-Hua Zhou. A brief introduction to weakly supervised learning. *National science review*, 5(1):44–53, 2018.

- [225] 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.
- [226] Zhaowei Zhu, Zihao Dong, and Yang Liu. Detecting corrupted labels without training a model to predict. In *International Conference on Machine Learning (ICML)*, 2022.
- [227] Zhaowei Zhu, Tongliang Liu, and Yang Liu. A second-order approach to learning with instance-dependent label noise. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021.
- [228] Zhaowei Zhu, Tongliang Liu, and Yang Liu. A second-order approach to learning with instance-dependent label noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10113–10123, 2021.
- [229] Zhaowei Zhu, Tianyi Luo, and Yang Liu. The rich get richer: Disparate impact of semi-supervised learning. In *International Conference on Learning Representations*, 2022.
- [230] Zhaowei Zhu, Yiwen Song, and Yang Liu. Clusterability as an alternative to anchor points when learning with noisy labels. In *Proceedings of the 38th International Conference on Machine Learning, ICML '21*, 2021.
- [231] Zhaowei Zhu, Jialu Wang, and Yang Liu. Beyond images: Label noise transition

- matrix estimation for tasks with lower-quality features. In *International Conference on Machine Learning (ICML)*. PMLR, 17–23 Jul 2022.
- [232] Zhaowei Zhu, Yuanshun Yao, Jiankai Sun, Hang Li, and Yang Liu. Weak proxies are sufficient and preferable for fairness with missing sensitive attributes. In *International Conference on Machine Learning (ICML)*, 2023.
- [233] Zhaowei Zhu, Jingxuan Zhu, Ji Liu, and Yang Liu. Federated bandit: A gossiping approach. In *Abstract Proceedings of the 2021 ACM SIGMETRICS/International Conference on Measurement and Modeling of Computer Systems*, pages 3–4, 2021.
- [234] Piotr Zielinski, Shankar Krishnan, and Satrajit Chatterjee. Explaining memorization and generalization: A large-scale study with coherent gradients. *arXiv preprint arXiv:2003.07422*, 2020.

Appendix A

More Details for Charter 2

Appendix A provides more theoretical details and experiment settings related to HOC (Section 2.2) and its extension (Section 2.3), which is organized as follows.

- Section A.1 presents the detailed examples and derivations of consensus equations.
- Section A.2 includes proofs and other details about our theoretical results. Particularly,
 - Section A.2.1 proves the uniqueness of \mathbf{T} .
 - Section A.2.2 justifies the feasibility of assumption $|E_3^*| = \Theta(N)$
 - Section A.2.3 shows the proof for Lemma 1
 - Section A.2.4 shows the proof for Theorem 2.
- Section A.3 presents more discussions, e.g., the soft 2-NN label clusterability, more details on local $\mathbf{T}(X)$, and the feasibility of our Assumption 1 & 2 to guarantee the uniqueness of \mathbf{T} .

- Section A.4 shows more experimental settings and results related to HOC.
- Section A.5.1 numerates some popular f -divergence functions and the corresponding optimal variational-conjugate pair (g^*, f^*) .
- Section A.5 shows the theoretical parts of HOC extension, where
 - Section A.5.2 shows the order-preserving property of using total variation.
 - Section A.5.3 shows the order-preserving property of using KL divergence.
- Section A.6 shows the more discussions for HOC extension, where
 - Section A.6.1 explains why we build our method on HOC.
 - Section A.6.2 discuss an interesting observation shown in Table 2.5 that high-noise settings may have lower errors.

A.1 Derivation of Consensus Equations

For the first-order consensus, we have

$$\mathbb{P}(\tilde{Y}_1 = j_1) = \sum_{i \in [K]} \mathbb{P}(\tilde{Y}_1 = j_1 | Y_1 = i) \mathbb{P}(Y_1 = i).$$

For the second-order consensus, we have

$$\begin{aligned}
& \mathbb{P}(\tilde{Y}_1 = j_1, \tilde{Y}_2 = j_2) \\
&= \sum_{i \in [K]} \mathbb{P}(\tilde{Y}_1 = j_1, \tilde{Y}_2 = j_2 | Y_1 = i, Y_2 = i) \mathbb{P}(Y_1 = Y_2 = i) \\
&\stackrel{(a)}{=} \sum_{i \in [K]} \mathbb{P}(\tilde{Y}_1 = j_1, \tilde{Y}_2 = j_2 | Y_1 = i, Y_2 = i) \cdot \mathbb{P}(Y_1 = i) \\
&\stackrel{(b)}{=} \sum_{i \in [K]} \mathbb{P}(\tilde{Y}_1 = j_1 | Y_1 = i) \cdot \mathbb{P}(\tilde{Y}_2 = j_2 | Y_2 = i) \cdot \mathbb{P}(Y_1 = i),
\end{aligned}$$

where equality (a) holds due to the 2-NN label clusterability, i.e., $Y_1 = Y_2 (= Y_3)$ w.p. 1, and equality (b) holds due to the conditional independency between \tilde{Y}_1 and \tilde{Y}_2 given their clean labels.

For the third-order consensus, we have

$$\begin{aligned}
& \mathbb{P}(\tilde{Y}_1 = j_1, \tilde{Y}_2 = j_2, \tilde{Y}_3 = j_3) \\
&= \sum_{i \in [K]} \mathbb{P}(\tilde{Y}_1 = j_1, \tilde{Y}_2 = j_2, \tilde{Y}_3 = j_3 | Y_1 = i, Y_2 = i, Y_3 = i) \mathbb{P}(Y_1 = Y_2 = Y_3 = i) \\
&\stackrel{(a)}{=} \sum_{i \in [K]} \mathbb{P}(\tilde{Y}_1 = j_1, \tilde{Y}_2 = j_2, \tilde{Y}_3 = j_3 | Y_1 = i, Y_2 = i, Y_3 = i) \mathbb{P}(Y_1 = i) \\
&\stackrel{(b)}{=} \sum_{i \in [K]} \mathbb{P}(\tilde{Y}_1 = j_1 | Y_1 = i) \mathbb{P}(\tilde{Y}_2 = j_2 | Y_2 = i) \mathbb{P}(\tilde{Y}_3 = j_3 | Y_3 = i) \mathbb{P}(Y_1 = i).
\end{aligned}$$

where equality (a) holds due to the 3-NN label clusterability, i.e., $Y_1 = Y_2 = Y_3$ w.p. 1, and equality (b) holds due to the conditional independency between \tilde{Y}_1 , \tilde{Y}_2 and \tilde{Y}_3 given their clean labels.

With the above analyses, there are 2 first-order equations,

$$\mathbb{P}(\tilde{Y}_1 = 1) = p_1(1 - e_1) + (1 - p_1)e_2,$$

$$\mathbb{P}(\tilde{Y}_1 = 2) = p_1e_1 + (1 - p_1)(1 - e_2).$$

There are 4 second-order equations for different combinations of \tilde{Y}_1, \tilde{Y}_2 , e.g.,

$$\mathbb{P}(\tilde{Y}_1 = 1, \tilde{Y}_2 = 1) = p_1(1 - e_1)^2 + (1 - p_1)e_2^2,$$

$$\mathbb{P}(\tilde{Y}_1 = 1, \tilde{Y}_2 = 2) = p_1(1 - e_1)e_1 + (1 - p_1)e_2(1 - e_2),$$

$$\mathbb{P}(\tilde{Y}_1 = 2, \tilde{Y}_2 = 1) = p_1(1 - e_1)e_1 + (1 - p_1)e_2(1 - e_2),$$

$$\mathbb{P}(\tilde{Y}_1 = 1, \tilde{Y}_2 = 1) = p_1e_1^2 + (1 - p_1)(1 - e_2)^2.$$

There are 8 third-order equations for different combinations of $\tilde{Y}_1, \tilde{Y}_2, \tilde{Y}_3$, e.g.,

$$\mathbb{P}(\tilde{Y}_1 = 1, \tilde{Y}_2 = 1, \tilde{Y}_3 = 1) = p_1(1 - e_1)^3 + (1 - p_1)e_2^3,$$

$$\mathbb{P}(\tilde{Y}_1 = 1, \tilde{Y}_2 = 1, \tilde{Y}_3 = 2) = p_1(1 - e_1)^2e_1 + (1 - p_1)e_2^2(1 - e_2),$$

$$\mathbb{P}(\tilde{Y}_1 = 1, \tilde{Y}_2 = 2, \tilde{Y}_3 = 1) = p_1(1 - e_1)^2e_1 + (1 - p_1)e_2^2(1 - e_2),$$

$$\mathbb{P}(\tilde{Y}_1 = 1, \tilde{Y}_2 = 2, \tilde{Y}_3 = 2) = p_1(1 - e_1)e_1^2 + (1 - p_1)e_2(1 - e_2)^2,$$

$$\mathbb{P}(\tilde{Y}_1 = 2, \tilde{Y}_2 = 1, \tilde{Y}_3 = 1) = p_1(1 - e_1)^2e_1 + (1 - p_1)e_2^2(1 - e_2),$$

$$\mathbb{P}(\tilde{Y}_1 = 2, \tilde{Y}_2 = 1, \tilde{Y}_3 = 2) = p_1(1 - e_1)e_1^2 + (1 - p_1)e_2(1 - e_2)^2,$$

$$\mathbb{P}(\tilde{Y}_1 = 2, \tilde{Y}_2 = 2, \tilde{Y}_3 = 1) = p_1(1 - e_1)e_1^2 + (1 - p_1)e_2(1 - e_2)^2,$$

$$\mathbb{P}(\tilde{Y}_1 = 2, \tilde{Y}_2 = 2, \tilde{Y}_3 = 2) = p_1e_1^3 + (1 - p_1)(1 - e_2)^3.$$

For a general K -class classification problem, we show one first-order consensus

below:

$$\begin{aligned} \mathbf{e}_j^\top \mathbf{c}^{[1]} &= \mathbb{P}(\tilde{Y}_1 = j) \\ &= \sum_{i \in [K]} \mathbb{P}(\tilde{Y}_1 = j | Y_1 = i) \mathbb{P}(Y_1 = i) \\ &= \sum_{i \in [K]} T_{ij} \cdot p_i = \mathbf{e}_j^\top \mathbf{T}^\top \mathbf{p}. \end{aligned}$$

The second-order consensus follows the example below:

$$\begin{aligned}
\mathbf{e}_j^\top \mathbf{c}_r^{[2]} &= \mathbb{P}(\tilde{Y}_1 = j, \tilde{Y}_2 = (j+r)_K) \\
&\stackrel{(a)}{=} \sum_{i \in [K]} \mathbb{P}(\tilde{Y}_1 = j | Y_1 = i) \mathbb{P}(\tilde{Y}_2 = (j+r)_K | Y_2 = i) \mathbb{P}(Y_1 = i) \\
&= \sum_{i \in [K]} T_{i,j} \cdot T_{i,(j+r)_K} \cdot p_i \stackrel{(b)}{=} \mathbf{e}_j^\top (\mathbf{T} \circ \mathbf{T}_r)^\top \mathbf{p},
\end{aligned}$$

where equality (a) holds again due to the 2-NN label clusterability the conditional independency (similar to binary cases), and equality (b) holds due to $\mathbf{T}_r[i, j] = T_{i,(j+r)_K}$.

We also show one third-order consensus below:

$$\begin{aligned}
\mathbf{e}_j^\top \mathbf{c}_r^{[3]} &= \mathbb{P}(\tilde{Y}_1 = j, \tilde{Y}_2 = (j+r)_K, \tilde{Y}_3 = (j+s)_K) \\
&\stackrel{(a)}{=} \sum_{i \in [K]} \mathbb{P}(\tilde{Y}_1 = j | Y_1 = i) \mathbb{P}(\tilde{Y}_2 = (j+r)_K | Y_2 = i) \mathbb{P}(\tilde{Y}_3 = (j+s)_K | Y_3 = i) \mathbb{P}(Y_1 = i) \\
&= \sum_{i \in [K]} T_{i,j} \cdot T_{i,(j+r)_K} \cdot T_{i,(j+s)_K} \cdot p_i \stackrel{(b)}{=} \mathbf{e}_j^\top (\mathbf{T} \circ \mathbf{T}_r \circ \mathbf{T}_s)^\top \mathbf{p},
\end{aligned}$$

where equality (a) holds again due to the 3-NN label clusterability the conditional independency (similar to binary cases), and equality (b) holds due to $\mathbf{T}_r[i, j] = T_{i,(j+r)_K}$, $\mathbf{T}_s[i, j] = T_{i,(j+s)_K}$.

A.2 Theoretical Guarantees for HOC

A.2.1 Uniqueness of \mathbf{T}

We need to prove the following equations have a unique solution when \mathbf{T} is non-singular and informative.

Consensus Equations

- First-order (K equations):

$$\mathbf{c}^{[1]} := \mathbf{T}^\top \mathbf{p},$$

- Second-order (K^2 equations):

$$\mathbf{c}_r^{[2]} := (\mathbf{T} \circ \mathbf{T}_r)^\top \mathbf{p}, \quad r \in [K],$$

- Third-order (K^3 equations):

$$\mathbf{c}_{r,s}^{[3]} := (\mathbf{T} \circ \mathbf{T}_r \circ \mathbf{T}_s)^\top \mathbf{p}, \quad r, s \in [K].$$

Firstly, we need the following Lemma for the Hadamard product of matrices:

Lemma 9. [73] *For column vectors \mathbf{x} and \mathbf{y} , and corresponding diagonal matrices $\mathbf{D}_\mathbf{x}$ and $\mathbf{D}_\mathbf{y}$ with these vectors as their main diagonals, the following identity holds:*

$$\mathbf{x}^*(\mathbf{A} \circ \mathbf{B})\mathbf{y} = \text{tr} \left(\mathbf{D}_\mathbf{x}^* \mathbf{A} \mathbf{D}_\mathbf{y} \mathbf{B}^\top \right),$$

where \mathbf{x}^* denotes the conjugate transpose of \mathbf{x} .

The following proof focuses on the second and third-order consensus. It is worth noting that, although the first-order consensus is not necessary for the derivation of the unique solution, it still helps improve the stability of solving for \mathbf{T} and \mathbf{p} numerically.

Step I: Transform the second-order equations. Denoted by $\mathbf{T}_r = \mathbf{T} \mathbf{S}_r$, where \mathbf{S}_r permutes particular columns of \mathbf{T} . Let \mathbf{e}_i be the column vector with only the i -th

element being 1 and 0 otherwise. With Lemma 9, the second-order consensus can be transformed as

$$\mathbf{e}_i^\top \mathbf{c}_r^{[2]} = \mathbf{e}_i^\top (\mathbf{T} \circ \mathbf{T}_r)^\top \mathbf{p} = \text{tr} \left(\mathbf{D}_{\mathbf{e}_i} \mathbf{T}^\top \mathbf{D}_p \mathbf{T} \mathbf{S}_r \right)$$

Then the $(i, (i+r)_K)$ -th element of matrix $\mathbf{T}^\top \mathbf{D}_p \mathbf{T}$ is

$$(\mathbf{T}^\top \mathbf{D}_p \mathbf{T})[i, (i+r)_K] = \mathbf{e}_i^\top \mathbf{c}_r^{[2]}.$$

With a fixed $\mathbf{e}_i^\top \mathbf{c}_r^{[2]}, \forall i, r \in [K]$, denote by

$$\mathbf{T}^\top \mathbf{D}_p \mathbf{T} = \mathbf{T}_\dagger, \tag{A.1}$$

where $\mathbf{T}_\dagger[i, (i+r)_K] = \mathbf{e}_i^\top \mathbf{c}_r^{[2]}$. Note \mathbf{T}_\dagger is fixed given $\mathbf{c}_r^{[2]}, \forall r \in [K]$.

Step II: Transform the third-order equations. Following the idea in Step I, we can also transform the third-order equations. First, notice that

$$\mathbf{e}_i^\top \mathbf{c}_{r,s}^{[3]} = \mathbf{e}_i^\top [(\mathbf{T} \circ \mathbf{T}_s) \circ \mathbf{T}_r]^\top \mathbf{p} = \text{tr} \left(\mathbf{D}_{\mathbf{e}_i} (\mathbf{T} \circ \mathbf{T}_s)^\top \mathbf{D}_p \mathbf{T} \mathbf{S}_r \right).$$

Then the $(i, (i+r)_K)$ -th element of matrix $(\mathbf{T} \circ \mathbf{T}_s)^\top \mathbf{D}_p \mathbf{T}$ is

$$((\mathbf{T} \circ \mathbf{T}_s)^\top \mathbf{D}_p \mathbf{T})[i, (i+r)_K] = \mathbf{e}_i^\top \mathbf{c}_{r,s}^{[3]}.$$

With a fixed $\mathbf{e}_i^\top \mathbf{c}_{r,s}^{[3]}, \forall i, r \in [K]$, denote by

$$(\mathbf{T} \circ \mathbf{T}_s)^\top \mathbf{D}_p \mathbf{T} = \mathbf{T}_{\dagger,s}^\top \Rightarrow \mathbf{T}^\top \mathbf{D}_p (\mathbf{T} \circ \mathbf{T}_s) = \mathbf{T}_{\dagger,s}^\top, \tag{A.2}$$

where $\mathbf{T}_{\dagger,s}[i, (i+r)_K] = \mathbf{e}_i^\top \mathbf{c}_{r,s}^{[3]}$. According to Eqn. (A.1), we have

$$\mathbf{T}^\top \mathbf{D}_p (\mathbf{T} \circ \mathbf{T}_s) = \mathbf{T}^\top \mathbf{D}_p \mathbf{T} \mathbf{T}^{-1} (\mathbf{T} \circ \mathbf{T}_s) = \mathbf{T}_\dagger \mathbf{T}^{-1} (\mathbf{T} \circ \mathbf{T}_s) = \mathbf{T}_{\dagger,s}^\top.$$

Thus

$$(\mathbf{T} \circ \mathbf{T}_s) = \mathbf{T} \mathbf{T}_\dagger^{-1} \mathbf{T}_{\dagger,s}^\top, \forall s \in [K]. \tag{A.3}$$

Step III: From matrices to vectors With Step I and Step II, we could transform the equations formulated by the second and the third-order consensus to a particular system of multivariate quadratic equations of \mathbf{T} in Eqn. (A.3). Generally, these equations could have up to 2^{K^2} solutions introduced by different combinations of each element in \mathbf{T} . To prove the uniqueness of \mathbf{T} , we need to exploit the structure of the equations in (A.3).

For a clear representation of the structure of equations and solutions, we first consider one subset of the equations in (A.3). Specifically, let $s = 0$ we have

$$(\mathbf{T} \circ \mathbf{T}) = \mathbf{T}\mathbf{T}_{\dagger}^{-1}\mathbf{T}_{\dagger}^{\top}. \quad (\text{A.4})$$

Then we need to study the number of feasible \mathbf{T} satisfying Eqn. (A.4). Denote by $\mathbf{A} = \mathbf{T}_{\dagger}(\mathbf{T}_{\dagger}^{-1})^{\top}$. Then each row of \mathbf{T} , denoted by \mathbf{u}^{\top} , is a solution to the equation

$$\mathbf{A}\mathbf{u} = \mathbf{D}_{\mathbf{u}}\mathbf{u} \quad (\text{a.k.a. } \mathbf{A}\mathbf{u} = \mathbf{u} \circ \mathbf{u}). \quad (\text{A.5})$$

Till now, in Step III, we split the matrix \mathbf{T} to several vectors \mathbf{u} , and transform our target from finding a matrix solution \mathbf{T} for (A.3) to a set of vector solutions \mathbf{u} for (A.5).

Assume there are M feasible \mathbf{u} vectors. We collect all the possible \mathbf{u} and define $\mathbf{U} := [\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_M], \mathbf{u}_i \neq \mathbf{u}_{i'}, \forall i, i' \in [M]$. If $M = K$, we know there exists at most $K!$ different \mathbf{T} (considering all the possible permutations of \mathbf{u}) that Eqn. (A.4) holds. Further, by considering an informative \mathbf{T} as Assumption 1, we can identify a particular permutation. Therefore, if $M = K$ and \mathbf{T} is informative, we know there exists and only exists one unique \mathbf{T} that Eqn. (A.4) holds.

Step IV: Constructing the M -th vector Supposing $M > K$, we have

$$\mathbf{A}\mathbf{U} = \mathbf{A}[\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_K, \dots, \mathbf{u}_M] = [\mathbf{D}_{\mathbf{u}_1}\mathbf{u}_1, \mathbf{D}_{\mathbf{u}_2}\mathbf{u}_2, \dots, \mathbf{D}_{\mathbf{u}_K}\mathbf{u}_K, \dots, \mathbf{D}_{\mathbf{u}_M}\mathbf{u}_M].$$

With a non-singular \mathbf{T} (Assumption 1), without loss of generality, we will assume the first K columns are full-rank. Then \mathbf{u}_M must be a linear combination of the first K columns, i.e., $\mathbf{u}_M = \sum_{i \in [K]} \lambda_i \mathbf{u}_i = \mathbf{U}\boldsymbol{\lambda}_0$, where $\boldsymbol{\lambda}_0 = [\lambda_1, \lambda_2, \dots, \lambda_K, 0, \dots, 0]$. According to the equation $\mathbf{A}\mathbf{u} = \mathbf{D}_{\mathbf{u}}\mathbf{u} = \mathbf{u} \circ \mathbf{u}$, we have

$$\mathbf{A}\mathbf{u}_M = \mathbf{D}_{\mathbf{u}_M}\mathbf{u}_M = \mathbf{D}_{\mathbf{U}\boldsymbol{\lambda}_0}\mathbf{U}\boldsymbol{\lambda}_0,$$

and

$$\mathbf{A}\mathbf{u}_M = \sum_{i \in [M]} \lambda_0[i] \mathbf{A}\mathbf{u}_i = \sum_{i \in [M]} \lambda_0[i] \mathbf{u}_i \circ \mathbf{u}_i = (\mathbf{U} \circ \mathbf{U})\boldsymbol{\lambda}_0.$$

Thus

$$(\mathbf{U} \circ \mathbf{U})\boldsymbol{\lambda}_0 = \mathbf{D}_{\mathbf{U}\boldsymbol{\lambda}_0}\mathbf{U}\boldsymbol{\lambda}_0 = (\mathbf{U}\boldsymbol{\lambda}_0) \circ (\mathbf{U}\boldsymbol{\lambda}_0).$$

Note that, the matrix \mathbf{U} can be written as $\mathbf{U} = [\mathbf{U}_K, \mathbf{U}_{M-K}]$, and the vector $\boldsymbol{\lambda}_0$ can be written as $\boldsymbol{\lambda}_0 = [\boldsymbol{\lambda}^\top, 0, \dots, 0]^\top$, where $\boldsymbol{\lambda} := [\lambda_1, \dots, \lambda_K]^\top$. Then the above equation can be transformed as follows:

$$(\mathbf{U}_K \circ \mathbf{U}_K)\boldsymbol{\lambda} = \mathbf{u}_M \circ \mathbf{u}_M, \text{ and } \mathbf{U}_K\boldsymbol{\lambda} = \mathbf{u}_M.$$

Similarly, $\forall s \in [K]$, we have

$$(\mathbf{U}_K \circ (\bar{\mathbf{S}}_s \mathbf{U}_K))\boldsymbol{\lambda} = \mathbf{u}_M \circ (\bar{\mathbf{S}}_s \mathbf{u}_M), \text{ and } \mathbf{U}_K\boldsymbol{\lambda} = \mathbf{u}_M,$$

where $\bar{\mathbf{S}}_s \mathbf{u}_M$ denotes a row circular shift such that $(\bar{\mathbf{S}}_s \mathbf{u}_M)[i] = \mathbf{u}_M[i + s]$. Note $\bar{\mathbf{S}}_s = \mathbf{S}_s^\top$.

Applying Lemma 9, we have

$$\text{tr}(\mathbf{D}_{e_i} \mathbf{U}_K \mathbf{D}_{\boldsymbol{\lambda}} \mathbf{U}_K^\top \bar{\mathbf{S}}_s^\top) = \text{tr}(\mathbf{D}_{e_i} \mathbf{U}_K \mathbf{D}_{\boldsymbol{\lambda}} \mathbf{U}_K^\top \mathbf{S}_s) = (\mathbf{u}_M \circ (\bar{\mathbf{S}}_s \mathbf{u}_M))[i]$$

Then the $(i, (i + s)_K)$ -th element of matrix $\mathbf{U}_K \mathbf{D}_\lambda \mathbf{U}_K^\top$ is

$$(\mathbf{U}_K \mathbf{D}_\lambda \mathbf{U}_K^\top)[i, (i + s)_K] = (\mathbf{u}_M \circ (\bar{\mathbf{S}}_s \mathbf{u}_M))[i] = \mathbf{u}_M[i] \cdot \mathbf{u}_M[(i + s)_K].$$

Then we have

$$\mathbf{U}_K \mathbf{D}_\lambda \mathbf{U}_K^\top = \mathbf{Q}, \text{ and } \mathbf{Q} = \mathbf{u}_M \mathbf{u}_M^\top.$$

When \mathbf{T} is non-singular, we know \mathbf{U} is invertible (full-rank), then

$$\mathbf{D}_\lambda = (\mathbf{U}_K^{-1} \mathbf{u}_M)(\mathbf{U}_K^{-1} \mathbf{u}_M)^\top.$$

Thus $\text{Rank}(\mathbf{D}_\lambda) = 1$. Recalling $\mathbf{1}^\top \boldsymbol{\lambda} = 1$, the vector $\boldsymbol{\lambda}$ could only be one-hot vectors, i.e. $\mathbf{e}_i, \forall i \in [K]$. This proves \mathbf{u}_M must be the same as one of $\mathbf{u}_i, i \in [K]$.

Wrapping-up: Unique \mathbf{T} From Step III, we know that, if $M = K$, we have a unique \mathbf{T} under the assumption that \mathbf{T} is informative and non-singular. Step IV proves the M -th ($M > K$) vector \mathbf{u} must be identical to one of $\mathbf{u}_i, i \in [K]$, indicating we only have $M = K$ non-repetitive \mathbf{u} vectors. Therefore, our consensus equations are sufficient for guaranteeing a unique \mathbf{T} . Besides, note there is no approximation applied during the whole proof. Thus with a perfect knowledge of $\mathbf{c}^{[\nu]}, \nu = 1, 2, 3$, the unique \mathbf{T} satisfying the consensus equations is indeed the true noise transition matrix.

A.2.2 Feasibility of Assumption $|E_3^*| = \Theta(N)$

We discuss the feasibility of our assumption on the number of 3-tuples. According to the definition of E_3^* , we know there are no more than $|E_3^*| \leq \lfloor N/3 \rfloor$ feasible 3-tuples. Strictly deriving the lower bound for $|E_3^*|$ is challenging due to the unknown distributions

of representations. To roughly estimate the order of $|E_3^*|$ (i.e., the maximum number of non-overlapping 3-tuples), we consider a special scenario where those high-dimensional representations could be mapped to a 2-D square of width $\sqrt{N/3}$, each grid of width 1 has exactly 3 mapped representations, and one mapped representation is at the center of each grid (also the center of each circle). Consider a particular construction of feasible 3-tuples as illustrated in Figure A.1. We require that, for each grid, the 2-NN fall in the corresponding circle. Otherwise, they may become the 2-NN of representations in other nearby grids. Assume the 2-NN are independently and uniformly distributed in the unit square, thus the probability of both 2-NN falling in the circle is $(\pi/4)^2$. Noting there are $N/3$ grids in the big square illustrated in Figure A.1, the expected number of feasible 3-tuples in this case is $\frac{\pi^2}{48} \cdot N = \Theta(N)$. Although this example only considers a special case, it demonstrates the order of $|E_3^*|$ could be $\Theta(N)$ with appropriate representations.

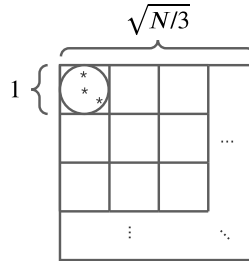


Figure A.1: Illustration of a special case.

A.2.3 Proof for Lemma 1

Then we present the proof for Lemma 1.

Proof. Recall in Eqn. (2.7), each high-order consensus pattern could be estimated by the sample mean of $|E_3^*|$ independent and identically distributed random variables, thus according to Hoeffding's inequality [70], w.p. $1 - \delta$, we have

$$|\hat{\mathbf{c}}^{[i]}[j] - \mathbf{c}^{[i]}[j]| \leq \sqrt{\frac{\ln \frac{2}{\delta}}{2|E_3^*|}}, i = 1, 2, 3, \forall j,$$

which is at the order of $O(\sqrt{\ln(1/\delta)/N})$. \square

A.2.4 Proof for Theorem 2

Consider a particular uniform off-diagonal matrix \mathbf{T} , where the off-diagonal elements are $T_{ij} = \frac{1-T_{ii}}{K-1}$. Recall the clean prior probability for the i -th class is p_i . To find the upper bound for the sample complexity, we can only consider a subset of our consensus equations. Specifically, we consider the equations related to the i -th element of Eqn. (2.2) and Eqn. (2.3) when $r = 0$. Then a solution to our consensus equations will need to satisfy at least the following two equations:

$$\hat{p}_i \hat{T}_{ii} + (1 - \hat{p}_i) \frac{1 - \hat{T}_{ii}}{K - 1} = \hat{c}_1, \quad (\text{A.6})$$

$$\hat{p}_i \hat{T}_{ii}^2 + (1 - \hat{p}_i) \frac{(1 - \hat{T}_{ii})^2}{(K - 1)^2} = \hat{c}_2, \quad (\text{A.7})$$

where \hat{p}_i and \hat{T}_{ii} denote the estimated clean prior probability and noisy transition matrix, \hat{c}_1 and \hat{c}_2 denote the corresponding estimates of first- and second-order statistics.

