Label Alchemy: Transforming Noisy Data into Precious Insights in Deep Learning

Ghiassi, S.

DOI
10.4233/uuid:9330d15b-c063-4908-9c9b-5bf510ecbba9

Publication date
2024

Document Version
Final published version

Citation (APA)
Ghiassi, S. (2024). *Label Alchemy: Transforming Noisy Data into Precious Insights in Deep Learning*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:9330d15b-c063-4908-9c9b-5bf510ecbba9

Label Alchemy: Transforming Noisy Data into Precious Insights in Deep Learning

LABEL ALCHEMY: TRANSFORMING NOISY DATA INTO PRECIOUS INSIGHTS IN DEEP LEARNING

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology,
by the authority of the Rector Magnificus prof. dr. ir. T.H.J.J. van der Hagen,
chair of the Board of Doctorates,
to be defended publicly on Wednesday 4 December 2024 at 17:30.

by

SeyedAmirMasoud GHIASSI

Master of Science in Information Technology Engineering,
Sharif University of Technology, Iran,
born in Tehran, Iran

This dissertation has been approved by the

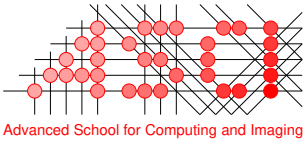
promotor: Prof. dr. ir. D.H.J. Epema
copromotor: Dr. L.Y. Chen

Composition of the doctoral committee:

Rector Magnificus,	Chairperson
Prof. dr. ir. D.H.J. Epema,	Delft University of Technology
Dr. L.Y. Chen,	Delft University of Technology

Independent members:

Prof. dr. M.T.J Spaan	Delft University of Technology
Prof. dr. M.E. van Dijk	Centrum Wiskunde & Informatica (CWI) and Vrije University
Dr. A. Anand	Delft University of Technology
Dr. P.Y. Chen	IBM Research, USA
Dr. R. Birke	University of Torino, Italy
Prof. dr. G. Smaragdakis	Delft University of Technology, reserve member



This work was carried out in the ASCI graduate school.
ASCI dissertation series number 460.

Keywords: Deep Neural Networks, Active learning, Multi-label learning, Robustness, Trusted data, Noisy labels, Noise transition matrix, Noise resilient loss

Printed by: Ridderprint (www.ridderprint.nl)

Cover: The design generated with Midjourney.

ISBN 978-94-6384-682-0

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

Science is what we have learned about
how to keep from fooling ourselves.

Richard Feynman

CONTENTS

SUMMARY

Labels are essential for training Deep Neural Networks (DNNs), guiding learning with fundamental ground truth. Label quality directly impacts DNN performance and generalization with accurate labels fostering robust predictions. Noisy labels introduce errors and hinder learning, affecting performance adversely. High-quality labels aid convergence, optimizing DNN training towards accurate data distribution representation. Ensuring label accuracy is vital for DNNs' effective learning, generalization, and real-world performance.

Undoubtedly, ensuring the quality of labels is not only critical but also demanding, often entailing considerable resources in terms of time and cost. As the scale of datasets grows, methods such as crowdsourcing have gained traction to expedite the labeling process. However, this approach comes with its own set of challenges, most notably the inherent susceptibility to errors and inaccuracies. For example, it was observed that the accuracy of AlexNet in classifying CIFAR-10 images plummeted from 77% to a mere 10% when labels were subjected to random flips. This stark drop in accuracy exemplifies the magnitude of influence that corrupted or erroneous labels can exert on the performance of DNNs. Such instances underscore the critical relationship between accurate labels and the efficacy of DNNs in understanding and effectively leveraging data.

Ensuring DNN robustness is vital, involving strategies like noise label identification, filtering, and integrating noise patterns into training for resilient models. Architectural and loss function design also combats label-related challenges, enhancing DNN adaptability across applications. This thesis investigates the pivotal role of labels in DNN training and their quality impact on model performance. Strategies spanning noise recovery, robust learning frameworks, and multi-label solutions contribute to DNN resilience against noisy labels, advancing both understanding and practical applications. Chapter 1 of this thesis introduces and explains the crucial elements involved in training DNNs, which include data, DNN models, and expert participation. It highlights the complexity introduced by label noise and sets the stage for the diverse methods designed in subsequent chapters to address these aspects comprehensively.

In Chapter 2, we delve into the complexities posed by label noise in training DNNs. To tackle this issue, two novel methods are introduced. The first method, named LABELNET, creatively harnesses noisy labels as auxiliary information to improve classification accuracy. By employing two interconnected networks, Amateur and Expert, LABELNET effectively transforms noisy labels into a learning advantage. The second method, TrustNet, focuses on estimating noise transition matrices using a small set of trusted data. This approach dynamically adjusts loss weights based on learning confidence, leading to improved model performance. Through empirical evaluations on various datasets, including challenging real-world scenarios, both LABELNET and TrustNet demonstrate their potential in enhancing the robustness of DNNs against noisy labels.

In Chapter 3, the focus shifts to addressing the challenge of learning from noisy labels under partial data supervision. The novel approach, Golden Symmetric Loss

(GSL), dynamically weighs regular and reverse cross-entropy based on an estimated corruption matrix. GSL effectively differentiates difficulty levels of classes and mitigates noise overfitting, outperforming state-of-the-art methods across various datasets. This chapter's contribution lies in the introduction of GSL, a dynamic approach that enhances the accuracy of deep learning models in the presence of noisy labels by considering noise rates and patterns, providing valuable insights into the realm of handling noisy labels in deep learning.

In Chapter 4, the thesis explores learning from noisy labels using oracle supervision. The proposed approach, Quality-driven Active Learning (QActor), combines quality models and active learning. QActor effectively filters data into clean and noisy categories, employing a noise-aware measure known as CENT for informative sample selection and dynamically adjusting query allocation based on training loss. This innovative framework achieves high accuracy with minimal oracle queries, surpassing existing methods. The chapter introduces the design of QActor, the CENT measure, and a dynamic learning strategy, addressing the dynamic allocation of an active learning budget in the presence of noisy labels, thereby offering a significant advancement in improving learning from noisy data.

In Chapter 5, we address label noise through a label aggregation approach. The proposed framework, LABNET, combines DNNs and a label aggregation algorithm to leverage knowledge from different sources and enhance label quality. The collaborative nature of LABNET bridges the gap between label aggregation and image classification, enabling iterative interactions between the two components. The DNN extracts features shared with the aggregation algorithm to enhance annotated labels, while the aggregated labels improve DNN training. The Expectation-Maximization algorithm is used for label aggregation, and an algorithm determines the optimal training timing for the DNN. LABNET demonstrates superior classification accuracy and lower label aggregation error rates on established image datasets. This chapter's contributions emphasize the collaborative essence of LABNET, offering a promising approach for improving classification accuracy while addressing label noise.

In Chapter 6, we focus on the complex realm of noisy multi-label classification. This chapter proposes two innovative mechanisms, Trusted Loss Correction for Multi-Label Learning (TLCM) and Multi-Label Loss Correction against Missing and Corrupted Labels (MLLSC), to enhance multi-label classifier accuracy in the presence of label noise. TLCM effectively estimates noise corruption matrices, improving classifier robustness against noisy labels. MLLSC mitigates the impact of missing and corrupted labels through a robust loss function. Empirical evaluations demonstrate significant improvements in mean average precision (mAP) for both methods across diverse datasets. This chapter extends the thesis's scope by addressing the challenges of noisy multi-label classification, providing comprehensive solutions that significantly enhance the accuracy and resilience of multi-label classifiers in the face of label noise.

Finally, Chapter 7 summarizes the contributions made in this thesis and outlines future research directions.

SAMENVATTING

Labels zijn essentieel voor het trainen van Deep Neural Networks (DNNs) voor het leren met fundamentele grondwaarheden. De kwaliteit van labels heeft een directe invloed op de prestaties en generalisatie van DNNs, waarbij nauwkeurige labels robuuste voorspellingen bevorderen. Labels met ruis introduceren fouten en belemmeren het leren, wat de prestaties negatief beïnvloedt. Kwalitatief hoogwaardige labels bevorderen de convergentie en de optimalisatie van het trainen van DNNs naar een nauwkeurige representatie van de distributie. Het garanderen van nauwkeurige labels is essentieel voor DNNs om effectief te leren, te generaliseren en in de echte wereld te presteren.

Het waarborgen van de kwaliteit van labels is niet alleen kritisch maar ook veeleisend en vergt vaak aanzienlijke middelen als het gaat om tijd en kosten. Naarmate de omvang van datasets toeneemt, zijn methodes zoals crowdsourcing populairder geworden om labelen van data te versnellen. Deze aanpak brengt echter eigen uitdagingen met zich mee die inherent de gevoeligheid voor fouten en onnauwkeurigheden vergroot. Zo werd bijvoorbeeld vastgesteld dat de nauwkeurigheid van AlexNet bij het classificeren van CIFAR-10 afbeeldingen verminderde van 77% naar slechts 10% toen de labels willekeurig werden omgedraaid. Deze sterke daling in nauwkeurigheid illustreert de grote invloed die beschadigde of foutieve labels kunnen hebben op de prestaties van DNNs. Dergelijke gevallen onderstrepen de cruciale relatie tussen nauwkeurige labels en de doeltreffendheid van DNNs bij het begrijpen en effectief benutten van gegevens.

Het waarborgen van de robuustheid van DNNs is van vitaal belang en omvat strategieën zoals labelidentificatie, filteren en integratie van ruispatronen in de training voor veerkrachtige modellen. Het ontwerp van architecturen en verliesfuncties gaat ook labelgerelateerde uitdagingen tegen en verbetert de aanpasbaarheid van DNN in verschillende toepassingen. Dit proefschrift onderzoekt de centrale rol van labels tijdens het trainen van DNNs en de impact op de modelprestaties. Strategieën voor ruisherstel, robuuste leerframeworks en multilabeloplossingen dragen bij aan de weerbaarheid van DNNs tegen labels met ruis, waardoor zowel het begrip als praktische toepassingen worden verbeterd.

Hoofdstuk 1 van dit proefschrift introduceert en verklaart de cruciale elementen die betrokken zijn bij het trainen van DNNs, waaronder data, DNN-modellen, en deelname van experts. Het belicht de complexiteit die labelruis met zich meebrengt en vormt de basis voor de diverse methoden die in de verdere hoofdstukken uitgebreid worden toegelicht.

In hoofdstuk 2 gaan we dieper in op de complexiteit van labelruis bij het trainen van DNNs. Om dit probleem aan te pakken worden twee nieuwe methoden geïntroduceerd. De eerste methode, genaamd LabelNet, maakt op creatieve wijze gebruik van ruis bij labels als aanvullende informatie om de classificatienauwkeurigheid te verbeteren. Door gebruik te maken van twee onderling verbonden netwerken, genaamd Amateur en Expert, transformeert LABELNET op effectieve wijze labels met ruis in een leervoordeel. De tweede

methode, TrustNet, richt zich op het schatten van ruis-overgangsmatrices met behulp van een kleine set vertrouwde gegevens. Deze aanpak past verliesgewichten dynamisch aan op basis van het leervertrouwen, wat leidt tot betere modelprestaties. Door middel van empirische evaluaties op verschillende datasets, waaronder uitdagende scenario's uit de echte wereld, tonen zowel LABELNET als TrustNet hun potentie aan in het verbeteren van de robuustheid van DNNs tegen labels met ruis.

In hoofdstuk 3 verschuift de aandacht naar de uitdaging van het leren van labels met ruis onder gedeeltelijke datatoezicht. De nieuwe aanpak, Golden Symmetric Loss (GSL), weegt op dynamische wijze reguliere en omgekeerde kruisentropie op basis van een geschatte corruptiematrix. GSL differentieert op effectieve wijze moeilijkheidsgraden van klassen en vermindert overfitting door ruis. Hierdoor presteert GSL beter dan state-of-the-art methoden in verschillende datasets. De bijdrage van dit hoofdstuk ligt in de introductie van GSL, een dynamische benadering die de nauwkeurigheid van DNNs verbetert in de aanwezigheid van labels met ruis door rekening te houden met ruispercentages en -patronen, en die waardevolle inzichten biedt in het omgaan met labels met ruis bij deep learning.

In hoofdstuk 4 onderzoekt het proefschrift het leren van labels met ruis met behulp van orakelbegeleiding. De voorgestelde aanpak, Kwaliteitsgestuurd Actief Leren (QActor), combineert kwaliteitsmodellen en actief leren. QActor filtert gegevens effectief in schone en lawaaiige categorieën, waarbij gebruik wordt gemaakt van een ruisbewuste maat die CENT wordt genoemd voor informatieve steekproefselectie en waarbij de query-toewijzing dynamisch wordt aangepast op basis van het trainingsverlies. Dit innovatieve framework bereikt een hoge nauwkeurigheid met minimale orakelvragen en overtreft daarmee bestaande methoden. Het hoofdstuk introduceert het ontwerp van QActor, de CENT-maatstaf, en een dynamische leerstrategie, die de dynamische toewijzing van een actief leerbudget aanpakt in de aanwezigheid van labels met ruis. QActor biedt daarmee een significante vooruitgang in het verbeteren van leren uit data met ruis.

In hoofdstuk 5 pakken we labelruis aan door middel van een benadering op basis van labelaggregatie. Het voorgestelde framework, LABNET, combineert DNNs en een algoritme voor labelaggregatie om kennis uit verschillende bronnen te benutten en de kwaliteit van labels te verbeteren. Het coöperatieve karakter van LABNET overbrugt de kloof tussen labelaggregatie en beeldclassificatie, waardoor iteratieve interacties tussen de twee componenten mogelijk worden. Het DNN extraheert kenmerken die worden gedeeld met het aggregatie-algoritme om geannoteerde labels te verbeteren, terwijl de geaggregeerde labels de training van DNNs verbeteren. Het verwachtingsmaximalisatie-algoritme wordt gebruikt voor labelaggregatie en een algoritme bepaalt de optimale trainingstijd voor het DNN. LABNET laat een superieure nauwkeurigheid zien op classificatie van bekende afbeeldingsdatasets en heeft tevens minder fouten bij labelaggregatie. De bijdragen in dit hoofdstuk benadrukt de essentie van LABNET en biedt een veelbelovende benadering voor het verbeteren van de classificatienauwkeurigheid terwijl labelruis wordt aangepakt.

In hoofdstuk 6 richten we ons op de complexe wereld van multi-label classificatie met veel ruis. Dit hoofdstuk stelt twee innovatieve mechanismen voor, Trusted Loss Correction for Multi-Label Learning (TLCM) en Multi-Label Loss Correction against Missing and Corrupted Labels (MLLSC), om de nauwkeurigheid van multi-label classificeerders te

verbeteren in aanwezigheid van labels met ruis. TLCM schat effectief de ruiscorruptiematrices, waardoor de robuustheid van de classificator tegen labels met ruis wordt verbeterd. MLLSC vermindert de impact van ontbrekende en beschadigde labels door middel van een robuuste verliesfunctie. Empirische evaluaties laten voor beide methoden, in diverse datasets, significante verbeteringen zien in gemiddelde precisie (mAP). Dit hoofdstuk breidt de reikwijdte van het proefschrift uit door de uitdagingen van multi-label classificatie met ruis aan te pakken, door uitgebreide oplossingen te bieden die de nauwkeurigheid en veerkracht van multi-label classificeerders aanzienlijk verbeteren in het licht van labels met ruis.

Tot slot geeft hoofdstuk 7 een samenvatting van de bijdragen in dit proefschrift en schetst toekomstige onderzoeksrichtingen.

1

INTRODUCTION

In the era of rapidly advancing technology, Deep Learning has become an integral part of our daily experiences. From powering voice-activated assistants to enabling the detection of anomalies, facilitating autonomous vehicles, and creating groundbreaking generative models like GPT and DALL-E, the applications of Deep Learning are diverse and impactful. At its core, Deep Learning, an advanced branch of Machine Learning, involves the utilization of sophisticated models trained on vast datasets. These models learn to discern intricate patterns and capture essential information, allowing them to generate novel content or accurately classify unseen data. This process has revolutionized various domains, making it a cornerstone of modern artificial intelligence.

Deep Learning operates on the principle of artificial neural networks, drawing inspiration from the structure and function of the human brain. These networks consist of layers of interconnected nodes, also known as neurons, where each connection holds a weight representing the strength of the relationship between two nodes. The process involves feeding input data into the network, propagating it through the layers, adjusting the weights based on the model's predictions, and fine-tuning the entire system through an iterative learning process. Deep Learning, as a subset of Machine Learning, relies heavily on substantial amounts of input data. The abundance of data is a critical factor in enabling these models to discern intricate patterns and relationships, ultimately enhancing their ability to make accurate predictions and generate valuable insights.

In Deep Learning, labels are important components in guiding Deep Neural Networks (DNNs) toward intelligence. These labels represent the final inference of DNNs for each data input and serve as ground truth during training, guiding the network toward accurate categorical classifications. They provide the network with precise instructions it needs to recognize meaningful patterns. The quality of the labels directly affects the accuracy of the DNN. If they are noisy or inaccurate, the network will learn incorrect patterns and produce poor results. The impact of noisy labels on DNN accuracy can be severe. In extreme cases, it can lead to the DNN learning completely incorrect patterns and producing completely useless results. For example, an autonomous vehicle is trained to identify stop signs. If the training data contains a lot of noisy labels, the vehicle may learn

to mistake other objects for stop signs, such as traffic cones or pedestrians. This could lead to the vehicle stopping in the middle of the road, causing a traffic accident.

This thesis aims to enhance the robustness of DNNs against the challenges posed by noisy labels by leveraging three key elements in training DNNs: data, DNN model, and expert supervision. We design, implement, and evaluate novel robust mechanisms that leverage these key elements by considering several aspects. The first aspect explores the relationship between clean and noisy labels. It involves utilizing DNNs to learn their correlation, investigating how the quantity of clean labels affects the learning of noise patterns, and exploring the use of DNN models for recovering correct labels from noisy ones. The next aspect focuses on training DNNs with noisy and limited clean labels while ensuring the accuracy of the DNNs remains uncompromised. This is achieved by leveraging their inherent capabilities, such as robustness in the loss function of DNNs and DNN models. In the last aspect, we consider the impact of combining key elements. We explore knowledge transfer between components, such as DNN models and expert supervision, to enhance the overall robustness of the system. Each mechanism is crafted to address specific facets of these three key elements, aiming to improve the resilience of DNNs in the face of label corruption.

1.1. BACKGROUND OF DEEP LEARNING

This thesis discusses various robust mechanisms against noisy labels. Therefore, it is important to understand how DNNs work, how labels assist models in learning various patterns within data, and finally how DNNs become noise-resilient.

1.1.1. DEEP NEURAL NETWORKS

Deep Neural Networks (DNNs) have become the workhorse of Machine Learning, particularly in areas such as computer vision and natural language processing. Their ability to learn from data and make predictions has made them invaluable tools for solving a wide range of problems. The fundamental building block of a Deep Learning model is the neural network layer [1]. It can be categorized into input, hidden, and output layers. The input layer receives raw data, which is then transformed and processed as it passes through the hidden layers. The output layer produces the final result, whether it be a classification, prediction, or generation of new content. During training, the model learns to recognize patterns and features in the input data by adjusting the weights of the connections between neurons. This adjustment is guided by a loss function, which measures the disparity between the model's predictions and the actual outcomes [1], [2]. The objective is to minimize this loss by optimizing the weights, a process achieved through backpropagation and gradient descent algorithms.

Machine Learning encompasses various types of learning paradigms, each serving distinct purposes in extracting insights from data. Supervised learning involves training a model on a labeled dataset, where the algorithm learns from both input data and corresponding target labels. This type of learning is common in tasks such as classification and regression. In contrast, unsupervised learning deals with unlabeled data, focusing on finding patterns or relationships within the input without explicit target labels. Clustering and dimensionality reduction are typical applications of unsupervised learning [1].

There are other types of learning, such as semi-supervised learning [3], which combines elements of both supervised and unsupervised learning. Reinforcement learning [4] is an interactive learning paradigm where an agent learns to make decisions by receiving feedback from its environment. Additionally, self-supervised learning [5] is a subset of unsupervised learning where the model generates its labels from the input data. Instead of relying on external labels, the algorithm creates its own supervisory signals, often by defining tasks that are easy to generate from the input. In this thesis, our focus is on supervised learning.

1.1.2. DEEP LEARNING: APPLICATIONS AND ETHICAL ISSUES

The rapid advancements in Artificial Intelligence (AI), particularly in the domain of DNNs, have ushered in a new era of innovation and transformation across diverse industries. In the healthcare sector, the application of DNNs has significantly improved medical image analysis, identifying abnormalities in medical images, enabling more accurate disease diagnosis, aiding in the discovery of novel drugs, and contributing to enhanced patient care and treatment planning [6]. In autonomous vehicles, these models play an essential role in enabling advanced driver assistance systems, enhancing navigation, and contributing to the development of self-driving cars [7]. DNNs revolutionize computer vision tasks like spotting objects, recognizing images, and segmenting scenes. They power tools that detect objects in real-time and classify images accurately [8], [9]. DNNs also help in identifying details in images and generating new visuals with specific styles, making them vital in various applications [10], [11]. Voice assistants, such as those found in smartphones and smart speakers, leverage Deep Learning for natural language processing and understanding, facilitating human-computer interaction [12]. In the field of finance, Deep Learning is employed for fraud detection, risk assessment, and algorithmic trading, where it can quickly analyze vast amounts of financial data and identify patterns or anomalies [13].

As AI rapidly evolves, critical discussions surrounding ethical considerations, including bias (prejudiced treatment based on certain characteristics), transparency (the ability to understand how AI systems reach decisions), and the potential for these systems to perpetuate unfairness, have become increasingly prominent. Biases can be introduced during the training process when the underlying data disproportionately represents specific demographics or situations. Training an image recognition system solely on pictures of cats, the resulting model would likely struggle to identify other animals. Similarly, AI systems trained on biased data might exhibit discriminatory behavior, potentially impacting high-stakes applications in healthcare, finance, or criminal justice. Mislabeling, a specific type of bias, occurs when data points are incorrectly labeled. This can happen unintentionally due to human error or intentionally through malicious manipulation. Mislabeled data disrupts the learning process, leading the AI system to develop inaccurate or unfair associations. Transparency in AI necessitates understanding not only the models' inner workings but also the quality of their training data. Noisy labels, encompassing mislabeling, incompleteness, and ambiguity, significantly hinder AI transparency. These errors introduce confusion during training, making it difficult to grasp how models reach their conclusions. In this thesis, our main focus lies on enhancing the robustness of DNNs against noisy labeled data. While we do not explicitly address bias and fairness of models,

our methods inherently contribute to mitigating these issues by improving the accuracy of the models. By addressing noisy labels, we indirectly tackle a significant source of potential bias within the data itself.

1.1.3. DEEP LEARNING: THE ROLE OF LABELS

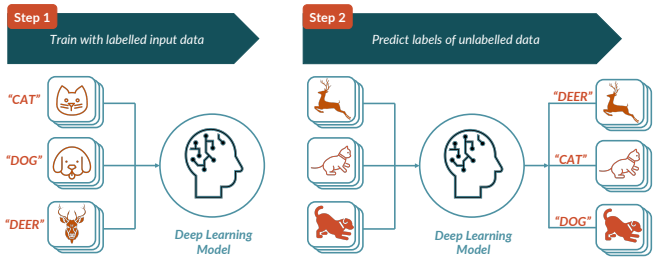
Labels are essential in supervised learning, as they provide the true reference data that guide ML models in making precise predictions and adapting to new, unseen information. In the realm of supervised learning tasks, a range of challenges arises, from straightforward single-label classification, where instances are assigned a single category, to more complex scenarios like multiple-label classification (instances with multiple labels), few-shot learning [14] (tasks with minimal labeled examples), and semi-supervised learning [3] (combining labeled and unlabeled data). In each case, labels play a central role, acting as a compass guiding machine learning algorithms towards accurate predictions and robust model performance. Despite its fundamental role, label collection is labor-intensive and resource-consuming. Consequently, labels are indispensable for initializing machine learning models, especially in supervised learning. Ensuring label cleanliness (accuracy) is imperative for training effective and dependable models.

Two important scenarios in supervised learning are single-label and multi-label learning. As shown in Figure 1.1(a), single-label classification is the most common form of supervised learning, where each data instance is associated with a single label representing its class or category. For example, in image classification, each image is labeled with a specific category like "cat" or "dog" [15]. Models are trained to predict the correct label for a given input instance. Multi-label classification (see Figure. 1.1(b)) deals with instances having multiple labels simultaneously, such as in text categorization. A news article might be labeled with both "politics" and "economy" if it covers aspects of both subjects [16]. It involves predicting the presence or absence of multiple labels for each input data point. In this scenario, the output is represented as a binary vector where each element corresponds to the presence or absence of a specific class or label. Handling *multiple labels* poses challenges like label correlation and imbalance, requiring specialized techniques for accurate predictions.

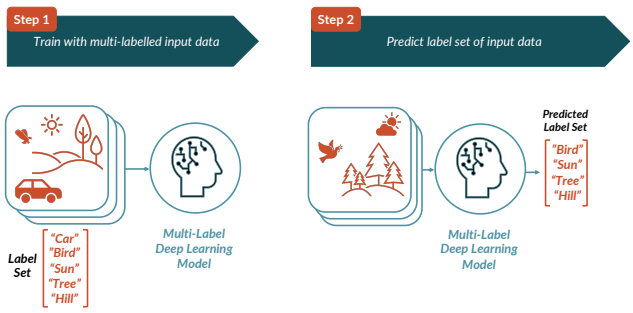
1.2. LABEL COLLECTION

Label collection can be a challenging and resource-intensive process in ML. Traditionally, labels are obtained through manual annotation by human experts, which is time-consuming and costly. However, with the advancement of technology, alternative approaches for label collection have emerged, including crowd-labeling and the use of machine learning algorithms to generate labels. Thus, there are three methods for data annotation, as illustrated in Figure 1.2: human annotation, crowdsourcing, and machine learning-based methods. Each of these methods can introduce errors into the label set, such as *corrupted* or *missing labels*.

Crowd-labeling [17], [18] involves outsourcing the labeling task to a crowd of workers, typically through online platforms. This approach allows for scalability and cost effectiveness, since multiple workers can annotate data simultaneously [19]. However, crowd-labeling comes with its own set of challenges, such as ensuring label quality and



(a) Single-label classification



(b) Multi-label classification

Figure 1.1: Training and inference processes in supervised learning for single-label and multi-label classification.

dealing with potential biases or inconsistencies among workers.

As for machine-generated labels, they can be obtained through various techniques. For example, in certain scenarios, machine learning algorithms can automatically infer labels based on patterns, clustering, or similarity measures within the data. This approach can be particularly useful when dealing with large datasets where manual annotation would be impractical [20]. It is important to note that machine-generated labeling, while efficient and scalable, can introduce errors and inaccuracies. Therefore, careful validation and quality assurance measures are essential to ensure the reliability of these labels.

In terms of cost, the process of label collection can be more expensive than gathering raw data itself. For instance, the creation of the ImageNet dataset, a widely used benchmark in computer vision, involved a significant investment of resources. Over 50,000 hours of human annotation were dedicated to labeling approximately 1.2 million images across thousands of categories [21], [22]. Another illustration of labeling expenses is Google Cloud's AI Platform Data Labeling Service, which offers human labeling for custom machine learning models. The pricing for this service is determined by the number

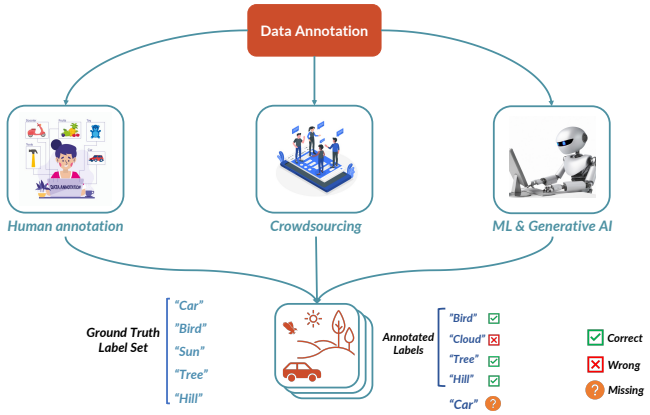


Figure 1.2: The three methods of data annotation with possible erroneous outputs.

of human labelers per data item and varies depending on the labeling task. For example, object tracking and bounding box labeling cost $86 per 1,000 units per human labeler, while event labeling in a 30-second video costs $214 per 1,000 units per human labeler [23]. Russakovsky et al. [22] explore the use of crowdsourcing platforms for image annotation and compare the cost-effectiveness of crowd-labeling with expert annotation. The study estimates that obtaining a single bounding box annotation can range from $0.03 to $10, depending on the platform and task complexity. Also, [24] estimate that labeling 1 million examples using crowd workers can cost approximately $30,000 to $50,000.

1.3. LABEL QUALITY

The quality of labels plays a crucial role in deep learning, significantly influencing the performance and reliability of models [25]–[27]. Labels can vary in quality, spanning from noisy labels, which contain errors or inaccuracies, to missing labels, where certain instances lack annotations altogether. Training accurate and robust models becomes challenging in the presence of noisy or missing labels [28]. According to IDC and IBM, the annual cost of bad data amounts to $3.1 trillion in the United States [29]. The high cost arises from the need for decision-makers, managers, knowledge workers, data scientists, and others to handle and clean noisy and missing labels, which is both time-consuming and expensive.

One prominent example highlighting the prevalence of noisy labels is the ImageNet dataset, a widely used benchmark in computer vision. Despite being a curated dataset, it is not immune to label quality issues. A study by [30] examines the quality of ImageNet labels and shows that approximately 5% of the images in the dataset have noisy labels. This indicates that labels have been annotated incorrectly due to human error or ambiguity in labeling criteria. The authors also note that certain categories, such as 'horses' and 'deer', have higher error rates compared to others, indicating the difficulty of accurately labeling similar object categories [30].

One study conducted by [31] analyzes the quality of labels in large-scale image

datasets, including ImageNet. They find that many images in these datasets are annotated with incorrect or inconsistent labels. The authors state, "The results show that even carefully curated datasets like ImageNet contain a substantial amount of labeling errors, which may affect the training and evaluation of machine learning models" [31]. Even popular and curated learning datasets include varying degrees of wrong labels, with bigger sets tending to have higher noise ratios, e.g., 10.12% for the QuickDraw dataset with 50M samples and 5.83% for the ImageNet dataset with 50K samples [26], [28].

In another investigation by [32], the authors explore the challenges of multi-label classification and analyze the quality of labels in the *MS-COCO* dataset, which encompasses a wide range of object categories. They discover that approximately 6% of the labels in the dataset are incorrect, emphasizing the importance of addressing label quality issues in multi-label classification tasks [32].

As mentioned before, machine learning algorithms themselves can contribute to label generation. Techniques such as weak supervision and active learning enable ML algorithms to generate labels based on patterns, heuristics, or human feedback. While these approaches provide scalability and cost-effectiveness, they also introduce the risk of label errors. For instance, [33] explores weak supervision techniques and highlights the inherent noise and potential errors in weakly supervised labels [33]. Hence, label quality varies considerably in machine learning.

1.4. TYPES OF NOISY LABELS

Ground-truth labels refer to the correct labels assigned to each sample in a dataset. These labels accurately represent the true class or classes to which each sample belongs, as determined by human annotators or domain experts. However, in practice, obtaining data samples with ground-truth labels is often impossible, and the available data samples may contain noisy labels. In the context of label quality, noisy labels can manifest in different forms, including corrupted labels with *symmetries* [34], [35], corrupted labels with *asymmetries* [36], and missing labels [28] in single-label (multi-class) and multi-label classification problems. Corrupted labels with symmetries refer to consistent mislabeling across instances, while corrupted labels with asymmetries involve inconsistent errors. Missing labels occur when certain labels are absent for specific samples. These challenges are further compounded in multi-label scenarios due to label skewness, where some labels appear more frequently than others. To depict the noise patterns on the labels, we utilize a corruption matrix that denotes the probability that each class is mistakenly classified as another class (see Figure 1.3).

As illustrated in Figure. 1.3(a), corrupted labels with symmetries refer to situations where label errors occur with the same probability across all the labels. This means that all labels are equally likely to be incorrectly assigned to every other class. For example, in a multi-class image classification task with images of cats, dogs, and cars, corrupted labels with symmetries would mean any cat image could be mislabeled as a dog or car with equal probability, and vice versa.

On the other hand, asymmetric label noise (as shown in Figure 1.3(b)) occurs when a specific class is more likely to be mislabeled into a particular class. In this case, the label errors are not uniformly distributed across all labels. Instead, certain classes may have a higher probability of being mislabeled than others. For instance, in the same multi-class

image classification task, if the 'cat' class is more likely to be mislabeled as 'dog' than any other class, it represents an instance of asymmetric label noise.

(a) Symmetric noise pattern (noise ratio = 40%)

(b) Asymmetric noise pattern (noise ratio = 40%)

Figure 1.3: Noise corruption matrix for a noise ratio of 0.4, illustrating both symmetric and asymmetric patterns.

Both symmetric and asymmetric label noise pose challenges in single-label and multi-label classifications. The presence of corrupted labels with symmetries can confuse the learning process and hinder the model's ability to accurately predict the true labels for each instance. Similarly, asymmetric label noise can introduce biases in the learning process, affecting the model's performance on specific classes that are more prone to mislabeling. Consequently, the model will be exposed to a disproportionate amount of incorrect information for these classes, leading to biased performance.

In multi-label classification, there are two groups of labels for every data point: negative and positive. Negative labels represent the classes not assigned to the data sample, indicating their absence [37]–[39]. Conversely, positive labels signify the presence of a class for a data point. Missing labels are common in multi-label classification when certain classes are not included in the label set for specific data samples while they should, resulting in *false negative* labels. For example, in a multi-label image classification task, an image containing a "tree" and a "sun" may only have labels for "tree" while the label for "sun" is missing. Corrupted labels can also occur, where *false positive* labels are assigned to data samples. Continuing with the previous example, a corrupted label could be assigning the "cloud" label to the image of the "sun" and "tree" (see Figure 1.1(b)). Another challenge in multi-label classification is *label skewness*, where some labels appear more frequently than others. Multi-label learning faces intrinsic label imbalances, as each image typically contains only a subset of all possible classes. This leads to a higher number of negative labels than positive labels, creating a label distribution imbalance [40]. Thus the contribution of negative labels to the loss function of DNNs is more significant, which poses difficulties in achieving accurate multi-label classification.

1.5. IMPACT OF NOISY LABELS ON DNNS

The impact of noisy labels in ML has been extensively studied across different learning paradigms, including both traditional standard learning approaches and deep learning models. In the presence of noisy labels, deep learning models are prone to the *memorization effect*, as observed in various studies [41]–[43]. This effect refers to the phenomenon where deep learning models tend to *overfit* to the noise present in the labels rather than capturing the true underlying patterns in the data. Consequently, this phenomenon leads to reduced generalization performance, as the models struggle to make accurate predictions on unseen data.

In the case of single-label classification, the presence of noisy labels can significantly affect model performance. A survey conducted by Zhang et al.[42] highlights the detrimental effects of label noise on standard classification algorithms and emphasizes the need for robust methods to mitigate the impact of corrupted labels and improve the reliability of learned models. In a study by Zhang et al. [44], it was found that the accuracy of AlexNet in classifying CIFAR-10 images dropped from 77% to 10% when labels were randomly flipped with a 40% noise ratio.

As shown in Figure. 1.4, empirical studies demonstrate the impact of bad labels on learning algorithms. Here, the DNN trained on noisy labels, with and without a regularization term in the loss function to mitigate the impact of noisy labels, achieves 100% training accuracy but only 50% testing accuracy, indicating overfitting. The significant performance gap between clean and noisy data highlights the need for robust learning systems or label recovery methods to improve accuracy.

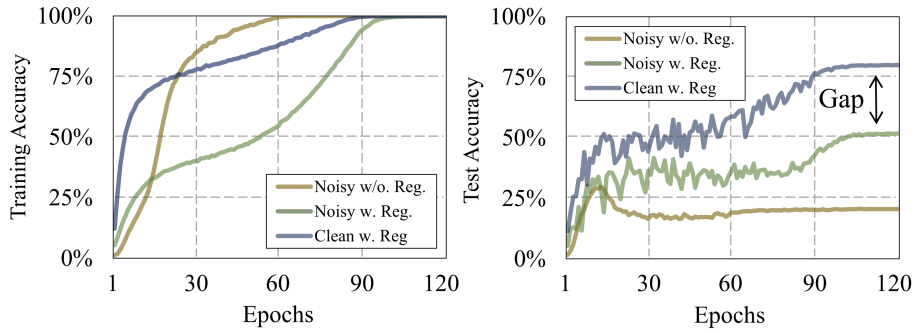


Figure 1.4: The training (left) and test (right) accuracy over epoch for the WideResNet-16-8 model trained on the CIFAR-100 dataset with 40% symmetric label noise with noisy data without regularization ('Noisy w/o. Reg.'), and noisy data with regularization ('Noisy w. Reg.'), and clean data with regularization ('Clean w. Reg.') using a standard training method [41].