Lemma 1 shows, with probability $1 - \delta$:

$$|\hat{c}_i - c_i| \leq O\left(\sqrt{\frac{\ln(1/\delta)}{N}}\right).$$

Multiplying both sides of Eqn. (A.6) by T_{ii} and adding Eqn. (A.7), we have

$$K(K-1)\hat{p}_i\hat{T}_{ii}^2 + (1-\hat{p}_i)(1-\hat{T}_{ii}) = (K-1)\hat{c}_1\hat{T}_{ii} + (K-1)^2\hat{c}_2.$$

Note the above equality also holds for the true values p_i, T_{ii}, c_1, c_2 . Taking the difference we have

$$\begin{aligned} & (\hat{T}_{ii} - T_{ii})(K(K-1)p_i(T_{ii} + \hat{T}_{ii}) - (1-p_i) - (K-1)c_1) \\ &= (K-1)^2(\hat{c}_2 - c_2) + (K-1)(\hat{c}_1 - c_1)\hat{T}_{ii} - K(K-1)\hat{T}_{ii}^2(\hat{p}_i - p_i) - (\hat{T}_{ii} - 1)(\hat{p}_i - p_i). \end{aligned}$$

Taking the absolute value for both sides yields

$$\begin{aligned} & |\hat{T}_{ii} - T_{ii}| \cdot |K(K-1)p_i(T_{ii} + \hat{T}_{ii}) - (1-p_i) - (K-1)c_1| \\ & \leq (K-1)^2|\hat{c}_2 - c_2| + (K-1)|\hat{c}_1 - c_1| + (K(K-1) + 1)|\hat{p}_i - p_i| \end{aligned}$$

From Eqn. (A.6), we have

$$\hat{p}_i = \frac{K-1}{K} \frac{\hat{c}_1 - 1/K}{\hat{T}_{ii} - 1/K} + \frac{1}{K}.$$

Thus

$$|\hat{p}_i - p_i| \leq \frac{K-1}{K} \frac{|\hat{c}_1 - c_1|}{\min(\hat{T}_{ii}, T_{ii}) - 1/K},$$

indicating $|\hat{p}_i - p_i|$ is at the order of $|\hat{c}_1 - c_1|$. Note that

$$K(K-1)p_i(T_{ii} + \hat{T}_{ii}) - (1-p_i) - (K-1)c_1 \geq K(K-1)p_iT_{ii} - (1-p_i) - (K-1)c_1.$$

When $K(K-1)p_iT_{ii} - (1-p_i) - (K-1)c_1 > 0$, we have

$$|\hat{T}_{ii} - T_{ii}| \leq \frac{(K-1)^2|\hat{c}_2 - c_2| + (K-1)|\hat{c}_1 - c_1| + (K(K-1) + 1)\frac{K-1}{K} \frac{|\hat{c}_1 - c_1|}{\min(\hat{T}_{ii}, T_{ii}) - 1/K}}{K(K-1)p_iT_{ii} - (1-p_i) - (K-1)c_1}.$$

Then by union bound we know, w.p. $1 - 2\delta$, the estimation error $|\hat{T}_{ii} - T_{ii}|$ is at the same order as $|\hat{c}_i - c_i|$, i.e. $O(\sqrt{\frac{\ln(1/\delta)}{N}})$.

A.3 More Discussions for HOC

A.3.1 Soft 2-NN Label Clusterability

The soft 2-NN label clusterability means one's 2-NN may have a certain (but small) probability of belonging to different clean classes. Statistically, if we use a new matrix \mathbf{T}^{soft} to characterize the probability of getting a different nearest neighbor, i.e. $T_{ij}^{\text{soft}} = \mathbb{P}(Y_2 = j | Y_1 = i) = \mathbb{P}(Y_3 = j | Y_1 = i)$, the second-order consensus becomes $\mathbf{c}_r^{[2]} := (\mathbf{T} \circ (\mathbf{T}^{\text{soft}} \mathbf{T}_r))^\top \mathbf{p}$ and the third-order consensus becomes $\mathbf{c}_{r,s}^{[3]} := (\mathbf{T} \circ (\mathbf{T}^{\text{soft}} \mathbf{T}_r) \circ (\mathbf{T}^{\text{soft}} \mathbf{T}_s))^\top \mathbf{p}$. Specifically, if $T_{ij}^{\text{soft}} = e, \forall i \neq j$ and $T_{ii}^{\text{soft}} = 1 - (K - 1)e, 0 \leq e < 1/K$, where e captures the small perturbation of the 2-NN assumption, our solution will likely output a transition matrix that affects the label noise between the effects of $\mathbf{T}^{\text{soft}} \mathbf{T}$ and \mathbf{T} . The above observation informs us that our estimation will be away from the true \mathbf{T} by at most a factor e . When $e = 0$, we recover the original 2-NN label clusterability condition.

A.3.2 Local $\mathbf{T}(X)$

Sparse regularizer Compared with estimating one global \mathbf{T} using the whole dataset of size N , each local estimation will have access to only M instances, where $M \ll N$. Thus the feasibility of returning an accurate $\mathbf{T}(x_n)$ requires more consideration. In some particular cases, e.g., HOC Local in Table 4.2, when \mathbf{p} is sparse due to the local datasets, we usually add a regularizer to ensure a sparse \mathbf{p} , such as $\sum_{i \in [K]} \ln(c_i + \varepsilon), \varepsilon \rightarrow 0_+$, where c_i is the i -th element of \mathbf{p} . Note the standard sparse regularizer, i.e. ℓ_1 -norm $\|\mathbf{p}\|_1$,

could not be applied here since $\|\mathbf{p}\|_1 = 1$. Therefore, with a regularizer that shrinks the search space and fewer variables, we could get an accurate estimate of $T(X)$ with a small M .

Other extensions Even with M -NN noise clusterability, estimating $\mathbf{T}(X)$ for the whole dataset requires executing Algorithm 3 a numerous number of times ($\sim N/M$). If equipped with prior knowledge that the label noise can be divided into several groups and $\mathbf{T} = \mathbf{T}(X)$ within each group [198, 170], we only need to estimate \mathbf{T} for each group by treating instances in each group as a local dataset and directly apply Algorithm 3. As a preliminary work on estimating \mathbf{T} relying on clusterability, the focus of this paper is to provide a generic method for estimating \mathbf{T} given a dataset. Designing efficient algorithms to split the original dataset into a tractable number of local datasets is interesting for future investigation.

A.3.3 Feasibility of Assumption 1 and Assumption 1

1. Denote the confusion matrix by $\mathbf{C}[h]$, where each element is $C_{ij}[h] := \mathbb{P}(Y = i, h(X) = j)$ and $h(X) = j$ represents the event that the classifier predicts j given feature X . Then the noisy confusion matrix could be written as $\tilde{\mathbf{C}}[h] := \mathbf{T}^\top \mathbf{C}[h]$. If \mathbf{T} is non-singular (a.k.a. invertible), statistically, we can always find the inverse matrix \mathbf{T}^{-1} such that the clean confusion matrix could be recovered as $\mathbf{C}[h] = (\mathbf{T}^{-1})^\top \tilde{\mathbf{C}}[h]$. Otherwise, we may think the label noise is too “much” such that the clean confusion matrix is not recoverable by \mathbf{T} . Then learning \mathbf{T} may not be

meaningful anymore. Therefore, Assumption 1 is effectively ensuring the necessity of estimating \mathbf{T} .

2. We require $T_{ii} > T_{ij}$ in Assumption 1 to ensure instances from observed class i (observed from noisy labels) are informative [123]. Intuitively, this assumption characterizes a particular permutation of row vectors in \mathbf{T} . Otherwise, there may exist $K!$ possible solutions by considering all the permutations of K rows [127].

A.4 More Detailed Experiment Settings for HOC

A.4.1 Generating the Instance-Dependent Label Noise

In this section, we introduce how to generate instance-based label noise, which is illustrated in Algorithm 8. Note this algorithm follows the state-of-the-art method [198, 227]. Define the noise rate (the global flipping rate) as η . To calculate the probability of x_n mapping to each class under certain noise conditions, we set sample instance flip rates q_n and sample parameters W . The size of W is $S \times K$, where S denotes the length of each feature.

First, we sample instance flip rates q_n from a truncated normal distribution $\mathbf{N}(\eta, 0.1^2, [0, 1])$ in Line 2. The average flipping rate (a.k.a. average noise rate) is η . q_n avoids all the instances having the same flip rate. Then, in Line 3, we sample parameters W from the standard normal distribution for generating the instance-dependent label noise. Each column of W acts as a projection vector. After acquiring q_n and W , we can calculate the probability of getting a wrong label for each instance (x_n, y_n) in Lines 4 – 6.

Note that in Line 5, we set $p_{y_n} = -\infty$, which ensures that x_n will not be mapped to its own true label. In addition, Line 7 ensures the sum of all the entries of p is 1. Suppose there are two features: x_i and x_j where $x_i = x_j$. Then the possibility p of these two features, calculated by $x \cdot W$, from the Algorithm 8, would be exactly the same. Thus the label noise is strongly instance-dependent.

Note Algorithm 8 cannot ensure $T_{ii}(X) > T_{ij}(X)$ when $\eta > 0.5$. To generate an informative dataset, we set $0.9 \cdot T_{ii}(X)$ as the upper bound of $T_{ij}(X)$ and distribute the remaining probability to other classes.

A.4.2 Basic Hyper-Parameters

To testify the classification performance, we adopt the flow: 1) Pre-training \rightarrow 2) Global Training \rightarrow 3) Local Training. Our HOC estimator is applied once at the beginning of each above step. Each training stage re-trains the model. In Stage-1, we load the standard ResNet50 model pre-trained on ImageNet to obtain basic representations. At the beginning of Stage-2 and Stage-3, we use the representations given by the current model. All experiments are repeated three times. *HOC Global* only employs one global T with $G = 50$ and $|E| = 15k$ as inputs of Algorithm 8. *HOC Local* uses 300 local matrices (250-NN noise clusterability, $|D_{h(n)}| = 250$, $G = 30$, $|E| = 100$) for CIFAR-10 and 5 local matrices (10k-NN noise clusterability, $|D_{h(n)}| = 10k$, $G = 30$, $|E| = 5k$) for CIFAR-100. Note the local matrices may not cover the whole dataset. For those uncovered instances, we simply apply T .

Algorithm 8 Instance-Dependent Label Noise Generation

Input:

- 1: Clean examples $(x_n, y_n)_{n=1}^N$; Noise rate: η ; Size of feature: $1 \times S$; Number of classes: K .

Iteration:

- 2: Sample instance flip rates q_n from the truncated normal distribution $\mathcal{N}(\eta, 0.1^2, [0, 1])$;

- 3: Sample $W \in \mathcal{R}^{S \times K}$ from the standard normal distribution $\mathcal{N}(0, 1^2)$;

for $n = 1$ to N **do**

- 4: $p = x_n \cdot W$ // Generate instance dependent flip rates. The size of p is $1 \times K$.

- 5: $p_{y_n} = -\infty$ // Only consider entries that are different from the true label

- 6: $p = q_n \cdot \text{SoftMax}(p)$ // Let q_n be the probability of getting a wrong label

- 7: $p_{y_n} = 1 - q_n$ // Keep clean w.p. $1 - q_n$

- 8: Randomly choose a label from the label space as noisy label \tilde{y}_n according to p ;

end for**Output:**

- 9: Noisy examples $(x_i, \tilde{y}_n)_{n=1}^N$.
-

Other hyperparameters:

- Batch size: 128 (CIFAR), 32 (Clothing1M)
- Learning rate:
 - CIFAR-10: Pre-training: 0.1 for 20 epochs \rightarrow 0.01 for 20 epochs. Global Training:

0.1 for 20 epochs \rightarrow 0.01 for 20 epochs. Local Training: 0.1 for 60 epochs \rightarrow 0.01 for 60 epochs \rightarrow 0.001 for 60 epochs.

- CIFAR-100: Pre-training: 0.1 for 30 epochs \rightarrow 0.01 for 30 epochs. Global Training: 0.1 for 30 epochs \rightarrow 0.01 for 30 epochs. Local Training: 0.1 for 30 epochs \rightarrow 0.01 for 30 epochs \rightarrow 0.001 for 30 epochs.
- Clothing1M: 0.01 for 25 epochs \rightarrow 0.001 for 25 epochs \rightarrow 0.0001 for 15 epochs \rightarrow 0.00001 for 15 epochs (Pre-training, Global training, and local training)
- Momentum: 0.9
- Weight decay: 0.0005 (CIFAR) and 0.001 (Clothing1M)
- Optimizer: SGD (Model training) and Adam with initial a learning rate of 0.1 (solving for \mathbf{T})

For each epoch in Clothing1M, we sample 1000 mini-batches from the training data while ensuring the (noisy) labels are balanced. The global \mathbf{T} is obtained by an average of \mathbf{T} from 5 random epochs. We only use $\mathbf{T}(X) = \mathbf{T}$ in local training. Estimating local transition matrices using HOC on Clothing1M is feasible, e.g., assuming M -NN noise clusterability, but it may be time-consuming to tune M . Noting our current performance is already satisfying, and the focus of this paper is on the ability to estimate \mathbf{T} , we leave the combination of $\mathbf{T}(X)$ with loss correction or other advanced techniques for future works.

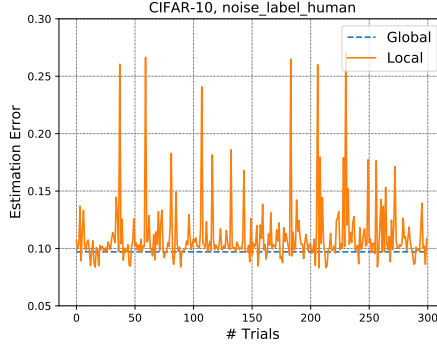


Figure A.2: Illustration of the global and local estimation errors.

A.4.3 Global and Local Estimation Errors on CIFAR-10 with Human Noise

Algorithm 9 details the generation of local datasets. Notice the fact that the i -th row of $\mathbf{T}(x_n)$ could be any feasible values when $p_i = 0$, so as the estimates $\hat{\mathbf{T}}_{\text{local}}$. In such a case, we need to refer to \mathbf{T} to complete the information. Particularly, we calculate the weighted average value with the corresponding $\hat{\mathbf{T}}$ as

$$\hat{\mathbf{T}}_{\text{local}}[i] = (1 - \zeta + \hat{p}_i)\hat{\mathbf{T}}_{\text{local}}[i] + (\zeta - \hat{p}_i)\hat{\mathbf{T}}[i],$$

where $\hat{\mathbf{T}}_{\text{local}}[i]$ and $\hat{\mathbf{T}}[i]$ denote the i -th row of estimates $\hat{\mathbf{T}}_{\text{local}}$ and $\hat{\mathbf{T}}$, \hat{p}_i denotes the estimated clean prior probability of class- i given the local dataset. We use $\zeta = 1$ for local estimates of CIFAR-10, and $\zeta = 0.5$ for local estimates of CIFAR-100.

Figure A.2 illustrates the variation of local estimation errors on CIFAR-10 with human noise using HOC. From the figure, we know the global estimation error is 0.0970, and the mean and standard deviation of local estimation errors are 0.1103 and 0.0278.

Algorithm 9 Local Datasets Generation

Input:

1: Maximal rounds: G' . Local dataset size: L . Noisy dataset: $\tilde{D} = \{(x_n, \tilde{y}_n)\}_{n \in [N]}$. Noisy dataset size: $|D|$.

Iteration:

2: Initialize the $|D|$ -dimensional index list: $S = \mathbf{1}$

for $k = 1$ to G' **do**

if(size($S[S > 0]$) > 0) **then**

3: $\text{Idx}_{\text{selected}} = \text{random.choice}(S[S > 0])$ // Choose a local center index randomly from the unselected index of \tilde{D} .

else

4: $\text{Idx}_{\text{selected}} = \text{random.randint}(0, |D|)$ // If the selected index has covered \tilde{D} , we choose local center randomly.

end if

5: $\text{Idx}_{\text{local}} = \text{SelectbyDist}(\text{Idx}_{\text{selected}}, L)$ // Select the index of L features closest to $\text{Idx}_{\text{selected}}$.

6: $S[\text{Idx}_{\text{local}}] = -1$ // Mark the state of the selected index in S to avoid duplicate selection.

7: $\tilde{D}_k = \tilde{D}[\text{Idx}_{\text{local}}]$ // Build a local dataset by selecting $(x_i, \tilde{y}_i), i \in \text{Idx}_{\text{local}}$.

end for

Output:

8: Local Datasets $\tilde{D}_k = \{(x_n, \tilde{y}_n)\} \cup \{(x_{n_1}, \tilde{y}_{n_1}), \dots, (x_{n_M}, \tilde{y}_{n_M})\}$, $n_i, k \in [L], i \in [M]$.

A.5 Theoretical Guarantees for HOC-extension

A.5.1 Common f -Divergence Functions

Following [145, 186], we show some common f -divergence functions in Table A.1.

Table A.1: List of popular f -divergences.

Name	Generator $f(v)$	Opt. variational func. g^*	dom_{f^*}	Opt. conjugate func. $f^*(u)$
Total Variation	$\frac{1}{2} v - 1 $	$\frac{1}{2} \text{sign} \left(\frac{p(v)}{q(v)} - 1 \right)$	$u \in [-\frac{1}{2}, \frac{1}{2}]$	u
KL	$v \log v$	$1 + \log \frac{p(v)}{q(v)}$	\mathbb{R}	e^{u-1}
Jenson-Shannon	$-(1+v) \log \frac{1+v}{2} + v \log v$	$\log \frac{2p(v)}{p(v)+q(v)}$	$u < \log 2$	$-\log(2 - e^u)$
Squared Hellinger	$(\sqrt{v} - 1)^2$	$1 - \sqrt{\frac{q(v)}{p(v)}}$	$u < 1$	$\frac{u}{1-u}$
Pearson \mathcal{X}^2	$(1-v)^2$	$2 \left(\frac{p(v)}{q(v)} - 1 \right)$	\mathbb{R}	$\frac{1}{4}u^2 + u$
Neyman \mathcal{X}^2	$\frac{(1-v)^2}{v}$	$1 - \left(\frac{q(v)}{p(v)} \right)^2$	$u < 1$	$2 - 2\sqrt{1-u}$
Reverse KL	$-\log v$	$-\frac{q(v)}{p(v)}$	\mathbb{R}_-	$-1 - \log(-u)$

A.5.2 Total-Variation

A.5.2.1 Proof for Lemma 3

Proof. Consider TV, we have:

$$\begin{aligned}
\text{VD}_f(\tilde{g}^*) &= \mathbb{E}_{V \sim P}[\tilde{g}^*(V)] - \mathbb{E}_{V \sim Q}[f^*(\tilde{g}^*(V))] \\
&= \frac{1}{2} \left[\mathbb{E}_{V \sim P} \left[\text{sign} \left(\frac{\tilde{p}(V)}{\tilde{q}(V)} - 1 \right) \right] - \mathbb{E}_{V \sim Q} \left[\text{sign} \left(\frac{\tilde{p}(V)}{\tilde{q}(V)} - 1 \right) \right] \right] \\
&= \frac{1}{2} \int_z \sum_{i \in \{1,2\}} [\mathbb{P}(Z = z, Y = i) - \mathbb{P}(Z = z)\mathbb{P}(Y = i)] \\
&\quad \cdot \text{sign} \left[\mathbb{P}(Z = z, \tilde{Y} = i) - \mathbb{P}(Z = z)\mathbb{P}(\tilde{Y} = i) \right] dz \\
&= \frac{1}{2} [\mathbb{P}(Z = z, Y = 1) - \mathbb{P}(Z = z)\mathbb{P}(Y = 1)] \\
&\quad \cdot \text{sign} \left[\mathbb{P}(Z = z, \tilde{Y} = 1) - \mathbb{P}(Z = z)\mathbb{P}(\tilde{Y} = 1) \right] \\
&\quad + \frac{1}{2} [\mathbb{P}(Z = z, Y = 2) - \mathbb{P}(Z = z)\mathbb{P}(Y = 2)] \\
&\quad \cdot \text{sign} \left[\mathbb{P}(Z = z, \tilde{Y} = 2) - \mathbb{P}(Z = z)\mathbb{P}(\tilde{Y} = 2) \right] dz \\
&= \frac{1}{2} [\mathbb{P}(Z = z, Y = 1) - \mathbb{P}(Z = z)\mathbb{P}(Y = 1)] \\
&\quad \cdot \text{sign} \left[\sum_{j \in \{1,2\}} \left(\mathbb{P}(\tilde{Y} = 1 | Z = z, Y = j)\mathbb{P}(Z = z, Y = j) \right. \right. \\
&\quad \quad \left. \left. - \mathbb{P}(Z = z)\mathbb{P}(\tilde{Y} = 1 | Y = j)\mathbb{P}(Y = j) \right) \right] \\
&\quad + \frac{1}{2} [\mathbb{P}(Z = z, Y = 2) - \mathbb{P}(Z = z)\mathbb{P}(Y = 2)] \\
&\quad \cdot \text{sign} \left[\sum_{j \in \{1,2\}} \left(\mathbb{P}(\tilde{Y} = 2 | Z = z, Y = j)\mathbb{P}(Z = z, Y = j) \right. \right. \\
&\quad \quad \left. \left. - \mathbb{P}(Z = z)\mathbb{P}(\tilde{Y} = 2 | Y = j)\mathbb{P}(Y = j) \right) \right] dz.
\end{aligned}$$

Note (assume class-dependent label noise)

$$\begin{aligned}
& \text{sign} \left[\sum_{j \in \{1,2\}} \left(\mathbb{P}(\tilde{Y} = 1 | Z = z, Y = j) \mathbb{P}(Z = z, Y = j) - \mathbb{P}(Z = z) \mathbb{P}(\tilde{Y} = 1 | Y = j) \mathbb{P}(Y = j) \right) \right] \\
&= \text{sign} [(1 - e_1) \mathbb{P}(Z = z, Y = 1) + e_2 \mathbb{P}(Z = z, Y = 2) - (1 - e_1) \mathbb{P}(Z = z) \mathbb{P}(Y = 1) - e_2 \mathbb{P}(Z = z) \mathbb{P}(Y = 2)] \\
&= \text{sign} [(1 - e_1) (\mathbb{P}(Z = z, Y = 1) - \mathbb{P}(Z = z) \mathbb{P}(Y = 1)) + e_2 (\mathbb{P}(Z = z, Y = 2) - \mathbb{P}(Z = z) \mathbb{P}(Y = 2))] \\
&= \text{sign} \left[(1 - e_1) (\mathbb{P}(Z = z, Y = 1) - \mathbb{P}(Z = z) \mathbb{P}(Y = 1)) \right. \\
&\quad \left. + e_2 (\mathbb{P}(Z = z) - \mathbb{P}(Z = z, Y = 1) - \mathbb{P}(Z = z)(1 - \mathbb{P}(Y = 1))) \right] \\
&= \text{sign}(1 - e_1 - e_2) \cdot \text{sign}(\mathbb{P}(Z = z, Y = 1) - \mathbb{P}(Z = z) \mathbb{P}(Y = 1)).
\end{aligned}$$

Thus

$$\begin{aligned}
& \text{VD}_f(\tilde{g}^*) \\
&= \frac{1}{2} \int_z [\mathbb{P}(Z = z, Y = 1) - \mathbb{P}(Z = z) \mathbb{P}(Y = 1)] \cdot \text{sign}(1 - e_1 - e_2) \cdot \text{sign}(\mathbb{P}(Z = z, Y = 1) - \mathbb{P}(Z = z) \mathbb{P}(Y = 1)) \\
&\quad + [\mathbb{P}(Z = z, Y = 2) - \mathbb{P}(Z = z) \mathbb{P}(Y = 2)] \cdot \text{sign}(1 - e_1 - e_2) \cdot \text{sign}(\mathbb{P}(Z = z, Y = 2) - \mathbb{P}(Z = z) \mathbb{P}(Y = 2)) dz.
\end{aligned}$$

When $\text{sign}(1 - e_1 - e_2) = 1$, i.e. $e_1 + e_2 < 1$, we have

$$\begin{aligned}
& \text{VD}_f(\tilde{g}^*) \\
&= \frac{1}{2} \int_z \sum_{i \in \{1,2\}} [\mathbb{P}(Z = z, Y = i) - \mathbb{P}(Z = z) \mathbb{P}(Y = i)] \\
&\quad \cdot \text{sign}(\mathbb{P}(Z = z, Y = i) - \mathbb{P}(Z = z) \mathbb{P}(Y = i)) dz \\
&= \text{VD}_f(g^*).
\end{aligned}$$

□

A.5.2.2 Proof for Theorem 3

Proof. Recall that, to show an f -mutual information metric is ϵ -order-preserving under label noise, we need to study how $\widetilde{\text{VD}}_f(\tilde{g}^*)$ differs from the order of $\text{VD}_f(g^*)$.

For total variation, with Lemma 2 and Lemma 3, we know

$$\widetilde{\text{VD}}_{\text{TV}}(\tilde{g}^*) = (1 - e_1 - e_2)\text{VD}_{\text{TV}}(\tilde{g}^*) = (1 - e_1 - e_2)\text{VD}_{\text{TV}}(g^*).$$

Therefore, when $e_1 + e_2 < 1$, $\widetilde{\text{VD}}_{\text{TV}}(\tilde{g}^*)$ always preserves the order of $\text{VD}_{\text{TV}}(g^*)$, indicating the total-variation-based mutual information is 0-order-preserving under class-dependent label noise. \square

A.5.3 KL Divergence

The definition of MI is

$$\begin{aligned} & I(Z, \tilde{Y}) \\ &= \sum_{j \in \{1,2\}} \int_z \mathbb{P}(Z = z, \tilde{Y} = j) \log \left(\frac{\mathbb{P}(Z = z, \tilde{Y} = j)}{\mathbb{P}(Z = z)\mathbb{P}(\tilde{Y} = j)} \right) dz \\ &= \sum_{j \in \{1,2\}} \int_z \mathbb{P}(Z = z, \tilde{Y} = j) \log \left(\mathbb{P}(Z = z, \tilde{Y} = j) \right) dz \\ &\quad - \sum_{j \in \{1,2\}} \int_z \mathbb{P}(Z = z, \tilde{Y} = j) \left(\log(\mathbb{P}(Z = z)) + \log(\mathbb{P}(\tilde{Y} = j)) \right) dz \\ &= \underbrace{\sum_{j \in \{1,2\}} \int_z \mathbb{P}(Z = z, \tilde{Y} = j) \log \left(\mathbb{P}(Z = z, \tilde{Y} = j) \right) dz}_{\text{Term-1: } -H(Z, \tilde{Y})} \\ &\quad - \underbrace{\int_z \mathbb{P}(Z = z) \log \mathbb{P}(Z = z) dz}_{\text{Term-2: } -H(Z)} - \underbrace{\sum_{j \in \{1,2\}} \mathbb{P}(\tilde{Y} = j) \log \mathbb{P}(\tilde{Y} = j) dz}_{\text{Term-3: } -H(\tilde{Y})}, \end{aligned}$$

where

$$\begin{aligned}
-H(Z, \tilde{Y}) &= \sum_{j \in \{1,2\}} \int_z \mathbb{P}(Z = z, \tilde{Y} = j) \log \mathbb{P}(Z = z, \tilde{Y} = j) dz \\
&= \int_z \mathbb{P}(Z = z, \tilde{Y} = 1) \log \mathbb{P}(Z = z, \tilde{Y} = 1) \\
&\quad + \mathbb{P}(Z = z, \tilde{Y} = 2) \log \mathbb{P}(Z = z, \tilde{Y} = 2) dz.
\end{aligned}$$

We first decouple term-1. Note

$$\begin{aligned}
&\int_z \mathbb{P}(Z = z, \tilde{Y} = 1) \log \mathbb{P}(Z = z, \tilde{Y} = 1) dz \\
&= \int_z \left[\mathbb{P}(\tilde{Y} = 1 | Z = z, Y = 1) \mathbb{P}(Z = z, Y = 1) + \mathbb{P}(\tilde{Y} = 1 | Z = z, Y = 2) \mathbb{P}(Z = z, Y = 2) \right] \\
&\quad \cdot \log \left[\mathbb{P}(\tilde{Y} = 1 | Z = z, Y = 1) \mathbb{P}(Z = z, Y = 1) + \mathbb{P}(\tilde{Y} = 1 | Z = z, Y = 2) \mathbb{P}(Z = z, Y = 2) \right] dz \\
&= \int_z [(1 - e_1) \mathbb{P}(Z = z, Y = 1) + e_2 \mathbb{P}(Z = z, Y = 2)] \\
&\quad \cdot \log [(1 - e_1) \mathbb{P}(Z = z, Y = 1) + e_2 \mathbb{P}(Z = z, Y = 2)] dz \\
&= \int_z [(1 - e_1 - e_2) \mathbb{P}(Z = z, Y = 1) + e_2 \mathbb{P}(Z = z)] \\
&\quad \cdot \log [(1 - e_1 - e_2) \mathbb{P}(Z = z, Y = 1) + e_2 \mathbb{P}(Z = z)] dz \\
&= \int_z [(1 - e_1 - e_2) \mathbb{P}(Z = z, Y = 1) + e_2 \mathbb{P}(Z = z)] \log \mathbb{P}(Z = z, Y = 1) \\
&\quad + [(1 - e_1 - e_2) \mathbb{P}(Z = z, Y = 1) + e_2 \mathbb{P}(Z = z)] \\
&\quad \cdot \log \left[\frac{(1 - e_1 - e_2) \mathbb{P}(Z = z, Y = 1) + e_2 \mathbb{P}(Z = z)}{\mathbb{P}(Z = z, Y = 1)} \right] dz \\
&= \int_z [(1 - e_1 - e_2) \mathbb{P}(Z = z, Y = 1) + e_2 \mathbb{P}(Z = z)] \log \mathbb{P}(Z = z, Y = 1) \\
&\quad + [(1 - e_1 - e_2) \mathbb{P}(Z = z, Y = 1) + e_2 \mathbb{P}(Z = z)] \log \left[1 - e_1 + e_2 \frac{\mathbb{P}(Z = z, Y = 2)}{\mathbb{P}(Z = z, Y = 1)} \right] dz.
\end{aligned}$$