Moving on to multi-label classification, the impact of bad labels becomes even more pronounced. In this setting, the presence of corrupted or missing labels can introduce complexities and affect the accuracy of predictions. A comprehensive survey conducted by Li et al. [45] delves into the challenges posed by bad labels in multi-label learning, encompassing issues such as label noise, label imbalance, and missing labels. The survey explores various techniques and strategies aimed at addressing these challenges and en-

hancing the performance and reliability of multi-label classification models. Experiments are conducted on the MS-COCO dataset to empirically evaluate the impact of label corruption in multi-label learning. Different noise levels are injected into the dataset during the training of a state-of-the-art multi-label classifier, ASL [38]. The results depicted in Figure 1.5 demonstrate that the mean Average Precision (mAP), a metric commonly used to evaluate multi-label classification by measuring the average precision across multiple classes or categories, significantly degrades with increasing noise levels (for more details, see §6).

By mitigating label noise, tackling label imbalance, and addressing missing labels, the overall performance of learning models can be significantly improved, spanning from standard learning approaches to deep learning paradigms.

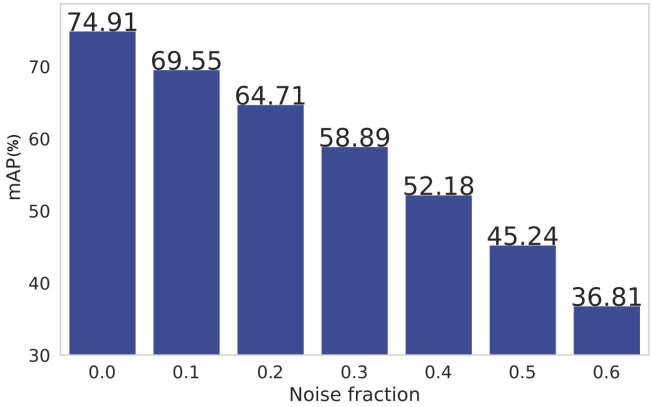


Figure 1.5: The mean Average Precision (mAP) of ASL, a multi-label classifier, under various noise fractions ranging from 0.0 to 0.6 on the MS-COCO dataset (see also Chapter 6).

1.6. LABEL QUALITY CHALLENGES FOR ROBUST LEARNING

Training robust DNNs heavily relies on the quality of labels used to train models. However, obtaining reliable labels can be challenging due to several factors. Here, the key challenges associated with label quality for robust learning are listed as follows:

1. High Cost and Time Consumption: Obtaining ground truth labels often requires human expertise, making the process expensive and time-consuming [21], [22]. This scarcity of high-quality labels significantly hinders the training of robust models.

2. Diverse and Unpredictable Noise Patterns: Labels can be corrupted by errors or inconsistencies, resulting in *noisy* labels that exhibit diverse and unpredictable patterns. Label noise can manifest in different forms, including *symmetric* and *asymmetric* noise, as well as label imbalance. This diversity complicates the task of identifying and addressing label noise, as different types of noise require different strategies for mitigation.

3. False-Positive and False-Negative Labels in Multi-Label Learning: Multi-label scenarios, where a data point can have multiple classes, introduce additional difficulties. False-positive labels (corrupted labels) and false-negative labels (missing labels) can significantly increase uncertainty during training, impacting the accuracy and reliability of the model.

4. Skewed Label Occurrence Distribution: The distribution of label occurrences is often highly skewed, with certain labels being more prevalent than others. This skewness can bias the learning process and affect the performance of DNN models, making it challenging to learn accurate representations of the data.

5. Identifying Samples with Noisy Labels: Determining which samples have noisy labels that require cleansing is a challenging task. In scenarios where noise patterns are intricate and diverse, pinpointing specific instances for correction becomes even more complex.

To address these challenges, robust learning methods have been developed to enhance the resilience of models to label noise, by both mitigating the impact of label noise and improving the reliability of learning algorithms. In the next section, we introduce the current solutions addressing the noisy labels and their challenges.

1.7. HOW TO COPE WITH NOISY LABELS?

The solutions for addressing the impact posed by noisy labels can be broadly categorized into three approaches: cleansing the labels without learning, cleansing the labels through learning techniques, and robust architectures of DNNs.

(i) **Cleansing the labels without learning**: This approach involves addressing the issue of bad labels without relying on the learning process itself. One common strategy is to utilize human labeling, where domain experts manually correct or refine the labels. Human labeling approaches can significantly improve the accuracy of the labels but can be time-consuming and expensive [46]. For example, Branson et al. [47] proposed a hybrid human-machine vision system that combines the expertise of human annotators with machine learning techniques for fine-grained categorization tasks.

Another approach is to leverage generative models to cleanse the labels. Generative models, such as generative adversarial networks (GANs), can be used to generate synthetic clean labels or augment the existing dataset with plausible examples. Reed et al. [48] introduce a generative adversarial text-to-image synthesis model, which can generate realistic images based on textual descriptions. By generating clean labels or additional data samples, these generative models can help improve the quality of the training data.

(ii) **Cleansing with learning**: We categorize this approach into two subgroups: (1) *Data correction*, and (2) *Data filtering*.

The *data correction* approach focuses on cleansing the labels during the training process itself. Techniques within the learning framework are incorporated to identify and mitigate the impact of bad labels [34], [49]–[51]. These techniques often involve iterative processes, such as bootstrapping [52], [53], self-training [54], or co-training [55]–[57]. Bootstrapping algorithms gradually refine the labels by iteratively selecting confident

predictions and using them as pseudo-labels for subsequent iterations. Self-training involves using the model's own predictions as additional labels to train a more robust model. Co-training leverages multiple views or modalities of the data to mutually reinforce the training process.

Data filtering approaches [27], [58]–[62] aim to identify and remove noisy samples or outliers from the dataset. This can be achieved through outlier detection algorithms or statistical analysis to identify samples with high uncertainty or inconsistencies. By filtering out noisy samples, the training data becomes more reliable and improves the model's performance.

(iii) **Robust Architectures of DNNs**: DNNs with noise-resilient architecture [49], [63]–[69] play a crucial role in mitigating the impact of label noise in machine learning models. These architectures encompass two key strategies: (a) *robust loss functions* and (b) *noise-tolerant layers*. Robust loss functions are specifically designed to address the challenges posed by label noise during training. These loss functions aim to reduce the influence of noisy or incorrect labels and enhance the model's resilience to such noise. For instance, the Forward Loss Correction proposed by Patrini et al. [70] modifies the loss function to assign different weights to reliable and noisy samples. By prioritizing reliable samples and down-weighting the impact of noisy samples, the model becomes more robust to label noise and can improve its generalization capability. Noise-tolerant layers [71], [72] are architectural components designed to handle label noise effectively. These layers are specifically engineered to be less sensitive to noisy labels and aid in improving the model's robustness. An example is the Noise-Adaptive Convolutional Layer [73], which adapts its behavior to handle noisy inputs more effectively. By incorporating such noise-tolerant layers, the model becomes more resilient to label noise and can maintain better performance in the presence of noisy data.

The prior art in addressing label noise and robust learning methods has made significant progress. However, there are still some limitations that need to be acknowledged, which are discussed in the next section.

1.8. LIMITATIONS IN CURRENT ROBUST MECHANISMS

Considering the challenges outlined in §1.6 and the current robust mechanisms detailed in §1.7, we identify several limitations in current approaches for managing noisy labels in DNNs:

- **Heavy Reliance on Ground Truth Labels**: Many existing mechanisms require access to a significant amount of ground truth data, which is clean and accurately labeled. Unfortunately, obtaining reliable ground truth labels is often an expensive and time-consuming process, especially for large-scale datasets. This dependence on clean labels hinders the applicability of these methods in real-world scenarios where such data may be scarce or costly.

- **Limited Noise Pattern Flexibility**: Current mechanisms often make simplifying assumptions about the nature of noise in labels. These assumptions might involve the noise being symmetric or asymmetric. Real-world noise, however, can be much more complex and may exhibit intricate patterns that these methods struggle to

handle effectively. This inflexibility limits the generalizability of existing approaches and can lead to performance degradation when dealing with unexpected noise distributions.

- **Trade-off in Filtering Methods**: One common approach to mitigating label noise involves filtering noisy samples from the training data. While this strategy aims to reduce the influence of noisy labels, it inherently faces a trade-off between sample selection and data exploration. By discarding unselected training examples, these methods achieve a reduction in the influence of noisy samples. However, this discarding process may also remove valuable training data containing correct labels, potentially leading to a decrease in overall model accuracy. Furthermore, filtering methods can limit the exploration of the training data's full feature space, which might hinder the model's ability to learn complex relationships within the data.

- **False-positive/false-negative Errors in Multi-Label Learning:** Existing robust approaches for multi-label learning often fall short in adequately addressing both false-positive (FP) and false-negative (FN) errors. In the context of multi-labels, FP denotes label corruption (incorrectly assigning a label that is not relevant) and FN signifies missing labels (failing to assign a relevant label). Current approaches typically focus on either FP or FN errors, with less emphasis on addressing them simultaneously. Methods designed for binary classification tasks might be applied to address label noise in multi-label settings, but these methods often overlook the crucial distinction between FP and FN errors in multi-label learning. This predominant focus on handling these errors individually rather than collectively fails to capture the intricate nature of label noise in multi-label scenarios, where both FP and FN errors can coexist and significantly impact model performance.

To gain deeper insights, we translate these challenges and limitations into five research questions that will guide our investigation in the next section. In response to these research questions, we propose seven robust mechanisms which are introduced in more detail in the following section. In Table 1.1, we illustrate the linkage between the research questions and each proposed mechanism.

1.9. RESEARCH QUESTIONS

This thesis investigates how to enhance the robustness of DNNs by exploring three key elements of their training process: data utilization across various ground truth ratios in conjunction with noisy labels, DNN models encompassing architecture and loss functions, and the integration of expert input like human expertise or an oracle. We aim to comprehend the collective impact of these key elements on the development of robust DNNs. For this purpose, we formulate the following five specific research questions:

[RQ1] How to effectively cleanse noisy labels? This research question revolves around finding effective methods to recover from or correct noisy labels when both the noisy labels and the corresponding ground truth labels are available. The aim is to explore approaches that leverage the availability of cleansed data to estimate and rectify the noise in the labels. The focus is on leveraging the information provided by the ground truth

labels to improve the quality of the training data and mitigate the negative impact of label noise.

[RQ2] How to effectively learn a robust classifier with only limited cleansed labels? This research question focuses on learning robust classifiers with minimal reliable data in the presence of label noise. The challenge is clear: most of our training data contains labeling errors, but we have only a small amount of accurately labeled data. The goal is to smartly use this limited correct data to enhance the accuracy and reliability of our deep learning models. These models should perform well even when facing labeling errors. This involves techniques like identifying incorrect labels, integrating correct ones into model training, and designing models that can handle errors without performance loss.

[RQ3] How to effectively leverage active learning and expert intervention mechanisms to handle noisy labels and construct a robust single-label classifier? This research question involves creating a system that utilizes quality models and active learning to select the best samples for training while managing the active learning budget wisely. The main challenge is addressing noisy labels that may require correction by experts (oracle). We aim to determine the effectiveness of this system compared to existing approaches and its contribution to enhancing our understanding of learning from noisy data.

[RQ4] How can label aggregation models be combined with DNNs to improve the accuracy of models trained on noisy labels? This research question delves into improving annotation quality and reducing errors in noisy labeled datasets through label aggregation. We are aiming to create methods that harness the power of label aggregation to make supervised learning models more accurate and dependable. This research contributes to our knowledge of how label aggregation can enhance learning from noisy data.

[RQ5] How to develop robust multi-label classifiers that are resilient to noisy and missing labels? This research question aims to develop robust multi-label classifiers that effectively handle label corruption and missing annotations in multi-label learning. The goal is to improve the accuracy and reliability of these classifiers in real-world scenarios. By addressing this question, we aim to advance techniques and approaches that mitigate the impact of label noise and incomplete annotations, ultimately enhancing the performance of multi-label classifiers.

1.10. RESEARCH METHODOLOGY

In this thesis, we use both theoretical and experimental methods for addressing each research question by designing, implementing, proving, and evaluating a robust mechanism capable of effectively handling noisy label data. As to theoretical methods, we formulate and present proofs of theorems, expanding upon existing ones in this research domain. In Chapter 2, we derive a closed-form equation that relates test accuracy to the noise ratio within the data. We utilize the lemmas and theorems introduced by [74] for our derivation. In Chapter 3, we introduce a novel bound for the noise rate, guaranteeing a noise-resilient cross-entropy loss function. To prove the new bound, we employ the method proposed by [75], [76]. Finally, in Chapter 4, we demonstrate that the average cross-entropy loss for noisy samples consistently exceeds that of clean samples.

On the experimental side, we implement all the mechanisms proposed for enhancing the robustness of DNNs in this thesis using Python, which is the predominant language in deep learning. Our primary tools and libraries for implementation include Keras,

Table 1.1: An overview of the mechanisms proposed, their code bases, and the deep learning frameworks used for each research question (proposed mechanism) in this thesis.

RQ	Mechanism	Codebase Reference	Deep Learning Framework
1	LABELNET, TrustNet	[77]	Keras, TensorFlow
2	GSL	[78]	Torch
3	QActor	[77]	Keras, TensorFlow
4	LABNET	[79]	Keras, TensorFlow
5	TLCM, MLLSC	[80]	Torch

TensorFlow, and Torch. We begin by cloning these state-of-the-art codebases, which serve as the foundation for our work. Table 1.1 details the specific repositories we clone and the deep learning framework used. Subsequently, we modify existing code and implement new functions and classes to develop our methodologies, aligning them with our research goals. To optimize performance, we then employ exhaustive hyperparameter tuning to identify the configuration that yields the best results. To ensure the generalizability of our results, we conduct multiple evaluations of the experiments in each chapter, typically ranging from 3 to 5 repetitions. We report the mean and standard deviation of these evaluations. Additionally, for each experiment, we employ a different seed for the random number generator, which is responsible for initializing the weights within DNNs. Each chapter includes a detailed description of the experiments we conduct. We train and deploy all DNNs on a machine equipped with an NVIDIA TITAN X for RQ1 and an NVIDIA GeForce RTX 2080 Ti for the remaining research questions.

We evaluate the performance of our mechanisms on a diverse set of public and widely used deep learning datasets. Table 1.2 provides an overview of the datasets employed in each research question. Since Clothing1M is the only available dataset with noisy labels, we synthetically generate noisy labels for evaluation in the other datasets in the table. The specific noise ratios and noise patterns used for each dataset are discussed in the corresponding chapters. To assess the performance of our proposed mechanisms, we evaluate their performance using the most common DNN metrics: test accuracy for single-label learning and mean average precision (mAP) for multi-label learning. We ensure the reproducibility of our experiments by providing detailed explanations of the methods in each chapter and making the codes publicly available on the 4TU repository[1].

1.11. THESIS OUTLINE AND CONTRIBUTIONS

In Chapters 2-6, we address the five research questions formulated in § 1.9. Figure 1.6 illustrates the seven robust mechanisms introduced in these five chapters, which are designed considering the three key elements involved in training DNNs and robust learning systems: data, DNN models, and experts (e.g., human expertise, crowdsourcing, etc). It shows the relationship between these mechanisms and the role of each key element in their learning process. In certain mechanisms, we leverage more than one of these learning elements. The following sections outline the content and contributions of each chapter.

[1] https://doi.org/10.4121/b00277a6-9431-47dc-9369-e9a477031e66

Table 1.2: An overview of public datasets used for the evaluation of each research question (proposed mechanism) and their properties.

RQ	Dataset	Data type	Number of samples	Number of classes
1	MNIST	Vision (single-label)	60K	10
1, 2, 3, 4	CIFAR-10	Vision (single-label)	60K	10
1, 2, 3, 4	CIFAR-100	Vision (single-label)	60K	100
1, 2	Clothing1M	Vision (single-label)	1M	14
2	Twitter	Text (single-label)	1827	25
2	Stanford Sentiment Treebank	Text (single-label)	9604	2
5	MS-COCO	Vision (multi-label)	122K	80
5	NUS-WIDE	Vision (multi-label)	269.6K	81
5	MIRFLICKR	Vision (multi-label)	25K	38

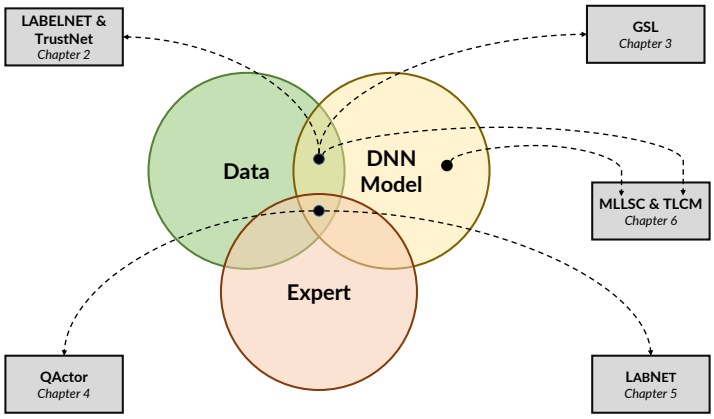


Figure 1.6: The seven mechanisms proposed in this thesis, their relation to each chapter, and how they enhance DNN robustness against noisy labels by leveraging the three key elements.

1.11.1. [CHAPTER 2] RECOVERING NOISY LABELS WITH CLEANSED DATA SUPERVISION

In this chapter, we address RQ1 and propose two novel mechanisms that leverage cleansed data, assuming both the noisy labels and the corresponding ground truth labels are available. By utilizing the ground truth labels, our mechanisms estimate the noise pattern and correct the noisy labels, thereby enhancing model accuracy.

Our first mechanism, named LABELNET, innovatively utilizes noisy labels as auxiliary information to enhance classification accuracy. Through the collaboration of two interconnected networks, Amateur and Expert, LABELNET effectively leverages clean and noisy labels, turning them into a learning asset. TrustNet, our second proposed mechanism, effectively estimates noise transition matrices using a small set of trusted data and LABELNET, and dynamically adjusts the loss weights, leading to improved model performance. Figure 1.7 shows an overview of our two proposed mechanisms. From our evaluation, we find that TrustNet surpasses state-of-the-art approaches in challenging scenarios and performs exceptionally well on large-scale real-world datasets. The combination of these two mechanisms highlights the significance of considering real-world noise patterns and

offers a promising solution to enhance the robustness of DNNs against noisy labels. This chapter is based on the following two publications:

AmirMasoud Ghiassi, Robert Birke, Rui Han and Lydia Y. Chen, "**LABELNET: Recoveries Noisy Labels**", *The International Joint Conference on Neural Networks (IJCNN)*, 2021. R. Han contributed to the proofreading.

AmirMasoud Ghiassi, Robert Birke, and Lydia Y.Chen, "**TrustNet: Learning from Trusted Data Against (A)symmetric Label Noise**", *IEEE/ACM International Conference on Big Data Computing, Applications, and Technologies (BDCAT)*, 2021.

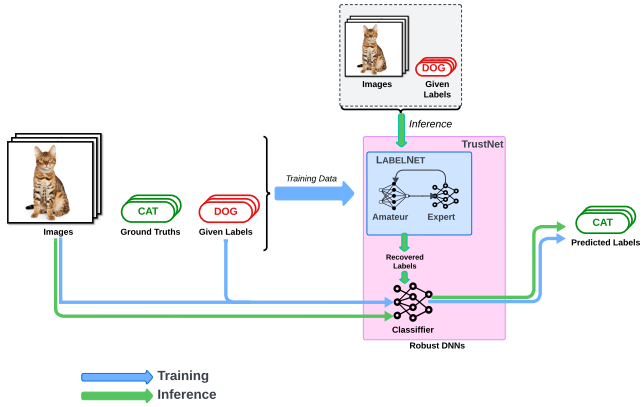


Figure 1.7: An overview of the mechanisms LABELNET and TrustNet in Chapter 2. The blue arrow depicts the training process, while the green arrow is exclusive to the inference process.

1.11.2. [CHAPTER 3] LEARNING FROM NOISY LABELS WITH PARTIAL CLEANSED DATA SUPERVISION

In this chapter, we address RQ2 and explore learning from noisy labels under partial cleansed data supervision. Partial cleansed data supervision involves only a small subset of the training data having trusted labels (clean labels), while most data are affected by noise, with no corresponding clean labels for each untrusted (noisy) one.

Annotation errors and adversarial attacks introduce inconsistencies in label quality, leading to reduced model accuracy due to overfitting to the noise structure. Existing methods employ techniques such as noisy data filtering, robust loss function derivation, or proactive label correction. Furthermore, there is a lack of methods that consider noise rates and patterns for weighting the loss for both noisy and clean labels. To overcome this, we introduce the Golden Symmetric Loss (GSL) (illustrated in Figure 1.8), which dynamically weighs regular and reverse cross-entropy using the estimated corruption matrix. GSL utilizes clean data to enhance the accuracy of the corruption matrix estimation. This mechanism tackles highly noisy labels for training deep learning models. By differentiating class difficulty levels and mitigating noise overfitting, GSL outperforms state-of-the-art methods across noise ratios in vision and text datasets. This chapter is based on the following paper:

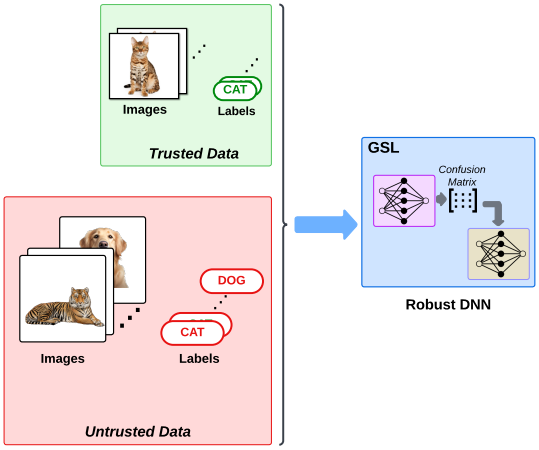


Figure 1.8: An overview of the mechanism Golden Symmetric Loss (GSL) in Chapter 3.

AmirMasoud Ghiassi, Robert Birke, and Lydia Y.Chen, "**Robust Learning via Golden Symmetric Loss of (un)Trusted Labels**", *SIAM International Conference on Data Mining (SDM)*, 2023.

1.11.3. [CHAPTER 4] LEARNING FROM NOISY LABELS WITH ORACLE SUPERVISION

In this chapter, we address RQ3 and focus on the learning from noisy labels using oracle supervision. This involves training data with noisy labels that require correction by an oracle (experts or humans). With a limited budget for obtaining correct labels from the oracle, it becomes crucial to select the most informative samples that contribute effectively to the training process.

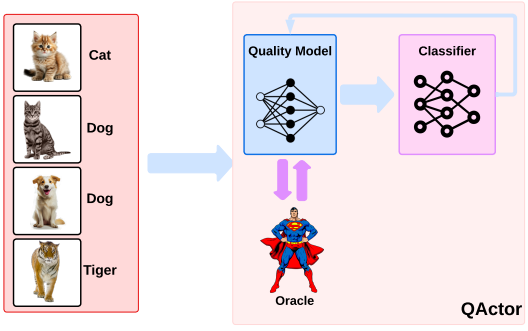


Figure 1.9: An overview of the mechanism QActor in Chapter 4.

We introduce Quality-driven Active Learning (QActor), a mechanism that combines quality models and active learning (see Figure 1.9). QActor filters data into clean and noisy categories, employs our novel proposed noise-aware measure called CENT for informative sample selection, and dynamically adjusts query allocation based on training loss. Our experimental results show that QActor achieves high accuracy with minimal oracle queries, outperforming existing methods. Contributions include QActor's design, CENT measure, and dynamic learning strategy, addressing the dynamic allocation of an active learning budget. This study is the first to address the dynamic allocation of an active learning budget in the presence of noisy labels, further enhancing the understanding and capabilities of learning from noisy data. This chapter is based on the following publication:

Taraneh Younesian, Zilong Zhao, AmirMasoud Ghiassi, Robert Birke, Lydia Y.Chen, "**QActor: Active learning on noisy labels**", *The 13th Asian Conference on Machine Learning (ACML)*, 2021. T. Younesian contributed to the idea, the proposed method, proof of the theorem, evaluation, and writing. Z. Zhao contributed to the evaluation of baselines. The author of this thesis contributed to the design of the proposed method, the proof of the theorem, evaluation, and writing.

1.11.4. [CHAPTER 5] LEARNING FROM NOISY LABELS WITH LABEL AGGREGATION APPROACH

In this chapter, we address RQ4 and focus on learning from noisy labels using a label aggregation approach, which considers multiple sources of labeling with varying capabilities. These sources may provide labels of different quality, some of which might be correct or corrupted.

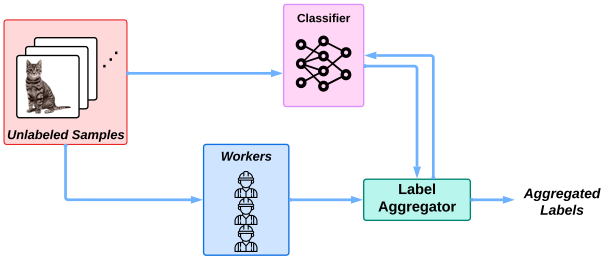


Figure 1.10: An overview of the mechanism LABNET in Chapter 5.

To tackle this scenario, we propose a collaborative mechanism called LABNET, which combines a deep neural network (DNN) model and a label aggregation method to leverage the knowledge from different sources and provide mutual feedback (see Figure 1.10). LABNET bridges the gap between label aggregation and image classification, enabling iterative interactions between the two components. The DNN extracts features that are shared with the label aggregation algorithm to enhance the quality of annotated labels, while the aggregated labels are used to train the DNN. We employ the Expectation-Maximization (EM) algorithm for label aggregation, utilizing the softmax output of the DNN as prior

probabilities. Furthermore, we introduce an algorithm to determine the optimal timing for training the DNN based on the cross-entropy between aggregated labels and previous predictions. Experimental evaluations on well-known image datasets demonstrate that LABNET achieves superior classification accuracy and lower label aggregation error rates compared to existing methods. The key features of this mechanism are rooted in the collaborative nature of LABNET, which incorporates both features and labels into the EM algorithm and DNN. Additionally, LABNET introduces an algorithm to determine the optimal timing for training the deep neural network. This algorithm relies on the cross-entropy between the aggregated labels and the labels predicted by the DNN in the previous training round. This chapter is based on the following publication:

AmirMasoud Ghiassi, Robert Birke, and Lydia Y.Chen, "**LABNET: A collaborative method for DNN training and label aggregation**", *International Conference on Agents and Artificial Intelligence (ICAART)*, 2022.

1.11.5. [CHAPTER 6] ROBUST MULTI-LABEL LEARNING

In this chapter, we address RQ5 and delve into the more complex and realistic problem of noisy multi-label classification. In multi-label learning, each data sample is associated with a set of labels that may suffer from errors or missing annotations. Our objective is to address this problem by extending the robustness of multi-class DNN methods to multi-label classifiers, ultimately enhancing the accuracy of multi-label classification in the presence of noisy labels (see Figure 1.11).

To achieve this goal, we propose two mechanisms that aim to improve the performance of multi-label classification models when dealing with noisy labels. The first method, known as Trusted Loss Correction for Multi-Label Learning (TLCM), leverages partially cleansed data and involves estimating the noise corruption matrix. By utilizing a small fraction of the trusted data, we can estimate the extent of label corruption and effectively tackle the label imbalance issue commonly encountered in multi-label learning. This approach specifically addresses the challenges posed by noisy and imbalanced label distributions, ultimately enhancing the accuracy of multi-label classifiers.

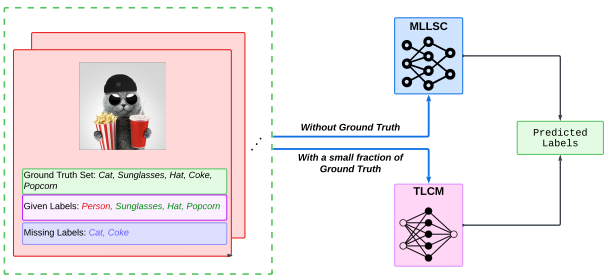


Figure 1.11: An overview of the proposed mechanisms MLLSC and TLCM in Chapter 6.

The second method we propose is a robust loss function called Multi-Label Loss Correction against Missing and Corrupted Labels (MLLSC). Unlike traditional loss functions, MLLSC does not rely on additional data or expert supervision. It is specifically designed to

mitigate the negative impact of noisy labels in multi-label classification. By incorporating robustness directly into the loss function, MLLSC improves the model's resilience to label noise, enabling enhanced performance even when clean or expert-labeled data is unavailable.

By combining these two mechanisms, we provide a comprehensive solution for effectively addressing noisy labels in multi-label classification. Our methods leverage both partial cleansed data supervision through TLCM and the robust loss function of MLLSC, allowing us to effectively handle label corruption and missing annotations. As a result, we significantly improve the accuracy and reliability of multi-label classifiers. The research presented in this chapter is based on the following papers:

AmirMasoud Ghiassi, Cosmin Octavian Pene, Robert Birke, and Lydia Y.Chen, "**Trusted Loss Correction for Noisy Multi-Label Learning**", *The 14th Asian Conference on Machine Learning (ACML)*, 2022. C.O. Pene contributed to the idea, proposed method, implementation, and evaluation.

AmirMasoud Ghiassi, Robert Birke, and Lydia Y.Chen, "**Multi Label Loss Correction against Missing and Corrupted Labels**", *The 14th Asian Conference on Machine Learning (ACML)*, 2022.

2

RECOVERING NOISY LABELS WITH CLEANSED DATA SUPERVISION

Today's availability of large-scale datasets from social media and open platforms presents significant opportunities and challenges for deep learning. However, these datasets often suffer from label noise and errors. In this work, we propose two novel approaches to enhance the robustness of deep learning models against noisy labels.

The first approach introduces LABELNET*, a framework that leverages noisy labels as learning features to improve classification performance.* LABELNET *consists of two components: Amateur and Expert. Through iterative learning, Amateur, a regular image classifier, receives feedback from Expert, simulating human expert corrections based on both noisy and ground truth labels. By effectively utilizing images and their noisy labels, the trained Amateur and Expert collaboratively infer image classes. Experimental evaluations on various datasets, including noisy versions of MNIST, CIFAR-10, CIFAR-100, and real-world data (Clothing1M), demonstrate the robustness of LabelNet in achieving accurate classification even in the presence of noise and with limited training data.*

The second approach addresses the challenge of label noise in large datasets and introduces TrustNet, a framework for robust weakly-supervised classifiers. TrustNet learns the patterns of noise corruption, including both symmetric and asymmetric patterns, from a small set of trusted data. It employs a robust loss function that weighs given labels against inferred labels derived from the learned noise pattern. Additionally, TrustNet dynamically adjusts the weights based on model uncertainty during training. Evaluations on synthetic label noise using CIFAR-10, CIFAR-100, and real-world datasets (Clothing1M) demonstrate the superior robustness of TrustNet across diverse noise patterns, outperforming state-of-the-art methods.

2.1. INTRODUCTION

The proliferation of big data systems has brought about a wealth of opportunities for deep learning. However, the presence of label noise and errors in large-scale datasets poses significant challenges to the effectiveness of deep learning models [81]. Label noise refers to the incorrect or corrupted labels associated with training data, which can adversely affect model performance. Addressing this issue has become a crucial research area, focusing on developing robust approaches that can handle label noise and improve the reliability and accuracy of deep learning models.

The first approach introduces the LabelNet framework, designed to enhance the robustness of deep models against noisy labels [82]. LabelNet consists of two components: Amateur and Expert. The Amateur component functions as a regular image classifier trained with feedback from the Expert component. The Expert component emulates the corrections that human experts would make to the predicted labels generated by the Amateur using the noise patterns derived from both the noisy and ground truth labels. By leveraging the images and their noisy labels, the trained Amateur and Expert collaboratively infer image classes, thereby improving classification performance even in the presence of label noise.

Meanwhile, the second approach addresses the challenges posed by label noise in large datasets [83]. The authors present the TrustNet framework, a robust learning approach specifically tailored for weakly-supervised classifiers. TrustNet starts by learning the patterns of noise corruption, encompassing both symmetric and asymmetric patterns, from a small set of trusted data. It utilizes a robust loss function that weighs the given labels against the inferred labels derived from the learned noise pattern. Additionally, TrustNet dynamically adjusts the weights based on model uncertainty during training, further enhancing its resilience against label noise.

While prior research has primarily focused on synthetic noise patterns, such as symmetric and asymmetric noise, both approaches acknowledge the importance of addressing real-world noise patterns prevalent in big data sets [25], [34], [84]. Real-world noise patterns often exhibit more intricate and diverse characteristics, necessitating specialized techniques to accurately model and handle them.

By presenting the first and second approaches, this paper aims to tackle the challenges of label noise in big data systems and enhance the robustness of deep learning models against noisy labels. The LabelNet and TrustNet frameworks offer promising solutions by incorporating the noisy labels as learning features and leveraging noise pattern modeling and dynamic weighting strategies. Through comprehensive evaluations on various datasets, including synthetic label noise and real-world big data sets, both approaches demonstrate their effectiveness in achieving higher accuracy and robustness compared to existing state-of-the-art methods [82], [83].

2.2. PROBLEM STATEMENT OF LABELNET

Motivated by the significant impact of noisy labels, the prior art [85] derives different robust deep networks with a central theme to distill the influence of noisy labels in the model training process without the knowledge of the label ground truth. As a result, the learned networks can robustly classify images in a stand-alone manner. D2L [34]

estimates the Local Intrinsic Dimension (LID) at each epoch as a proxy to indicate the existence and impact of dirty labels. Co-teaching [61] trains two networks simultaneously by exchanging the weights updated from possibly clean data. Forward [70] uses a noise-aware correction matrix to correct labels and train the network. Bootstrap [52] has a loss function which combines predicted and noisy labels.

While prior art significantly improves the robustness of deep networks, the pre-assumed scenarios overlook the opportunity of noisy labels. On the one hand, today's image data are often bundled with labels of questionable quality and detrimental impact on the learning. On the other hand, such labels provide auxiliary information which can compliment the learnt knowledge of deep networks trained solely on image inputs. The core idea behind visual-semantic models, e.g., DeVise [86], is to combine the learning capacities of labeled images and annotated data. C-GAN [87] (conditional generative adversarial network) improves the quality of images synthesized by the generator network via additional label information and RC-GAN [88] further addresses the challenge of dirty labels for C-GAN.

In this chapter, we advocate to leverage the noisy labels as an additional feature to derive a stronger classifier. We consider learning scenarios where at training time both the ground truth and noisy labels are available, and only noisy labels at inference time. In particular difficult classification problems, whose labels require a high degree of expertise, can fit this scenario well. One such example is cancer detection from medical images. This is a daunting task, and even trained experts are prone to make errors. Hence, these images are evaluated by multiple doctors of varying expertise. In such a setting, both noisy (first evaluation by one expert) and true labels (e.g., stemming from a committee or subsequent in-depth exams) are available at the same time.

We derive a robust network, namely LABELNET, composed of Amateur and Expert, where the former classifies images based on the feedback from Expert and the latter learns how to correct the output of Amateur like human experts. Both models are trained simultaneously at each minibatch. Amateur learns to classify the input images to the corrected labels from Expert, and the softmax output of Amateur plus the given labels are inputs to train Expert to match the ground truth. Amateur can be seen as a regular image classifier, which Expert helps it to be aware of the presence of noisy labels. Such trained Amateur and Expert can then classify images based on the image and corresponding noisy label.

We empirically evaluate LABELNET on synthetic noise injected into MNIST, CIFAR-10 and CIFAR-100, and noise drawn from real world contained in Clothing1M. For a fair comparison with state-of-the-art robust deep models, we present the classification accuracy in both Amateur only and complete LABELNET model under different subsets of training inputs. LABELNET consistently outperforms existing image-only models, i.e., D2L, Co-teaching, Forward and Boostrap, especially for CIFAR-100. When using the same amount of training data, LABELNET can achieve absolute accuracy improvements of 5% up to 30%. LABELNET reaches similar or higher accuracy than image-only models even with just 20% of training data in the case of CIFAR-100.

Our contribution can be summarized as follows. First, we derive a novel network framework, i.e., LABELNET, that turns noisy labels into auxiliary learning advantages via imitating human experts. Secondly, we significantly improve the robustness of deep

networks against noisy labels compared to models based on images only.