Let $\alpha = \mathbb{P}(Z = z, Y = 1) / \mathbb{P}(Z = z, Y = 2) \in [0, +\infty)$ (note α is actually a

function of (Z, Y)). Then

$$\begin{aligned}
& \int_z \mathbb{P}(Z = z, \tilde{Y} = 1) \log \mathbb{P}(Z = z, \tilde{Y} = 1) dz \\
&= \int_z (1 - e_1 - e_2) \mathbb{P}(Z = z, Y = 1) \log \mathbb{P}(Z = z, Y = 1) dz \\
&\quad + \int_z e_2 \mathbb{P}(Z = z) [\log \alpha + \log \mathbb{P}(Z = z, Y = 2)] \\
&\quad + [(1 - e_1 - e_2)\alpha \mathbb{P}(Z = z, Y = 2) + e_2 \mathbb{P}(Z = z)] \log \left(1 - e_1 + \frac{e_2}{\alpha}\right) dz
\end{aligned}$$

and

$$\begin{aligned}
& \int_z \mathbb{P}(Z = z, \tilde{Y} = 2) \log \mathbb{P}(Z = z, \tilde{Y} = 2) dz \\
&= \int_z (1 - e_1 - e_2) \mathbb{P}(Z = z, Y = 2) \log \mathbb{P}(Z = z, Y = 2) dz \\
&\quad + \int_z e_1 \mathbb{P}(Z = z) \log \mathbb{P}(Z = z, Y = 2) \\
&\quad + [(1 - e_1 - e_2) \mathbb{P}(Z = z, Y = 2) + e_1 \mathbb{P}(Z = z)] \log (1 - e_2 + e_1 \alpha) dz.
\end{aligned}$$

Thus

$$\begin{aligned}
& \sum_{j \in \{1,2\}} \int_z \mathbb{P}(Z = z, \tilde{Y} = j) \log \mathbb{P}(Z = z, \tilde{Y} = j) dz \\
&= (1 - e_1 - e_2) \sum_{i \in \{1,2\}} \int_z \mathbb{P}(Z = z, Y = i) \log \mathbb{P}(Z = z, Y = i) dz \\
&\quad + \int_z e_2 \mathbb{P}(Z = z) \log \alpha + (e_1 + e_2) \mathbb{P}(Z = z) \log \mathbb{P}(Z = z, Y = 2) \\
&\quad + (1 - e_1 - e_2) \mathbb{P}(Z = z, Y = 2) \left[\alpha \log \left(1 - e_1 + \frac{e_2}{\alpha} \right) + \log(1 - e_2 + e_1 \alpha) \right] \\
&\quad + \mathbb{P}(Z = z) \left[e_1 \log(1 - e_2 + e_1 \alpha) + e_2 \log \left(1 - e_1 + \frac{e_2}{\alpha} \right) \right] dz \\
&= (1 - e_1 - e_2) \sum_{i \in \{1,2\}} \int_z \mathbb{P}(Z = z, Y = i) \log \mathbb{P}(Z = z, Y = i) dz \\
&\quad + \int_z e_2 \mathbb{P}(Z = z) \log \alpha - (e_1 + e_2) \mathbb{P}(Z = z) \log(\alpha + 1) + (e_1 + e_2) \mathbb{P}(Z = z) \log \mathbb{P}(Z = z) \\
&\quad + \frac{(1 - e_1 - e_2)}{\alpha + 1} \mathbb{P}(Z = z) \left[\alpha \log \left(1 - e_1 + \frac{e_2}{\alpha} \right) + \log(1 - e_2 + e_1 \alpha) \right] \\
&\quad + \mathbb{P}(Z = z) \left[e_1 \log(1 - e_2 + e_1 \alpha) + e_2 \log \left(1 - e_1 + \frac{e_2}{\alpha} \right) \right] dz \\
&= (1 - e_1 - e_2) \sum_{i \in \{1,2\}} \int_z \mathbb{P}(Z = z, Y = i) \log \mathbb{P}(Z = z, Y = i) dz \quad \text{(Term 1.1)} \\
&\quad + \int_z (e_1 + e_2) \mathbb{P}(Z = z) \log \mathbb{P}(Z = z) + \mathbb{P}(Z = z) \Delta_{\text{Bias}}(\alpha, e_1, e_2) dz, \quad \text{(Term 1.2)}
\end{aligned}$$

where in Term 1.2:

$$\begin{aligned}
\Delta_{\text{Bias}}(\alpha, e_1, e_2) &= e_2 \log \alpha - (e_1 + e_2) \log(\alpha + 1) \\
&\quad + \frac{(1 - e_1 - e_2)}{\alpha + 1} \left[\alpha \log \left(1 - e_1 + \frac{e_2}{\alpha} \right) + \log(1 - e_2 + e_1 \alpha) \right] \quad \text{(A.8)} \\
&\quad + \left[e_1 \log(1 - e_2 + e_1 \alpha) + e_2 \log \left(1 - e_1 + \frac{e_2}{\alpha} \right) \right].
\end{aligned}$$

In Term 1.1, recalling $\alpha = \mathbb{P}(Z = z, Y = 1)/\mathbb{P}(Z = z, Y = 2)$, we have

$$\begin{aligned}
& (1 - e_1 - e_2) \sum_{i \in \{1,2\}} \int_z \mathbb{P}(Z = z, Y = i) \log \mathbb{P}(Z = z, Y = i) dz \\
&= (1 - e_1 - e_2) \int_z \frac{\mathbb{P}(Z = z)}{\alpha + 1} \log \frac{\mathbb{P}(Z = z)}{\alpha + 1} + \frac{\mathbb{P}(Z = z)\alpha}{\alpha + 1} \log \frac{\mathbb{P}(Z = z)\alpha}{\alpha + 1} dz \\
&= (1 - e_1 - e_2) \int_z \frac{\mathbb{P}(Z = z)}{\alpha + 1} \log \mathbb{P}(Z = z) + \frac{\mathbb{P}(Z = z)}{\alpha + 1} \log \frac{1}{\alpha + 1} \\
&\quad + \frac{\mathbb{P}(Z = z)\alpha}{\alpha + 1} \log \mathbb{P}(Z = z) + \frac{\mathbb{P}(Z = z)\alpha}{\alpha + 1} \log \frac{\alpha}{\alpha + 1} dz \\
&= (1 - e_1 - e_2) \int_z \mathbb{P}(Z = z) \log \mathbb{P}(Z = z) dz \\
&\quad + (1 - e_1 - e_2) \int_z \mathbb{P}(Z = z) \left[\frac{\alpha}{\alpha + 1} \log \frac{\alpha}{\alpha + 1} + \frac{1}{\alpha + 1} \log \frac{1}{\alpha + 1} \right] dz.
\end{aligned}$$

Denote the effective part of MI by

$$\Delta_{\text{MI}}(\alpha, e_1, e_2) = (1 - e_1 - e_2) \left[\frac{\alpha}{\alpha + 1} \log \frac{\alpha}{\alpha + 1} + \frac{1}{\alpha + 1} \log \frac{1}{\alpha + 1} \right]. \quad (\text{A.9})$$

We have

$$\begin{aligned}
-H(Z, \tilde{Y}) &= \sum_{j \in \{1,2\}} \int_z \mathbb{P}(Z = z, \tilde{Y} = j) \log \mathbb{P}(Z = z, \tilde{Y} = j) dz \\
&= \int_z \mathbb{P}(Z = z) \log \mathbb{P}(Z = z) dz + \int_z \mathbb{P}(Z = z) \left[\Delta_{\text{MI}}(\alpha, e_1, e_2) \right. \\
&\quad \left. + \Delta_{\text{Bias}}(\alpha, e_1, e_2) \right] dz,
\end{aligned}$$

and

$$I(Z, \tilde{Y}) = \int_z \mathbb{P}(Z = z) [\Delta_{\text{MI}}(\alpha, e_1, e_2) + \Delta_{\text{Bias}}(\alpha, e_1, e_2)] dz + H(\tilde{Y}).$$

Define $\Delta_{\text{Bias,MI}}(\alpha, e_1, e_2) = \Delta_{\text{MI}}(\alpha, e_1, e_2) + \Delta_{\text{Bias}}(\alpha, e_1, e_2)$. Then

$$\begin{aligned}
& \Delta_{\text{Bias,MI}}(\alpha, e_1, e_2) \\
&= \Delta_{\text{MI}}(\alpha, e_1, e_2) + \Delta_{\text{Bias}}(\alpha, e_1, e_2) \\
&= (1 - e_1 - e_2) \left[\frac{\alpha}{\alpha + 1} \log \frac{\alpha}{\alpha + 1} + \frac{1}{\alpha + 1} \log \frac{1}{\alpha + 1} \right] + e_2 \log \alpha - (e_1 + e_2) \log(\alpha + 1) \\
&\quad + \frac{(1 - e_1 - e_2)}{\alpha + 1} \left[\alpha \log \left(1 - e_1 + \frac{e_2}{\alpha} \right) + \log(1 - e_2 + e_1 \alpha) \right] \\
&\quad + \left[e_1 \log(1 - e_2 + e_1 \alpha) + e_2 \log \left(1 - e_1 + \frac{e_2}{\alpha} \right) \right] \\
&= (1 - e_1 - e_2) \left[\frac{\alpha}{\alpha + 1} \log \alpha + \log \frac{1}{\alpha + 1} \right] + e_2 \log \alpha + (e_1 + e_2) \log \frac{1}{\alpha + 1} + \\
&\quad \left[\frac{(1 - e_1 - e_2)\alpha}{\alpha + 1} + e_2 \right] \log \left(1 - e_1 + \frac{e_2}{\alpha} \right) + \left[\frac{(1 - e_1 - e_2)}{\alpha + 1} + e_1 \right] \log(1 - e_2 + e_1 \alpha) \\
&= \left[\frac{(1 - e_1 - e_2)\alpha}{\alpha + 1} + e_2 \right] \log \alpha + \log \frac{1}{\alpha + 1} + \\
&\quad \left[\frac{(1 - e_1 - e_2)\alpha}{\alpha + 1} + e_2 \right] \log \left(1 - e_1 + \frac{e_2}{\alpha} \right) + \left[\frac{(1 - e_1 - e_2)}{\alpha + 1} + e_1 \right] \log(1 - e_2 + e_1 \alpha) \\
&= \log \frac{1}{\alpha + 1} + \frac{1}{\alpha + 1} [(1 - e_1 - e_2)\alpha + e_2(\alpha + 1)] \log(\alpha(1 - e_1) + e_2) \\
&\quad + \frac{1}{\alpha + 1} [(1 - e_1 - e_2) + e_1(\alpha + 1)] \log(1 - e_2 + e_1 \alpha) \\
&= \left[\frac{\alpha(1 - e_1) + e_2}{\alpha + 1} + \frac{(\alpha + 1) - (\alpha(1 - e_1) + e_2)}{\alpha + 1} \right] \log \frac{1}{\alpha + 1} \\
&\quad + \frac{1}{\alpha + 1} [\alpha(1 - e_1) + e_2] \log(\alpha(1 - e_1) + e_2) \\
&\quad + \frac{1}{\alpha + 1} [(\alpha + 1) - (\alpha(1 - e_1) + e_2)] \log((\alpha + 1) - (\alpha(1 - e_1) + e_2)) \\
&= \frac{\alpha(1 - e_1) + e_2}{\alpha + 1} \log \frac{\alpha(1 - e_1) + e_2}{\alpha + 1} + \left[1 - \frac{\alpha(1 - e_1) + e_2}{\alpha + 1} \right] \log \left[1 - \frac{\alpha(1 - e_1) + e_2}{\alpha + 1} \right].
\end{aligned}$$

Note

$$\frac{\alpha(1 - e_1) + e_2}{\alpha + 1} = (1 - e_1 - e_2) \cdot \frac{\alpha}{\alpha + 1} + e_2.$$

Let $\beta = \alpha/(1 + \alpha) \in [0, 1)$. Note β is a function of z . We drop notation z for ease of

notation. Then

$$\begin{aligned}\Delta_{\text{Bias,MI}}(\beta, e_1, e_2) &= ((1 - e_1 - e_2)\beta + e_2) \log((1 - e_1 - e_2)\beta + e_2) \\ &\quad + [1 - ((1 - e_1 - e_2)\beta + e_2)] \log[1 - ((1 - e_1 - e_2)\beta + e_2)].\end{aligned}$$

The bias caused by label noise is

$$\begin{aligned}\Delta_{\text{Bias}}(\beta, e_1, e_2) &= \Delta_{\text{Bias,MI}}(\beta, e_1, e_2) - \Delta_{\text{MI}}(\beta, e_1, e_2) \\ &= ((1 - e_1 - e_2)\beta + e_2) \log((1 - e_1 - e_2)\beta + e_2) \\ &\quad + [1 - ((1 - e_1 - e_2)\beta + e_2)] \log[1 - ((1 - e_1 - e_2)\beta + e_2)] \\ &\quad - (1 - e_1 - e_2) [\beta \log \beta + (1 - \beta) \log(1 - \beta)].\end{aligned}\tag{A.10}$$

To get $\arg \max_{\beta \in [0,1]}$, we check the first derivative:

$$\frac{\partial \Delta_{\text{Bias}}(\beta, e_1, e_2)}{\partial \beta} = (1 - e_1 - e_2) \left[\log \frac{(1 - e_1 - e_2)\beta + e_2}{1 - (1 - e_1 - e_2)\beta - e_2} - \log \frac{\beta}{1 - \beta} \right].$$

Let $\partial \Delta_{\text{Bias}}(\beta, e_1, e_2) = 0$, we have

$$\beta^* = \frac{e_2}{e_1 + e_2}.$$

By checking $\partial^2 \Delta_{\text{Bias}}(\beta, e_1, e_2) / \partial \beta^2$, we can find that $\Delta_{\text{Bias}}(\beta, e_1, e_2)$ is increasing when $\beta \in [0, \beta^*]$, and decreasing when $\beta \in [\beta^*, 1]$. Thus $\beta^* = e_2 / (e_1 + e_2)$ is the global maximum and the upper bound for $\Delta_{\text{Bias}}(\beta, e_1, e_2)$ is

$$\Delta_{\text{Bias}}(\beta, e_1, e_2) \leq e_1 \log e_1 + e_2 \log e_2 - (e_1 + e_2) \log(e_1 + e_2).$$

Assume $\beta \in [\underline{\beta}, \bar{\beta}]$ in practice. The lower bound for $\Delta_{\text{Bias}}(\beta, e_1, e_2)$ is

$$\Delta_{\text{Bias}}(\beta, e_1, e_2) \geq \min(\Delta_{\text{Bias}}(\underline{\beta}, e_1, e_2), \Delta_{\text{Bias}}(\bar{\beta}, e_1, e_2)).$$

A looser bound that holds for all the possible $\beta \in [0, 1]$ is:

$$\Delta_{\text{Bias}}(\beta, e_1, e_2) \geq \min(e_1 \log e_1 + (1 - e_1) \log(1 - e_1), e_2 \log e_2 + (1 - e_2) \log(1 - e_2)).$$

Note (when $e_1 = e_2 = 0$)

$$I(Z, Y) = \int_z \mathbb{P}(Z = z) [\Delta_{\text{MI}}(\alpha, 0, 0)] dz + H(Y),$$

and $\Delta_{\text{MI}}(\beta, e_1, e_2) = (1 - e_1 - e_2)\Delta_{\text{MI}}(\alpha, 0, 0)$.

Hence (note β is actually a function of Z),

$$\begin{aligned} I(Z, \tilde{Y}) &= \int_z \mathbb{P}(Z = z) [\Delta_{\text{MI}}(\alpha, e_1, e_2) + \Delta_{\text{Bias}}(\alpha, e_1, e_2)] dz + H(\tilde{Y}) \\ &= \int_z \mathbb{P}(Z = z) [(1 - e_1 - e_2)\Delta_{\text{MI}}(\beta, 0, 0) + \Delta_{\text{Bias}}(\beta, e_1, e_2)] dz + H(\tilde{Y}) \\ &= (1 - e_1 - e_2) \int_z \mathbb{P}(Z = z) \Delta_{\text{MI}}(\beta, 0, 0) dz + \int_z \Delta_{\text{Bias}}(\beta, e_1, e_2) dz + H(\tilde{Y}) \\ &= (1 - e_1 - e_2)I(Z, Y) + \int_z \Delta_{\text{Bias}}(\beta, e_1, e_2) dz - (1 - e_1 - e_2)H(Y) + H(\tilde{Y}) \\ &= (1 - e_1 - e_2)I(Z, Y) + C(e_1, e_2, Y, \tilde{Y}) + \Delta_Z(e_1, e_2), \end{aligned}$$

where

$$\begin{aligned} C(e_1, e_2, Y, \tilde{Y}) &= \min(e_1 \log e_1 + (1 - e_1) \log(1 - e_1), e_2 \log e_2 + (1 - e_2) \log(1 - e_2)) \\ &\quad - (1 - e_1 - e_2)H(Y) + H(\tilde{Y}) \end{aligned}$$

is a constant for given Y and \tilde{Y} .

The other part is in the range $\Delta_Z(e_1, e_2) \in [0, \mathbf{Gap}_Z(e_1, e_2)]$, and

$$\begin{aligned} \mathbf{Gap}_Z(e_1, e_2) &= e_1 \log e_1 + e_2 \log e_2 - (e_1 + e_2) \log(e_1 + e_2) \\ &\quad - \min(e_1 \log e_1 + (1 - e_1) \log(1 - e_1), e_2 \log e_2 + (1 - e_2) \log(1 - e_2)). \end{aligned}$$

Note $\Delta_Z(e_1, e_2)$ may be different for Z_μ and Z_ν , $\mu \neq \nu$.

Therefore, when

$$|I_f(Z_\mu; \tilde{Y}) - I_f(Z_\nu; \tilde{Y})| > \mathbf{Gap}_Z(e_1, e_2),$$

we have

$$\text{sign} \left[I_f(Z_\mu; \tilde{Y}) - I_f(Z_\nu; \tilde{Y}) \right] = \text{sign} [I_f(Z_\mu; Y) - I_f(Z_\nu; Y)], \forall \mu \in [d] \nu \in [d].$$

Now we take a further look at the gap $\mathbf{Gap}_Z(e_1, e_2)$. Assume $e_1 \geq e_2 \Rightarrow e_2 = \delta e_1$, where $\delta \in [0, 1]$. Then $H(e_1) \leq H(e_2)$, and

$$\begin{aligned} \mathbf{Gap}_Z(e_1, e_2) &= e_1 \log e_1 + e_2 \log e_2 - (e_1 + e_2) \log(e_1 + e_2) - \min(H(e_1), H(e_2)) \\ &= e_2 \log e_2 - (e_1 + e_2) \log(e_1 + e_2) - (1 - e_1) \log(1 - e_1) \\ &= e_2 \log \frac{e_2}{e_1 + e_2} + e_1 \log \frac{1 - e_1}{e_1 + e_2} - \log(1 - e_1) \\ &= \delta e_1 \log \frac{\delta}{1 + \delta} + e_1 \log \frac{1 - e_1}{e_1(1 + \delta)} - \log(1 - e_1) \\ &= e_1 \left[\delta \log \frac{\delta}{1 + \delta} + \log \frac{1}{(1 + \delta)} \right] + e_1 \log \frac{(1 - e_1)}{e_1} - \log(1 - e_1) \\ &= e_1 [\delta \log \delta - (1 + \delta) \log(1 + \delta)] - (1 - e_1) \log(1 - e_1) - e_1 \log e_1 \\ &= e_1 [\delta \log \delta - (1 + \delta) \log(1 + \delta)] + H(e_1) \end{aligned}$$

Note: We can also roughly estimate $\delta \log \delta - (1 + \delta) \log(1 + \delta)$ by the best quadratic fit (rooted mean squared error ≈ 0.02) and get

$$\delta \log \delta - (1 + \delta) \log(1 + \delta) \approx 0.9124\delta^2 - 2.14\delta - 0.1202.$$

A.6 More Discussions for HOC-extension

A.6.1 Rationale for building on HOC

The rationale for building our analyses on HOC is described as follows. Our major concern is that the learning-based approaches usually require more effort in tuning their hyperparameters. The effect of reweighing feature variables will be entangled with the training procedure, making things more complicated. On the other hand, the training-free approach seems to be more lightweight to employ the reweighing treatment. In particular, HOC consistently achieves lower estimation error, as shown in Figure 2.3.

A.6.2 More Experiments

High-noise settings may have lower errors Table 2.5 shows the estimation error may decrease for higher noise rate settings. This observations mainly due to two reasons:

- Reason-1: The original dataset may be noisy. Notably, the original Yelp dataset contains lots of noisy reviews [129]. Now we analyze the issues caused by an originally noise dataset with a toy example.

Example: Consider a binary classification with inherent 20% noise. Define two noise settings: 1) *Low noise*: Add 10% symmetric label noise ($e_1 = e_2 = 0.1$). 2) *High noise*: Add 40% symmetric label noise ($e_1 = e_2 = 0.4$). For the low noise setting, the real average noise rate is:

$$\epsilon_{\text{low, real}} = 0.1 \times 0.8 + 0.2 \times 0.9 = 0.26.$$

For the high noise setting, the real average noise rate is:

$$e_{\text{high, real}} = 0.4 \times 0.8 + 0.2 \times 0.6 = 0.44.$$

Note $e_{\text{low, synthetic}} = 0.1$ and $e_{\text{high, synthetic}} = 0.4$. Thus the gap between the real noise rate and the synthetic noise rate is

$$e_{\text{low, real}} - e_{\text{low, synthetic}} = 0.26 - 0.1 = 0.16, \quad e_{\text{high, real}} - e_{\text{high, synthetic}} = 0.44 - 0.4 = 0.04.$$

Therefore, with inherent label noise exists, the perfectly estimated \mathbf{T} will have an error of 0.16 for the low-noise setting and an error of 0.04 for the high-noise setting, which accounts for our current observation.

- Reason-2: The tolerance of noise rates for different settings are different. Consider a binary classification with symmetric label noise, i.e., $e_1 = e_2 = e$. Let the random guess be $e_1 = e_2 = 0.5$. Define the estimation error caused by the random guess as the tolerance. Thus the tolerances when $e = 0.1$ and $e = 0.4$ are 0.4 and 0.1, respectively. From this aspect, an error of 0.1 will not destroy the low-noise case since

$$|\hat{e}_{1,\text{low}} - e_{1,\text{low}}| = 0.1 \Rightarrow \hat{e}_{1,\text{low}} = 0.2.$$

But an error of 0.1 may destroy the high-noise case since

$$|\hat{e}_{1,\text{high}} - e_{1,\text{high}}| = 0.1 \Rightarrow \hat{e}_{1,\text{high}} = 0.3 \quad \text{or} \quad 0.5.$$

Therefore, although the error of high-noise settings seems low, it may cause severe problems.

Table A.2: The calibrated estimation error ($\times 100$) on Yelp-5.

TEXT DATASETS		METHOD	
$(d, [N_1, \dots, N_K])$	NOISE RATE	OURS- A -KL	OURS- A -TV
	LOW	3.56	4.01
YELP-5 (BERT)	MEDIUM	3.46	2.59
(768, [130K \times 5])	HIGH	8.59	8.56

A preliminary test on calibrating inherent errors We do the following experiment to help explain Reason-1 by calibrating the inherent label noise in Yelp-5. Note the label noise accumulation follows:

$$\mathbf{T}_{\text{real}} = \mathbf{T}_{\text{org}} \mathbf{T}_{\text{synthetic}},$$

where $\mathbf{T}_{\text{synthetic}} = \mathbf{T}$. If we know \mathbf{T}_{org} , we can calibrate $\mathbf{T}_{\text{synthetic}}$ and evaluate the error by $\text{Error}(\mathbf{T}_{\text{org}} \mathbf{T}, \mathbf{T}_{\text{org}} \hat{\mathbf{T}})$. Unfortunately, we cannot find the ground truth \mathbf{T}_{org} for Yelp-5. For a preliminary test, we estimate \mathbf{T}_{org} by applying OURS-**A**-KL on the original Yelp dataset. We show the calibrated error in Table A.2, where we can find the high-noise settings are indeed more challenging (higher error) compared with the low-noise setting. The average noise rate follows $e = 1/(1 + r/\sqrt{K-1})$. Particularly, we have: LOW: $r = 8$. MEDIUM: $r = 4$. HIGH: $r = 1.5$.

Appendix B

More Details for Charter 3

Appendix B provides more theoretical details and experiment settings related to CORES² (Section 3.1) and SimiFeat (Section 3.2), which is organized as follows.

- Section B.1 shows the proof for theorems in CORES².
- Section B.2 shows the proof for theorems in SimiFeat.
- Section B.3 shows more experiment settings in SimiFeat.

B.1 CORES²: Proof for Theorems

In this section, we firstly present the proof for Theorem 7 (our main theorem) in Section B.1.1, which provides a generic machinery for anatomizing noisy datasets. Then we will respectively prove Theorem 5 in Section B.1.2, Theorem 6 in Section B.1.3, and Theorem 8 in Section B.1.4 according to the order they appear.

B.1.1 Proof for Theorem 7

Proof. The expected form of traditional CE loss on noisy distribution $\tilde{\mathcal{D}}$ can be written

as

$$\begin{aligned}
& \mathbb{E}_{\tilde{\mathcal{D}}}[\ell(f(X), \tilde{Y})] \\
&= \sum_{j \in [K]} \sum_{i \in [K]} \mathbb{P}(Y = i) \mathbb{E}_{\mathcal{D}|Y=i}[T_{ij}(X) \ell(f(X), j)] \\
&= \sum_{j \in [K]} \sum_{i \in [K]} \mathbb{P}(Y = i) T_{ij} \mathbb{E}_{\mathcal{D}|Y=i}[\ell(f(X), j)] \\
&\quad + \sum_{j \in [K]} \sum_{i \in [K]} \mathbb{P}(Y = i) \text{Cov}_{\mathcal{D}|Y=i}(T_{ij}(X), \ell(f(X), j)).
\end{aligned}$$

The first term could be transformed as:

$$\begin{aligned}
& \sum_{j \in [K]} \sum_{i \in [K]} \mathbb{P}(Y = i) T_{ij} \mathbb{E}_{\mathcal{D}|Y=i}[\ell(f(X), j)] \\
&= \sum_{j \in [K]} \left[T_{jj} \mathbb{P}(Y = j) \mathbb{E}_{\mathcal{D}|Y=j}[\ell(f(X), j)] + \sum_{i \in [K], i \neq j} T_{ij} \mathbb{P}(Y = i) \mathbb{E}_{\mathcal{D}|Y=i}[\ell(f(X), j)] \right] \\
&= \underline{T} \mathbb{E}_{\mathcal{D}}[\ell(f(X), Y)] + \bar{\Delta} \mathbb{E}_{\mathcal{D}_{\Delta}}[\ell(f(X), Y)] \\
&\quad + \sum_{j \in [K]} \sum_{i \in [K], i \neq j} T_{ij} \mathbb{P}(Y = i) \mathbb{E}_{\mathcal{D}|Y=i}[\ell(f(X), j)],
\end{aligned}$$

where

$$\underline{T} := \min_{j \in [K]} T_{jj}, \quad \bar{\Delta} := \sum_{j \in [K]} \Delta_j \mathbb{P}(Y = j), \quad \Delta_j := T_{jj} - \underline{T},$$

and

$$\mathbb{E}_{\mathcal{D}_{\Delta}}[\ell(f(X), Y)] := \begin{cases} \sum_{j \in [K]} \frac{\Delta_j \mathbb{P}(Y=j)}{\bar{\Delta}} \mathbb{E}_{\mathcal{D}|Y=j}[\ell(f(X), j)], & \text{if } \bar{\Delta} > 0, \\ 0 & \text{if } \bar{\Delta} = 0. \end{cases}$$

Then

$$\begin{aligned}
& \mathbb{E}_{\tilde{\mathcal{D}}}[l(f(X), \tilde{Y})] \\
&= \underline{T}\mathbb{E}_{\mathcal{D}}[l(f(X), Y)] + \bar{\Delta}\mathbb{E}_{\mathcal{D}_{\Delta}}[l(f(X), Y)] + \sum_{j \in [K]} \sum_{i \in [K], i \neq j} T_{ij} \mathbb{P}(Y = i) \mathbb{E}_{\mathcal{D}|Y=i}[l(f(X), j)], \\
&+ \sum_{j \in [K]} \sum_{i \in [K]} \mathbb{P}(Y = i) \text{Cov}_{\mathcal{D}|Y=i}(T_{ij}(X), l(f(X), j)) \\
&= \underline{T}\mathbb{E}_{\mathcal{D}}[l(f(X), Y)] + \bar{\Delta}\mathbb{E}_{\mathcal{D}_{\Delta}}[l(f(X), Y)] + \sum_{j \in [K]} \sum_{i \in [K], i \neq j} T_{ij} \mathbb{P}(Y = i) \mathbb{E}_{\mathcal{D}|Y=i}[l(f(X), j)], \\
&+ \sum_{j \in [K]} \sum_{i \in [K], i \neq j} \mathbb{P}(Y = i) \mathbb{E}_{\mathcal{D}|Y=i}[(T_{ij}(X) - T_{ij})(l(f(X), j) - \mathbb{E}_{\mathcal{D}|Y=i}[l(f(X), j)])] \\
&+ \sum_{j \in [K]} \mathbb{P}(Y = j) \mathbb{E}_{\mathcal{D}|Y=j}[(T_{jj}(X) - T_{jj})(l(f(X), j) - \mathbb{E}_{\mathcal{D}|Y=j}[l(f(X), j)])] \\
&= \underline{T}\mathbb{E}_{\mathcal{D}}[l(f(X), Y)] + \bar{\Delta}\mathbb{E}_{\mathcal{D}_{\Delta}}[l(f(X), Y)] \\
&+ \sum_{j \in [K]} \sum_{i \in [K], i \neq j} \mathbb{P}(Y = i) \\
&\quad \cdot \mathbb{E}_{\mathcal{D}|Y=i}[(T_{ij}(X) - T_{ij})(l(f(X), j) - \mathbb{E}_{\mathcal{D}|Y=i}[l(f(X), j)]) + T_{ij}l(f(X), j)] \\
&+ \sum_{j \in [K]} \mathbb{P}(Y = j) \mathbb{E}_{\mathcal{D}|Y=j}[(T_{jj}(X) - T_{jj})(l(f(X), j) - \mathbb{E}_{\mathcal{D}|Y=j}[l(f(X), j)])] \\
&= \underline{T}\mathbb{E}_{\mathcal{D}}[l(f(X), Y)] + \bar{\Delta}\mathbb{E}_{\mathcal{D}_{\Delta}}[l(f(X), Y)] \\
&\quad + \sum_{j \in [K]} \sum_{i \in [K], i \neq j} \mathbb{P}(Y = i) \mathbb{E}_{\mathcal{D}|Y=i}[T_{ij}(X)l(f(X), j)] \\
&\quad + \sum_{j \in [K]} \mathbb{P}(Y = j) \mathbb{E}_{\mathcal{D}|Y=j}[(T_{jj}(X) - T_{jj})l(f(X), j)] \\
&= \underline{T}\mathbb{E}_{\mathcal{D}}[l(f(X), Y)] + \bar{\Delta}\mathbb{E}_{\mathcal{D}_{\Delta}}[l(f(X), Y)] + \sum_{j \in [K]} \sum_{i \in [K]} \mathbb{P}(Y = i) \mathbb{E}_{\mathcal{D}|Y=i}[U_{ij}(X)l(f(X), j)],
\end{aligned}$$

where

$$U_{ij}(X) = T_{ij}(X), \forall i \neq j, \quad U_{jj}(X) = T_{jj}(X) - T_{jj}.$$

The expected form of ℓ_{CR} on noisy distribution $\tilde{\mathcal{D}}$ can be written as

$$\begin{aligned}
\mathbb{E}_{\tilde{\mathcal{D}}}[\ell_{\text{CR}}(f(x_i))] &= -\beta \mathbb{E}_{\tilde{\mathcal{D}}} \left[\mathbb{E}_{\mathcal{D}_{\tilde{Y}|\tilde{D}}}[\ell(f(x_i), \tilde{Y})] \right] \\
&= -\beta \int_{\tilde{\mathcal{D}}} \left[\mathbb{P}(\tilde{D}) \mathbb{E}_{\mathcal{D}_{\tilde{Y}|\tilde{D}}}[\ell(f(x_i), \tilde{Y})] \right] \\
&= -\beta \sum_{j \in [K]} \mathbb{P}(\tilde{Y} = j) \mathbb{E}_{\mathcal{D}_X}[\ell(f(x_i), j)] \\
&= - \sum_{j \in [K]} \sum_{i \in [K]} \mathbb{P}(Y = i) \mathbb{E}_{\mathcal{D}|Y=i}[\beta \mathbb{P}(\tilde{Y} = j) \ell(f(x_i), j)].
\end{aligned}$$

Thus the expected form of the new regularized loss is

$$\begin{aligned}
\mathbb{E}_{\tilde{\mathcal{D}}} \left[\ell(f(X), \tilde{Y}) + \ell_{\text{CR}}(f(x_i)) \right] &= \underline{T} \mathbb{E}_{\mathcal{D}}[\ell(f(X), Y)] + \bar{\Delta} \mathbb{E}_{\mathcal{D}_{\Delta}}[\ell(f(X), Y)] \\
+ \sum_{j \in [K]} \sum_{i \in [K]} \mathbb{P}(Y = i) \mathbb{E}_{\mathcal{D}|Y=i} &[(U_{ij}(X) - \beta \mathbb{P}(\tilde{Y} = j)) \ell(f(X), j)].
\end{aligned} \tag{B.1}$$

□

B.1.2 Proof for Theorem 5

Proof. Let $\ell(\cdot)$ be the CE loss. Note this proof does not rely on whether the data distribution is clean or not. We use \mathcal{D} to denote any data distribution and D to denote the corresponding dataset. This notation applies only to this proof. For any data distribution \mathcal{D} , we have

$$\begin{aligned}
&\mathbb{E}_{\mathcal{D}} \left[\ell(f(X), Y) - \mathbb{E}_{\mathcal{D}_{Y|D}}[\ell(f(x_n), Y)] \right] \\
&= \mathbb{E}_{\mathcal{D}}[\ell(f(X), Y)] - \mathbb{E}_{\mathcal{D}_Y}[\mathbb{E}_{\mathcal{D}_X}[\ell(f(X), Y)]] \\
&= - \int_{\mathcal{D}_X} dx \sum_{y \in [K]} \mathbb{P}(x, y) \ln f_x[y] + \int_{\mathcal{D}_X} dx \sum_{y \in [K]} \mathbb{P}(x) \mathbb{P}(y) \ln f_x[y] \\
&= - \int_{\mathcal{D}_X} dx \sum_{y \in [K]} \ln f_x[y] [\mathbb{P}(x, y) - \mathbb{P}(x) \mathbb{P}(y)].
\end{aligned}$$

The dynamical analyses are based on the following three assumptions:

- A1. The model capacity is infinite (i.e., it can realize arbitrary variation).
- A2. The model is updated using the gradient descent algorithm (i.e. updates follow the direction of decreasing $\mathbb{E}_{\mathcal{D}} [\ell(f(X), Y)] - \mathbb{E}_{\mathcal{D}_Y} [\mathbb{E}_{\mathcal{D}_X} [\ell(f(X), Y)]]$).
- A3. The derivative of network function $\frac{\partial f(x;w)}{\partial w_i}$ is smooth (i.e. the network function has no singular point), where w_i 's are model parameters.

Denote the variations of $f_x[y]$ during one gradient descent update by $\Delta_y(x)$.

From Lemma 10, it can be explicitly written as

$$\Delta_y(x) = f_x[y] \cdot \eta \int_{\mathcal{D}_X} dx' \sum_{y' \in [K]} [\mathbb{P}(x', y') - \mathbb{P}(x')\mathbb{P}(y')] \sum_{i \in [K]} G_i(x, y) G_i(x', y'), \quad (\text{B.2})$$

where η is the learning rate,

$$G_i(x, y) = -\frac{\partial g_y(x)}{\partial w_i} + \sum_{y' \in [K]} f_x[y'] \frac{\partial g_{y'}(x)}{\partial w_i},$$

and $g_y(x)$ is the network output before the softmax activation. i.e.

$$f_x[y] = \frac{\exp(g_y(x))}{\sum_{y' \in [K]} \exp(g_{y'}(x))}.$$

With $\Delta_y(x)$, the variation of the regularized loss is

$$\Delta \mathbb{E}_{\mathcal{D}} [\ell(f(X), Y) + \ell_{\text{CR}}] = - \int_{\mathcal{D}_X} dx \mathbb{P}(x) \sum_{y \in [K]} \Delta_y(x) \frac{\mathbb{P}(y|x) - \mathbb{P}(y)}{f_x[y]}. \quad (\text{B.3})$$

If the training reaches a steady state (a.k.a. local optimum), we have

$$\Delta \mathbb{E}_{\mathcal{D}} [\ell(f(X), Y) + \ell_{\text{CR}}] = 0.$$

To check the property of this variation, consider the following example. For a particular x_0 , define

$$F(x_0) := \sum_{y \in [K]} \Delta_y(x_0) \frac{\mathbb{P}(y|x_0) - \mathbb{P}(y)}{f_{x_0}[y]}.$$

Split the labels y into the following two sets (without loss of generality, we ignore the $\mathbb{P}(y|x_0) - \mathbb{P}(y) = 0$ cases):

$$\mathcal{Y}_{x_0;-} = \{y : \mathbb{P}(y|x_0) - \mathbb{P}(y) < 0\}$$

and

$$\mathcal{Y}_{x_0;+} = \{y : \mathbb{P}(y|x_0) - \mathbb{P}(y) > 0\}.$$

By assigning $\Delta_y(x_0) = a_y < 0, \forall y \in \mathcal{Y}_{x_0;-}$ and $\Delta_y(x_0) = b_y > 0, \forall y \in \mathcal{Y}_{x_0;+}$, one finds $F(x_0) > 0$ since $f_{x_0}[y] > 0$. Note we have an extra constraint $\sum_y \Delta_y(x_0) = 0$ to ensure $\sum_{y \in [K]} f_{x_0}[y] = 1$ after update. It is easy to check our assigned a_y and b_y could maintain this constraint by introducing a weight N_{ab} to scale b'_y as follows.

$$\sum_{y \in \mathcal{Y}_-} a_y + N_{ab} \sum_{y \in \mathcal{Y}_+} b'_y = 0, \quad b_y = N_{ab} b'_y.$$

Let $B_\epsilon(x_0)$ be a ϵ -neighbourhood of x_0 . Since $f_x[y]$ is continuous, we can set $\Delta_y(x) = \frac{1}{2}(1 + \cos \frac{\pi \|x-x_0\|}{\epsilon})\Delta_y(x_0), \forall x \in B_\epsilon(x_0)$ and 0 otherwise. The coefficient $\frac{1}{2}(1 + \cos \frac{\pi \|x-x_0\|}{\epsilon})$ is added so that the continuity of $f_x[y]$ preserves. This choice will lead to $\Delta \mathbb{E}_{\mathcal{D}} [\ell(f(X), Y) + \ell_{\text{CR}}] < 0$. Therefore, for any $\ell_{\text{CA}}(f(x_n), y_n)$ with solution $f_{x_n}[i] > 0, \forall i \in [K]$, we can always find a decreasing direction, indicating the solution is not (steady) locally optimal. Note \mathcal{D} can be any distribution in this proof. Thus the result holds for the noisy distribution $\tilde{\mathcal{D}}$. \square

Lemma 10.

$$\Delta_y(x) = f_x[y] \cdot \eta \int_{\mathcal{D}_X} dx' \sum_{y' \in [K]} [\mathbb{P}(x', y') - \mathbb{P}(x')\mathbb{P}(y')] \sum_{i \in [K]} G_i(x, y) G_i(x', y').$$

Proof. We need to take into account the actual form of activation function, i.e., the softmax function, as well as the SGD algorithm to demonstrate the correctness of this lemma. The variation $\Delta_{y_0}(x_0)$ is caused by the change in network parameters $\{w_i\}$, i.e.,

$$\Delta_{y_0}(x_0) = \sum_{i \in [K]} \frac{\partial f_{x_0}[y_0]}{\partial w_i} \delta w_i, \quad (\text{B.4})$$

where δw_i are determined by the SGD algorithm

$$\begin{aligned} \delta w_i &= -\eta \frac{\partial \mathbb{E}_{\mathcal{D}} [\ell(f(X), Y) + \ell_{\text{CR}}]}{\partial w_i} \\ &= \eta \int \frac{\mathbb{P}(x, y) - \mathbb{P}(x)\mathbb{P}(y)}{f_x[y]} \frac{\partial f_x[y]}{\partial w_i}. \end{aligned}$$

Plugging back to (B.4) yields

$$\Delta_{y_0}(x_0) = \eta \int \frac{\mathbb{P}(x, y) - \mathbb{P}(x)\mathbb{P}(y)}{f_x[y]} \sum_{i \in [K]} \frac{\partial f_{x_0}[y_0]}{\partial w_i} \frac{\partial f_x[y]}{\partial w_i}.$$

To proceed, we need to expand $\frac{\partial f_x[y]}{\partial w_i}$. Taking into account the activation function, one has

$$f_x[y] = \frac{\exp(g_y(x))}{\sum_{y' \in [K]} \exp(g_{y'}(x))},$$

where $g_y(x)$ refers to the network output before passed to the activation function. Recall that, by our assumption, derivatives $\frac{\partial f(x; w)}{\partial w_i}$ are not singular. Now we have

$$\begin{aligned} \frac{\partial f_x[y]}{\partial w_i} &= \frac{\partial e^{-g_y(x)}}{\partial w_i} \frac{1}{\sum_{y' \in [K]} e^{-g_{y'}(x)}} + e^{-g_y(x)} \frac{\partial}{\partial w_i} \left(\frac{1}{\sum_{y' \in [K]} e^{-g_{y'}(x)}} \right) \\ &= \frac{-e^{-g_y(x)}}{\sum_{y' \in [K]} e^{-g_{y'}(x)}} \frac{\partial g_y(x)}{\partial w_i} + \frac{e^{-g_y(x)}}{\left(\sum_{y'' \in [K]} e^{-g_{y''}(x)} \right)^2} \sum_{y' \in [K]} e^{-g_{y'}(x)} \frac{\partial g_{y'}(x)}{\partial w_i} \\ &= f_x[y] \left[-\frac{\partial g_y(x)}{\partial w_i} + \sum_{y' \in [K]} f_x[y'] \frac{\partial g_{y'}(x)}{\partial w_i} \right]. \end{aligned}$$

For simplicity, we can rewrite the above result as

$$\frac{\partial f_x[y]}{\partial w_i} = f_x[y]G_i(x, y),$$

where

$$G_i(x, y) := -\frac{\partial g_y(x)}{\partial w_i} + \sum_{y'} f_x[y'] \frac{\partial g_{y'}(x)}{\partial w_i}$$

is a smooth function.

Combining all the above gives $\Delta_{y_0}(x_0)$ as follows.

$$\Delta_{y_0}(x_0) = f_{x_0}[y_0] \cdot \eta \int_{x, y} [\mathbb{P}(x, y) - \mathbb{P}(x)\mathbb{P}(y)] \sum_i G_i(x_0, y_0) G_i(x, y)$$

□

B.1.3 Proof for Theorem 6

Proof. Let y_n be the true label corresponding to feature x_n . For a clean sample, we have $\tilde{y}_n = y_n$. Consider an arbitrary DNN model f . With the CE loss, we have $\ell(f(x_n), y_n) = -\ln(f_{x_n}[y_n])$. According to Equation (3.4) in the paper, the necessary and sufficient condition of $v_n > 0$ is

$$\begin{aligned} \ell(f(x_n), \tilde{y}_n) + \ell_{\text{CR}}(f(x_n)) < \alpha_n &\Leftrightarrow -\ln(f_{x_n}[y_n]) < -\frac{1}{K} \sum_{y \in [K]} \ln(f_{x_n}[y]) \\ &\Leftrightarrow -\ln(f_{x_n}[y_n]) < -\frac{1}{K-1} \sum_{y \in [K], y \neq y_n} \ln(f_{x_n}[y]). \end{aligned}$$

By Jensen's inequality we have

$$-\ln\left(\frac{1 - f_{x_n}[y_n]}{K-1}\right) = -\ln\left(\frac{\sum_{y \in [K], y \neq y_n} f_{x_n}[y]}{K-1}\right) \leq -\frac{1}{K-1} \sum_{y \in [K], y \neq y_n} \ln(f_{x_n}[y]).$$

Therefore, when (sufficient condition)

$$-\ln(f_{x_n}[y_n]) < -\ln\left(\frac{1-f_{x_n}[y_n]}{K-1}\right) \Leftrightarrow f_{x_n}[y_n] > \frac{1}{K},$$

we have $v_n > 0$. Inequality $f_{x_n}[y_n] > \frac{1}{K}$ indicates the model prediction is better than random guess. □

B.1.4 Proof for Theorem 8

Before proving Theorem 8, we need to show the effect of adding Term-2 to Term-1 in (3.5). Let $\epsilon_X < 0.5$ be the measure of separation among classes w.r.t feature X in distribution \mathcal{D} , i.e., $\mathbb{P}(Y = Y^*|X) = 1 - \epsilon_X$, $(X, Y) \sim \mathcal{D}$, where $Y^* := \arg \max_{i \in [K]} \mathbb{P}(Y = i|X)$ is the Bayes optimal label. Let \mathcal{D}' be the shifted distribution by adding Term-2 to Term-1 and Y' be the shifted label. Then $\mathbb{P}(X|Y) = \mathbb{P}(X|Y')$, $\forall (X, Y) \sim \mathcal{D}, (X, Y') \sim \mathcal{D}'$ but $\mathbb{P}(Y')$ may be different from $\mathbb{P}(Y)$. Lemma 11 shows the invariant property of this label shift.

Lemma 11. *Label shift does not change the Bayes optimal label of feature X when $\epsilon_X < \min_{\forall i, j \in [K]} \left(\frac{T_{jj}}{T_{ii} + T_{jj}}\right)$.*

Proof. Consider the shifted distribution \mathcal{D}' . Let

$$\underline{T}\mathbb{E}_{\mathcal{D}}[\ell(f(X), Y)] + \bar{\Delta}\mathbb{E}_{\mathcal{D}_{\Delta}}[\ell(f(X), Y)] = C\mathbb{E}_{\mathcal{D}'}[\ell(f(X), Y)],$$

where

$$\mathbb{E}_{\mathcal{D}'}[\ell(f(X), Y)] := \sum_{j \in [K]} \mathbb{P}(Y' = j) \mathbb{E}_{\mathcal{D}'|Y'=j}[\ell(f(X), j)],$$

and

$$\mathbb{P}(Y' = j) := \frac{T_{jj}\mathbb{P}(Y = j)}{C},$$

where $C := \sum_{j \in [K]} T_{jj}\mathbb{P}(Y = j)$ is a constant for normalization. For each possible $Y = i$, we have $\mathbb{P}(Y = i|X) \in [0, \epsilon_X] \cup \{1 - \epsilon_X\}$, $\epsilon_X < 0.5$. Thus

$$\mathbb{P}(X|Y = i) = \frac{\mathbb{P}(Y = i|X)\mathbb{P}(X)}{\mathbb{P}(Y = i)} \in [0, \frac{\epsilon_X\mathbb{P}(X)}{\mathbb{P}(Y = i)}] \cup \{\frac{\mathbb{P}(X)(1 - \epsilon_X)}{\mathbb{P}(Y = i)}\}.$$

Compare \mathcal{D}' and \mathcal{D} , we know there is a label shift [3, 160], where $\mathbb{P}(X|Y = i) = \mathbb{P}(X|Y' = i)$ but $\mathbb{P}(Y)$ and $\mathbb{P}(Y')$ may be different. To ensure the label shift does not change the Bayes optimal label, we need

$$Y^* = \arg \max_{i \in [K]} \mathbb{P}(Y' = i|X) = \arg \max_{i \in [K]} \frac{\mathbb{P}(X|Y' = i)\mathbb{P}(Y' = i)}{\mathbb{P}(X)}, (X, Y') \sim \mathcal{D}.$$

One sufficient condition is

$$\frac{\epsilon_X\mathbb{P}(Y' = i)}{\mathbb{P}(Y = i)} < \frac{(1 - \epsilon_X)\mathbb{P}(Y' = j)}{\mathbb{P}(Y = j)} \Rightarrow \epsilon_X < \min_{\forall i, j \in [K]} \left(\frac{T_{jj}}{T_{ii} + T_{jj}} \right)$$

□

With Lemma 11, Assumption 3, and Assumption 4, we present the proof for Theorem 8 as follows.

Proof. It is easy to check $\epsilon_X = 0, \forall X \sim \mathcal{D}_X$ when Assumption 3 holds. Thus adding Term-2 to Term-1 in (3.5) does not change the Bayes optimal label. With Assumption 3, the Bayes optimal classifier on the clean distribution should satisfy $f^*(X)[Y] = 1, \forall (X, Y) \sim \mathcal{D}$. On one hand, when $\beta \geq \max_{i, j \in [K], X \sim \mathcal{D}_X} U_{ij}(X)/\mathbb{P}(\tilde{Y} = j)$, we have

$$\beta_{ij}(X) := U_{ij}(X) - \beta\mathbb{P}(\tilde{Y} = j) \leq 0, \forall i, j \in [K], X \sim \mathcal{D}_X.$$

In this case, minimizing the regularization term results in confident predictions. On the other hand, to make it unbiased to clean results, β could not be arbitrarily large. We need to find the upper bound on β such that f^* also minimizes the loss defined in the latter regularization term. Assume there is no loss on confident true predictions and there is one miss-prediction on example $(x_n, y_n = j_1)$, i.e., the prediction changes from the Bayes optimal prediction $f_{x_n}[j_1] = 1$ to $f_{x_n}[j_2] = 1, j_2 \neq j_1$. Compared to the optimal one, the first two terms in the right side of (3.5) is increased by $T_{j_1, j_1} \ell_0$, where $\ell_0 > 0$ is the regret of one confident wrong prediction. Accordingly, the last term in the right side of (3.5) is increased by $(\beta_{j_1, j_1}(X) - \beta_{j_1, j_2}(X))\ell_0$. It is supposed that

$$T_{j_1, j_1} \ell_0 + (\beta_{j_1, j_1}(x_n) - \beta_{j_1, j_2}(x_n))\ell_0 \geq 0, \forall j_1, j_2 \in [K],$$

which is equivalent to

$$\beta(\mathbb{P}(\tilde{Y} = j_1) - \mathbb{P}(\tilde{Y} = j_2)) \leq T_{j_1, j_1}(x_n) - T_{j_1, j_2}(x_n), \forall j_1, j_2 \in [K].$$

Thus

$$\beta \leq \min_{\mathbb{P}(\tilde{Y}=j_1) > \mathbb{P}(\tilde{Y}=j_2), X \sim \mathcal{D}_X} \frac{T_{j_1, j_1}(X) - T_{j_1, j_2}(X)}{\mathbb{P}(\tilde{Y} = j_1) - \mathbb{P}(\tilde{Y} = j_2)}.$$

By mathematical inductions, it can be generalized to the case with multiple miss-predictions in the CE term. □

B.2 SimiFeat: Theoretical Analyses

B.2.1 Proof for Proposition 1

Now we derive a lower bound for the probability of getting true detection with majority vote:

$$\begin{aligned}
\mathbb{P}(\text{Vote is correct}|k) &\geq (1 - \delta_k) \cdot \left[p \sum_{l=0}^{\lceil (k+1)/2 \rceil - 1} \binom{k+1}{l} e^l (1-e)^{k+1-l} \right. \\
&\quad \left. + (1-p) \sum_{l=0}^{\lceil (k+1)/2 \rceil - 1} \binom{k+1}{l} e^l (1-e)^{k+1-l} \right] \\
&= (1 - \delta_k) \cdot \left[p \cdot I_{1-e}(k+1-k', k'+1) \right. \\
&\quad \left. + (1-p) \cdot I_{1-e}(k+1-k', k'+1) \right]
\end{aligned}$$

where $I_{1-e}(k+1-k', k'+1)$ is the regularized incomplete beta function defined as

$$I_{1-e}(k+1-k', k'+1) = (k+1-k') \binom{k+1}{k'} \int_0^{1-e} t^{k-k'} (1-t)^{k'} dt,$$

and $k' = \lceil (k+1)/2 \rceil - 1$.

B.2.2 Proof for Theorem 9

Proof. Now we derive the worst-case error bound. We first repeat the notations defined in Section 3.2.2.2 as follows.

Denote random variable S by the score of each instance being clean. A higher score S indicates the instance is more likely to be clean. Denote the score of a true/false

instance by

$$S_{n,j}^{\text{true}} := \text{Score}(\hat{\mathbf{y}}_n, j) \mid (\tilde{y}_n = y_n = j),$$

$$S_{n',j}^{\text{false}} := \text{Score}(\hat{\mathbf{y}}_{n'}, j) \mid (\tilde{y}_{n'} = j, y_{n'} \neq j).$$

Both are scalars. Then for instances in \mathcal{N}_j , we have two set of random variables $\mathbb{S}_j^{\text{true}} := \{S_{n,j}^{\text{true}} \mid n \in \mathcal{N}_j, \tilde{y}_n = y_n = j\}$ and $\mathbb{S}_j^{\text{false}} := \{S_{n',j}^{\text{false}} \mid n' \in \mathcal{N}_j, \tilde{y}_{n'} = j, y_{n'} \neq j\}$. Recall $\mathcal{N}_j := \{n \mid \tilde{y}_n = j\}$ are the set of indices that correspond to noisy class j . Intuitively, the score $S_{n,j}^{\text{true}}$ should be greater than $S_{n',j}^{\text{false}}$. Suppose their means, which depend on noise rates, are bounded, i.e.,

$$\mathbb{E}[S_{n,j}^{\text{true}}] \geq \mu_j^{\text{true}}, \quad \mathbb{E}[S_{n',j}^{\text{false}}] \leq \mu_j^{\text{false}}$$

for all feasible n, n' . Assume there exists a feasible v such that both S_j^{true} and S_j^{false} follow sub-Gaussian distributions with variance proxy $\frac{\Delta^2}{2v}$ [21, 233] such that:

$$\mathbb{P}(\mu_j^{\text{true}} - S_{n,j}^{\text{true}} \geq t) \leq e^{-\frac{vt^2}{\Delta^2}}, \mathbb{P}(S_{n',j}^{\text{false}} - \mu_j^{\text{false}} \geq t) \leq e^{-\frac{vt^2}{\Delta^2}},$$

and the probability density satisfies $\mathbb{P}(S_j^{\text{true}} = \mu_j^{\text{true}}) = \mathbb{P}(S_j^{\text{false}} = \mu_j^{\text{false}}) = 1/\Delta$, where $1/\Delta$ is the ‘‘height’’ of both distributions, v is the decay rate of tails. Let N_j^- (N_j^+) be the number of indices in $\mathbb{S}_j^{\text{false}}$ ($\mathbb{S}_j^{\text{true}}$).

For ease of notations, we omit the subscript j in this proof since the detection is performed on each j individually.

Let S^{false} (S^{true}) be an arbitrary random variable in $\mathbb{S}^{\text{false}}$ (\mathbb{S}^{true}). Denote the order statistics of random variables in set $\mathbb{S}^{\text{false}}$ by $S_{(1)}^{\text{false}}, \dots, S_{(N^-)}^{\text{false}}$, where $S_{(1)}^{\text{false}}$ is the

smallest order statistic and $S_{(N^-)}^{\text{false}}$ is the largest order statistic. The following lemma motivates the performance of the rank-based method.

Lemma 12. *The F_1 -score of detecting corrupted labels in \mathcal{N}_j by the rank-based method will be no less than $1 - \alpha/N^-$ when the true probability $\mathbb{P}(Y = j|\tilde{Y} = j)$ is known and $S_{(N^-)}^{\text{false}} < S_{(\alpha+1)}^{\text{true}}$.*

Lemma 12 connects the upper bound for the number of wrongly detected corrupted instances with order statistics. There are two cases that can cause detection errors:

Case-1:

$0 \leq \mu^{\text{true}} - S^{\text{true}} < \Delta$ and $0 \leq S^{\text{false}} - \mu^{\text{false}} < \Delta$: at most α errors when $S_{(N^-)}^{\text{false}} < S_{(\alpha+1)}^{\text{true}}$.

and **Case-2:**

$\mu^{\text{true}} - S^{\text{true}} \geq \Delta$ or $S^{\text{false}} - \mu^{\text{false}} \geq \Delta$: at most $\max(N_-, N_+)$ errors

We analyze each case as follows.

Case-1: When Case-1 holds, we have

$$\mathbb{P}(\mu^{\text{true}} - S^{\text{true}} = x) \leq 1/\Delta, x \in [0, \Delta]$$

and

$$\mathbb{P}(S^{\text{false}} - \mu^{\text{true}} = x) \leq 1/\Delta, x \in [0, \Delta].$$

The above two inequalities show that the left tail of S^{true} and the right tail of S^{false} can be upper bounded by uniform distributions. Denote the corresponding uniform

distribution by $U^{\text{true}} \sim \text{Unif}(\mu^{\text{true}} - \Delta, \mu^{\text{true}})$ and $U^{\text{false}} \sim \text{Unif}(\mu^{\text{false}}, \mu^{\text{false}} + \Delta)$.