The problem considered here is as follows. Images collected in the public domain are tagged with pre-existing noisy labels, whose true classes can be corrupted. We assume that label noises follow random distribution. We illustrate in Figure. 2.1 (black elements only) the learning procedure that is commonly deployed by robust deep networks [34], [52], [61], [70], [89]. The deep networks are trained by a set of images and labels, which are noisy, meaning with incorrect label classes. The objective of the training process is to minimize the loss function, which may be modified to be noise tolerant [34]. The network architecture may consist of different components, e.g., two networks that train each other in parallel [61] or sequentially [70] via stochastic gradient descent. In the inference phase, images are then fed into the trained network, and the prediction accuracy is computed based on true labels. The core idea behind such a learning process is to filter out the negative impact of noisy labels during training and learn a model from *clean* information.

In contrast to indirectly learning the noise dynamics of the label, our core idea is to leverage noisy labels as part of the training and inference input, as shown in Figure. 2.1 (including green elements), to directly learn the noisy label dynamics and incorporate that as auxiliary input into the training process. To such an end, the ground truth of labels is assumed from human experts or oracles and provided as part of the training input. Essentially, the networks are trained by three inputs: images, their noisy labels, and the ground truth labels. Afterward, the trained network will be tested on images and their noisy labels. The classifier can then classify images based on image inputs and limited label info.

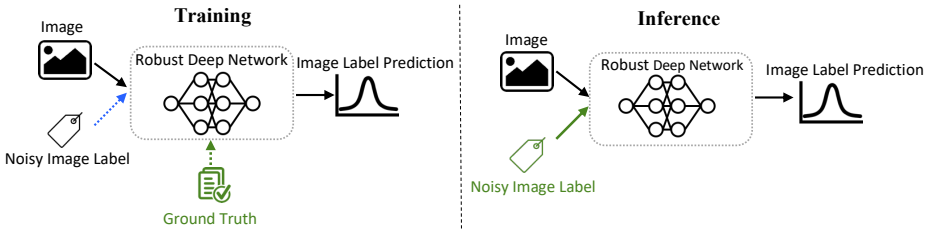


Figure 2.1: Training (left) and inference (right) with noisy labels for regular noise resilient methods (without green elements) vs. LABELNET (with green elements).

2.3. PROBLEM STATEMENT OF TRUSTNET

Nowadays big data systems allow collecting and processing immense datasets which shifts the bottleneck for deep learning from computing resources to providing high quality labels [81]. Big data systems allowed for a surge of massive self-generated data. However, it is shown that big data training sets collected from the wild can contain corrupted labels as high as 40% [25]. Even popular and curated learning datasets include varying degrees of wrong labels with bigger sets tending to have higher noise ratios, e.g., 10.12% for the QuickDraw dataset with 50M samples and 5.83% for the ImageNet dataset with 50K samples [26], [28]. The high learning capacity of deep neural networks can memorize the pattern of correct data and, unfortunately, dirty data as well [43]. As a result, when

training on data with non-negligible dirty labels, the learning accuracy of deep neural networks can significantly drop [44].

While the prior art deems it imperative to derive robust neural networks that are resilient to label noise, there is a disparity in which noise patterns to consider and evaluate. The majority of deep networks robust against dirty labels focuses on synthetic label noise, which can be symmetric or asymmetric. The former case [74] assumes noise labels can be corrupted into any other classes with equal probability. The later case [63] assumes only a particular set of classes are swapped, e.g., truck images are often mislabeled as automobile class in CIFAR-10. Patterns of noisy labels observed from real-life big data sets, e.g., Clothing1M [25], exhibit not only high percentages of label noise but also more complicated patterns mixing symmetric and asymmetric noises. Moreover, there is disagreement among related work on which noise patterns are more detrimental and difficult to defend against for regular networks [34], [84].

Noise patterns are commonly captured in transition matrices [74], which describe the probability of how a true label is corrupted into another fake and observable label. A large body of prior art estimates such a label transition matrix without knowing the true labels and incorporates such information into the learning process [70]. Accurate estimation of the transition matrix can improve the robustness of neural networks, but it is extremely complicated when lacking the information on true labels and encountering sophisticated noise patterns [83], [90].

Joint training on clean and adversarial examples with known ground truth is shown effective [91], [92] to enhance the robustness of deep models against noisy and poisonous labels. Nonetheless, it is costly to obtain label ground truth. To take advantage of adversarial examples and avoid its high overhead, we advocate to use only a fraction of *trusted data* that contain not only given labels but also the expert-validated true labels for the same. Moreover, we opt to use such small set to mainly supervise the training of transition matrix, instead of supervising the classifier directly as done in most adversarial learning.

In this paper, we first develop a thorough understanding of the noise patterns, ranging from symmetric and asymmetric. We extend the analysis from [74] and derive the generalized analysis for classification test accuracy under any given noise pattern. Our theoretical analysis compares real-world noise patterns against synthetic, symmetric, and simple asymmetric, noise. Our findings on a diverse set of noise patterns lead us to focus on challenging cases where existing robust networks [34], [70], [93] may fall short of defending against. The second contribution of this paper is to introduce a new robust learning framework TrustNet. Specifically, we adopt the idea in LABELNET [82] to estimate the noise transition matrix via training on a small set of trusted data, i.e., 10% of the training data and provide estimated labels – additional label information. TrustNet extends LABELNET by weighting the loss of the given labels and inferred labels to enhance the model performance. The specific weights are dynamically adjusted every epoch, based on the model confidence. Thirdly, we evaluate TrustNet on multiple big data vision sets. We use the curated CIFAR-10 and CIFAR-100 sets with labels corrupted by synthetically generated noise transition patterns. TrustNet is able to achieve higher accuracy than SCL [63], D2L [34], Boostrap [94], Forward [70], and Co-teaching+ [95] in all most challenging scenarios. We also demonstrate the effectiveness of TrustNet on a big data vision set collected in the wild, i.e., Clothing1M, again achieving higher accuracy

than state-of-the-art baselines.

2.4. RELATED WORK

The problem of noisy labeled data has been addressed in several recent studies. We first summarize the impact of noise patterns, followed by the defense strategies that specifically leverage noise patterns.

2.4.1. NOISY LABELS IN BIG DATA

The existence of wrong labels in Big Data sets is inevitable [28]. Several studies indicate the presence of noisy labels in both training sets [96]–[98] and testing sets [22], [26], [28]. The amount of errors, i.e. noise level, varies according to the label collection method, the annotators expertise, and, most relevantly, the size of the dataset [99]. For instance, [100] and [97], [101] study the noisy labels in the WebVision and ImageNet datasets, respectively, two of popular big vision datasets with over 24.M and 14M images. However, this phenomena goes beyond image labelling. Recent studies [26], [28] find label errors in many even highly popular learning datasets from diverse domains.

2.4.2. IMPACT OF NOISE PATTERNS

Understanding the effect of label noise on the performance of the learning models is crucial to make them robust. The impact of label noise in deep neural networks is first characterized [74] by the theoretical testing accuracy over a limited set of noise patterns. We generalize the theoretical test accuracy proposed by [74] for different noise patterns by using a generic transition matrix. [84] suggests an undirected graphical model for modeling label noise in deep neural networks, indicating the symmetric noise to be more challenging than asymmetric. Multiple untrusted data sources are studied by [102], considering label noise as one of the attributes of mistrust. However, it remains unclear how various kinds of noise patterns impact learning.

2.4.3. NOISE RESILIENT NETWORKS

SYMMETRIC NOISE

The following studies tackle the problem of symmetric label noise, meaning that corrupted labels can be any of the remaining classes with equal probability. One approach is to train the network based on noise resilient loss functions. D2L [34] monitors the changes in Local Intrinsic Dimension (LID) and incorporates LID into their loss function for the symmetric label noise. [49] introduces a loss correction technique and estimates a label corruption matrix for symmetric and asymmetric noise. Leveraging two different neural networks is another method to overcome label noise. Co-teaching [61] and Co-teaching+ [95] trains two neural networks while crossing the samples with the smallest loss between the networks for both noise patterns. [85] combats uniform label flipping via a curriculum provided by the MentorNet for the StudentNet. However, these works do not explicitly model the noise pattern in their resilient models. Although LABELNET [82] learns the noise pattern by training a DNN with ground truth and noisy labels, it requires the ground truth of all the samples. We aim to solve this issue by reducing the dependency on the ground truth via TrustNet.

ASYMMETRIC NOISE

Another stream of related work considers both symmetric and asymmetric noise. One key idea is to differentiate clean and noisy samples by exploring their dissimilarity. [103], [104] introduce class prototypes for each class and compare the samples with the prototypes to detect noisy and clean samples. Decoupling [105] uses two neural networks and updates the networks when a disagreement happens between the networks. Estimation of the noise transition matrix is another line of research to overcome label noise, introduced in Masking [106] and Forward [70] to correct the labels. However, these studies fail to consider the information in the noisy labels to estimate the matrix. Building a robust loss function against label noise has been studied in the following works, although the dynamics of the learning model seem to be neglected. SCL [63] and [107] provide robust loss function by adding regularization term. Bootstrapping [94] combines perceptual consistency with the prediction objective by using a reconstruction loss. Meta-Weight-Net [57] uses multi-layer perceptron to re-weight samples during learning process in the loss function. With the same perspective, [108] re-weights samples based on their similarity to a clean validation set. The studies [109], [110] changes the architecture of the neural network to tackle the problem. In this work, we study both symmetric and various kinds of asymmetric label noise. We leverage the information of the trusted data, containing both noisy labels and ground truth, to accurately estimate the noise transition matrix. Furthermore, we benefit from a dynamic update in our proposed loss function to tackle the label noise problem.

2.5. UNDERSTANDING DNNS TRAINED WITH NOISY LABELS

In this section, we present theoretical analysis on the test accuracy of deep neural networks assumed to have high learning capacity. Test accuracy is a common metric defined as the probability that the predicted label is equal to the given label. We extend prior art results [74] by deriving test accuracy for generic label noise distributions. We apply our formulation on three exemplary study cases and verify the theoretical values against experimental results. Finally, we compare test accuracy curves for different noise patterns providing insights on their difficulty for regular networks.

2.5.1. PRELIMINARIES

Consider the classification problem having dataset $\mathscr{D} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), ..., (\boldsymbol{x}_N, y_N)\}$ where $\boldsymbol{x}_k$ denotes the k^{th} observed sample, and $y_k \in C := \{0, ..., c-1\}$ the corresponding given class label over c classes affected by label noise. Let $\mathscr{F}(\cdot, \boldsymbol{\theta})$ denote a neural network parameterized by $\boldsymbol{\theta}$, and $y^{\mathscr{F}}$ denote the predicted label of $\boldsymbol{x}$ given by the network $y^{\mathscr{F}} = \mathscr{F}(\boldsymbol{x}, \boldsymbol{\theta})$. The label corruption process is characterised by a transition matrix $T_{ij} = P(y = j | \hat{y} = i)$ where $\hat{y}$ is the true label. Synthetic noise patterns are expressed as a label corruption probability ε plus a noise label distribution. For example, symmetric noise is defined by ε describing the corruption probability, i.e. $T_{ii} = 1 - \varepsilon, \forall i \in C$, plus a uniform label distribution across the other labels, i.e. $T_{ij} = \frac{\varepsilon}{c-1}, \forall i \neq j \in C$.

2.5.2. GENERALIZATION OF TEST ACCURACY

To generalize the previous test accuracy [74], we first consider the case where all classes are affected by the same noise ratio. We then further extend to the case where only a subset of classes is affected by noise. To derive the following Lemmas we assume that $\mathscr{F}$ is a perfect Deep Neural Network (DNN) having sufficient high capacity to learn the given pattern with high accuracy. This is the same assumption used by related work, i.e. [44], [74].

All class noise: All classes are affected by the same noise ratio ε, i.e., meaning only $1-\varepsilon$ percentage of given labels are the true labels.

Lemma 1 *For noise with fixed noise ratio ε and any given label distribution with probability function $P(y=j), \forall j \neq i$, where $i \in C$ is the true label, the test accuracy is*

$$P(y^{\mathscr{F}} = y) = (1-\varepsilon)^2 + \varepsilon^2 \sum_{j \neq i}^{C} P^2(y=j) \tag{2.1}$$

Proof: We have that $T_{ii} = 1-\varepsilon, \forall i \in C$ since all classes are affected by the same noise ratio. Moreover, the probability of selecting noisy class labels is scaled by the noise ratio $T_{ij} = \varepsilon\, P(y=j), j \neq i \in C$. Now:

$$\begin{aligned} P(y^{\mathscr{F}} = y) &= \sum_{i}^{C} P(\hat{y}=i) P(y^{\mathscr{F}} = y | \hat{y}=i) = \sum_{i}^{C} P(\hat{y}=i) \sum_{j}^{C} T_{ij}^2 \\ &= \sum_{i}^{C} P(\hat{y}=i)[T_{ii}^2 + \sum_{j \neq i}^{C} T_{ij}^2] = \sum_{i}^{C} P(\hat{y}=i)[(1-\varepsilon)^2 + \varepsilon^2 \sum_{j \neq i}^{C} P^2(y=j)]. \end{aligned} \tag{2.2}$$

Since $\sum_{i}^{C} P(\hat{y}=i) = 1$, we obtain Eq. 2.1. □

Partial class noise: in this pattern only a subset S of class labels are affected by a noise ratio, whereas the set $U = C \setminus S$ is unaffected by any label noise.

Lemma 2 *For partial class noise with equal class label probability, where S is the set affected by noise with ratio ε and U is the set of unaffected labels, for any true label $i \in C$ and any given label distribution with probability function $P(y=j), \forall j \neq i$, the test accuracy is*

$$P(y^{\mathscr{F}} = y) = \frac{|U|}{|C|} + \frac{|S|}{|C|}[(1-\varepsilon)^2 + \varepsilon^2 \sum_{j \neq i}^{S} P^2(y=j)] \tag{2.3}$$

Proof: We have that for affected labels in S the same noise transition definitions hold, i.e. $T_{ii} = 1-\varepsilon, \forall i \in S$ and $T_{ij} = \varepsilon\, P(y=j), j \neq i \in S$. For unaffected labels we have that $\varepsilon = 0$ hence $T_{ii} = 1, \forall i \in U$ and $T_{ij} = 0, j \neq i \in U$. Moreover, $P(\hat{y}=i) = \frac{1}{|C|}$ assuming all

class labels are equally probable. Now:

$$
\begin{aligned}
P(y^f = y) &= \sum_{i}^{C} P(\hat{y} = i) P(y^f = y | \hat{y} = i) \\
&= \sum_{i}^{|U|} P(\hat{y} = i) P(y^f = y | \hat{y} = i) + \sum_{i'}^{|S|} P(\hat{y} = i') P(y^f = y | \hat{y} = i') \\
&= \sum_{i}^{U} P(\hat{y} = i) \sum_{j}^{U} T_{ij}^2 + \sum_{i'}^{S} P(\hat{y} = i') \sum_{j'}^{S} T_{i'j'}^2 \\
&= \sum_{i}^{U} P(\hat{y} = i) [T_{ii}^2 + \sum_{j \neq i}^{U} T_{ij}^2] + \sum_{i'}^{S} P(\hat{y} = i') [T_{i'i'}^2 + \sum_{j' \neq i'}^{S} T_{i'j'}^2] \\
&= \frac{1}{|C|} \sum_{i}^{U} [T_{ii}^2 + \sum_{j \neq i}^{U} T_{ij}^2] + \frac{1}{|C|} \sum_{i'}^{S} [T_{i'i'}^2 + \sum_{j' \neq i'}^{S} T_{i'j'}^2] \\
&= \frac{1}{|C|} \sum_{i}^{U} 1 + \frac{1}{|C|} \sum_{i'}^{S} [(1-\varepsilon)^2 + \varepsilon^2 \sum_{j' \neq i'}^{S} P^2(y = j')] \\
&= \frac{|U|}{|C|} + \frac{|S|}{|C|} [(1-\varepsilon)^2 + \varepsilon^2 \sum_{j' \neq i'}^{S} P^2(y = j')]
\end{aligned}
$$

□

The goal of Lemma 1 and Lemma 2 is to generalize the test accuracy proposed by [74] to noises characterized by a generic transition matrix T_{ij}.

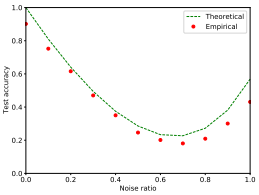
2.5.3. VALIDATION OF THEORETICAL ANALYSIS

We validate our extension of test accuracy on three various noise patterns for CIFAR-10 under different noise ratios and comparing the theoretical estimation with empirical accuracy results.

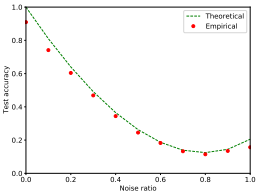
As the first new noise pattern, we consider noisy class labels following a truncated normal distribution $\mathcal{N}^T(\mu, \sigma, a, b)$. This noise pattern is motivated by the targeted adversarial attacks [111]. We scale $\mathcal{N}^T(\mu, \sigma, a, b)$ by the number of classes and center it around a target class $\tilde{c}$ by setting $\mu = \tilde{c}$ and use σ to control how spread out the noise is. a and b simply define the class label boundaries, i.e. $a = 0$ and $b = c - 1$. To compute the test accuracy, we estimate the empirical distribution at the different classes and apply Eq. 2.1. The second noise pattern extends our previous case. This distribution, referred in short as bimodal hereon, combines two truncated normal distributions. It has two peaks in μ_1 and μ_2 with two different shapes controlled by σ_1 and σ_2. The peaks are centered on two different target classes $\mu_1 = \tilde{c}_1$ and $\mu_2 = \tilde{c}_2$. The third noise pattern considers partial targeted noise where only a subset of classes, $[2, 3, 4, 5, 9]$ in our example, are affected by targeted noise, i.e. swapped with a specific other class. Here we rely on Eq. 2.3 to estimate test accuracy. This noise pattern has been studied in [63].

Figure. 2.2 summarizes the results. The first row compares the theoretical curves against the empirical results obtained by corrupting CIFAR-10 dataset with different noise ratios from clean to fully corrupted data: $0 \leq \varepsilon \leq 1$. The highest deviation between theoretical (lines) and empirical (points) results occurs for truncated normal noise around

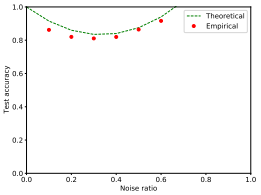
$\varepsilon = 1.0$. Here the theoretical accuracy is 13.46% points worse than the measured accuracy. For the other two, the deviation is at most 7.69% and 6.73% (without considering $\varepsilon = 0.0$) for bimodal and partial targeted noise, respectively. Overall, the theoretical and empirical values match well across the whole range of noise ratios.