With true $\mathbb{P}(Y = j | \tilde{Y} = j)$, the detection errors only exist in the cases when the left tail of S^{true} and the right tail of S^{false} are overlapped. When the tails are upper bounded by uniform distributions, we have

$$\begin{aligned} \mathbb{P}(S_{(N^-)}^{\text{false}} < S_{(\alpha+1)}^{\text{true}}) &\geq \mathbb{P}(U_{(N^-)}^{\text{false}} < U_{(\alpha+1)}^{\text{true}}) \\ &= \mathbb{P}\left(\left[U^{\text{false}} - \mu^{\text{false}}\right]_{(N^-)} + \mu^{\text{false}} < \left[U^{\text{true}} - (\mu^{\text{true}} - \Delta)\right]_{(\alpha+1)} + (\mu^{\text{true}} - \Delta)\right) \\ &= \mathbb{P}\left(\left[U^{\text{false}} - \mu^{\text{false}}\right]_{(N^-)} - \left[U^{\text{true}} - (\mu^{\text{true}} - \Delta)\right]_{(\alpha+1)} < \mu^{\text{true}} - \mu^{\text{false}} - \Delta\right). \end{aligned}$$

Note

$$\left[U^{\text{false}} - \mu^{\text{false}}\right]_{(N^-)} \sim \text{Beta}(N^-, 1),$$

and

$$\left[U^{\text{true}} - (\mu^{\text{true}} - \Delta)\right]_{(\alpha+1)} \sim \text{Beta}(\alpha + 1, N^+ - \alpha),$$

where *Beta* denotes the Beta distribution. Both variables are independent. Thus the PDF of the difference is

$f(p) =$

$$\begin{cases} B(N^+ - \alpha, 1)p^{N^+ - \alpha}(1-p)^{\alpha+1}F(1, N^- + N^+, 1 - N^-; \alpha + 2; 1 - p, 1 - p^2)/A, & 0 < p \leq 1 \\ B(N^- - \alpha, N^+ - \alpha)(-p)^{N^+ - \alpha}(1+p)^{N^- + N^+ - \alpha - 1}F(N^+ - \alpha, -\alpha, N^- + N^+; N^- + N^+ - \alpha; 1 - p^2, 1 + p)/A, & -1 \leq p < 0 \\ B(N^- + \alpha, N^+ - \alpha)/A, & p = 0, \end{cases}$$

where $A = B(N^-, 1)B(\alpha + 1, N^+ - \alpha)$, $B(a, b) = \int_0^1 t^{a-1}(1-t)^{b-1}dt$

$$F(a, b_1, b_2; c; x, y) = \frac{\Gamma(c)}{\Gamma(a)\Gamma(c-a)} \int_0^1 t^{a-1}(1-t)^{c-a-1}(1-xt)^{-b_1}(1-yt)^{-b_2} dt.$$

Therefore, we have

$$\mathbb{P}(S_{(N^-)}^{\text{false}} < S_{(\alpha+1)}^{\text{true}}) \geq \int_{-1}^{\mu^{\text{true}} - \mu^{\text{false}} - \Delta} f(p) dp.$$

Case-2 The other part, we have no more than $e^{-v} \cdot \max(N^-, N^+)$ corrupted instances that may have higher scores than one clean instance.

Wrap-up From the above analyses, we know, w.p. at least $\int_{-1}^{\mu^{\text{true}} - \mu^{\text{false}} - \Delta} f(p) dp$, there are at most $e^{-v} \max(N^-, N^+) + \alpha$ errors in detection corrupted instances. Note Precision = Recall if we detect with the best threshold $N_j \mathbb{P}(Y = j | \tilde{Y} = j)$. Therefore, the corresponding F_1 -score would be at least $1 - \frac{e^{-v} \max(N^-, N^+) + \alpha}{N^-}$.

□

B.3 SimiFeat: Experiment Settings on Clothing1M

We first perform noise detection on 1 million noisy training instances then train only with the selected clean data to check the effectiveness. Particularly, in each epoch of the noisy detection, we use a batch size of 32 and sample 1,000 mini-batches from 1M training instances while ensuring the (noisy) labels are balanced. We repeat noisy detection for 600 epochs to ensure full coverage of 1 million training instances. Parameter k is set to 10.

Feature Extractor: We tested three different feature extractors in Table 4.3: R50-Img, ViT-B/32-CLIP, and R50-Img Warmup-1. The former two feature extractors are the same as the ones used in Table 3.4. Particularly, R50-Img means the feature extractor is the standard ResNet50 encoder (removing the last linear layer) pre-trained on ImageNet [38]. ViT-B/32-CLIP indicates the feature extractor is a vision transformer pre-trained

by CLIP [151]. Noting that Clothing1M is a fine-grained dataset. To get better domain-specific fine-grained visual features, we slightly train the ResNet50 pre-trained with ImageNet for one epoch, i.e., 1,000 mini-batches (batch size 32) randomly sampled from 1M training instances while ensuring the (noisy) labels are balanced. The learning rate is 0.002.

Training with the selected clean instances: Given the selected clean instances from our approach, we directly apply the Cross-Entropy loss to train a ResNet50 initialized by standard ImageNet pre-trained parameters. We **did not** apply any sophisticated training techniques, e.g., mixup [215], dual networks [108, 63], loss-correction [119, 139, 146], and robust loss functions [124, 30, 227, 186]. We train the model for 80 epochs with a batch size of 32. We sample 1,000 mini-batches per epoch randomly selected from 1M training instances. Note Table 4.3 does not apply balanced sampling. Only the pure cross-entropy loss is applied. We also test the performance with balanced training, i.e., in each epoch, ensure the noisy labels from each class are balanced. Our approach can be consistently benefited by balanced training, and achieves an accuracy of 73.97 in the best epoch, outperforming many baselines such as HOC 73.39% [230], GCE+SimCLR 73.35% [56], CORES 73.24% [30], GCE 69.75% [221]. We believe the performance could be further improved by using some sophisticated training techniques mentioned above.

Table B.1: Experiments on Clothing1M [200] with or without balanced sampling.

DATA SELECTION	# TRAINING SAMPLES	BEST EPOCH	LAST 10 EPOCHS	LAST EPOCH
NONE (STANDARD BASELINE) (UNBALANCED)	1M (100%)	70.32	69.44 ± 0.13	69.53
NONE (STANDARD BASELINE) (BALANCED)	1M (100%)	72.20	71.40 ± 0.31	71.22
R50-IMG (UNBALANCED)	770K (77.0%)	72.37	71.95 ± 0.08	71.89
R50-IMG (BALANCED)	770K (77.0%)	72.42	72.06 ± 0.16	72.24
ViT-B/32-CLIP (UNBALANCED)	700K (70.0%)	72.54	72.23 ± 0.17	72.11
ViT-B/32-CLIP (BALANCED)	700K (70.0%)	72.99	72.76 ± 0.15	72.91
R50-IMG WARMUP-1 (UNBALANCED)	767K (76.7%)	73.64	73.28 ± 0.18	73.41
R50-IMG WARMUP-1 (BALANCED)	767K (76.7%)	73.97	73.37 ± 0.03	73.35

Appendix C

More Details for Charter 4

Appendix C provides more theoretical details and experiment settings related to CAL (Section 4.2), where Sections C.1–Section C.3 show the detailed theoretical proofs and Section C.4 present more discussions.

C.1 Proof for Lemmas

C.1.1 Proof for Lemma 4

Proof. We try to build the connection between noisy distribution $\tilde{\mathcal{D}}$ and the underlying Bayes optimal distribution \mathcal{D}^* by the noise rates e_+ and e_- . The primary difference

from the proof of Lemma 2 in [124] is the usage of \mathcal{D}^* . Note:

$$\begin{aligned}
& \mathbb{E}_{\tilde{\mathcal{D}}}[l(f(X), \tilde{Y})] \\
&= \mathbb{E}_{\mathcal{D}^*} \left[\sum_{j \in \{-1, +1\}} \mathbb{P}(\tilde{Y} = j | X, Y^*) l(f(X), j) \right] \\
&= \mathbb{E}_{\mathcal{D}^*} \left[\sum_{j \in \{-1, +1\}} \mathbb{P}(\tilde{Y} = j | Y^*) l(f(X), j) \right] \\
&= \sum_{i \in \{-1, +1\}} \mathbb{P}(Y^* = i) \mathbb{E}_{\mathcal{D}^* | Y^* = i} [\mathbb{P}(\tilde{Y} = +1 | Y^* = i) l(f(X), +1) \\
&\quad + \mathbb{P}(\tilde{Y} = -1 | Y^* = i) l(f(X), -1)] \\
&= \mathbb{P}(Y^* = +1) \mathbb{E}_{\mathcal{D}^* | Y^* = +1} [(1 - e_+) l(f(X), +1) + e_+ l(f(X), -1)] \\
&\quad + \mathbb{P}(Y^* = -1) \mathbb{E}_{\mathcal{D}^* | Y^* = -1} [(1 - e_-) l(f(X), -1) + e_- l(f(X), +1)].
\end{aligned}$$

Similarly, following the proof of Lemma 2 in [124], we can prove this lemma. \square

C.1.2 Proof for Lemma 13

Peer Loss on the Bayes Optimal Distribution Recall our goal is to learn a classifier f from the noisy distribution $\tilde{\mathcal{D}}$ which also minimizes the loss on the corresponding Bayes optimal distribution \mathcal{D}^* , i.e. $\mathbb{E}[\mathbf{1}(f(X), Y^*)]$, $(X, Y^*) \sim \mathcal{D}^*$. Before considering the case with label noise, we need to prove peer loss functions induce the Bayes optimal classifier when minimizing the 0-1 loss on \mathcal{D}^* as in Lemma 13.

Lemma 13. *Given the Bayes optimal distribution \mathcal{D}^* , the optimal peer classifier defined below:*

$$f_{peer}^* = \arg \min_f \mathbb{E}_{\mathcal{D}^*} [\mathbf{1}_{PL}(f(X), Y^*)]$$

also minimizes $\mathbb{E}_{\mathcal{D}^*}[\mathbf{1}(f(X), Y^*)]$.

See the proof below. It has been shown in [124] that Lemma 13 holds for the clean distribution \mathcal{D} when the clean dataset is class-balanced, i.e. $\mathbb{P}(Y = -1) = \mathbb{P}(Y = +1) = 0.5$. For the Bayes optimal distribution \mathcal{D}^* , as shown in Lemma 13, there is *no requirement* for the prior $p^* := \mathbb{P}(Y^* = +1)$.

Proof. Recall Y^* is the Bayes optimal label defined as

$$Y^*|X := \arg \max_Y \mathbb{P}(Y|X), (X, Y) \sim \mathcal{D}.$$

We need to prove that the “optimal peer classifier” defined below:

$$f_{\text{peer}}^* = \arg \min_f \mathbb{E}_{\mathcal{D}^*}[\mathbf{1}_{\text{PL}}(f(X), Y^*)]$$

is the same as the Bayes optimal classifier f^* . To see this, suppose the claim is wrong.

Denote by (notations ϵ_+ and ϵ_- are defined only for this proof):

$$\epsilon_+ := \mathbb{P}(f_{\text{peer}}^*(X) = -1 | f^*(X) = +1), \quad \epsilon_- := \mathbb{P}(f_{\text{peer}}^*(X) = +1 | f^*(X) = -1)$$

and denote by $p^* := \mathbb{P}(f^*(X) = +1)$. Then

$$\begin{aligned}
& \mathbb{E}_{\mathcal{D}^*}[\mathbb{1}_{\text{PL}}(f_{\text{peer}}^*(X), Y^*)] \\
&= \mathbb{P}(f_{\text{peer}}^*(X) \neq Y^*) - p^* \cdot \mathbb{P}(f_{\text{peer}}^*(X) \neq +1) - (1 - p^*) \cdot \mathbb{P}(f_{\text{peer}}^*(X) \neq -1) \\
&= p^* \cdot \epsilon_+ + (1 - p^*) \cdot \epsilon_- - p^* \cdot \mathbb{P}(f_{\text{peer}}^*(X) \neq +1) - (1 - p^*) \cdot \mathbb{P}(f_{\text{peer}}^*(X) \neq -1) \\
&= p^* \cdot \epsilon_+ + (1 - p^*) \cdot \epsilon_- \\
&\quad - p^* \cdot \left(\mathbb{P}(f_{\text{peer}}^*(X) \neq +1 | f^*(X) \neq +1) \mathbb{P}(f^*(X) \neq +1) \right. \\
&\quad \quad \left. + \mathbb{P}(f_{\text{peer}}^*(X) \neq +1 | f^*(X) \neq -1) \mathbb{P}(f^*(X) \neq -1) \right) \\
&\quad - (1 - p^*) \cdot \left(\mathbb{P}(f_{\text{peer}}^*(X) \neq -1 | f^*(X) \neq +1) \mathbb{P}(f^*(X) \neq +1) \right. \\
&\quad \quad \left. + \mathbb{P}(f_{\text{peer}}^*(X) \neq -1 | f^*(X) \neq -1) \mathbb{P}(f^*(X) \neq -1) \right) \\
&= p^* \cdot \epsilon_+ + (1 - p^*) \cdot \epsilon_- \\
&\quad - p^* \cdot \mathbb{P}(f^*(X) \neq +1)(1 - \epsilon_-) - p^* \cdot \mathbb{P}(f^*(X) \neq -1) \cdot \epsilon_+ \\
&\quad - (1 - p^*) \cdot \mathbb{P}(f^*(X) \neq -1)(1 - \epsilon_+) - (1 - p^*) \cdot \mathbb{P}(f^*(X) \neq +1) \cdot \epsilon_- \\
&= 0 - p^* \cdot \mathbb{P}(f^*(X) \neq +1) - (1 - p^*) \cdot \mathbb{P}(f^*(X) \neq -1) \\
&\quad + p^*(\epsilon_+ + \mathbb{P}(f^*(X) \neq +1)\epsilon_- - \mathbb{P}(f^*(X) \neq -1)\epsilon_+) \\
&\quad + (1 - p^*)(\epsilon_- + \mathbb{P}(f^*(X) \neq -1)\epsilon_+ - \mathbb{P}(f^*(X) \neq +1)\epsilon_-) \\
&> 0 - p^* \cdot \mathbb{P}(f^*(X) \neq +1) - (1 - p^*) \cdot \mathbb{P}(f^*(X) \neq -1) \\
&= \mathbb{E}_{\mathcal{D}^*}[\mathbb{1}_{\text{PL}}(f^*(X), Y^*)]
\end{aligned}$$

contradicting the optimality of f_{peer}^* . Thus our claim is proved. \square

C.2 Proof for Theorems

C.2.1 Proof for Theorem 10

Proof. The covariance $\text{Cov}(\cdot, \cdot)$ in this proof is taken over the Bayes optimal distribution \mathcal{D}^* . The following proof is built on the result of Theorem 11, i.e. Eq. (4.2). First note

$$\begin{aligned} & \text{Cov}(Z_1(X), \mathbf{1}(f_1(X), Y^*) - \mathbf{1}(f_2(X), Y^*)) \\ &= \mathbb{E}[(Z_1(X) - \mathbb{E}[Z_1(X)]) \cdot (\mathbf{1}(f_1(X), Y^*) - \mathbf{1}(f_2(X), Y^*))] \\ &\leq \mathbb{E}[|(Z_1(X) - \mathbb{E}[Z_1(X)])|] \\ &\leq \mathbb{E}|e_+(X) - \mathbb{E}[e_+(X)]| + \mathbb{E}|e_-(X) - \mathbb{E}[e_-(X)]| \end{aligned}$$

Similarly, one can show that

$$\begin{aligned} & \text{Cov}(Z_2(X), \mathbf{1}(f_1(X), -1) - \mathbf{1}(f_2(X), -1)) \\ &\leq \mathbb{E}|e_+(X) - \mathbb{E}[e_+(X)]| + \mathbb{E}|e_-(X) - \mathbb{E}[e_-(X)]| \end{aligned}$$

Now with bounded variance in the error rates, suppose:

$$\mathbb{E}|e_+(X) - \mathbb{E}[e_+(X)]| \leq \epsilon_+, \quad \mathbb{E}|e_-(X) - \mathbb{E}[e_-(X)]| \leq \epsilon_-$$

Note

$$\begin{aligned}
\tilde{f}_{\text{peer}}^* &:= \arg \min_f \mathbb{E}_{\tilde{\mathcal{D}}} \left[\mathbf{1}_{\text{PL}}(f(X), \tilde{Y}) \right] \\
&= \arg \min_f \left[(1 - e_+ - e_-) \mathbb{E}_{\mathcal{D}^*} [\mathbf{1}_{\text{PL}}(f(X), Y^*) + \text{Cov}(Z_1(X), \mathbf{1}(f(X), Y^*)) \right. \\
&\quad \left. + \text{Cov}(Z_2(X), \mathbf{1}(f(X), -1))] \right] \\
&= \arg \min_f \left[(1 - e_+ - e_-) \right. \\
&\quad \cdot (\mathbb{E}_{\mathcal{D}^*} [\mathbf{1}(f(X), Y^*) - p^* \cdot \mathbb{E}_{\mathcal{D}^*} [\mathbf{1}(f(X), +1)] - (1 - p^*) \cdot \mathbb{E}_{\mathcal{D}^*} [\mathbf{1}(f(X), -1)]] \\
&\quad \left. + \text{Cov}(Z_1(X), \mathbf{1}(f(X), Y^*)) + \text{Cov}(Z_2(X), \mathbf{1}(f(X), -1)) \right].
\end{aligned}$$

Then

$$\begin{aligned}
&\mathbb{E}_{\mathcal{D}^*} \left[\mathbf{1}(\tilde{f}_{\text{peer}}^*(X), Y^*) \right] + \frac{\text{Cov}(Z_1(X), \mathbf{1}(\tilde{f}_{\text{peer}}^*(X), Y^*)) + \text{Cov}(Z_2(X), \mathbf{1}(\tilde{f}_{\text{peer}}^*(X), -1))}{1 - e_+ - e_-} \\
&= \mathbb{E}_{\mathcal{D}^*} \left[\mathbf{1}(\tilde{f}_{\text{peer}}^*(X), Y^*) \right] - 0.5 \cdot \mathbb{E}_X [\mathbf{1}(\tilde{f}_{\text{peer}}^*(X), +1)] - 0.5 \cdot \mathbb{E}_X [\mathbf{1}(\tilde{f}_{\text{peer}}^*(X), -1)] + 0.5 \\
&\quad + \frac{\text{Cov}(Z_1(X), \mathbf{1}(\tilde{f}_{\text{peer}}^*(X), Y^*)) + \text{Cov}(Z_2(X), \mathbf{1}(\tilde{f}_{\text{peer}}^*(X), -1))}{1 - e_+ - e_-} \\
&\leq \mathbb{E}_{\mathcal{D}^*} \left[\mathbf{1}(\tilde{f}_{\text{peer}}^*(X), Y^*) \right] \\
&\quad - p^* \cdot \mathbb{E}_X [\mathbf{1}(\tilde{f}_{\text{peer}}^*(X), +1)] - (1 - p^*) \cdot \mathbb{E}_X [\mathbf{1}(\tilde{f}_{\text{peer}}^*(X), -1)] + |p^* - 0.5| + 0.5 \\
&\quad + \frac{\text{Cov}(Z_1(X), \mathbf{1}(\tilde{f}_{\text{peer}}^*(X), Y^*)) + \text{Cov}(Z_2(X), \mathbf{1}(\tilde{f}_{\text{peer}}^*(X), -1))}{1 - e_+ - e_-} \\
&\leq \mathbb{E}_{\mathcal{D}^*} \left[\mathbf{1}(f^*(X), Y^*) \right] - p^* \cdot \mathbb{E}_X [\mathbf{1}(f^*(X), +1)] - (1 - p^*) \cdot \mathbb{E}_X [\mathbf{1}(f^*(X), -1)] + |p^* - 0.5| + 0.5 \\
&\quad + \frac{\text{Cov}(Z_1(X), \mathbf{1}(f^*(X), Y^*)) + \text{Cov}(Z_2(X), \mathbf{1}(f^*(X), -1))}{1 - e_+ - e_-} \\
&\leq \mathbb{E}_{\mathcal{D}^*} \left[\mathbf{1}(f^*(X), Y^*) \right] + \frac{\text{Cov}(Z_1(X), \mathbf{1}(f^*(X), Y^*)) + \text{Cov}(Z_2(X), \mathbf{1}(f^*(X), -1))}{1 - e_+ - e_-} + 2|p^* - 0.5|.
\end{aligned}$$

Thus

$$\begin{aligned}
& \mathbb{E}_{\mathcal{D}^*} \left[\mathbf{1}(\tilde{f}_{\text{peer}}^*(X), Y^*) - \mathbf{1}(f^*(X), Y^*) \right] \\
& \leq \frac{\text{Cov}(Z_1(X), \mathbf{1}(f^*(X), Y^*) - \mathbf{1}(\tilde{f}_{\text{peer}}^*(X), Y^*))}{1 - e_+ - e_-} \\
& \quad + \frac{\text{Cov}(Z_2(X), \mathbf{1}(f^*(X), -1) - \mathbf{1}(\tilde{f}_{\text{peer}}^*(X), -1))}{1 - e_+ - e_-} + 2|p^* - 0.5| \\
& \leq 2 \frac{\mathbb{E}|e_+(X) - \mathbb{E}[e_+(X)]| + \mathbb{E}|e_-(X) - \mathbb{E}[e_-(X)]|}{1 - e_+ - e_-} + 2|p^* - 0.5| \\
& \leq \frac{2(\epsilon_+ + \epsilon_-)}{1 - e_+ - e_-} + 2|p^* - 0.5|.
\end{aligned}$$

Noting $\mathbf{1}(f^*(X), Y^*) = 0$, we finish the proof. □

C.2.2 Proof for Theorem 11

Proof. The covariance $\text{Cov}(\cdot, \cdot)$ in this proof is taken over the Bayes optimal distribution

\mathcal{D}^* . Recall

$$e_+(X) := \mathbb{P}(\tilde{Y} = -1 | Y^* = +1, X), \quad e_-(X) := \mathbb{P}(\tilde{Y} = +1 | Y^* = -1, X)$$

and

$$e_+ := \mathbb{E}_X[e_+(X)], \quad e_- := \mathbb{E}_X[e_-(X)]$$

We first have the following equality:

$$\begin{aligned}
\mathbb{E}_{\tilde{\mathcal{D}}}[\mathbf{1}_{\text{PL}}(f(X), \tilde{Y})] &= \mathbb{E}_{\mathcal{D}^*}[(1 - e_+(X) - e_-(X))\mathbf{1}(f(X), Y^*)] && \text{(Term-A)} \\
&+ \mathbb{E}_X[e_+(X)\mathbf{1}(f(X), -1) + e_-(X)\mathbf{1}(f(X), +1)] && \text{(Term-B)} \\
&- (1 - e_+ - e_-) \cdot \mathbb{E}_{\mathcal{D}^*}[\mathbf{1}(f(X), Y_p^*)] && \text{(Term-C)} \\
&- \mathbb{E}_X[e_+ \cdot \mathbf{1}(f(X), -1) + e_- \cdot \mathbf{1}(f(X), +1)] && \text{(Term-D)}
\end{aligned}$$

Term-B can be transformed to:

$$\begin{aligned}
& \mathbb{E}_X[e_+(X) \cdot \mathbf{1}(f(X), -1) + e_-(X) \cdot \mathbf{1}(f(X), +1)] \\
&= \mathbb{E}_X[e_+(X) \cdot \mathbf{1}(f(X), -1) + e_-(X) \cdot (1 - \mathbf{1}(f(X), -1))] \\
&= \mathbb{E}_X[(e_+(X) - e_-(X)) \cdot \mathbf{1}(f(X), -1) + e_-(X)].
\end{aligned}$$

Similarly, Term-D turns to

$$\mathbb{E}_X[e_+ \cdot \mathbf{1}(f(X), -1) + e_- \cdot \mathbf{1}(f(X), +1)] = (e_+ - e_-) \cdot \mathbb{E}_X[\mathbf{1}(f(X), -1)] + e_-.$$

Define two random variables

$$Z_1(X) := 1 - e_+(X) - e_-(X), \quad Z_2(X) = e_+(X) - e_-(X).$$

Then Term-A becomes

$$\begin{aligned}
& \mathbb{E}_{\mathcal{D}^*}[(1 - e_+(X) - e_-(X))\mathbf{1}(f(X), Y^*)] \\
&= \mathbb{E}[Z_1(X)] \cdot \mathbb{E}_{\mathcal{D}^*}[\mathbf{1}(f(X), Y^*)] + \text{Cov}(Z_1(X), \mathbf{1}(f(X), Y^*)) \\
&= (1 - e_+ - e_-) \cdot \mathbb{E}_{\mathcal{D}^*}[\mathbf{1}(f(X), Y^*)] + \text{Cov}(Z_1(X), \mathbf{1}(f(X), Y^*))
\end{aligned}$$

Similarly, Term-B can be further transformed to

$$\begin{aligned}
& \mathbb{E}_X[(e_+(X) - e_-(X)) \cdot \mathbf{1}(f(X), -1) + e_-(X)] \\
&= \mathbb{E}[Z_2(X)]\mathbb{E}_X[\mathbf{1}(f(X), -1)] + \text{Cov}(Z_2(X), \mathbf{1}(f(X), -1)) + e_- \\
&= (e_+ - e_-)\mathbb{E}_X[\mathbf{1}(f(X), -1)] + \text{Cov}(Z_2(X), \mathbf{1}(f(X), -1)) + e_-
\end{aligned}$$

Combining the above results, we have

$$\begin{aligned}
\mathbb{E}_{\tilde{\mathcal{D}}}[1_{\text{PL}}(f(X), \tilde{Y})] &= (1 - e_+ - e_-) \cdot \mathbb{E}_{\mathcal{D}^*}[1(f(X), Y^*)] \\
&\quad + (e_+ - e_-) \mathbb{E}_X[1(f(X), -1)] + e_- \\
&\quad - (1 - e_+ - e_-) \cdot \mathbb{E}_{\mathcal{D}^*}[1(f(X), Y_p^*)] \\
&\quad - (e_+ - e_-) \cdot \mathbb{E}_X[1(f(X), -1)] - e_- \\
&\quad + \text{Cov}(Z_1(X), 1(f(X), Y)) + \text{Cov}(Z_2(X), 1(f(X), -1)) \\
&= (1 - e_+ - e_-) \mathbb{E}_{\mathcal{D}^*}[1_{\text{PL}}(f(X), Y^*)] \\
&\quad + \text{Cov}(Z_1(X), 1(f(X), Y^*)) + \text{Cov}(Z_2(X), 1(f(X), -1))
\end{aligned}$$

□

C.2.3 Proof for Theorem 12

Proof. From Theorem 11, we know

$$\begin{aligned}
&\left[\mathbb{E}_{\tilde{\mathcal{D}}}[1_{\text{PL}}(f(X), \tilde{Y})] - \text{Cov}(Z_1(X), 1(f(X), Y^*)) - \text{Cov}(Z_2(X), 1(f(X), -1)) \right] \\
&= (1 - e_- - e_+) \cdot \mathbb{E}_{\mathcal{D}^*}[1_{\text{PL}}(f(X), Y^*)].
\end{aligned}$$

With Lemma 13, we can finish the proof. □

C.2.4 Proof for Theorem 13

Proof. Recall $\tau \in [0, 1]$ is the expected ratio (a.k.a. probability) of correct examples in \hat{D}^τ , i.e. $\tau = \mathbb{E}[1\{(X, \hat{Y}) \in \hat{D}^\tau | (X, Y^*) \in D^*\}] = \mathbb{P}((X, \hat{Y}) \sim \hat{D}^\tau | (X, Y^*) \sim \mathcal{D}^*)$. With

\hat{D}^τ , the classifier learned by minimizing the 0-1 CAL loss is

$$\begin{aligned} \tilde{f}_{\text{CAL-}\tau}^* := \arg \min_f \mathbb{E}_{\tilde{\mathcal{D}}} \left[\mathbb{1}_{\text{PL}}(f(X), \tilde{Y}) \right. \\ \left. - \text{Cov}_{\hat{\mathcal{D}}^\tau}(Z_1(X), \mathbb{1}(f(X), \hat{Y})) - \text{Cov}_{\hat{\mathcal{D}}^\tau}(Z_2(X), \mathbb{1}(f(X), -1)) \right]. \end{aligned}$$

Note

$$\begin{aligned} & \text{Cov}_{\hat{\mathcal{D}}^\tau}(Z_1(X), \mathbb{1}(f(X), Y)) \\ &= \mathbb{E}_{\hat{\mathcal{D}}^\tau} \left[(Z_1(X) - \mathbb{E}_{\hat{\mathcal{D}}^\tau}[Z_1(X)]) (\mathbb{1}(f(X), Y) - \mathbb{E}_{\hat{\mathcal{D}}^\tau}[\mathbb{1}(f(X), Y)]) \right] \\ &= \mathbb{E}_{\hat{\mathcal{D}}^\tau} \left[(Z_1(X) - \mathbb{E}_{\hat{\mathcal{D}}^\tau}[Z_1(X)]) \mathbb{1}(f(X), Y) \right] \\ &= \mathbb{P}((X, Y) \in D^* | (X, Y) \in \hat{D}^\tau) \mathbb{E}_{\hat{\mathcal{D}}^\tau} \left[(Z_1(X) - \mathbb{E}_{\hat{\mathcal{D}}^\tau}[Z_1(X)]) \mathbb{1}(f(X), Y) | (X, Y) \in D^* \right] \\ & \quad + \mathbb{P}((X, Y) \notin D^* | (X, Y) \in \hat{D}^\tau) \mathbb{E}_{\hat{\mathcal{D}}^\tau} \left[(Z_1(X) - \mathbb{E}_{\hat{\mathcal{D}}^\tau}[Z_1(X)]) \mathbb{1}(f(X), Y) | (X, Y) \notin D^* \right]. \end{aligned}$$

Similarly,

$$\begin{aligned} & \text{Cov}_{\mathcal{D}^*}(Z_1(X), \mathbb{1}(f(X), Y)) \\ &= \mathbb{E}_{\mathcal{D}^*} \left[(Z_1(X) - \mathbb{E}_{\mathcal{D}^*}[Z_1(X)]) \mathbb{1}(f(X), Y) \right] \\ &= \mathbb{P}((X, Y) \in \hat{D}^\tau | (X, Y) \in \mathcal{D}^*) \mathbb{E}_{\mathcal{D}^*} \left[(Z_1(X) - \mathbb{E}_{\mathcal{D}^*}[Z_1(X)]) \mathbb{1}(f(X), Y) | (X, Y) \in \hat{D}^\tau \right] \\ & \quad + \mathbb{P}((X, Y) \notin \hat{D}^\tau | (X, Y) \in \mathcal{D}^*) \mathbb{E}_{\mathcal{D}^*} \left[(Z_1(X) - \mathbb{E}_{\mathcal{D}^*}[Z_1(X)]) \mathbb{1}(f(X), Y) | (X, Y) \notin \hat{D}^\tau \right]. \end{aligned}$$

When D^* , \hat{D}^τ and \tilde{D} have the same feature set, we have

$$\mathbb{P}((X, Y) \in D^* | (X, Y) \in \hat{D}^\tau) = \mathbb{P}((X, Y) \in \hat{D}^\tau | (X, Y) \in D^*) = \tau,$$

$$\mathbb{P}((X, Y) \notin D^* | (X, Y) \in \hat{D}^\tau) = \mathbb{P}((X, Y) \notin \hat{D}^\tau | (X, Y) \in D^*) = 1 - \tau.$$

Therefore,

$$\text{Cov}_{\hat{\mathcal{D}}^\tau}(Z_1(X), \mathbb{1}(f(X), Y)) - \text{Cov}_{\mathcal{D}^*}(Z_1(X), \mathbb{1}(f(X), Y)) \leq 2(1 - \tau)(\epsilon_+ + \epsilon_-).$$

The rest of the proof can be accomplished by following the proof of Theorem 10. \square

C.3 Proof for Corollaries

C.3.1 Proof for Corollary 4

Proof.

$$\mathbb{E}_{\tilde{\mathcal{D}}}[l_{\text{PL}}(f(X), \tilde{Y})] = \mathbb{E}_{\tilde{\mathcal{D}}}[\ell(f(X), \tilde{Y})] - \mathbb{E}_{\tilde{\mathcal{D}}_Y} \left[\mathbb{E}_{\mathcal{D}_X}[\ell(f(X_p), \tilde{Y}_p)] \right]. \quad (\text{C.1})$$

The first term in (C.1) is

$$\begin{aligned} & \mathbb{E}_{\tilde{\mathcal{D}}}[\ell(f(X), \tilde{Y})] \\ = & \mathbb{E}_{\mathcal{D}^*} \left[\sum_{j \in [K]} \mathbb{P}(\tilde{Y} = j | X, Y^*) \ell(f(X), j) \right] \\ = & \sum_{j \in [K]} \sum_{i \in [K]} \mathbb{P}(Y^* = i) \mathbb{E}_{\mathcal{D}^* | Y^* = i} [T_{ij}(X) \ell(f(X), j)] \\ = & \sum_{j \in [K]} \sum_{i \in [K]} \mathbb{P}(Y^* = i) [T_{ij} \mathbb{E}_{\mathcal{D}^* | Y^* = i} [\ell(f(X), j)] + \text{Cov}_{\mathcal{D}^* | Y^* = i} [T_{ij}(X), \ell(f(X), j)]] \\ = & \sum_{j \in [K]} \left[\mathbb{P}(Y^* = j) \left(1 - \sum_{i \neq j, i \in [K]} T_{ji} \right) \mathbb{E}_{\mathcal{D}^* | Y^* = j} [\ell(f(X), j)] + \sum_{i \in [K], i \neq j} \mathbb{P}(Y^* = i) T_{ij} \mathbb{E}_{\mathcal{D}^* | Y^* = i} [\ell(f(X), j)] \right] \\ & + \sum_{j \in [K]} \sum_{i \in [K]} P(Y^* = i) \text{Cov}_{\mathcal{D}^* | Y^* = i} [T_{ij}(X), \ell(f(X), j)] \\ = & \sum_{j \in [K]} \left[\mathbb{P}(Y^* = j) \left(1 - \sum_{i \neq j, i \in [K]} e_i \right) \mathbb{E}_{\mathcal{D}^* | Y^* = j} [\ell(f(X), j)] + \sum_{i \in [K], i \neq j} \mathbb{P}(Y^* = i) e_j \mathbb{E}_{\mathcal{D}^* | Y^* = i} [\ell(f(X), j)] \right] \\ & + \sum_{j \in [K]} \sum_{i \in [K]} P(Y^* = i) \text{Cov}_{\mathcal{D}^* | Y^* = i} [T_{ij}(X), \ell(f(X), j)] \\ = & \left(1 - \sum_{i \in [K]} e_i \right) \mathbb{E}_{\mathcal{D}^*} [\ell(f(X), Y^*)] + \sum_{j \in [K]} \sum_{i \in [K]} \mathbb{P}(Y^* = i) e_j \mathbb{E}_{\mathcal{D}^* | Y^* = i} [\ell(f(X), j)] \\ & + \sum_{j \in [K]} \sum_{i \in [K]} P(Y^* = i) \text{Cov}_{\mathcal{D}^* | Y^* = i} [T_{ij}(X), \ell(f(X), j)] \end{aligned}$$

The rest of the proofs can be done following standard multi-class peer loss derivations [124].

□

C.4 More Discussions

C.4.1 Setting Thresholds L_{\min} and L_{\max}

In a high level, there are two strategies for setting L_{\min} and L_{\max} : 1) $L_{\min} < L_{\max}$ and 2) $L_{\min} = L_{\max}$.

Strategy-1: $L_{\min} < L_{\max}$: This strategy may provide a higher ratio of true Bayes optimal labels among feasible examples in \hat{D} since some ambiguous examples are dropped. However, dropping examples changes the distribution of X (as well as the distribution of the unobservable Y^*), a.k.a. covariate shift [76, 33]. Importance re-weighting with weight $\gamma(X)$ is necessary for correcting the covariate shift, i.e. the weight of each feasible example $(x, \hat{y}) \in \hat{D}$ should be changed from 1 to $\gamma(x)$. Let \mathcal{D}_X and $\hat{\mathcal{D}}_X$ be the marginal distributions of \mathcal{D} and $\hat{\mathcal{D}}$ on X . With a particular kernel $\Phi(X)$, the optimization problem is:

$$\begin{aligned} \min_{\gamma(X)} \quad & \|\mathbb{E}_{\mathcal{D}_X}[\Phi(X)] - \mathbb{E}_{\hat{\mathcal{D}}_X}[\gamma(X)\Phi(X)]\| \\ \text{s.t.} \quad & \gamma(X) > 0 \quad \text{and} \quad \mathbb{E}_{\hat{\mathcal{D}}_X}[\gamma(X)] = 1. \end{aligned} \tag{C.2}$$

The optimal solution is supposed to be $\gamma^*(X) = \frac{\mathbb{P}_{\mathcal{D}_X}(X)}{\mathbb{P}_{\hat{\mathcal{D}}_X}(X)}$. Note the selection of kernel $\Phi(\cdot)$ is non-trivial, especially for complicated features [47] in DNN solutions. Using this strategy, with appropriate L_{\min} and L_{\max} such that all the examples in \hat{D} are Bayes optimal, the covariance could be guaranteed to be optimal when each example in \hat{D} is re-weighted by $\gamma^*(X)$.

Strategy-2: $L_{\min} = L_{\max}$: Compared with Strategy-1, we effectively lose one degree of freedom for getting a better \hat{D} . However, this is not entirely harmful since \hat{D} and D^* have the same feature set, indicating estimating $\gamma(X)$ is no longer necessary and $\gamma(X) = 1$ is an optimal solution for (C.2) with this strategy.

Strategy selection When we can get a high-quality \hat{D} by fine-tuning L_{\min} and L_{\max} or \hat{D} is already provided from other sources, we may solve the optimization problem in (C.2) to find the optimal weight $\gamma(X)$. However, considering the fact that estimating $\gamma(X)$ introduces extra computation and potentially extra errors, we focus on Strategy-2 in this paper. Using Strategy-2 also reduces the effort on tuning hyperparameters. Besides, the proposed CAL loss is tolerant of an imperfect \hat{D} (shown theoretically in Section 4.2.3).

C.4.2 Generation of Instance-Dependent Label Noise

The generation process follows Section A.4.1.

C.4.3 More Implementation Details on Clothing1M

Construct \hat{D} We first train the network for 120 epochs on 1 million noisy training images using the method in [30]. The batch-size is set to 32. The initial learning rate is set as 0.01 and reduced by a factor of 10 at 30, 60, 90 epochs. We sample 1000 mini-batches from the training data for each epoch while ensuring the (noisy) labels are balanced. Mixup [215] is adopted for data augmentations. Hyperparameter β is set to 0

at first 80 epochs, and linearly increased to 0.4 for next 20 epochs and kept as 0.4 for the rest of the epochs. We construct \hat{D} with the best model.

Train with CAL We change the loss to the CAL loss after getting \hat{D} and continue training the model (without mixup) with an initial learning rate of 10^{-5} for 120 epochs (reduced by a factor of 10 at 30, 60, 90 epochs). We also tested re-train the model with \hat{D} and get an accuracy of 73.56. A randomly-collected balanced dataset with 18,976 noisy examples in each class is employed in training with CAL. Examples that are not in this balanced dataset are removed from \hat{D} for ease of implementation.

Appendix D

More Details for Charter 5

Appendix D provides more theoretical details related to the disparate impact of SSL and estimating fairness with missing sensitive attributes, which is organized as follows.

Disparate Impact in SSL The Appendix is organized as follows.

- Section D.1 presents the detailed derivations for generalization bounds.
 - Section D.1.1 proves the upper bound for Term-1.
 - Section D.1.2 proves the lower bound for Term-1.
 - Section D.1.2 proves the upper bound for Term-2 (also for Theorem 14).
 - Section D.1.4 proves Lemma 6.
 - Section D.1.5 proves Corollary 5.

Estimate Fairness with Missing Sensitive Attributes The Appendix is organized as follows.

- Section D.2 presents a summary of notations, more fairness definitions, and a clear statement of the assumption that is common in the literature. Note our algorithm does *not* rely on this assumption.
- Section D.3 presents the full version of our theorems (for DP, EOd, EOp), corollaries, and the corresponding proofs.
- Section D.4 shows how HOC works and analyzes why other learning-centric methods in the noisy label literature may not work in our setting.

D.1 Theoretical Results

D.1.1 Term-1 Upper Bound

$$\begin{aligned}
& R_{\mathcal{D}}(f) - R_{\tilde{\mathcal{D}}}(f) \\
&= \int_X \left(\mathbb{P}(f(X) \neq Y|X) - \mathbb{P}(f(X) \neq \tilde{Y}|X) \right) \mathbb{P}(X) dX \\
&= \int_X \left(\mathbb{P}(f(X) = \tilde{Y}|X) - \mathbb{P}(f(X) = Y|X) \right) \mathbb{P}(X) dX \\
&\leq \frac{1}{2} \int_X \left(\left| \mathbb{P}(f(X) = \tilde{Y}|X) - \mathbb{P}(f(X) = Y|X) \right| \right. \\
&\quad \left. + \left| \mathbb{P}(f(X) \neq \tilde{Y}|X) - \mathbb{P}(f(X) \neq Y|X) \right| \right) \mathbb{P}(X) dX \\
&\stackrel{(a)}{=} \int_X \text{TD}(\tilde{Y}_f(X) || Y_f(X)) \mathbb{P}(X) dX \\
&\stackrel{(b)}{\leq} \int_X \text{TD}(\tilde{Y}(X) || Y(X)) \mathbb{P}(X) dX \\
&= \frac{1}{2} \int_X \sum_{i \in [K]} \left| \mathbb{P}(\tilde{Y} = i|X) - \mathbb{P}(Y = i|X) \right| \mathbb{P}(X) dX \\
&= \int_X \eta(X) \mathbb{P}(X) dX \\
&= \eta
\end{aligned}$$

where in **equality (a)**, given model f and feature X , we can treat $\tilde{Y}_f(X)$ as a Bernoulli random variable such that

$$\mathbb{P}(\tilde{Y}_f(X) = +) = \mathbb{P}(f(X) = \tilde{Y}|X) \quad \text{and} \quad \mathbb{P}(\tilde{Y}_f(X) = -) = \mathbb{P}(f(X) \neq \tilde{Y}|X).$$

Then according to the definition of total variation of two distributions, i.e.,

$$\text{TD}(P||Q) := \frac{1}{2} \int_u \left| \frac{dP}{du} - \frac{dQ}{du} \right| du,$$

we can summarize the integrand as the total variation between $\tilde{Y}_f(X)$ and $Y_f(X)$.

Inequality (b) holds due to the data processing inequality since the probabilities $[\mathbb{P}(\tilde{Y} = f(X)), \mathbb{P}(\tilde{Y} \neq f(X))]$ are generated by $[\mathbb{P}(\tilde{Y} = i), \forall i \in [K]]$, and the probabilities $[\mathbb{P}(Y = f(X)), \mathbb{P}(Y \neq f(X))]$ are generated by $[\mathbb{P}(Y = i), \forall i \in [K]]$.

$\eta(X)$ is the accuracy of labels on feature X (in the cases specified in Lemma 6).

D.1.2 Term-1 Lower Bound

Let $e(X) := \mathbb{P}(Y \neq \tilde{Y}|X)$ be the feature-dependent error rate, $\tilde{A}_f(X) := \mathbb{P}(f(X) = \tilde{Y}|X)$ be the accuracy of prediction $f(X)$ on noisy dataset \tilde{D} . Note $e(X)$ is independent of $\tilde{A}_f(X)$. Denote their expectations (over X) by $\bar{e} := \mathbb{E}_X[e(X)]$,

$\tilde{A}_f = \mathbb{E}_X[\tilde{A}_f(X)]$. We have:

$$\begin{aligned}
& R_{\mathcal{D}}(f) - R_{\tilde{\mathcal{D}}}(f) \\
&= \int_X \left(\mathbb{P}(f(X) \neq Y|X) - \mathbb{P}(f(X) \neq \tilde{Y}|X) \right) \mathbb{P}(X) \\
&= \int_X \left(\mathbb{P}(f(X) = \tilde{Y}|X) - \mathbb{P}(f(X) = Y|X) \right) \mathbb{P}(X) \\
&= \int_X \left(\underbrace{\mathbb{P}(f(X) = \tilde{Y}|f(X) = \tilde{Y}, X)}_{\text{w.p. 1}} - \underbrace{\mathbb{P}(f(X) = Y|f(X) = \tilde{Y}, X)}_{\mathbb{P}(\tilde{Y}=Y|X)} \right) \mathbb{P}(f(X) = \tilde{Y}|X) \mathbb{P}(X) \\
&\quad + \int_X \left(\underbrace{\mathbb{P}(f(X) = \tilde{Y}|f(X) \neq \tilde{Y}, X)}_{\text{w.p. 0}} - \mathbb{P}(f(X) = Y|f(X) \neq \tilde{Y}, X) \right) \mathbb{P}(f(X) \neq \tilde{Y}|X) \mathbb{P}(X) \\
&= \int_X \underbrace{\left(1 - \mathbb{P}(\tilde{Y} = Y|X) \right)}_{\text{denoted by } e(X)} \underbrace{\mathbb{P}(f(X) = \tilde{Y}|X)}_{\text{denoted by } \tilde{A}_f(X)} \mathbb{P}(X) \\
&\quad + \int_X \left(0 - \mathbb{P}(f(X) = Y|f(X) \neq \tilde{Y}, X) \right) \mathbb{P}(f(X) \neq \tilde{Y}|X) \mathbb{P}(X) \\
&= \int_X e(X) \tilde{A}_f(X) \mathbb{P}(X) \\
&\quad - \int_X \left(\mathbb{P}(f(X) = Y|f(X) \neq \tilde{Y}, Y = \tilde{Y}, X) \mathbb{P}(Y = \tilde{Y}|f(X) \neq \tilde{Y}, X) \right) \mathbb{P}(f(X) \neq \tilde{Y}|X) \mathbb{P}(X) \\
&\quad - \int_X \left(\underbrace{\mathbb{P}(f(X) = Y|f(X) \neq \tilde{Y}, Y \neq \tilde{Y}, X)}_{\leq 1} \mathbb{P}(Y \neq \tilde{Y}|f(X) \neq \tilde{Y}, X) \right) \mathbb{P}(f(X) \neq \tilde{Y}|X) \mathbb{P}(X) \\
&\geq \int_X e(X) \tilde{A}_f(X) \mathbb{P}(X) \\
&\quad - \int_X \left(0 \cdot \mathbb{P}(Y = \tilde{Y}|f(X) \neq \tilde{Y}, X) \right) \mathbb{P}(f(X) \neq \tilde{Y}|X) \mathbb{P}(X) \\
&\quad - \int_X \left(1 \cdot \underbrace{\mathbb{P}(Y \neq \tilde{Y}|f(X) \neq \tilde{Y}, X)}_{=\mathbb{P}(Y \neq \tilde{Y}|X) \text{ due to independency}} \right) \mathbb{P}(f(X) \neq \tilde{Y}|X) \mathbb{P}(X) \\
&= \int_X e(X) \tilde{A}_f(X) \mathbb{P}(X) - \int_X \underbrace{\mathbb{P}(Y \neq \tilde{Y}|X)}_{\text{denoted by } e(X)} \underbrace{\mathbb{P}(f(X) \neq \tilde{Y}|X)}_{\text{denoted by } 1 - \tilde{A}_f(X)} \mathbb{P}(X) \\
&= \int_X e(X) (2\tilde{A}_f(X) - 1) \mathbb{P}(X) \\
&= (2\tilde{A}_f - 1)e.
\end{aligned}$$

D.1.3 Term-2 Upper Bound

We adopt the same technique to prove Theorem 14 and Lemma 7. The proof follows [124]. We prove Theorem 14 as follows.

Denote the expected error rate of classifier f on distribution \mathcal{D} by

$$R_{\mathcal{D}}(f) := \mathbb{E}_{\mathcal{D}}[\mathbb{1}(f(X), Y)].$$

Let \hat{f}_D denote the classifier trained by minimizing 0-1 loss with clean dataset D , i.e.,

$$\hat{f}_D := \arg \min_f \hat{R}_D(f),$$

where

$$\hat{R}_D(f) := \frac{1}{N} \sum_{n \in [N]} \mathbb{1}(f(x_n), y_n).$$

The Bayes optimal classifier is denoted by

$$f_{\mathcal{D}}^* := \arg \min_f R_{\mathcal{D}}(f).$$

The expected error given by $f_{\mathcal{D}}^*$ is written as

$$R^* := R_{\mathcal{D}}(f_{\mathcal{D}}^*).$$

Denote by $Y^*|X := \arg \max_{i \in [K]} \mathbb{P}(Y=i|X)$ the Bayes optimal label on clean distribution

\mathcal{D} . With probability at least $1 - \delta$, we have:

$$\begin{aligned} & R_{\mathcal{D}}(\hat{f}_D) - R_{\mathcal{D}}(f_{\mathcal{D}}^*) \\ &= \hat{R}_D(\hat{f}_D) - \hat{R}_D(f_{\mathcal{D}}^*) + \left(R_{\mathcal{D}}(\hat{f}_D) - \hat{R}_D(\hat{f}_D) \right) + \left(\hat{R}_D(f_{\mathcal{D}}^*) - R_{\mathcal{D}}(f_{\mathcal{D}}^*) \right) \\ &\stackrel{(a)}{\leq} 0 + |\hat{R}_D(\hat{f}_D) - R_{\mathcal{D}}(\hat{f}_D)| + |\hat{R}_D(f_{\mathcal{D}}^*) - R_{\mathcal{D}}(f_{\mathcal{D}}^*)| \\ &\stackrel{(b)}{\leq} \sqrt{\frac{2 \log(4/\delta)}{N}}, \end{aligned}$$

where inequality (a) holds since 1) $\hat{R}_D(\hat{f}_D)$ achieves the minimum empirical risk according to its definition, thus $\hat{R}_D(\hat{f}_D) - \hat{R}_D(f_D^*) \leq 0$; 2) each of the following two terms will be no greater than the corresponding gap $|\hat{R}_D(f) - R_D(f)|$. Specifically, inequality (b) holds due to Hoeffding's inequality, i.e., given any classifier \hat{f}_D , and f_D^* , with probability at least $1 - \delta/2$, we have the following bounds independently:

$$|\hat{R}_D(\hat{f}_D) - R_D(\hat{f}_D)| \leq \sqrt{\frac{\log(4/\delta)}{2N}}, \quad |\hat{R}_D(f_D^*) - R_D(f_D^*)| \leq \sqrt{\frac{\log(4/\delta)}{2N}}.$$

By the union bound, we have inequality (b) with probability at least $1 - \delta$.

Noting $R_D(f_D^*) = \mathbb{P}(Y \neq Y^*)$, we can prove Theorem 14. But substituting \mathcal{D} for $\tilde{\mathcal{D}}$, we can also prove Lemma 7.

D.1.4 Proof for Lemma 6

Proof. Note

$$\mathbb{P}(\tilde{Y} \neq Y|X) = 1 - \mathbb{P}(\tilde{Y} = Y|X) = 1 - \sum_{i \in [K]} \mathbb{P}(\tilde{Y} = i|X)\mathbb{P}(Y = i|X).$$

and

$$\eta(X) := \frac{1}{2} \sum_{i \in [K]} |\mathbb{P}(\tilde{Y} = i|X) - \mathbb{P}(Y = i|X)|.$$

Assume $\mathbb{P}(Y = i'|X) = 1$. We have $\mathbb{P}(\tilde{Y} \neq Y|X) = 1 - \mathbb{P}(\tilde{Y} = i'|X)$ and

$$\eta(X) := \frac{1}{2} \left(1 - \mathbb{P}(\tilde{Y} = i'|X) + \sum_{i \in [K], i \neq i'} \mathbb{P}(\tilde{Y} = i|X) \right) = 1 - \mathbb{P}(\tilde{Y} = i'|X).$$

□

D.1.5 Proof for Corollary 5

$$\begin{aligned}\widehat{\text{BR}}(\mathcal{P}) &= \frac{\eta + \sqrt{\frac{2 \log(4/\delta)}{N_{\mathcal{P}}}} + \mathbb{P}(\tilde{Y} \neq \tilde{Y}^*) - \sqrt{\frac{2 \log(4/\delta)}{N_{\mathcal{P}_L}}} - \mathbb{P}(Y \neq Y^*)}{\sqrt{\frac{2 \log(4/\delta)}{N_{\mathcal{P}}}} - \sqrt{\frac{2 \log(4/\delta)}{N_{\mathcal{P}_L}}}} \\ &= \frac{\Delta(N_{\mathcal{P}}, N_{\mathcal{P}_L}) - \eta}{\Delta(N_{\mathcal{P}}, N_{\mathcal{P}_L})}.\end{aligned}$$

D.2 More Definitions and Assumptions

D.2.1 Summary of Notations

Table D.1: Summary of key notations

Notation	Explanation
$\mathcal{G} := \{g_1, \dots, g_C\}$	C proxy models for generating noisy sensitive attributes
X, Y, A , and $\tilde{A} := g(X)$	Random variables of feature, label, ground-truth sensitive attribute, and noisy sensitive attributes
x_n, y_n, a_n	The n -th feature, label, and ground-truth sensitive attribute in a dataset
N, K, M	The number of instances, label classes, categories of sensitive attributes
$[N] := \{1, \dots, N\}$	A set counting from 1 to N
$\mathcal{X}, f : \mathcal{X} \rightarrow [K]$	Space of X , target model
$D^\circ := \{(x_n, y_n) n \in [N]\}$	Target dataset
$D := \{(x_n, y_n, a_n) n \in [N]\}$	D° with ground-truth sensitive attributes
$\tilde{D} := \{(x_n, y_n, (\tilde{a}_n^1, \dots, \tilde{a}_n^C)) n \in [N]\}$	D° with noisy sensitive attributes
$(X, Y, A) \sim \mathcal{D}, (X, Y, \tilde{A}) \sim \tilde{\mathcal{D}}$	Distribution of D and \tilde{D}
$u \in \{\text{DP}, \text{EOd}, \text{EOp}\}$	A unified notation of fairness definitions, e.g., EOd, EOp, EOd
$\Delta^u(\mathcal{D}, f), \tilde{\Delta}^u(\tilde{\mathcal{D}}, f), \hat{\Delta}^u(\tilde{\mathcal{D}}, f)$	True, (direct) noisy, and calibrated group fairness metrics on data distributions
$\Delta^u(D, f), \tilde{\Delta}^u(\tilde{D}, f), \hat{\Delta}^u(\tilde{D}, f)$	True, (direct) noisy, and calibrated group fairness metrics on datasets
$\mathbf{H}, \mathbf{H}[a], \mathbf{H}[:, k], \mathbf{H}[a, k]$	Fairness matrix, its a -th row, k -th column, (a, k) -th element
$\tilde{\mathbf{H}}$	Noisy fairness matrix with respect to \tilde{A}
$\mathbf{T}, T[a, \tilde{a}] := \mathbb{P}(\tilde{A} = \tilde{a} A = a)$	Global noise transition matrix
$\mathbf{T}_k, T_k[a, \tilde{a}] := \mathbb{P}(\tilde{A} = \tilde{a} A = a, f(X) = k)$	Local noise transition matrix
$\mathbf{p} := [\mathbb{P}(A = 1), \dots, \mathbb{P}(A = M)]^\top$	Clean prior probability
$\tilde{\mathbf{p}} := [\mathbb{P}(\tilde{A} = 1), \dots, \mathbb{P}(\tilde{A} = M)]^\top$	Clean prior probability

D.2.2 Common Conditional Independence Assumption in the Literature

We present below a common conditional independence assumption in the literature [8, 148, 49]. Note our algorithm successfully drops this assumption.

Assumption 5 (Conditional Independence). \tilde{A} and $f(X)$ are conditionally independent given A (and Y for EOd , EOp):

$$DP: \mathbb{P}(\tilde{A} = \tilde{a} | f(X) = k, A = a) = \mathbb{P}(\tilde{A} = \tilde{a} | A = a), \forall a, \tilde{a} \in [M], k \in [K].$$

$$(i.e. \tilde{A} \perp\!\!\!\perp f(X) | A).$$

$$EOd / EOp: \mathbb{P}(\tilde{A} = \tilde{a} | f(X) = k, Y = y, A = a)$$

$$= \mathbb{P}(\tilde{A} = \tilde{a} | Y = y, A = a), \forall a, \tilde{a} \in [M], k, y \in [K]. (i.e. \tilde{A} \perp\!\!\!\perp f(X) | Y, A).$$

D.3 Proofs

D.3.1 Full Version of Theorem 16 and Its Proof

Denote by \mathbf{T}_y the attribute noise transition matrix with respect to label y , whose (a, \tilde{a}) -th element is $T_y[a, \tilde{a}] := \mathbb{P}(\tilde{A} = \tilde{a} | A = a, Y = y)$. Note it is different from \mathbf{T}_k . Denote by $\mathbf{T}_{k \otimes y}$ the attribute noise transition matrix when $f(X) = k$ and $Y = y$, where the (a, \tilde{a}) -th element is $\mathbf{T}_{k \otimes y}[a, \tilde{a}] := \mathbb{P}(\tilde{A} = \tilde{a} | f(X) = k, Y = y, A = a)$. Denote by $\mathbf{p}_y := [\mathbb{P}(A = 1 | Y = y), \dots, \mathbb{P}(A = K | Y = y)]^\top$ and $\tilde{\mathbf{p}}_y := [\mathbb{P}(\tilde{A} = 1 | Y = y), \dots, \mathbb{P}(\tilde{A} = K | Y = y)]^\top$ the clean prior probabilities and noisy prior probability, respectively.

Theorem 3.2 (Error Upper Bound of Noisy Metrics) Denote by $\text{Err}_u^{\text{raw}} := |\Delta^u(\tilde{\mathcal{D}}, f) - \Delta^u(\mathcal{D}, f)|$ the estimation error of the directly measured noisy fairness metrics.

Its upper bound is:

- DP:

$$\text{Err}_{\text{DP}}^{\text{raw}} \leq \frac{2}{K} \sum_{k \in [K]} \left(\underbrace{\bar{h}_k \|\Lambda_{\tilde{\mathbf{p}}}(\mathbf{T}^{-1} \mathbf{T}_k - \mathbf{I}) \Lambda_{\tilde{\mathbf{p}}}^{-1}\|_1}_{\text{cond. indep. violation}} + \delta_k \underbrace{\|\Lambda_{\mathbf{p}} \mathbf{T}_k \Lambda_{\tilde{\mathbf{p}}}^{-1} - \mathbf{I}\|_1}_{\text{error of } g} \right).$$

where $\bar{h}_k := \frac{1}{M} \sum_{a \in [M]} H[a, k]$, $\delta_k := \max_{a \in [M]} |H[a, k] - \bar{h}_k|$.