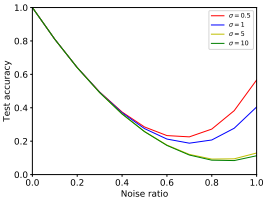
(a) Truncated normal, $\mu = 1, \sigma = 0.5$



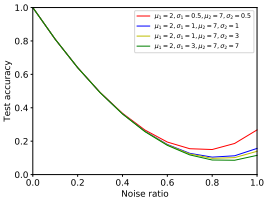
(b) Bimodal, $\mu_1 = 2, \sigma_1 = 0.5$, $\mu_2 = 7, \sigma_2 = 5$



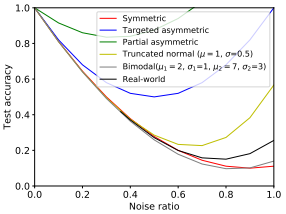
(c) Partial targeted



(d) Truncated normal: $\mu = 1$, different σ



(e) Bimodal: $\mu_1 = 2, \mu_2 = 7$, different σ_1, σ_2



(f) All

Figure 2.2: Test Accuracy under Different Noise Ratios on CIFAR-10: The top row showcases theoretical (lines) vs. empirical (points) under varying noise ratios for (a) truncated normal, (b) bimodal, and (c) partial targeted. The bottom row analyzes noise parameters variation for (d) truncated normal and (e) bimodal patterns, and (f) compares all patterns to real-world noise comprehensively.

2.5.4. IMPACT OF DIFFERENT NOISE PATTERNS

We conclude by using our theoretical analysis to compare the impact on test accuracy of different noise patterns. First, we consider different parameters for truncated normal and bimodal noises and finish with comparing all noise patterns from here, in [74] and the real-world noise pattern from [25].

Figure. 2.2, the second row shows all results. We start with truncated normal noise with a fixed target class and different σ. Higher values of σ result in a wider spread of label noise across adjacent classes as shown in Figure. 2.2(d). Under lower noise ratios, e.g., $\varepsilon < 0.5$, the impact of varying σ is negligible, as shown by the overlapping curves. After that, we see that the most challenging cases are with high values of σ due to the wider spread of corrupted labels deviating from their true classes. Similarly to the previous analysis, for bimodal noise, we fix the target classes, i.e., μ_1 and μ_2, while varying the variances around the two peaks, i.e., σ_1 and σ_2. Overall the results are similar to truncated normal noise, but we can observe that the sensitivity to σ is lower (see Figure. 2.2(e)) even if on average test accuracy of truncated normal is higher than bimodal noise. For instance, in case of $\varepsilon = 1.0$ the difference between $\sigma = 0.5$ and $\sigma = 1$ is 16.26% for truncated normal,

but only 11.11% for bimodal. Hence, bimodal tends to be more challenging since lines for different σ are all more condensed around low values of accuracy with respect to truncated normal noise.

To conclude, we compare all synthetic symmetric and asymmetric noise patterns considered against the real-world noise pattern observed on the Clothing1M dataset [25] (see Figure. 2.2(f)). The measured noise ratio of this dataset is $\varepsilon = 0.41$. To estimate the test accuracy, we scale the noise pattern to different ε by redistributing the noise, such as to maintain all relative ratios between noise transition matrix elements per class. This imposes a lower limit on the noise ratio of $\varepsilon = 0.36$ to be able to keep all elements within the range $[0, 1]$. As intuition can suggest, partial targeted noise has the least impact since it only affects a fraction of classes. More interestingly, we see that the decrease in accuracy for all asymmetric noise patterns is not monotonic. When noise ratios are high, another class becomes dominant, and thus it is easier to counter the noise pattern. On the contrary, all curves tend to overlap at smaller noise ratios, i.e., noise patterns play a weaker role compared to at higher noise ratios. Finally, the real-world noise pattern almost overlaps with bimodal. This might be due that errors in Clothing1M often are between two classes sharing visual patterns [25].

2.6. LABELNET METHODOLOGY

Consider the classification problem having training set $\mathcal{T} = \{(\boldsymbol{x}_1, y_1, \hat{y}_1), (\boldsymbol{x}_2, y_2, \hat{y}_1), ..., (\boldsymbol{x}_N, y_N, \hat{y}_N)\}$ where $\boldsymbol{x}_i$ denotes the i^{th} observed sample, and $\hat{y}_i \in \{0, 1\}^K$ and $y_i \in \{0, 1\}^K$ the corresponding label vectors over K classes representing the clean ground truth and noisy given classes, respectively. Traditional classification problems only use the sample $\boldsymbol{x}_i$ and its true label $\hat{y}_i$. However, real-world datasets are typically affected to various degrees by label noise. Hence, for some samples, the given label y_i is different from the true label $\hat{y}_i$ even in the training set. The core contribution of the paper is a model, named LABELNET, which leverages both $\boldsymbol{x}_i$ and y_i to predict $\hat{y}_i$, instead of only $\boldsymbol{x}_i$. Using this additional information enables the model to significantly boost the accuracy.

2.6.1. LABELNET ARCHITECTURE

We address this problem via LABELNET comprising two neural networks complete with a traditional softmax output layer named Amateur $\mathscr{A}$ and Expert $\mathscr{E}$. Figure. 2.3 shows how the two networks are interconnected. The goal of the Amateur is to predict the label $\hat{y}_i^{\mathscr{A}}$ of an observed sample $\boldsymbol{x}_i$ while Expert aims at correcting, if necessary, this prediction based on the output of the Amateur and the given label y_i. The label corrected by the Expert $\hat{y}_i^{\mathscr{E}}$ is provided as feedback to the Amateur during training closing the loop.

Expert acts as a supervisor which reviews and corrects the predictions of Amateur by comparing it to another label source: the given labels y. We need Expert because both label sources are affected by errors stemming from an imperfect model for the former and from label noise for the latter. From this point of view, we can consider the given labels y as the output of a second external independent imperfect model which is prone to make different errors than Amateur. The idea of having Expert is to leverage not only the intrinsic properties of single models, as most related work does, but also the differences

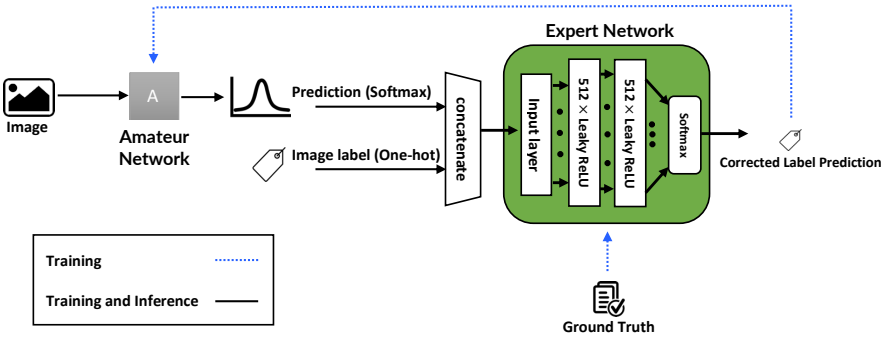


Figure 2.3: LABELNET Architecture including Amateur and Expert networks.

across the models. In the simplest case, if model A is good at classifying dogs and model B in classifying cats, we could learn to trust more model A when predicting dogs and model B when predicting cats. However, cases are rarely as easy, and we resort to the Expert model to learn these patterns.

To decide the type of information to exchange between Amateur and Expert, we consider that the output layer of neural networks is traditionally a softmax transformation $\sigma(z_k) = e^{z_k}/\sum_{j=i}^{K} e^{z_j}$. This ensures that the output vector elements are all in the range $z_j \in 0,1, j = 1...K$ and their sum is $\sum_{j=1}^{K} z_j = 1$ satisfying the properties of a probability distribution. This probability distribution is more informative about the correctness of the prediction [91] because it intrinsically includes information on how confident, i.e. how sharp, or how insecure, i.e. spread out, the model is on the prediction of the most likely class. Hence, we use this as the input to Expert from Amateur rather than the sole predicted class.

The task of Amateur is to classify images. This fits well the classic state-of-the-art DNN vision-models. In our evaluation, we use the CNN defined in [112] having three blocks of two convolutional layers plus one pooling layer followed by a fully connected layer and the softmax output layer. Instead, the task of Expert is to decide the correct label based on the concatenation of the class probability and given label vectors. Here we use a shallower multilayer perceptron.

2.6.2. LABELNET TRAINING

Let $\mathcal{F}^{\mathcal{A}}(\cdot;\boldsymbol{\theta}^{\mathcal{A}})$ parameterized by $\boldsymbol{\theta}^{\mathcal{A}}$ and $\mathcal{F}^{\mathcal{E}}(\cdot;\boldsymbol{\theta}^{\mathcal{E}})$ parameterized by $\boldsymbol{\theta}^{\mathcal{E}}$ be the prediction functions. $\mathcal{F}^{\mathcal{A}}()$ and $\mathcal{F}^{\mathcal{E}}()$ output the class probabilities predicted by the final softmax layer of Amateur and Expert, respectively. The training loss functions can be written as follows:

$$l^{\mathcal{A}} = \min_{\boldsymbol{\theta}^{\mathcal{A}}} \sum_{i=1}^{N} \mathcal{L}(\mathcal{F}^{\mathcal{E}}(<\mathcal{F}^{\mathcal{A}}(\boldsymbol{x}_i), y_i>;\boldsymbol{\theta}^{\mathcal{E}}), \mathcal{F}^{\mathcal{A}}(\boldsymbol{x}_i;\boldsymbol{\theta}^{\mathcal{A}})) \tag{2.4}$$

Algorithm 1: Training LABELNET

Input : Training set $\mathscr{T}$ made of: Observed samples $\boldsymbol{x}$, Given labels y, True labels $\hat{y}$
Output: Trained Amateur $\mathscr{A}$ and Expert $\mathscr{E}$

1 Initialize $\mathscr{A}$ and $\mathscr{E}$ with random $\boldsymbol{\theta}^{\mathscr{A}}$ and $\boldsymbol{\theta}^{\mathscr{E}}$
2 **for** *training iteration* **do**
3 **for** *each batch* $B\{\boldsymbol{x}, y, \hat{y}\}$ *from* $\mathscr{T}$ **do**
4 $\hat{y}^{\mathscr{A}}$:= Predict label probabilities of $\boldsymbol{x}$ by $\mathscr{A}$
5 $\boldsymbol{z}$:= **concatenate** $< \hat{y}^{\mathscr{A}}, y >$
6 Train $\mathscr{E}$ with pair $(\boldsymbol{z}, \hat{y})$ updating $\boldsymbol{\theta}^{\mathscr{E}}$
7 $\hat{y}^{\mathscr{E}}$:= Predict corrected label probabilities from $\boldsymbol{z}$ by $\mathscr{E}$
8 Train $\mathscr{A}$ with pair $(\boldsymbol{x}, \hat{y}^{\mathscr{E}})$ updating $\boldsymbol{\theta}^{\mathscr{A}}$
9 **end**
10 **end**

$$l^{\mathscr{E}} = \min_{\boldsymbol{\theta}^{\mathscr{E}}} \sum_{i=1}^{N} \mathscr{L}(\hat{y}_i, \mathscr{F}^{\mathscr{E}}(< \mathscr{F}^{\mathscr{A}}(\boldsymbol{x}_i), y_i >; \boldsymbol{\theta}^{\mathscr{E}})) \tag{2.5}$$

where $< \cdot, \cdot >$ is the concatenation function of two vectors and $\mathscr{L}$ the loss over the K classes. For both networks we use $\mathscr{L}$ equal to the cross-entropy loss fitting well the probabilistic output of softmax layer. $\mathscr{L}$ increases as predicted probability diverges from expected label.

To train the model we use the alternating minimization approach on batches of data. We first train the Expert based on the output of the Amateur then the Amateur based on the feedback from the Expert. Algorithm 1 details this process. After random initialization of the weights $\boldsymbol{\theta}^{\mathscr{A}}$ and $\boldsymbol{\theta}^{\mathscr{E}}$ (Step 1) for each training step and data batch, we use $\mathscr{A}$ to predict the labels $\hat{y}^{\mathscr{A}}$ of the observed images $\boldsymbol{x}$ (Step 4). $\hat{y}^{\mathscr{A}}$ is concatenated with the given labels y (Step 5) as input to train $\mathscr{E}$ together with the true labels $\hat{y}$ (Step 6). After that in turn we use $\mathscr{E}$ to predict the corrected labels $\hat{y}^{\mathscr{E}}$ (Step 7) and train $\mathscr{A}$ based on the pair $(\boldsymbol{x}, \hat{y}^{\mathscr{E}})$ (Step 8). We use stochastic gradient descent with momentum and learning rate decay to update $\boldsymbol{\theta}^{\mathscr{E}}$ and $\boldsymbol{\theta}^{\mathscr{A}}$.

2.7. TRUSTNET METHODOLOGY

In this section, we present our proposed robust learning framework, TrustNet, featuring on a light weight estimation of noise patterns and a robust loss function.

2.7.1. TRUSTNET ARCHITECTURE

Consider extending the classification problem from Preliminaries section with a set of trusted data, $\mathscr{T} = \{(\boldsymbol{x}_1, y_1, \hat{y}_1), (\boldsymbol{x}_2, y_2, \hat{y}_1), ..., (\boldsymbol{x}_N, y_N, \hat{y}_N)\}$. $\mathscr{T}$ is validated by experts and has for each sample $\boldsymbol{x}$ both given y and true $\hat{y}$ class labels. Hence, our classification problem comprises two types of datasets: $\mathscr{T}$ and $\mathscr{D}$, where $\mathscr{D}$ has only the given class label y. The given class labels y in both data sets are affected by the same noise pattern and noise ratio. Further, we assume that $\mathscr{T}$ is small compared to $\mathscr{D}$, i.e. $|\mathscr{T}| << |\mathscr{D}|$, due to the cost of experts' advise. Corresponding to the two datasets, TrustNet consists of two

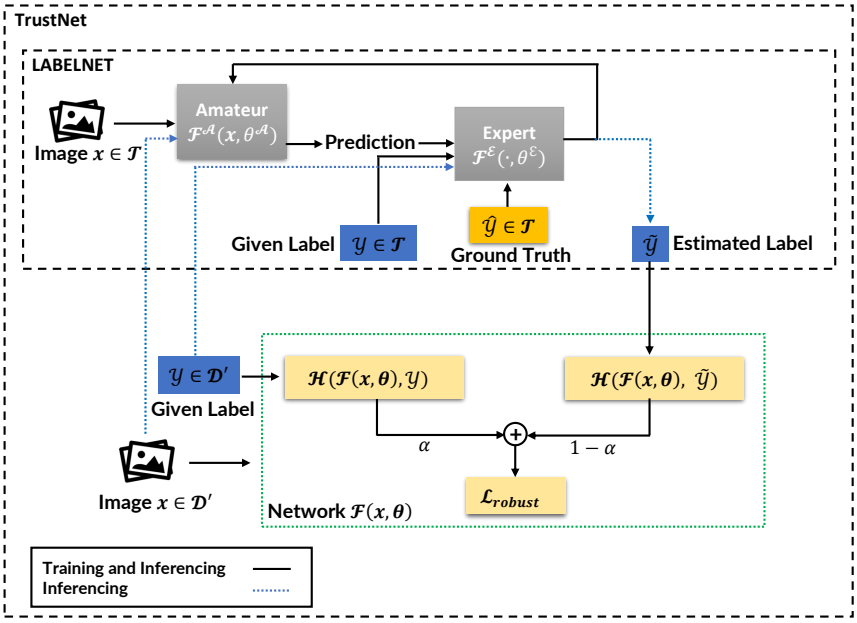


Figure 2.4: Overview of the TrustNet architecture and integration with LABELNET for generating estimated labels.

training routines highlighted by the top and bottom halves of Figure. 3.3.

First (top half), we adopt the architecture of LABELNET [82] and leverage the trusted dataset to learn the underlying noise transition matrix. TrustNet uses LABELNET to learn the noise transition matrix and estimate the true labels for the untrusted data. LABELNET is a deep neural network jointly trained on the given and true labels, however, it requires the ground truth of all the data. Since acquiring the ground truth of the data in real-world scenarios is extremely expensive and time consuming, we reduce this cost by modifying LABELNET via introducing a weighted loss function, which we describe in § 2.7.3.

Second (bottom half), the trained LABELNET is used to derive a dataset $\mathscr{D}'$ from $\mathscr{D}$ by enriching it with estimated class labels $\tilde{y}$ inferred by LABELNET (blue path). Hence $\mathscr{D}' = \{(\boldsymbol{x}_1, y_1, \tilde{y}_1), (\boldsymbol{x}_2, y_2, \tilde{y}_2), ..., (\boldsymbol{x}_N, y_N, \tilde{y}_N)\}$. Then, we train a deep neural network, $\mathscr{F}(\cdot, \boldsymbol{\theta})$, on $\mathscr{D}'$ using the proposed robust loss function from Noise Robust Loss Function section. We note that the trusted data is used only to train LABELNET, not $\mathscr{F}(\cdot, \boldsymbol{\theta})$.

As we mentioned in § 2.5, the proposed Lemma 1 and Lemma 2 show an extension of the DNN memorization effect on test accuracy [44], [74] to the noise patterns used to evaluate TrustNet. TrustNet intends to reduce the memorization effect for noise with a two-stage approach. The first stage corrects the noisy labels and the second stage uses a weighted loss function on the given and the corrected labels.

2.7.2. ESTIMATING NOISE TRANSITION MATRIX

As we mentioned in § 2.6 during training, first Amateur provides for a sample $\boldsymbol{x}_k$ a prediction of the class probabilities $\boldsymbol{y}_k^{\mathscr{A}}$ to Expert. Expert uses $\boldsymbol{y}_k^{\mathscr{A}}$ concatenated with the given class label y_k to learn to predict the ground truth class label $\hat{y}_k$. In turn, the predicted label from Expert $y_k^{\mathscr{E}}$ is provided as feedback to train Amateur. In summary, training tries to minimize recursively the following two loss functions for Amateur, described by $\mathscr{F}^{\mathscr{A}}(\cdot,\boldsymbol{\theta}^{\mathscr{A}})$ and Expert, described by $\mathscr{F}^{\mathscr{E}}(\cdot,\boldsymbol{\theta}^{\mathscr{E}})$:

$$\min_{\boldsymbol{\theta}^{\mathscr{A}}} \mathscr{L}(\mathscr{F}^{\mathscr{A}}(\boldsymbol{x}_k,\boldsymbol{\theta}^{\mathscr{A}}), y_k^{\mathscr{E}}) \tag{2.6}$$

$$\min_{\boldsymbol{\theta}^{\mathscr{E}}} \mathscr{L}(\mathscr{F}^{\mathscr{E}}(<\boldsymbol{y}_k^{\mathscr{A}}, y_k>,\boldsymbol{\theta}^{\mathscr{E}}), \hat{y}_k) \tag{2.7}$$

where $<\cdot,\cdot>$ represents vector concatenation.