- EOd:

$$\begin{aligned} & \text{Err}_{\text{EOd}}^{\text{raw}} \\ & \leq \frac{2}{K^2} \sum_{k \in [K], y \in [K]} \left(\underbrace{\bar{h}_{k \otimes y} \|\Lambda_{\tilde{\mathbf{p}}_y} (\mathbf{T}_y^{-1} \mathbf{T}_{k \otimes y} - \mathbf{I}) \Lambda_{\tilde{\mathbf{p}}_y}^{-1}\|_1}_{\text{cond. indep. violation}} + \underbrace{\delta_{k \otimes y} \|\Lambda_{\mathbf{p}_y} \mathbf{T}_{k \otimes y} \Lambda_{\tilde{\mathbf{p}}_y}^{-1} - \mathbf{I}\|_1}_{\text{error of } g} \right). \end{aligned}$$

where $\bar{h}_{k \otimes y} := \frac{1}{M} \sum_{a \in [M]} H[a, k \otimes y]$, $\delta_{k \otimes y} := \max_{a \in [M]} |H[a, k \otimes y] - \bar{h}_{k \otimes y}|$.

- EO_p: We obtain the result for EO_p by simply letting $k = 1$ and $y = 1$, i.e.,

$$\text{Err}_{\text{EO}_p}^{\text{raw}} \leq 2 \sum_{k=1, y=1} \left(\underbrace{\bar{h}_{k \otimes y} \|\Lambda_{\tilde{\mathbf{p}}_y} (\mathbf{T}_y^{-1} \mathbf{T}_{k \otimes y} - \mathbf{I}) \Lambda_{\tilde{\mathbf{p}}_y}^{-1}\|_1}_{\text{cond. indep. violation}} + \underbrace{\delta_{k \otimes y} \|\Lambda_{\mathbf{p}_y} \mathbf{T}_{k \otimes y} \Lambda_{\tilde{\mathbf{p}}_y}^{-1} - \mathbf{I}\|_1}_{\text{error of } g} \right).$$

where $\bar{h}_{k \otimes y} := \frac{1}{M} \sum_{a \in [M]} H[a, k \otimes y]$, $\delta_{k \otimes y} := \max_{a \in [M]} |H[a, k \otimes y] - \bar{h}_{k \otimes y}|$.

Proof. The following proof builds on the relationship derived in the proof for Theorem 17.

We encourage readers to check Appendix D.3.2 before the following proof.

Recall $\mathbf{T}_y[a, a'] := \mathbb{P}(\tilde{A} = a' | A = a, Y = y)$. Note

$$\Lambda_{\tilde{\mathbf{p}}_y} \mathbf{1} = \mathbf{T}_y^\top \Lambda_{\mathbf{p}_y} \mathbf{1} \Leftrightarrow (\mathbf{T}_y^\top)^{-1} \Lambda_{\tilde{\mathbf{p}}_y} \mathbf{1} = \Lambda_{\mathbf{p}_y} \mathbf{1}.$$

Denote by

$$\mathbf{H}[:, k \otimes y] = \bar{h}_{k \otimes y} \mathbf{1} + \mathbf{v}_{k \otimes y},$$

where $\bar{h}_{k \otimes y} := \frac{1}{M} \sum_{a \in [M]} \mathbb{P}(f(X) = k | A = a, Y = y)$. We have

$$\Lambda_{\mathbf{p}_y} \mathbf{H}[:, k \otimes y] = \bar{h}_{k \otimes y} \Lambda_{\mathbf{p}_y} \mathbf{1} + \Lambda_{\mathbf{p}_y} \mathbf{v}_{k \otimes y} = \bar{h}_{k \otimes y} (\mathbf{T}_y^\top)^{-1} \Lambda_{\tilde{\mathbf{p}}_y} \mathbf{1} + \Lambda_{\mathbf{p}_y} \mathbf{v}_{k \otimes y}.$$

We further have

$$\begin{aligned}
& \widetilde{\mathbf{H}}[:, k \otimes y] \\
&= \left(\Lambda_{\tilde{\mathbf{p}}_y}^{-1} \mathbf{T}_{k \otimes y}^\top \Lambda_{\mathbf{p}_y} - \mathbf{I} \right) \mathbf{H}[:, k \otimes y] + \mathbf{H}[:, k \otimes y] \\
&= \bar{h}_{k \otimes y} \Lambda_{\tilde{\mathbf{p}}_y}^{-1} \mathbf{T}_{k \otimes y}^\top (\mathbf{T}_y^\top)^{-1} \Lambda_{\tilde{\mathbf{p}}_y} \mathbf{1} + \Lambda_{\tilde{\mathbf{p}}_y}^{-1} \mathbf{T}_{k \otimes y}^\top \Lambda_{\mathbf{p}_y} \mathbf{v}_{k \otimes y} - \bar{h}_{k \otimes y} \mathbf{1} - \mathbf{v}_{k \otimes y} + \mathbf{H}[:, k \otimes y] \\
&= \bar{h}_{k \otimes y} \Lambda_{\tilde{\mathbf{p}}_y}^{-1} \left(\mathbf{T}_{k \otimes y}^\top (\mathbf{T}_y^\top)^{-1} - \mathbf{I} \right) \Lambda_{\tilde{\mathbf{p}}_y} \mathbf{1} + \left(\Lambda_{\tilde{\mathbf{p}}_y}^{-1} \mathbf{T}_{k \otimes y}^\top \Lambda_{\mathbf{p}_y} - \mathbf{I} \right) \mathbf{v}_{k \otimes y} + \mathbf{H}[:, k \otimes y].
\end{aligned}$$

Noting $|A| - |B| \leq |A + B| \leq |A| + |B|$, we have $||A + B| - |B|| \leq |A|$.

Therefore,

$$\begin{aligned}
& \left| \left| (\mathbf{e}_{\tilde{a}} - \mathbf{e}_{\tilde{a}'})^\top \widetilde{\mathbf{H}}[:, k \otimes y] \right| - \left| (\mathbf{e}_{\tilde{a}} - \mathbf{e}_{\tilde{a}'})^\top \mathbf{H}[:, k \otimes y] \right| \right| \\
& \leq \bar{h}_{k \otimes y} \left| (\mathbf{e}_{\tilde{a}} - \mathbf{e}_{\tilde{a}'})^\top \Lambda_{\tilde{\mathbf{p}}_y}^{-1} (\mathbf{T}_y^{-1} \mathbf{T}_{k \otimes y} - \mathbf{I})^\top \Lambda_{\tilde{\mathbf{p}}_y} \mathbf{1} \right| \quad (\text{Term 1}) \\
& \quad + \left| (\mathbf{e}_{\tilde{a}} - \mathbf{e}_{\tilde{a}'})^\top \left(\Lambda_{\tilde{\mathbf{p}}_y}^{-1} \mathbf{T}_{k \otimes y}^\top \Lambda_{\mathbf{p}_y} - \mathbf{I} \right) \mathbf{v}_{k \otimes y} \right|. \quad (\text{Term 2})
\end{aligned}$$

Term-1 and Term-2 can be upper bounded as follows.

Term 1: With the Hölder's inequality, we have

$$\begin{aligned}
& \bar{h}_{k \otimes y} \left| (\mathbf{e}_{\tilde{a}} - \mathbf{e}_{\tilde{a}'})^\top \Lambda_{\tilde{\mathbf{p}}_y}^{-1} (\mathbf{T}_y^{-1} \mathbf{T}_{k \otimes y} - \mathbf{I})^\top \Lambda_{\tilde{\mathbf{p}}_y} \mathbf{1} \right| \\
& \leq \bar{h}_{k \otimes y} \|\mathbf{e}_{\tilde{a}} - \mathbf{e}_{\tilde{a}'}\|_1 \left\| \Lambda_{\tilde{\mathbf{p}}_y}^{-1} (\mathbf{T}_y^{-1} \mathbf{T}_{k \otimes y} - \mathbf{I})^\top \Lambda_{\tilde{\mathbf{p}}_y} \mathbf{1} \right\|_\infty \\
& \leq 2\bar{h}_{k \otimes y} \left\| \Lambda_{\tilde{\mathbf{p}}_y}^{-1} (\mathbf{T}_y^{-1} \mathbf{T}_{k \otimes y} - \mathbf{I})^\top \Lambda_{\tilde{\mathbf{p}}_y} \mathbf{1} \right\|_\infty \\
& \leq 2\bar{h}_{k \otimes y} \left\| \Lambda_{\tilde{\mathbf{p}}_y}^{-1} (\mathbf{T}_y^{-1} \mathbf{T}_{k \otimes y} - \mathbf{I})^\top \Lambda_{\tilde{\mathbf{p}}_y} \right\|_\infty \\
& = 2\bar{h}_{k \otimes y} \left\| \Lambda_{\tilde{\mathbf{p}}_y} (\mathbf{T}_y^{-1} \mathbf{T}_{k \otimes y} - \mathbf{I}) \Lambda_{\tilde{\mathbf{p}}_y}^{-1} \right\|_1
\end{aligned}$$

Term 2: Denote by $\delta_{k \otimes y} := \max_{a \in [M]} |H[a, k \otimes y] - \bar{h}_{k \otimes y}|$, which is the largest absolute offset from its mean. With the Hölder's inequality, we have

$$\begin{aligned}
& \left| (\mathbf{e}_{\tilde{a}} - \mathbf{e}_{\tilde{a}'})^\top \left(\Lambda_{\tilde{\mathbf{p}}_y}^{-1} \mathbf{T}_{k \otimes y}^\top \Lambda_{\mathbf{p}_y} - \mathbf{I} \right) \mathbf{v}_{k \otimes y} \right| \\
& \leq \| \mathbf{e}_{\tilde{a}} - \mathbf{e}_{\tilde{a}'} \|_1 \left\| \left(\Lambda_{\tilde{\mathbf{p}}_y}^{-1} \mathbf{T}_{k \otimes y}^\top \Lambda_{\mathbf{p}_y} - \mathbf{I} \right) \mathbf{v}_{k \otimes y} \right\|_\infty \\
& \leq 2 \left\| \left(\Lambda_{\tilde{\mathbf{p}}_y}^{-1} \mathbf{T}_{k \otimes y}^\top \Lambda_{\mathbf{p}_y} - \mathbf{I} \right) \mathbf{v}_{k \otimes y} \right\|_\infty \\
& \leq 2 \delta_{k \otimes y} \left\| \Lambda_{\tilde{\mathbf{p}}_y}^{-1} \mathbf{T}_{k \otimes y}^\top \Lambda_{\mathbf{p}_y} - \mathbf{I} \right\|_\infty \\
& = 2 \delta_{k \otimes y} \left\| \Lambda_{\mathbf{p}_y} \mathbf{T}_{k \otimes y} \Lambda_{\tilde{\mathbf{p}}_y}^{-1} - \mathbf{I} \right\|_1
\end{aligned}$$

Wrap-up:

$$\begin{aligned}
& \left| \left| (\mathbf{e}_{\tilde{a}} - \mathbf{e}_{\tilde{a}'})^\top \widetilde{\mathbf{H}}[: k \otimes y] \right| - \left| (\mathbf{e}_{\tilde{a}} - \mathbf{e}_{\tilde{a}'})^\top \mathbf{H}[: k \otimes y] \right| \right| \\
& \leq 2 \bar{h}_{k \otimes y} \left\| \Lambda_{\tilde{\mathbf{p}}_y} (\mathbf{T}_y^{-1} \mathbf{T}_{k \otimes y} - \mathbf{I}) \Lambda_{\tilde{\mathbf{p}}_y}^{-1} \right\|_1 + 2 \delta_{k \otimes y} \left\| \Lambda_{\mathbf{p}_y} \mathbf{T}_{k \otimes y} \Lambda_{\tilde{\mathbf{p}}_y}^{-1} - \mathbf{I} \right\|_1.
\end{aligned}$$

Denote by $\tilde{\Delta}_{k \otimes y}^{\tilde{a}, \tilde{a}'} := |\widetilde{\mathbf{H}}[\tilde{a}, k \otimes y] - \widetilde{\mathbf{H}}[\tilde{a}', k \otimes y]|$ the noisy disparity and $\Delta_{k \otimes y}^{\tilde{a}, \tilde{a}'} := |\mathbf{H}[\tilde{a}, k \otimes y] - \mathbf{H}[\tilde{a}', k \otimes y]|$ the clean disparity between attributes \tilde{a} and \tilde{a}' in the case

when $f(X) = k$ and $Y = y$. We have

$$\begin{aligned}
& \left| \tilde{\Delta}^{\text{EOd}}(\tilde{\mathcal{D}}, f) - \Delta^{\text{EOd}}(\mathcal{D}, f) \right| \\
& \leq \frac{1}{M(M-1)K^2} \sum_{\tilde{a}, \tilde{a}' \in [M], k, y \in [K]} \left| \tilde{\Delta}_{k \otimes y}^{\tilde{a}, \tilde{a}'} - \Delta_{k \otimes y}^{\tilde{a}, \tilde{a}'} \right| \\
& \leq \frac{2}{M(M-1)K^2} \\
& \quad \cdot \sum_{\tilde{a}, \tilde{a}' \in [M], k, y \in [K]} \left(\bar{h}_{k \otimes y} \left\| \Lambda_{\tilde{\mathbf{p}}_y} (\mathbf{T}_y^{-1} \mathbf{T}_{k \otimes y} - \mathbf{I}) \Lambda_{\tilde{\mathbf{p}}_y}^{-1} \right\|_1 + \delta_{k \otimes y} \left\| \Lambda_{\mathbf{p}_y} \mathbf{T}_{k \otimes y} \Lambda_{\tilde{\mathbf{p}}_y}^{-1} - \mathbf{I} \right\|_1 \right) \\
& = \frac{2}{K^2} \sum_{k, y \in [K]} \left(\bar{h}_{k \otimes y} \left\| \Lambda_{\tilde{\mathbf{p}}_y} (\mathbf{T}_y^{-1} \mathbf{T}_{k \otimes y} - \mathbf{I}) \Lambda_{\tilde{\mathbf{p}}_y}^{-1} \right\|_1 + \delta_{k \otimes y} \left\| \Lambda_{\mathbf{p}_y} \mathbf{T}_{k \otimes y} \Lambda_{\tilde{\mathbf{p}}_y}^{-1} - \mathbf{I} \right\|_1 \right).
\end{aligned}$$

The results for DP can be obtained by dropping the dependence on $Y = y$, and the results for EOp can be obtained by letting $k = 1$ and $y = 1$. \square

D.3.2 Full Version of Theorem 17 and Its Proof

Recall \mathbf{p} , $\tilde{\mathbf{p}}$, \mathbf{T} and \mathbf{T}_k are clean prior, noisy prior, global transition matrix, and local transition matrix defined in Sec. 1.2.1. Denote by $\Lambda_{\tilde{\mathbf{p}}}$ and $\Lambda_{\mathbf{p}}$ the square diagonal matrices constructed from $\tilde{\mathbf{p}}$ and \mathbf{p} .

Theorem 4.1 (Closed-form relationship (DP,EOd,EOp)). The relationship between the true fairness vector \mathbf{h}^u and the corresponding noisy fairness vector $\tilde{\mathbf{h}}^u$ writes as

$$\mathbf{h}^u = (\mathbf{T}^{u \top} \Lambda_{\mathbf{p}^u})^{-1} \Lambda_{\tilde{\mathbf{p}}^u} \tilde{\mathbf{h}}^u, \quad \forall u \in \{\text{DP}, \text{EOd}, \text{EOp}\},$$

where $\Lambda_{\tilde{\mathbf{p}}^u}$ and $\Lambda_{\mathbf{p}^u}$ denote the square diagonal matrix constructed from $\tilde{\mathbf{p}}^u$ and \mathbf{p}^u , u unifies different fairness metrics. Particularly,

- DP ($\forall k \in [K]$): $\mathbf{p}^{\text{DP}} := [\mathbb{P}(A = 1), \dots, \mathbb{P}(A = M)]^\top$, $\tilde{\mathbf{p}}^{\text{DP}} := [\mathbb{P}(\tilde{A} = 1), \dots, \mathbb{P}(\tilde{A} = M)]^\top$. $\mathbf{T}^{\text{DP}} := \mathbf{T}_k$, where the (a, \tilde{a}) -th element of \mathbf{T}_k is $T_k[a, \tilde{a}] := \mathbb{P}(\tilde{A} = \tilde{a} | f(X) = k, A = a)$.

$$\mathbf{h}^{\text{DP}} := \mathbf{H}[:, k] := [\mathbb{P}(f(X) = k | A = 1), \dots, \mathbb{P}(f(X) = k | A = M)]^\top$$

$$\tilde{\mathbf{h}}^{\text{DP}} := \tilde{\mathbf{H}}[:, k] := [\mathbb{P}(f(X) = k | \tilde{A} = 1), \dots, \mathbb{P}(f(X) = k | \tilde{A} = M)]^\top.$$

- EOd and EO ρ ($\forall k, y \in [K], u \in \{\text{EOd}, \text{EO}\rho\}$): $\forall k, y \in [K]$: $k \otimes y := K(k - 1) + y$, $\mathbf{p}^u := \mathbf{p}_y := [\mathbb{P}(A = 1 | Y = y), \dots, \mathbb{P}(A = M | Y = y)]^\top$, $\tilde{\mathbf{p}}^u := \tilde{\mathbf{p}}_y := [\mathbb{P}(\tilde{A} = 1 | Y = y), \dots, \mathbb{P}(\tilde{A} = M | Y = y)]^\top$. $\mathbf{T}^u := \mathbf{T}_{k \otimes y}$, where the (a, \tilde{a}) -th element of $\mathbf{T}_{k \otimes y}$ is $T_{k \otimes y}[a, \tilde{a}] := \mathbb{P}(\tilde{A} = \tilde{a} | f(X) = k, Y = y, A = a)$.

$$\mathbf{h}^u := \mathbf{H}[:, k \otimes y] := [\mathbb{P}(f(X) = k | Y = y, A = 1), \dots, \mathbb{P}(f(X) = k | Y = y, A = M)]^\top$$

$$\tilde{\mathbf{h}}^u := \tilde{\mathbf{H}}[:, k \otimes y] := [\mathbb{P}(f(X) = k | Y = y, \tilde{A} = 1), \dots, \mathbb{P}(f(X) = k | Y = y, \tilde{A} = M)]^\top.$$

Proof. We first prove the theorem for DP, then for EOd and EO ρ .

Proof for DP. In DP, each element of $\tilde{\mathbf{h}}^{\text{DP}}$ satisfies:

$$\begin{aligned} & \mathbb{P}(f(X) = k | \tilde{A} = \tilde{a}) \\ &= \frac{\sum_{a \in [M]} \mathbb{P}(f(X) = k, \tilde{A} = \tilde{a}, A = a)}{\mathbb{P}(\tilde{A} = \tilde{a})} \\ &= \frac{\sum_{a \in [M]} \mathbb{P}(\tilde{A} = \tilde{a} | f(X) = k, A = a) \cdot \mathbb{P}(A = a) \cdot \mathbb{P}(f(X) = k | A = a)}{\mathbb{P}(\tilde{A} = \tilde{a})} \end{aligned}$$

Recall \mathbf{T}_k is the attribute noise transition matrix when $f(X) = k$, where the (a, \tilde{a}) -th element is $T_k[a, \tilde{a}] := \mathbb{P}(\tilde{A} = \tilde{a} | f(X) = k, A = a)$. Recall $\mathbf{p} := [\mathbb{P}(A = 1), \dots, \mathbb{P}(A = M)]^\top$ and $\tilde{\mathbf{p}} := [\mathbb{P}(\tilde{A} = 1), \dots, \mathbb{P}(\tilde{A} = M)]^\top$ the clean prior probabilities

and noisy prior probability, respectively. The above equation can be re-written as a matrix form as

$$\widetilde{\mathbf{H}}[:, k] = \mathbf{\Lambda}_{\tilde{\mathbf{p}}}^{-1} \mathbf{T}_k^\top \mathbf{\Lambda}_{\mathbf{p}} \mathbf{H}[:, k],$$

which is equivalent to

$$\mathbf{H}[:, k] = ((\mathbf{T}_k^\top) \mathbf{\Lambda}_{\mathbf{p}})^{-1} \mathbf{\Lambda}_{\tilde{\mathbf{p}}} \widetilde{\mathbf{H}}[:, k].$$

Proof for EOd, EOp. In EOd or EOp, each element of $\tilde{\mathbf{h}}^u$ satisfies:

$$\begin{aligned} & \mathbb{P}(f(X) = k | Y = y, \tilde{A} = \tilde{a}) \\ &= \frac{\mathbb{P}(f(X) = k, Y = y, \tilde{A} = \tilde{a})}{\mathbb{P}(Y = y, \tilde{A} = \tilde{a})} \\ &= \frac{\sum_{a \in [M]} \mathbb{P}(f(X) = k, Y = y, \tilde{A} = \tilde{a}, A = a)}{\mathbb{P}(Y = y, \tilde{A} = \tilde{a})} \\ &= \frac{\sum_{a \in [M]} \mathbb{P}(\tilde{A} = \tilde{a} | f(X) = k, Y = y, A = a) \mathbb{P}(Y = y, A = a) \mathbb{P}(f(X) = k | Y = y, A = a)}{\mathbb{P}(Y = y, \tilde{A} = \tilde{a})} \end{aligned}$$

Denote by $\mathbf{T}_{k \otimes y}$ the attribute noise transition matrix when $f(X) = k$ and $Y = y$, where the (a, \tilde{a}) -th element is $\mathbf{T}_{k \otimes y}[a, \tilde{a}] := \mathbb{P}(\tilde{A} = \tilde{a} | f(X) = k, Y = y, A = a)$. Denote by $\mathbf{p}_y := [\mathbb{P}(A = 1 | Y = y), \dots, \mathbb{P}(A = K | Y = y)]^\top$ and $\tilde{\mathbf{p}}_y := [\mathbb{P}(\tilde{A} = 1 | Y = y), \dots, \mathbb{P}(\tilde{A} = K | Y = y)]^\top$ the clean prior probabilities and noisy prior probability, respectively. The above equation can be re-written as a matrix form as

$$\widetilde{\mathbf{H}}[:, k] = \mathbf{\Lambda}_{\tilde{\mathbf{p}}_y}^{-1} \mathbf{T}_{k \otimes y}^\top \mathbf{\Lambda}_{\mathbf{p}_y} \mathbf{H}[:, k],$$

which is equivalent to

$$\mathbf{H}[:, k] = (\mathbf{T}_{k \otimes y}^\top \mathbf{\Lambda}_{\mathbf{p}_y})^{-1} \mathbf{\Lambda}_{\tilde{\mathbf{p}}_y} \widetilde{\mathbf{H}}[:, k].$$

Wrap-up. We can conclude the proof by unifying the above two results with u . \square

D.3.3 Proof for Corollary 6

Proof. When the conditional independence (Assumption 5)

$$\mathbb{P}(\tilde{A} = a' | A = a, Y = y) = \mathbb{P}(\tilde{A} = a' | A = a, f(X) = k, Y = y), \forall a', a \in [M]$$

holds, we have $\mathbf{T}_y = \mathbf{T}_{k \otimes y}$ and Term-1 in Theorem 16 can be dropped. For Term-2, to get a tight bound in this specific case, we apply the Hölder's inequality by using l_∞ norm on $\mathbf{e}_{\tilde{a}} - \mathbf{e}_{\tilde{a}'}$, i.e.,

$$\begin{aligned} & \left| (\mathbf{e}_{\tilde{a}} - \mathbf{e}_{\tilde{a}'})^\top \left(\mathbf{\Lambda}_{\tilde{\mathbf{p}}_y}^{-1} \mathbf{T}_{k \otimes y}^\top \mathbf{\Lambda}_{\mathbf{p}_y} - \mathbf{I} \right) \mathbf{v}_{k \otimes y} \right| \\ & \leq \| \mathbf{e}_{\tilde{a}} - \mathbf{e}_{\tilde{a}'} \|_\infty \left\| \left(\mathbf{\Lambda}_{\tilde{\mathbf{p}}_y}^{-1} \mathbf{T}_{k \otimes y}^\top \mathbf{\Lambda}_{\mathbf{p}_y} - \mathbf{I} \right) \mathbf{v}_{k \otimes y} \right\|_1 \\ & = \left\| \left(\mathbf{\Lambda}_{\tilde{\mathbf{p}}_y}^{-1} \mathbf{T}_{k \otimes y}^\top \mathbf{\Lambda}_{\mathbf{p}_y} - \mathbf{I} \right) \mathbf{v}_{k \otimes y} \right\|_1 \\ & \leq K \cdot \delta_{k \otimes y} \left\| \mathbf{\Lambda}_{\tilde{\mathbf{p}}_y}^{-1} \mathbf{T}_{k \otimes y}^\top \mathbf{\Lambda}_{\mathbf{p}_y} - \mathbf{I} \right\|_1 \\ & = K \cdot \delta_{k \otimes y} \left\| \mathbf{\Lambda}_{\mathbf{p}_y} \mathbf{T}_{k \otimes y} \mathbf{\Lambda}_{\tilde{\mathbf{p}}_y}^{-1} - \mathbf{I} \right\|_\infty \end{aligned}$$

Therefore,

$$\begin{aligned} & \left| \tilde{\Delta}^{\text{EOd}}(\tilde{\mathcal{D}}, f) - \Delta^{\text{EOd}}(\mathcal{D}, f) \right| \\ & \leq \frac{1}{K} \sum_{k, y \in [K]} \delta_{k \otimes y} \left\| \mathbf{\Lambda}_{\mathbf{p}_y} \mathbf{T}_{k \otimes y} \mathbf{\Lambda}_{\tilde{\mathbf{p}}_y}^{-1} - \mathbf{I} \right\|_\infty \\ & = \frac{1}{K} \sum_{k, y \in [K]} \delta_{k \otimes y} \left\| \mathbf{\Lambda}_{\mathbf{p}_y} \mathbf{T}_y \mathbf{\Lambda}_{\tilde{\mathbf{p}}_y}^{-1} - \mathbf{I} \right\|_\infty \\ & = \frac{1}{K} \sum_{k, y \in [K]} \delta_{k \otimes y} \left\| \check{\mathbf{T}}_y - \mathbf{I} \right\|_\infty, \end{aligned}$$

where $\check{T}_y[a, \tilde{a}] = \mathbb{P}(A = a | \tilde{A} = \tilde{a}, Y = y)$.

Special binary case in DP In addition to the conditional independence, when the sensitive attribute is binary and the label class is binary, considering DP, we have

$$\left| \tilde{\Delta}^{\text{DP}}(\tilde{\mathcal{D}}, f) - \Delta^{\text{DP}}(\mathcal{D}, f) \right| \leq 2\delta_k \|\check{\mathbf{T}} - \mathbf{I}\|_{\infty},$$

where $\check{T}_y[a, \tilde{a}] = \mathbb{P}(A = a | \tilde{A} = \tilde{a})$. Let $\check{T}_y[1, 2] = e_1, \check{T}_y[2, 1] = e_2$, we know

$$\check{\mathbf{T}} := \begin{pmatrix} 1 - e_2 & e_1 \\ e_2 & 1 - e_1 \end{pmatrix}$$

and

$$\left| \tilde{\Delta}^{\text{DP}}(\tilde{\mathcal{D}}, f) - \Delta^{\text{DP}}(\mathcal{D}, f) \right| \leq 2\delta_k \cdot (e_1 + e_2).$$

Note the equality in above inequality always holds. To prove it, firstly we note

$$\begin{aligned} & \mathbb{P}(f(X) = k | \tilde{A} = \tilde{a}) \\ &= \frac{\sum_{a \in [M]} \mathbb{P}(f(X) = k, \tilde{A} = \tilde{a}, A = a)}{\mathbb{P}(\tilde{A} = \tilde{a})} \\ &= \frac{\sum_{a \in [M]} \mathbb{P}(\tilde{A} = \tilde{a} | f(X) = k, A = a) \cdot \mathbb{P}(A = a) \cdot \mathbb{P}(f(X) = k | A = a)}{\mathbb{P}(\tilde{A} = \tilde{a})} \\ &= \frac{\sum_{a \in [M]} \mathbb{P}(\tilde{A} = \tilde{a} | A = a) \cdot \mathbb{P}(A = a) \cdot \mathbb{P}(f(X) = k | A = a)}{\mathbb{P}(\tilde{A} = \tilde{a})} \\ &= \sum_{a \in [M]} \mathbb{P}(A = a | \tilde{A} = \tilde{a}) \cdot \mathbb{P}(f(X) = k | A = a), \end{aligned}$$

i.e., $\tilde{\mathbf{H}}[:, k] = \check{\mathbf{T}}^{\top} \mathbf{H}[:, k]$. Denote by $\mathbf{H}[:, 1] = [h, h']^{\top}$. We have ($\tilde{a} \neq \tilde{a}'$)

$$\left| (\mathbf{e}_{\tilde{a}} - \mathbf{e}_{\tilde{a}'})^{\top} \tilde{\mathbf{H}}[:, 1] \right| = |h - h'| \cdot |1 - e_1 - e_2|,$$

and

$$\left| (\mathbf{e}_{\tilde{a}} - \mathbf{e}_{\tilde{a}'})^\top \mathbf{H}[:, 1] \right| = |h - h'|.$$

Therefore, letting $\tilde{a} = 1, \tilde{a}' = 2$, we have

$$\begin{aligned} & \left| \tilde{\Delta}^{\text{DP}}(\tilde{\mathcal{D}}, f) - \Delta^{\text{DP}}(\mathcal{D}, f) \right| \\ &= \frac{1}{2} \sum_{k \in \{1, 2\}} \left| \left| (\mathbf{e}_1 - \mathbf{e}_2)^\top \tilde{\mathbf{H}}[:, k] \right| - \left| (\mathbf{e}_1 - \mathbf{e}_2)^\top \mathbf{H}[:, k] \right| \right| \\ &= \left| \left| (\mathbf{e}_1 - \mathbf{e}_2)^\top \tilde{\mathbf{H}}[:, 1] \right| - \left| (\mathbf{e}_1 - \mathbf{e}_2)^\top \mathbf{H}[:, 1] \right| \right| \\ &= |h - h'| \cdot |e_1 + e_2| \\ &= \delta \cdot (e_1 + e_2), \end{aligned}$$

where $\delta = |\mathbb{P}(f(X) = 1|A = 1) - \mathbb{P}(f(X) = 1|A = 2)|$. Therefore, the equality holds. \square

D.3.4 Proof for Theorem 18

Theorem 4.5 (Error upper bound of calibrated metrics). Denote the error of the calibrated fairness metrics by $\text{Err}_u^{\text{cal}} := |\hat{\Delta}^u(\tilde{\mathcal{D}}, f) - \Delta^u(\mathcal{D}, f)|$. It can be upper bounded as:

- DP:

$$\text{Err}_{\text{DP}}^{\text{cal}} \leq \frac{2}{K} \sum_{k \in [K]} \|\mathbf{\Lambda}_p^{-1}\|_1 \|\mathbf{\Lambda}_p \mathbf{H}[:, k]\|_\infty \varepsilon(\hat{\mathbf{T}}_k, \hat{\mathbf{p}}),$$

where $\varepsilon(\hat{\mathbf{T}}_k, \hat{\mathbf{p}}) := \|\mathbf{\Lambda}_{\hat{\mathbf{p}}}^{-1} \mathbf{\Lambda}_p - \mathbf{I}\|_1 \|\mathbf{T}_k \hat{\mathbf{T}}_k^{-1}\|_1 + \|\mathbf{I} - \mathbf{T}_k \hat{\mathbf{T}}_k^{-1}\|_1$ is the error induced by calibration.