The trained LABELNET can estimate the true label from an image $\boldsymbol{x_k}$:

$$\tilde{y}_k = \mathscr{F}^{\mathscr{E}}(<\mathscr{F}^{\mathscr{A}}(\boldsymbol{x}_k,\boldsymbol{\theta}^{\mathscr{A}}), y_k>,\boldsymbol{\theta}^{\mathscr{E}}). \tag{2.8}$$

Specifically, we use the trained LABELNET to enrich and transform $\mathscr{D}$ in $\mathscr{D}'$ by incorporating for each image $\boldsymbol{x}_k$ the inferred class label $\tilde{y}_k$. Subsequently, we use $\mathscr{D}'$ to train $\mathscr{F}(\cdot,\boldsymbol{\theta})$ via the loss function robust to noise from Noise Robust Loss Function section.

2.7.3. NOISE ROBUST LOSS FUNCTION

The given labels are corrupted by noise. Directly training on the given labels results in highly degraded performance as the neural network is not able to easily discern between clean and corrupted labels. To make the learning more robust to noise, TrustNet proposes to modify the loss function to leverage both given labels y and inferred labels $\tilde{y}$ from LABELNET to train $\mathscr{F}(\cdot,\boldsymbol{\theta})$.

The predicted label of $\mathscr{F}(\cdot,\boldsymbol{\theta})$ is compared, e.g., via cross-entropy loss, against both the given label and inferred label. The challenge is how to combine these two loss values. Ideally, for samples for which LABELNET and $\mathscr{F}(\cdot,\boldsymbol{\theta})$ are highly accurate, the inferred label can be trusted more. On the contrary, for samples for which LABELNET and $\mathscr{F}(\cdot,\boldsymbol{\theta})$ have low accuracy, the given labels can be trusted more. Specifically, TrustNet uses a weighted average between the loss of the predicted label from $\mathscr{F}(\boldsymbol{x}_k,\boldsymbol{\theta})$ against both the given label y_k and the LABELNET's inferred label $\tilde{y}_k$ with per sample weights α_k and $(1-\alpha_k)$ for all samples $\boldsymbol{x}_k$ in $\mathscr{D}'$. Moreover, TrustNet dynamically adjusts α_k after each epoch based on the observed learning performance of $\mathscr{F}(\boldsymbol{x}_k,\boldsymbol{\theta})$.

In detail we use cross-entropy H as standard loss measure to train our deep neural network $\mathscr{F}(\boldsymbol{x}_k,\boldsymbol{\theta})$:

$$\mathscr{H}(\mathscr{F}(\boldsymbol{x}_k,\boldsymbol{\theta}), y_k) = -\sum_{i=0}^{c-1} \mathbb{1}(y_k, c) \log \mathscr{F}(\boldsymbol{x}_k,\boldsymbol{\theta}) \tag{2.9}$$

where $\mathbb{1}(y_k, c)$ is an indicator function equal to 1 if $y_k = c$ and 0 otherwise. For each data point $\boldsymbol{x}_k$ in $\mathscr{D}'$, we assign weights of α_k and $(1-\alpha_k)$ to the cross-entropy of the given y_k and inferred $\tilde{y}_k$ labels, respectively. We let $\alpha_k \in [0,1]$. Hence, we write the robust loss function $\mathscr{L}_{robust}$ as following:

$$\begin{aligned}\mathscr{L}_{robust}(\mathscr{F}(\boldsymbol{x}_k,\boldsymbol{\theta}), y_k, \tilde{y}_k) = \alpha_k\, &\mathscr{H}(\mathscr{F}(\boldsymbol{x}_k,\boldsymbol{\theta}), y_k) \\ &+ (1-\alpha_k)\, \mathscr{H}(\mathscr{F}(\boldsymbol{x}_k,\boldsymbol{\theta}), \tilde{y}_k).\end{aligned} \tag{2.10}$$

When the weight factor is low, we put more weight on the cross-entropy of inferred labels, and vice versa. In the following, we explain how to dynamically set α_k per epoch.

DYNAMIC α_k

Here we adjust α_k based on the uncertainty of TrustNet and LABELNET. When the learning capacities of LABELNET and TrustNet are higher (lower values of loss function), we have more confidence on the inferred labels and put more weight on the second term of Eq. 2.10, i.e., smaller α_k values. As a rule of thumb, at the beginning α_k values are high since TrustNet experiences higher losses at the start of training. Then α_k values gradually decrease with the growing capacity of TrustNet.

Let $\alpha_{k,e}$ be the weight of the k^{th} image at epoch e. We initialize $\alpha_{k,0}$ based on the entropy value S from inferred class probabilities $\tilde{\boldsymbol{y}}_k$ of LABELNET:

$$S(\tilde{\boldsymbol{y}}_k) = -\sum_{i=0}^{c-1} \tilde{y}_k^i \, log \, \tilde{y}_k^i$$

where c is the number of classes and $\tilde{y}_k^i$ is the i^{th} class probability of $\tilde{\boldsymbol{y}}_k$. We use LABELNET since we do not have yet any predictions from TrustNet's own neural network.

For subsequent epochs, $e > 0$, we switch to TrustNet as source of entropy values. We gradually adjust $\alpha_{k,e}$ based on the relative difference between current and previous epoch values:

$$\alpha_{k,e} = \alpha_{k,e-1} \cdot (1 + \frac{S(\boldsymbol{y}_k^{\mathscr{F}}(e)) - S(\boldsymbol{y}_k^{\mathscr{F}}(e-1))}{S(\boldsymbol{y}_k^{\mathscr{F}}(e-1))}) \quad \forall e > 0, \tag{2.11}$$

where $\boldsymbol{y}_k^{\mathscr{F}}(e)$ are the class probabilities predicted by $\mathscr{F}(\cdot, \boldsymbol{\theta})$ for the k^{th} image at epoch e. When the entropy values decrease, we gain more confidence in TrustNet and the weights on the inferred labels (1-(1 − α)) increase.

We summarize the training procedure of TrustNet in Algorithm 2. Training LABELNET consists of training two neural networks: Expert, $\mathscr{F}^{\mathscr{E}}(\cdot, \boldsymbol{\theta}^{\mathscr{E}})$, and Amateur, $\mathscr{F}^{\mathscr{A}}(\cdot, \boldsymbol{\theta}^{\mathscr{A}})$, using the trusted data $\mathscr{T}$ for E_{LABELNET} epochs (line 1-4). Then we need to compute the inferred labels for all data points in $\mathscr{D}$ to produce $\mathscr{D}'$ (line 5). Finally, we train TrustNet for $E_{TrustNet}$ epochs (line 6-14). The initialization of α_k is via the entropy of the inferred labels (line 9) and then updated by the entropy of predicted labels (line 11). The robust loss function is computed accordingly (line 13).

2.8. LABELNET EVALUATION

2.8.1. EXPERIMENTS SETUP

Datasets. Our evaluation is based on four benchmarking datasets: MNIST [113], CIFAR-10 [114], CIFAR-100 [115] and Clothing1M [25]. MNIST consists of 28 × 28-pixel black-and-white images of handwritten digits from zero to nine. The dataset contains 60000 training and 10000 validation images. CIFAR-10 and CIFAR-100 include 32 × 32-pixel color images organized in 10 and 100 classes, respectively. The image classes range from animals to vehicles. Both datasets contain 50000 training and 10000 validation images. Clothing1M contains images collected from the Internet classified into 14 classes based on the surrounding text. It is representative of real world noise (average noise rate of

Algorithm 2: TrustNet training

Input : Trusted dataset $\mathscr{T}$, Untrusted dataset $\mathscr{D}$; Epochs E_{LABELNET},$E_{TrustNet}$.
Untrusted dataset $\mathscr{D}$ made of: Observed samples $\boldsymbol{x}$, Given labels y
Trusted dataset $\mathscr{T}$ made of: Observed samples $\boldsymbol{x}$, Given labels y, True labels $\hat{y}$

Output : Trained TrustNet $\mathscr{F}(\boldsymbol{x},\boldsymbol{\theta})$

Initialize $\mathscr{F}^{\mathscr{A}}$ and $\mathscr{F}^{\mathscr{E}}$ with random $\boldsymbol{\theta}^{\mathscr{A}}$ and $\boldsymbol{\theta}^{\mathscr{E}}$
for $e = 0, 1, ..., E_{\text{LABELNET}}$ *on* $\mathscr{T}$ **do**
  Train $\mathscr{F}^{\mathscr{E}}$ and $\mathscr{F}^{\mathscr{A}}$ `#LABELNET training`
end
$\mathscr{D}' = \mathscr{D}$ extended with $\tilde{y} = \mathscr{F}^{\mathscr{E}}(<\mathscr{F}^{\mathscr{A}}(\boldsymbol{x},\boldsymbol{\theta}^{\mathscr{A}}), y>, \boldsymbol{\theta}^{\mathscr{E}})$ `#LABELNET inference`
Initialize $\mathscr{F}$ with random $\boldsymbol{\theta}$ `#TrustNet training`
for $e = 0, 1, ..., E_{TrustNet}$ *on* $\mathscr{D}'$ **do**
  if $e == 0$ **then**
    $\alpha_{k,0} = S(\tilde{\boldsymbol{y}}_k)$
  else
    $\alpha_{k,e} = \alpha_{k,e-1} \cdot (1 + \frac{S(\boldsymbol{y}_k^{\mathscr{F}}(e)) - S(\boldsymbol{y}_k^{\mathscr{F}}(e-1))}{S(\boldsymbol{y}_k^{\mathscr{F}}(e-1))})$
  end
  Train $\mathscr{F}(\boldsymbol{x},\boldsymbol{\theta}_e)$ with $\alpha_{k,e}\,\mathscr{H}(\mathscr{F}(\boldsymbol{x}_k,\boldsymbol{\theta}_e), y_k) + (1-\alpha_{k,e})\,\mathscr{H}(\mathscr{F}(\boldsymbol{x}_k,\boldsymbol{\theta}_e), \tilde{y}_k)$ for each sample k
end

39.5%). Here we use the cleansed training, validation and testing sets of 47K, 14K and 10K samples, respectively.

Label noise. For MNIST, CIFAR-10 and CIFAR-100 we use the original labels as true labels $\boldsymbol{t}$. We generate the noisy given labels $\boldsymbol{y}$ by injecting symmetric label noise where the original label is flipped to one of the other classes with uniform probability. We use different noise ratios corresponding to flipping probabilities of 0.2, 0.3, 0.4, and 0.5. Such generating principles are applied for both training and inferences images. Since the ground truth of 1 million training image labels in Clothing1M is not available, we use cleansed labels (47K samples) available in the dataset and then generate given (noisy) labels by using the estimated noise confusion matrix which is provided by [25].

LABELNET parameters. For MNIST, CIFAR-10 and CIFAR-100 Amateur is the 12-layer CNN architecture used in [112] with ReLU activation functions. Expert is a feed-forward 4-layer neural network with Leaky ReLU activation functions in the hidden layers and sigmoid in the last layer. Both networks are implemented using Keras v2.2.4 and Tensorflow v1.13 and trained using stochastic gradient descent with momentum 0.9, weight decay 10^{-4}, and learning rate 0.01. We train our model for 100, 120 and 200 epochs for MNIST, CIFAR-10 and CIFAR-100, respectively. For Clothing1M, we resize each image to 256×256 pixels and crop the center to 224×224. We use ResNet50 for Amateur with SGD optimizer and momentum of 0.9. The weight decay factor is 5×10^{-3}, and the batch size is 16. The initial learning rate is 0.002 and decreased by 10 every 5 epochs. The total training epochs are 50. The Expert architecture remains the same. All experiments run on

servers equipped with 8-cores @ 2.4GHz, 64GB of RAM, and an NVIDIA TITAN X GPU.

2.8.2. COMPETING METHODS

We consider the following four methods, which aim to filter out the (impact of) noisy labels by altering the loss function, selecting the clean labels, and inferring the noise transition matrix. Competing models are based on their original code and settings.

D2L [34]: uses the Local Intrinsic Dimensionality (LID) to detect points of noisy data and modifies the loss function based on the LID score.

Co-teaching [61]: uses two neural networks to teach each other by selecting and exchanging the more informative data batches where the selection leverages the memory effect of neural networks.

Bootstrap [52]: uses a weighted combination of the original label and prediction of the model as the final prediction.

Forward [70]: uses the noise transition matrix to correct the labels before training.

Comparison modes For a fair comparison, we compare the competing models against LABELNET under the two following scenarios:

- **LABELNET**: This is the complete system. Here we use the given labels $\boldsymbol{y}$ both during training and inference and the predictions are taken from the output of Expert.
- **LABELNET-NGL**: This is a reduced system: LABELNET No Given Labels (NGL). Here we forgo the use of the given labels (and Expert) during inference. The given labels and Expert are used only during training. In this case the predictions are taken from the output of Amateur.

Additionally, we evaluate the effect of decreasing amounts of training data which range from randomly selected 100% to 20% of the training samples for each dataset. Experiment across competing models all use the same training and validation sets.

Metrics of interests We present the inference accuracy of LABELNET, LABELNET-NGL and all four competing methods. As a performance metric, we use the accuracy evaluated on the validation data computed as the ratio of the number of correct predictions, i.e. equal to the original true labels $\boldsymbol{t}$, divided by the total number of validation samples. Such convention are used in Table 2.1, 2.2 and 2.3. The difference between the accuracy values of LABELNET and LABELNET-NGL represents the auxiliary learning capacity of Expert.

2.8.3. RESULTS

MNIST Among all three benchmarks, MNIST is the easiest dataset due to a smaller number of classes and image complexity. Results in Table 2.1 show that LABELNET achieves exceptional inference accuracy, i.e., consistently above 95% even in the case of 50% noise and just 20% training available. The difference between LABELNET and LABELNET-NGL is rather modest, as the learning capacity of trained Amateur (via the guidance of Expert) is sufficient to distill the impact of noise labels. The other four baselines can infer the images classes with high accuracy, when the training data is 100%, but do not necessarily sustain good accuracy with the reduction of training data.

Under large amounts of training data and low noise LABELNET, LABELNET-NGL, Co-Teaching, and Forward all reach an almost perfect accuracy score. Specifically, LABELNET, LABELNET-NGL and Co-teaching are the models which are least affected by noise and diminishing amounts of training data. Even in the worst case, with 50% noise and 20%

Table 2.1: Testing accuracy (%) of LABELNET and LABELNET-NGL compared to 4 baselines under varying label noise ratios and sizes of training data on MNIST.

Training data	LABELNET	LABELNET-NGL	D2L	Co-Teaching	Bootstrap	Forward
Noise Ratio = 20%						
100%	**99.31**	99.18	89.69	97.59	87.83	97.76
80%	**99.36**	99.09	89.40	97.40	84.70	94.43
60%	**99.16**	99.01	89.47	97.29	78.11	81.08
40%	**99.22**	98.72	88.94	96.67	60.27	61.49
20%	**98.38**	97.66	88.59	96.04	10.04	15.17
Noise Ratio = 30%						
100%	**99.38**	99.13	86.15	95.72	79.47	95.33
80%	**99.14**	98.99	82.33	95.95	72.09	86.05
60%	**99.12**	98.90	81.01	95.79	58.31	63.64
40%	**98.82**	98.66	80.11	95.22	28.61	48.84
20%	**97.84**	97.41	69.83	94.89	5.13	9.80
Noise Ratio = 40%						
100%	**99.08**	98.97	74.52	95.11	68.30	93.01
80%	**98.94**	98.86	72.78	94.72	58.09	71.05
60%	**98.83**	98.75	70.39	94.55	37.59	42.84
40%	**98.52**	98.31	71.02	94.00	10.11	15.09
20%	**97.70**	96.08	70.61	92.79	3.15	9.45
Noise Ratio = 50%						
100%	**98.90**	98.45	70.47	93.42	62.71	90.85
80%	**98.68**	98.21	70.13	93.02	55.41	68.25
60%	**98.56**	98.02	69.49	92.76	33.36	39.78
40%	**98.24**	97.68	68.32	92.41	10.01	14.26
20%	**97.38**	95.78	67.83	92.06	3.12	9.11

training data, the accuracy is above 97% and 95% for LABELNET and LABELNET-NGL, respectively. Co-teaching follows with 92%. Moreover, even if D2L and LABELNET-NGL share the same CNN architecture, LABELNET-NGL outperforms D2L by 10%+ growing with the level of noise. This underlines how effective the feedback from Expert is even if used only during training.

Finally, we highlight the exceptional results of LABELNET by comparing its inference accuracy achieved by a small fraction of training data, i.e., 20%, with competing methods that are trained on 100% training data. LABELNET is still able to outperform them across all noise ratios by up to 35 percent points.

CIFAR-10 We report accuracy results under all noise levels and amounts of training data in Table 2.2. Starting with 100% training data, one can see that LABELNET achieves the accuracy of 89.23%, 88.30%, 84.36%, and 80.73% for the cases of 20%, 30%, 40%, and 50% noise ratio. These results are significantly better than the competing image-only models. The accuracy of LABELNET is 1.79% to 11.92% higher under all considered noise ratios. We attribute the superior accuracy to the Expert that leverages well the information from both Amateur and the given labels. If the given labels are not available, the simpler LABELNET-NGL still benefits from the feedback of Expert during training. As a result, even the simpler LABELNET-NGL is able to compete well and surpass some of the competing models. For the same scenarios, LABELNET-NGL alone still achieves the accuracy of 82.29%, 81.85%, 79.53%, and 76.45% consistently beating Forward and Bootstrap and sometimes Co-teaching.

In case of reduced training data, the accuracy of LABELNET decreases with the amount of data. However, LABELNET is still the best across all competing models. Even with more

training data, e.g. 60% data and 20% noise, we achieve 86.27% and 79.18% accuracy in LABELNET and LABELNET-NGL, respectively. The best rival achieves only 79.73%. In an extreme case, with only 20% training data, LABELNET is significantly better than all others. For fair comparison, we summarize the training data required by LABELNET to reach similar or higher accuracy than the four competing methods each trained with 100% training data. LABELNET achieves this using only 40%, 40%, 60%, and 80% training data under 20%, 30%, and 40% and 50% noise, respectively. One effect of diminishing training data is that the difference between LABELNET and LABELNET-NGL becomes more significant. We interpret this as the lower the amount of available data, the more one should use any possibly source of information, e.g., the given labels, even during inference. The training loss for both Amateur and Expert converge to a lower bound, but the evolution of Expert loss is smoother than its Amateur equivalent. This indicates that LABELNET makes it easier for Expert to correct labels than for Amateur to classify images.