- EOd:

$$\text{Err}_{\text{EOd}}^{\text{cal}} \leq \frac{2}{K^2} \sum_{k \in [K], y \in [K]} \left\| \Lambda_{\mathbf{p}_y}^{-1} \right\|_1 \left\| \Lambda_{\mathbf{p}_y} \mathbf{H}[:, k \otimes y] \right\|_{\infty} \varepsilon(\widehat{\mathbf{T}}_{k \otimes y}, \hat{\mathbf{p}}_y),$$

where $\varepsilon(\widehat{\mathbf{T}}_{k \otimes y}, \hat{\mathbf{p}}_y) := \left\| \Lambda_{\hat{\mathbf{p}}_y}^{-1} \Lambda_{\mathbf{p}_y} - \mathbf{I} \right\|_1 \left\| \mathbf{T}_{k \otimes y} \widehat{\mathbf{T}}_{k \otimes y}^{-1} \right\|_1 + \left\| \mathbf{I} - \mathbf{T}_{k \otimes y} \widehat{\mathbf{T}}_{k \otimes y}^{-1} \right\|_1$ is the error induced by calibration.

- EO_p:

$$\text{Err}_{\text{EO}_p}^{\text{cal}} \leq 2 \sum_{k=1, y=1} \left\| \Lambda_{\mathbf{p}_y}^{-1} \right\|_1 \left\| \Lambda_{\mathbf{p}_y} \mathbf{H}[:, k \otimes y] \right\|_{\infty} \varepsilon(\widehat{\mathbf{T}}_{k \otimes y}, \hat{\mathbf{p}}_y),$$

where $\varepsilon(\widehat{\mathbf{T}}_{k \otimes y}, \hat{\mathbf{p}}_y) := \left\| \Lambda_{\hat{\mathbf{p}}_y}^{-1} \Lambda_{\mathbf{p}_y} - \mathbf{I} \right\|_1 \left\| \mathbf{T}_{k \otimes y} \widehat{\mathbf{T}}_{k \otimes y}^{-1} \right\|_1 + \left\| \mathbf{I} - \mathbf{T}_{k \otimes y} \widehat{\mathbf{T}}_{k \otimes y}^{-1} \right\|_1$ is the error induced by calibration.

Proof. We prove with EO_d.

Consider the case when $f(X) = k$ and $Y = y$. For ease of notations, we use $\widehat{\mathbf{T}}$ to denote the estimated local transition matrix (should be $\widehat{\mathbf{T}}_{k \otimes y}$). Denote the noisy (clean) fairness vectors with respect to $f(X) = k$ and $Y = y$ by $\tilde{\mathbf{h}}$ (\mathbf{h}). The error can be decomposed by

$$\begin{aligned} & \left| \left| (\mathbf{e}_a - \mathbf{e}_{a'})^\top \left(\Lambda_{\hat{\mathbf{p}}_y}^{-1} (\widehat{\mathbf{T}}^\top)^{-1} \Lambda_{\tilde{\mathbf{p}}_y} \tilde{\mathbf{h}} \right) \right| - \left| (\mathbf{e}_a - \mathbf{e}_{a'})^\top \left(\Lambda_{\mathbf{p}_y}^{-1} (\mathbf{T}_{k \otimes y}^\top)^{-1} \Lambda_{\tilde{\mathbf{p}}_y} \tilde{\mathbf{h}} \right) \right| \right| \\ &= \underbrace{\left| (\mathbf{e}_a - \mathbf{e}_{a'})^\top \left((\Lambda_{\hat{\mathbf{p}}_y}^{-1} - \Lambda_{\mathbf{p}_y}^{-1}) (\widehat{\mathbf{T}}^\top)^{-1} \Lambda_{\tilde{\mathbf{p}}_y} \tilde{\mathbf{h}} \right) \right|}_{\text{Term-1}} \\ &+ \underbrace{\left| \left| (\mathbf{e}_a - \mathbf{e}_{a'})^\top \left(\Lambda_{\hat{\mathbf{p}}_y}^{-1} (\widehat{\mathbf{T}}^\top)^{-1} \Lambda_{\tilde{\mathbf{p}}_y} \tilde{\mathbf{h}} \right) \right| - \left| (\mathbf{e}_a - \mathbf{e}_{a'})^\top \left(\Lambda_{\mathbf{p}_y}^{-1} (\mathbf{T}_{k \otimes y}^\top)^{-1} \Lambda_{\tilde{\mathbf{p}}_y} \tilde{\mathbf{h}} \right) \right| \right|}_{\text{Term-2}}. \end{aligned}$$

Now we upper bound them respectively.

Term-1:

$$\begin{aligned}
& \left| (\mathbf{e}_a - \mathbf{e}_{a'})^\top \left((\boldsymbol{\Lambda}_{\hat{\mathbf{p}}_y}^{-1} - \boldsymbol{\Lambda}_{\mathbf{p}_y}^{-1}) (\widehat{\mathbf{T}}^\top)^{-1} \boldsymbol{\Lambda}_{\hat{\mathbf{p}}_y} \tilde{\mathbf{h}} \right) \right| \\
& \stackrel{(a)}{=} \left| (\mathbf{e}_a - \mathbf{e}_{a'})^\top \left((\boldsymbol{\Lambda}_{\hat{\mathbf{p}}_y}^{-1} - \boldsymbol{\Lambda}_{\mathbf{p}_y}^{-1}) (\mathbf{T}_{k \otimes y} \widehat{\mathbf{T}}^{-1})^\top \boldsymbol{\Lambda}_{\mathbf{p}_y} \mathbf{H}[:, k \otimes y] \right) \right| \\
& \stackrel{(b)}{=} \left| (\mathbf{e}_a - \mathbf{e}_{a'})^\top \left((\boldsymbol{\Lambda}_{\hat{\mathbf{p}}_y}^{-1} \boldsymbol{\Lambda}_{\mathbf{p}_y} - \mathbf{I}) \boldsymbol{\Lambda}_{\mathbf{p}_y}^{-1} \mathbf{T}_\delta^\top \boldsymbol{\Lambda}_{\mathbf{p}_y} \mathbf{H}[:, k \otimes y] \right) \right| \\
& \leq 2 \left\| \boldsymbol{\Lambda}_{\hat{\mathbf{p}}_y}^{-1} \boldsymbol{\Lambda}_{\mathbf{p}_y} - \mathbf{I} \right\|_\infty \left\| \boldsymbol{\Lambda}_{\mathbf{p}_y}^{-1} \right\|_\infty \|\mathbf{T}_\delta\|_1 \left\| \boldsymbol{\Lambda}_{\mathbf{p}_y} \mathbf{H}[:, k \otimes y] \right\|_\infty \\
& = 2 \left\| \boldsymbol{\Lambda}_{\hat{\mathbf{p}}_y}^{-1} \right\|_\infty \left\| \boldsymbol{\Lambda}_{\mathbf{p}_y} \mathbf{H}[:, k \otimes y] \right\|_\infty \left(\left\| \boldsymbol{\Lambda}_{\hat{\mathbf{p}}_y}^{-1} \boldsymbol{\Lambda}_{\mathbf{p}_y} - \mathbf{I} \right\|_\infty \|\mathbf{T}_\delta\|_1 \right),
\end{aligned}$$

where equality (a) holds due to

$$\boldsymbol{\Lambda}_{\hat{\mathbf{p}}_y} \widehat{\mathbf{H}}[:, k \otimes y] = \mathbf{T}_{k \otimes y}^\top \boldsymbol{\Lambda}_{\mathbf{p}_y} \mathbf{H}[:, k \otimes y]$$

and equality (b) holds because we denote the error matrix by \mathbf{T}_δ , i.e.,

$$\widehat{\mathbf{T}} = \mathbf{T}_\delta^{-1} \mathbf{T}_{k \otimes y} \Leftrightarrow \mathbf{T}_\delta = \mathbf{T}_{k \otimes y} \widehat{\mathbf{T}}^{-1}.$$

Term-2: Before preceeding, we introduce the Woodbury matrix identity:

$$(\mathbf{A} + \mathbf{UCV})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1} \mathbf{U} (\mathbf{C}^{-1} + \mathbf{VA}^{-1} \mathbf{U})^{-1} \mathbf{VA}^{-1}$$

Let $\mathbf{A} := \mathbf{T}_{k \otimes y}^\top$, $\mathbf{C} = \mathbf{I}$, $\mathbf{V} := \mathbf{I}$, $\mathbf{U} := \widehat{\mathbf{T}}^\top - \mathbf{T}_{k \otimes y}^\top$. By Woodbury matrix identity, we have

$$\begin{aligned}
& (\widehat{\mathbf{T}}^\top)^{-1} \\
& = (\widehat{\mathbf{T}}_{k \otimes y}^\top + (\widehat{\mathbf{T}}^\top - \mathbf{T}_{k \otimes y}^\top))^{-1} \\
& = (\mathbf{T}_{k \otimes y}^\top)^{-1} - (\mathbf{T}_{k \otimes y}^\top)^{-1} (\widehat{\mathbf{T}}^\top - \mathbf{T}_{k \otimes y}^\top) \left(\mathbf{I} + (\mathbf{T}_{k \otimes y}^\top)^{-1} (\widehat{\mathbf{T}}^\top - \mathbf{T}_{k \otimes y}^\top) \right)^{-1} (\mathbf{T}_{k \otimes y}^\top)^{-1}
\end{aligned}$$

Term-2 can be upper bounded as:

$$\begin{aligned}
& \left| (e_a - e_{a'})^\top (\Lambda_{\mathcal{P}_y}^{-1} (\hat{\mathbf{T}}^\top)^{-1} \Lambda_{\tilde{\mathcal{P}}_y} \tilde{\mathbf{h}}) \right| - \left| (e_a - e_{a'})^\top (\Lambda_{\mathcal{P}_y}^{-1} (\mathbf{T}_{k \otimes y}^\top)^{-1} \Lambda_{\tilde{\mathcal{P}}_y} \tilde{\mathbf{h}}) \right| \\
\stackrel{(a)}{=} & \left| (e_a - e_{a'})^\top \left(\Lambda_{\mathcal{P}_y}^{-1} \left((\mathbf{T}_{k \otimes y}^\top)^{-1} - (\mathbf{T}_{k \otimes y}^\top)^{-1} (\hat{\mathbf{T}}^\top - \mathbf{T}_{k \otimes y}^\top) (\mathbf{I} + (\mathbf{T}_{k \otimes y}^\top)^{-1} (\hat{\mathbf{T}}^\top - \mathbf{T}_{k \otimes y}^\top))^{-1} (\mathbf{T}_{k \otimes y}^\top)^{-1} \right) \Lambda_{\tilde{\mathcal{P}}_y} \tilde{\mathbf{h}} \right) \right| \\
& - \left| (e_a - e_{a'})^\top (\Lambda_{\mathcal{P}_y}^{-1} (\mathbf{T}_{k \otimes y}^\top)^{-1} \Lambda_{\tilde{\mathcal{P}}_y} \tilde{\mathbf{h}}) \right| \\
\leq & \left| (e_a - e_{a'})^\top \left(\Lambda_{\mathcal{P}_y}^{-1} (\mathbf{T}_{k \otimes y}^\top)^{-1} (\hat{\mathbf{T}}^\top - \mathbf{T}_{k \otimes y}^\top) (\mathbf{I} + (\mathbf{T}_{k \otimes y}^\top)^{-1} (\hat{\mathbf{T}}^\top - \mathbf{T}_{k \otimes y}^\top))^{-1} (\mathbf{T}_{k \otimes y}^\top)^{-1} \Lambda_{\tilde{\mathcal{P}}_y} \tilde{\mathbf{h}} \right) \right| \\
\stackrel{(b)}{\leq} & \|e_a - e_{a'}\|_1 \left\| \Lambda_{\mathcal{P}_y}^{-1} (\mathbf{T}_{k \otimes y}^\top)^{-1} (\hat{\mathbf{T}}^\top - \mathbf{T}_{k \otimes y}^\top) (\mathbf{I} + (\mathbf{T}_{k \otimes y}^\top)^{-1} (\hat{\mathbf{T}}^\top - \mathbf{T}_{k \otimes y}^\top))^{-1} (\mathbf{T}_{k \otimes y}^\top)^{-1} \Lambda_{\tilde{\mathcal{P}}_y} \tilde{\mathbf{h}} \right\|_\infty \\
\leq & 2 \left\| \Lambda_{\mathcal{P}_y}^{-1} \right\|_\infty \left\| (\mathbf{T}_{k \otimes y}^\top)^{-1} (\hat{\mathbf{T}}^\top - \mathbf{T}_{k \otimes y}^\top) (\mathbf{I} + (\mathbf{T}_{k \otimes y}^\top)^{-1} (\hat{\mathbf{T}}^\top - \mathbf{T}_{k \otimes y}^\top))^{-1} (\mathbf{T}_{k \otimes y}^\top)^{-1} \Lambda_{\tilde{\mathcal{P}}_y} \tilde{\mathbf{h}} \right\|_\infty \\
= & 2 \left\| \Lambda_{\mathcal{P}_y}^{-1} \right\|_\infty \left\| (\mathbf{I} + (\mathbf{T}_{k \otimes y}^\top)^{-1} (\hat{\mathbf{T}}^\top - \mathbf{T}_{k \otimes y}^\top) - \mathbf{I}) (\mathbf{I} + (\mathbf{T}_{k \otimes y}^\top)^{-1} (\hat{\mathbf{T}}^\top - \mathbf{T}_{k \otimes y}^\top))^{-1} (\mathbf{T}_{k \otimes y}^\top)^{-1} \Lambda_{\tilde{\mathcal{P}}_y} \tilde{\mathbf{h}} \right\|_\infty \\
= & 2 \left\| \Lambda_{\mathcal{P}_y}^{-1} \right\|_\infty \left\| \left[\mathbf{I} - (\mathbf{I} + (\mathbf{T}_{k \otimes y}^\top)^{-1} (\hat{\mathbf{T}}^\top - \mathbf{T}_{k \otimes y}^\top))^{-1} \right] (\mathbf{T}_{k \otimes y}^\top)^{-1} \Lambda_{\tilde{\mathcal{P}}_y} \tilde{\mathbf{h}} \right\|_\infty \\
= & 2 \left\| \Lambda_{\mathcal{P}_y}^{-1} \right\|_\infty \left\| (\mathbf{I} - \mathbf{T}_{k \otimes y} \hat{\mathbf{T}}^{-1})^\top (\mathbf{T}_{k \otimes y}^\top)^{-1} \Lambda_{\tilde{\mathcal{P}}_y} \tilde{\mathbf{h}} \right\|_\infty \\
\stackrel{(c)}{\leq} & 2 \left\| \Lambda_{\mathcal{P}_y}^{-1} \right\|_\infty \| \mathbf{I} - \mathbf{T}_\delta \|_1 \left\| (\mathbf{T}_{k \otimes y}^\top)^{-1} \Lambda_{\tilde{\mathcal{P}}_y} \tilde{\mathbf{h}} \right\|_\infty \\
\stackrel{(d)}{=} & 2 \left\| \Lambda_{\mathcal{P}_y}^{-1} \right\|_\infty \| \mathbf{I} - \mathbf{T}_\delta \|_1 \left\| \Lambda_{\mathcal{P}_y} \mathbf{H}[:, k \otimes y] \right\|_\infty,
\end{aligned}$$

where the key steps are:

- (a): Woodbury identity.
- (b): Hölder's inequality.
- (c): $\hat{\mathbf{T}} = \mathbf{T}_\delta^{-1} \mathbf{T}_{k \otimes y}$ and triangle inequality
- (d):

$$\begin{aligned}
\widetilde{\mathbf{H}}[:, k \otimes y] &= \Lambda_{\tilde{\mathcal{P}}_y}^{-1} \mathbf{T}_{k \otimes y}^\top \Lambda_{\mathcal{P}_y} \mathbf{H}[:, k \otimes y] \\
&\Leftrightarrow (\mathbf{T}_{k \otimes y}^\top)^{-1} \Lambda_{\tilde{\mathcal{P}}_y} \widetilde{\mathbf{H}}[:, k \otimes y] = \Lambda_{\mathcal{P}_y} \mathbf{H}[:, k \otimes y].
\end{aligned}$$

Wrap-up Combining the upper bounds of Term-1 and Term-2, we have (recovering full notations)

$$\begin{aligned}
& \left| \left| (e_a - e_{a'})^\top \left(\Lambda_{\hat{p}_y}^{-1} (\hat{\mathbf{T}}^\top)^{-1} \Lambda_{\tilde{p}_y} \tilde{\mathbf{h}} \right) \right| - \left| (e_a - e_{a'})^\top \left(\Lambda_{\mathbf{p}_y}^{-1} (\mathbf{T}_{k \otimes y}^\top)^{-1} \Lambda_{\tilde{p}_y} \tilde{\mathbf{h}} \right) \right| \right| \\
& \leq 2 \left\| \Lambda_{\mathbf{p}_y}^{-1} \right\|_\infty \left\| \Lambda_{\mathbf{p}_y} \mathbf{H}[:, k \otimes y] \right\|_\infty \left(\left\| \Lambda_{\hat{p}_y}^{-1} \Lambda_{\mathbf{p}_y} - \mathbf{I} \right\|_\infty \left\| \mathbf{T}_\delta \right\|_1 + \left\| \mathbf{I} - \mathbf{T}_\delta \right\|_1 \right) \\
& = 2 \left\| \Lambda_{\mathbf{p}_y}^{-1} \right\|_\infty \left\| \Lambda_{\mathbf{p}_y} \mathbf{H}[:, k \otimes y] \right\|_\infty \left(\left\| \Lambda_{\hat{p}_y}^{-1} \Lambda_{\mathbf{p}_y} - \mathbf{I} \right\|_\infty \left\| \mathbf{T}_{k \otimes y} \hat{\mathbf{T}}_{k \otimes y}^{-1} \right\|_1 + \left\| \mathbf{I} - \mathbf{T}_{k \otimes y} \hat{\mathbf{T}}_{k \otimes y}^{-1} \right\|_1 \right).
\end{aligned}$$

Denote by $\hat{\Delta}_{k \otimes y}^{\tilde{a}, \tilde{a}'} := |\widehat{\mathbf{H}}[\tilde{a}, k \otimes y] - \widehat{\mathbf{H}}[\tilde{a}', k \otimes y]|$ the calibrated disparity and $\Delta_{k \otimes y}^{\tilde{a}, \tilde{a}'} := |\mathbf{H}[\tilde{a}, k \otimes y] - \mathbf{H}[\tilde{a}', k \otimes y]|$ the clean disparity between attributes \tilde{a} and \tilde{a}' in the case when $f(X) = k$ and $Y = y$. We have

$$\begin{aligned}
& \left| \hat{\Delta}^{\text{EOd}}(\tilde{\mathcal{D}}, f) - \Delta^{\text{EOd}}(\mathcal{D}, f) \right| \\
& \leq \frac{1}{M(M-1)K^2} \sum_{\tilde{a}, \tilde{a}' \in [M], k, y \in [K]} \left| \hat{\Delta}_{k \otimes y}^{\tilde{a}, \tilde{a}'} - \Delta_{k \otimes y}^{\tilde{a}, \tilde{a}'} \right| \\
& \leq \frac{2}{K^2} \sum_{k, y \in [K]} 2 \left\| \Lambda_{\mathbf{p}_y}^{-1} \right\|_\infty \left\| \Lambda_{\mathbf{p}_y} \mathbf{H}[:, k \otimes y] \right\|_\infty \\
& \quad \left(\left\| \Lambda_{\hat{p}_y}^{-1} \Lambda_{\mathbf{p}_y} - \mathbf{I} \right\|_\infty \left\| \mathbf{T}_{k \otimes y} \hat{\mathbf{T}}_{k \otimes y}^{-1} \right\|_1 + \left\| \mathbf{I} - \mathbf{T}_{k \otimes y} \hat{\mathbf{T}}_{k \otimes y}^{-1} \right\|_1 \right).
\end{aligned}$$

The above inequality can be generalized to DP by dropping dependency on y and to EOp by requiring $k = 1$ and $y = 1$.

□

D.3.5 Proof for Corollary 7

Proof. Consider DP. Denote by $\mathbf{H}[:, k = 1] = [h, h']^\top$. We know $\delta = |h - h'|/2 = \Delta^{\text{DP}}(\mathcal{D}, f)/2$. Suppose $p \leq 1/2$, $\|\mathbf{\Lambda}_p^{-1}\|_\infty = 1/p$ and

$$\|\mathbf{\Lambda}_p \mathbf{H}[:, k]\|_\infty = \max(ph, (1-p)h').$$

Recall

$$\varepsilon(\widehat{\mathbf{T}}_k, \hat{\mathbf{p}}) := \|\mathbf{\Lambda}_{\hat{\mathbf{p}}}^{-1} \mathbf{\Lambda}_p - \mathbf{I}\|_1 \|\mathbf{T}_k \widehat{\mathbf{T}}_k^{-1}\|_1 + \|\mathbf{I} - \mathbf{T}_k \widehat{\mathbf{T}}_k^{-1}\|_1.$$

By requiring the error upper bound in Theorem 18 less than the exact error in Corollary 6, we have (when $k = 1$)

$$\begin{aligned} \|\mathbf{\Lambda}_p^{-1}\|_\infty \|\mathbf{\Lambda}_p \mathbf{H}[:, k]\|_\infty \varepsilon(\widehat{\mathbf{T}}_k, \hat{\mathbf{p}}) &\leq \delta \cdot (e_1 + e_2) \\ \Leftrightarrow \varepsilon(\widehat{\mathbf{T}}_k, \hat{\mathbf{p}}) &\leq \frac{\delta \cdot (e_1 + e_2)}{\|\mathbf{\Lambda}_p^{-1}\|_\infty \|\mathbf{\Lambda}_p \mathbf{H}[:, k]\|_\infty} \\ \Leftrightarrow \varepsilon(\widehat{\mathbf{T}}_k, \hat{\mathbf{p}}) &\leq \frac{\delta \cdot (e_1 + e_2)}{\max(h, (1-p)h'/p)}. \end{aligned}$$

If $p = 1/2$, noting $\max(h, h') = (|h + h'| + |h - h'|)/2$, we further have (when $k = 1$)

$$\varepsilon(\widehat{\mathbf{T}}_k, \hat{\mathbf{p}}) \leq \frac{|h - h'| \cdot (e_1 + e_2)}{|h - h'| + |h + h'|} = \frac{e_1 + e_2}{1 + \frac{h+h'}{|h-h'|}} = \frac{e_1 + e_2}{1 + \frac{h+h'}{\Delta^{\text{DP}}(\mathcal{D}, f)}}.$$

To make the above equality holds for all $k \in \{1, 2\}$, we have

$$\varepsilon(\widehat{\mathbf{T}}_k, \hat{\mathbf{p}}) \leq \max_{k' \in \{1, 2\}} \frac{e_1 + e_2}{1 + \frac{\|\mathbf{H}[:, k']\|_1}{\Delta^{\text{DP}}(\mathcal{D}, f)}}, \forall k \in \{1, 2\}.$$

□

D.3.6 Differential Privacy Guarantee

We explain how we calculate the differential privacy guarantee.

Suppose $\mathbb{P}(\tilde{A} = a|A = a, X) \leq 1 - \epsilon_0$ and $\mathbb{P}(\tilde{A} = a|A = a', X) \geq \epsilon_1, \forall X, a \in$

$[M], a' \in [M], a \neq a'$. Then following the result of [53], we have

$$\begin{aligned} \frac{\mathbb{P}(\text{RandResponse}(a) = \tilde{a})}{\mathbb{P}(\text{RandResponse}(a') = \tilde{a})} &\leq \frac{\mathbb{P}(\tilde{A} = \tilde{a}|A = a, X)}{\mathbb{P}(\tilde{A} = \tilde{a}|A = a', X)} \leq \frac{\max \mathbb{P}(\tilde{A} = a|A = a, X)}{\min \mathbb{P}(\tilde{A} = a|A = a', X)} \leq \frac{1 - \epsilon_0}{\epsilon_1} \\ &= e^\epsilon. \end{aligned}$$

Then we know $\epsilon = \ln(\frac{1-\epsilon_0}{\epsilon_1})$. In practice, if proxies are too strong, i.e., $\ln(\frac{1-\epsilon_0}{\epsilon_1})$ is too large, we can add additional noise to reduce their informativeness and therefore better protect privacy. For example, when we add 40% of random noise, the the corresponding privacy guarantee is at least 0.41-DP. To get this value, noting the proxy model's accuracy of individual feature is not clear, we consider a native worst case that the model has an accuracy of 1 on some feature. Then by adding 40% of the random noise (random response), we have

$$\epsilon = \ln \frac{1 - 0.4}{0.4} < 0.41,$$

corresponding to at least 0.41-DP.

D.4 More Discussions on Transition Matrix Estimators

In this section, we extend HOCFair to a general form which can be used for EOd and EOp. For readers who are interested in details about HOC, we have provided more details in Chapter 2. We also encourage the readers to read the original papers [230, 231].

Consider a general fairness metric depending on both $f(X)$ and Y . According to the full Version of Theorem 17 in Appendix D.3.2, we need to estimate $\mathbf{T}_{k \otimes y}$ and \mathbf{p}_y ,

$\forall k \in [K], y \in [K]$. We summarize the general form of HOCFair in Algorithm 10. In this general case, our Global method in experiments adopt $\mathbf{T}_{k \otimes y} \approx \hat{\mathbf{T}}$ and $\mathbf{p}_y \approx \hat{\mathbf{p}}_y, \forall y \in [K]$. For example, considering EOp with binary attributes and binary label classes, we will estimate 4 noise transition matrices and 2 clean prior probabilities for Local, and 1 noise transition and 2 clean prior probabilities for Global.

Algorithm 10 StatEstimator: HOCFair (General)

1: **Input:** Noisy dataset \tilde{D} . Target model f .

Get the number of noisy attributes (i.e., # proxy models)

2: $C \leftarrow \# \text{Attribute}(\tilde{D})$

Get 2-Nearest-Neighbors of x_n and save their attributes as x_n 's attribute

3: **if** $C < 3$ **then**

4: $\{(x_n, y_n, (\tilde{a}_n^1, \dots, \tilde{a}_n^{3C})) | n \in [N]\} \leftarrow \text{Get2NN}(\tilde{D})$

5: $\tilde{D} \leftarrow \{(x_n, y_n, (\tilde{a}_n^1, \dots, \tilde{a}_n^{3C})) | n \in [N]\}$

6: **end if**

Randomly sample 3 noisy attributes for each instance

7: $\{(\tilde{a}_n^1, \tilde{a}_n^2, \tilde{a}_n^3) | n \in [N]\} \leftarrow \text{Sample}(\tilde{D})$

Get estimates $\mathbf{T}_k \approx \hat{\mathbf{T}}$ and $\mathbf{p} \approx \hat{\mathbf{p}}$

8: $(\hat{\mathbf{T}}, \hat{\mathbf{p}}) \leftarrow \text{HOC}(\{(\tilde{a}_n^1, \tilde{a}_n^2, \tilde{a}_n^3) | n \in [N]\})$

Get estimates $\mathbf{T}_{k \otimes y} \approx \hat{\mathbf{T}}_{k \otimes y}$, and $\mathbf{p}_y = \hat{\mathbf{p}}_y$

9: **for** $y \in [K]$ **do**

10: $(\hat{\mathbf{T}}_{k \otimes y}, \hat{\mathbf{p}}_y) \leftarrow \text{HOC}(\{(\tilde{a}_n^1, \tilde{a}_n^2, \tilde{a}_n^3) | n \in [N], f(x_n) = k, Y = y\}), \quad \forall k \in [K]$

11: **end for**

Return the estimated statistics

12: **Output:** $\hat{\mathbf{T}}, \{\hat{\mathbf{T}}_{k \otimes y} \mid k \in [K], y \in [K]\}, \{\hat{\mathbf{p}}_y \mid y \in [K]\}$