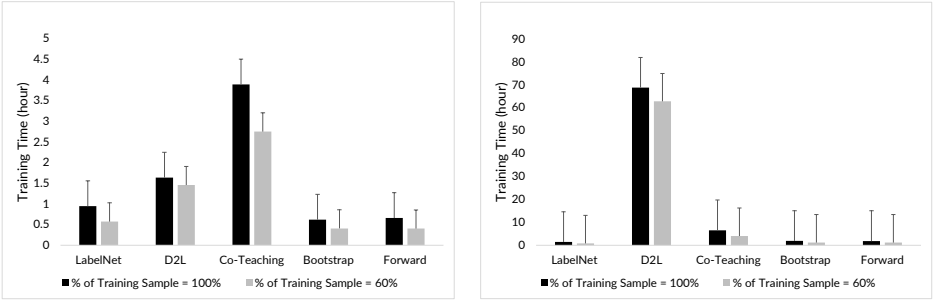
Our proposed LABELNET not only has excellent accuracy but with diminished training data also shorter training times than competing models. Fig. 2.5(a) shows the different training times under two percentage of training data. The bar represents the mean while the whiskers the standard deviation across three repetitions. The closest competitors of LABELNET are D2L and Co-teaching. However both are slower to train than LABELNET by a factor 1.7x to 4.1x. Therefore, we achieve high accuracy with faster training time.

Table 2.2: Testing accuracy (%) of LABELNET and LABELNET-NGL compared to 4 baselines under varying label noise ratios and sizes of training data on CIFAR-10.

Training data	LABELNET	LABELNET-NGL	D2L	Co-Teaching	Bootstrap	Forward
Noise Ratio = 20%						
100%	**89.23**	82.29	84.75	82.45	81.80	83.11
80%	**88.61**	81.74	82.85	81.57	79.98	81.43
60%	**86.27**	79.18	79.73	79.30	74.19	75.40
40%	**86.01**	75.11	77.94	77.09	63.82	60.31
20%	**82.23**	67.45	70.47	70.37	23.82	31.35
Noise Ratio = 30%						
100%	**88.30**	81.85	82.45	80.29	77.14	81.68
80%	**85.83**	79.87	81.27	79.16	75.60	79.38
60%	**86.32**	77.73	79.14	76.87	70.09	68.35
40%	**82.12**	71.88	75.68	72.87	35.13	37.87
20%	**75.44**	61.06	70.78	67.37	20.99	28.99
Noise Ratio = 40%						
100%	**84.36**	79.53	80.69	77.28	72.44	78.12
80%	**81.76**	76.16	79.11	75.74	71.50	72.76
60%	**79.33**	73.35	76.63	73.95	58.02	55.67
40%	**74.78**	67.32	71.84	70.47	25.11	31.88
20%	67.88	56.33	**67.96**	63.08	18.91	21.16
Noise Ratio = 50%						
100%	**80.73**	76.45	78.94	74.47	70.14	76.23
80%	**78.48**	72.24	76.43	71.52	57.84	63.33
60%	**75.06**	69.91	73.73	68.80	31.98	37.01
40%	**69.34**	61.86	68.63	64.25	22.95	26.87
20%	**61.37**	50.96	59.08	57.11	15.63	17.89

CIFAR-100 is significantly more difficult than CIFAR-10. First, the number of classes increases by a factor 10. Second, the training samples per class reduce by a factor 10.

Consequently, the achieved accuracy scores shown in Table 2.3 are lower. Even so LABELNET achieves 86.72%, 79.92%, 73.87%, and 66.11% for noise ratios of 20%, 30%, 40%, and 50%, respectively. Moreover, the advantage of using Expert as guidance is more pronounced. Not only no other model except LABELNET is able to reach 60% accuracy, but also LABELNET-NGL consistently reaches similar performance as D2L except under amounts of training data below 60%. However, LABELNET-NGL is in line with Forward. Another positive result is that LABELNET seems to be the least affected by diminishing training data. Unfortunately, the same does not hold for increasing noise levels. For a fair comparison, we identify the minimum required training data for LABELNET to achieve similar or higher accuracy as the other four approaches that leverage 100% training data. Under all noise ratios, LABELNET only needs to have 20% training data to outperform other methods by large margins, ranging from 15% to 27% absolute accuracy improvements.



(a) CIFAR-10 (b) CIFAR-100

Figure 2.5: Training time comparison of LABELNET and baselines using 100% training samples (in black) and 60% of training samples (in gray) on CIFAR-10 and CIFAR-100 datasets.

LABELNET achieves remarkable inference accuracy in the presence of noise labels on significantly smaller sets of training data, compared to state-of-the-art methods. The effective design of LABELNET is particularly evident for more difficult benchmarks, such as CIFAR-100. The combination of Amateur and Expert outperforms other methods even when learning from just 20% of training data used for the others.

Finally, Fig. 2.5(b) compares the training times. All models take longer to train than under CIFAR-10, but LABELNET, while being the most accurate, is still faster than D2L and Co-Teaching. Here especially D2L takes long to train: 68 hours on average. For comparison, LABELNET takes a little under 1.5 hours. Moreover, the training time of LABELNET can be further reduced by decreasing the training data.

Clothing1M We summarize results in Table 2.4 with full training data and randomly selected 50% data. When using all the training set, LABELNET achieves 83.42% accuracy, which is 13 points higher than the second best approach, i.e., Forward at 70.04%. This is due to the capacity of Expert to learn the real world noise pattern. When halving the training set, LABELNET still achieves 69.83%, which is roughly the result the four competing methods reach using 100% training data. In other words, having the ground truth for half of the data LABELNET can still outperform other approaches which do not leverage the knowledge of noise patterns. LABELNET has the best relative performance on

Table 2.3: Testing accuracy (%) of LABELNET and LABELNET-NGL compared to 4 baselines under varying label noise ratios and sizes of training data on CIFAR-100.

Training data	LABELNET	LABELNET-NGL	D2L	Co-Teaching	Bootstrap	Forward
Noise Ratio = 20%						
100%	**86.72**	59.24	55.70	52.74	52.58	59.87
80%	**85.38**	54.56	51.26	50.01	48.95	55.51
60%	**84.85**	50.01	48.33	42.82	41.62	50.76
40%	**82.51**	44.13	42.48	36.75	32.68	48.04
20%	**80.74**	31.11	31.19	27.93	24.01	32.65
Noise Ratio = 30%						
100%	**79.92**	56.87	51.13	45.68	44.99	54.18
80%	**78.61**	52.98	48.26	44.36	41.47	53.66
60%	**76.18**	46.34	43.79	39.71	35.46	48.46
40%	**73.05**	39.87	40.28	33.83	30.21	47.71
20%	**71.72**	26.45	27.98	24.21	21.78	29.77
Noise Ratio = 40%						
100%	**73.87**	53.04	49.50	41.87	40.11	49.44
80%	**71.06**	49.91	45.17	39.88	37.24	47.23
60%	**68.80**	41.76	39.89	33.54	30.63	44.22
40%	**64.33**	34.96	36.82	28.33	22.77	41.35
20%	**61.89**	23.12	24.33	19.92	18.45	26.66
Noise Ratio = 50%						
100%	**66.11**	48.65	43.65	35.89	39.84	46.06
80%	**63.45**	43.61	37.98	33.69	33.79	41.23
60%	**58.56**	35.55	33.37	29.14	28.98	37.05
40%	**53.02**	25.51	30.76	23.96	15.80	34.27
20%	**51.01**	16.87	17.54	16.85	10.54	21.87

CIFAR-100, followed by Clothing1M, and CIFAR-10, reflecting the decreasing importance and difficulty to learn the noise patterns. Clothing1M results further accentuate the idea of LABELNET that learning from both images and (noisy) labels can strengthen the robustness and data efficiency of deep neural networks.

2.8.4. DISCUSSION ON LABELNET

The core idea of LABELNET is to leverage a fraction of the ground truth of noisy labels and imitate how experts correct such noise data. However, in the real world scenarios, noisy labels exhibit dynamic patterns, i.e., the noise ratios fluctuate. This presents a new challenge to LABELNET on how to select representative noisy data in both training and testing phase, reflecting truthfully the reality. To cope with such scenarios, we envision to rely on transfer learning to generalize the learnt models.

We apply transfer learning by, first, training LABELNET under a base noise ratio. Second, we freeze the weights of both Expert and Amateur of all hidden layers except the last one. Third, we fine tune the weights of the last layer by training on data from a different target noise ratio.

We transfer the trained models of CIFAR-10 and CIFAR-100 from 20% to 40% noise ratios and summarize the results in Table 2.5 under different amounts of retraining data. The performance of the base model without transfer learning is included as reference. One can see that applying a model to a different noise ratio degrades the performance significantly. For CIFAR-10, applying the 20% noise base model yields only 81.02% accuracy

Table 2.4: Test accuracy (%) of LABELNET compared to various baselines, utilizing 50% and 100% of the training data on the real-world noisy dataset Clothing1M.

Methods	Accuracy(%)	
	Training data = 50%	Training data = 100%
LABELNET	**69.83**	**83.42**
D2L	49.05	69.43
Co-Teaching	50.11	69.92
Forward	51.26	70.04
Bootstrap	48.94	68.77

Table 2.5: Test accuracy (%) and training time [sec/epoch] for the Transfer Learning (TL) scenario, initially trained on a 20% noise ratio and transferred to a 40% noise ratio. The 'Retraining data' in the table indicates the percentage of additional data used for retraining the transfer learning model under 40% noise. The 'No TL' column represents the scenario without Transfer Learning.

Dataset	Retraining data	No TL	5%	10%	20%	100%
CIFAR-10	Accuracy (%)	81.02	87.15	87.24	87.51	88.04
	Time [s/epoch]	-	0.84	1.69	3.45	17.32
CIFAR-100	Accuracy (%)	58.67	74.65	75.44	76.68	78.04
	Time [s/epoch]	-	0.88	1.74	3.55	17.51

instead of 84.36% achieved by applying the 40% noise base model for CIFAR-10 (shown in Table 2.2). Similarly, such a degradation can be observed for CIFAR-100, i.e., dropping from 73.87% (shown in Table 2.3) to 58.67%. However, with even little, e.g., 5%, retraining data, we are able to adapt the model to the 40% noise ratio bringing up the accuracy to 87.15% and 74.65% for CIFAR-10 and CIFAR-100, respectively. Moreover, these accuracy values surpass the performance achieved by their base model counterparts by 2.8% and 0.8%. Increasing the amount of retraining data improves the results, but only marginally.

In addition to a better generalization of the base models, transfer learning also significantly decreases the training time (see average time per training epoch in Table 2.5). The reason is twofold. First, it is sufficient to train on fewer data as seen previously and the training time per epoch scales linearly with the amount of data. Second, we only train the weights of the last layer of Amateur and Expert, all others being frozen. Using 100% data and compared to the training of the corresponding base models, fine tuning offers a speed up of 1.6x for both CIFAR-10 and CIFAR-100.

2.9. TRUSTNET EVALUATION

In this section, we empirically compare TrustNet against the state of the art noise, under both synthetic and real-world noises. We aim to show the effectiveness of TrustNet via testing accuracy on diverse and challenging noise patterns.

2.9.1. EXPERIMENTS SETUP

We consider three datasets: CIFAR-10 [114], CIFAR-100 [115] and Clothing1M [25]. For CIFAR-10 and CIFAR-100, we assume that 10% of the training set forms the trusted

data with access to the clean labels used as ground truth. We use this trusted set to learn the noise transition via LABELNET. In turn, LABELNET infers the estimated labels for the remaining training data. The whole training set is then used to train TrustNet. Clothing1M contains 1 million images scrapped from the Internet which we resize and crop to 224 × 224 pixels. These labels are affected by real-world noise stemming from the automatic labelling. Out of the 1 million images, a subset of trusted expert-validated images contains the ground truth labels. This subset consists of 47K and 10K images for training and testing, respectively. As for CIFAR-10 and CIFAR-100, we use the trusted set to train LABELNET and infer the estimated labels for the rest of the dataset to train TrustNet. Note that for all three datasets, only training set is subject to label noise, not testing set.

The architecture of Expert consists of a 4-layer feed-forward neural network with Leaky ReLU activation functions in the hidden layers and sigmoid in the last layer. This Expert architecture is used across all datasets. TrustNet and Amateur use the same architecture, which depends on the dataset. For CIFAR-10 TrustNet and Amateur consist in an 8-layer CNN with 6 convolutional layers followed by 2 fully connected layers with ReLU activation functions as in [112]. For CIFAR-100 both rely on the ResNet44 architecture. Finally, Clothing1M uses pretrained ResNet101 with ImageNet. TrustNet (LABELNET) is trained for 120 (150) and 200 (180) for CIFAR-10 and CIFAR-100, respectively, using SGD optimizer with batch size 128, momentum 0.9, weight decay 10^{-4}, and learning rate 0.01. Finally, Clothing1M uses 50 (35) epochs and batch size 32, momentum 0.9, weight decay 5×10^{-3} and learning rate 2×10^{-3} divided by 10 every 5 epochs.

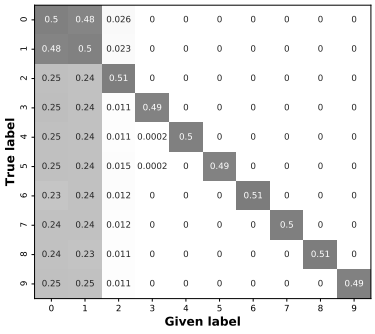
Our target evaluation metric is the accuracy achieved on the clean testing set, i.e. not affected by noise. We compare TrustNet against six noise resilient networks from the state of the art: SCL [63], D2L [34], Forward [70], Bootstrap [94], Co-teaching+ [95], and Co-teaching [61]. We do not compare TrustNet to LABELNET because primarily LABELNET requires labels in the inference process and training. Moreover, LABELNET needs accessing the ground truth for the whole data samples. All training uses Keras v2.2.4 and Tensorflow v1.13. We use 10% of the dataset as the trusted samples for the pre-training of baselines to have a fair comparison.

2.9.2. SYNTHETIC NOISE PATTERNS

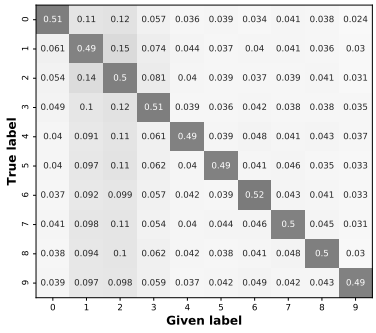
For CIFAR-10 and CIFAR-100, we inject synthetic noise. We focus on asymmetric noise patterns following a truncated normal and bimodal distribution, and symmetric noise, as discussed in Understanding DNNs. section. We inject noises with average rates $\varepsilon = 0.4$, 0.5 and 0.6. For truncated normal the target classes and variances are class 1 with $\sigma = 0.5$ or $\sigma = 5$ and 10 with $\sigma = 1$ or $\sigma = 10$ for CIFAR-10 and CIFAR-100, respectively. For bimodal we use $\mu_1 = 2$, $\sigma_1 = 1$ plus $\mu_2 = 7$, $\sigma_2 = 3$ and $\mu_1 = 20$, $\sigma_1 = 10$ plus $\mu_2 = 70$, $\sigma_2 = 5$ for CIFAR-10 and CIFAR-100, respectively. We illustrate the noise transition matrix of these noise patterns in Figure 2.6.

CIFAR-10

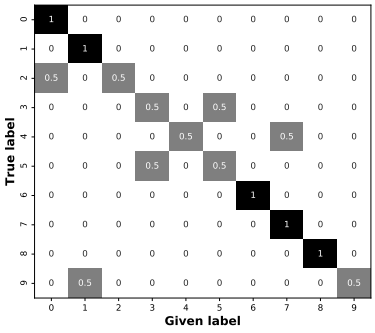
We summarize the results of CIFAR-10 in Table 2.6. We report the average and standard deviation across three runs. Overall the results are stable across different runs as seen from the low values of standard deviation.



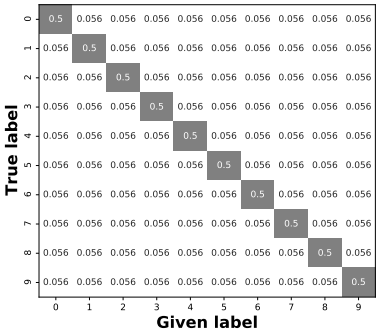
(a) Truncated normal, $\mu = 1, \sigma = 0.5$



(b) Bim., $\mu_1 = 2\ \sigma_1 = 1\ \mu_2 = 7\ \sigma_2 = 3$



(c) Partial targeted



(d) Symmetric

Figure 2.6: Transition matrices illustrating four different noise patterns on CIFAR-10 with a noise ratio of 0.5 across 10 classes.

TrustNet achieves the highest accuracy for bimodal noises, which is one of the most difficult noise patterns based on Understanding DNNs. section. Here the accuracy of TrustNet is consistently the best beating the second best method by increasing 2.4%, 21.1%, and 27.2% for 40%, 50%, and 60% noise ratios, respectively. At the same time, TrustNet is the second best method for symmetric and truncated normal asymmetric noise. Here the best method is often SCL, which also leverages a modified loss function to enhance the per class accuracy using symmetric cross-entropy. This design targets direct symmetric noise where SCL outperforms TrustNet. Considering the asymmetric truncated normal noise, the difference is smaller and decreasing with increasing noise ratio. At 60% noise SCL is only marginally better by, on average, 2.9%. Finally, test accuracy variations are not noticeable with increasing σ values. All other baselines perform worse.

Table 2.6: Test accuracy (%) of TrustNet compared to baselines on clean testing set for CIFAR-10 under 0.4 and 0.6 noise rates and patterns: i) symmetric, ii) bimodal with $\mu_1 = 2$, $\sigma_1 = 1$, $\mu_2 = 7$, $\sigma_2 = 3$ and iii) truncated normal with $\mu = 1$, $\sigma = [0.5, 5]$. The best results are in bold.

CIFAR-10								
Methods	Symmetric		Bimodal Asymmetric		Truncated Normal Asymmetric			
	$\varepsilon = 0.4$	$\varepsilon = 0.6$	$\varepsilon = 0.4$	$\varepsilon = 0.6$	$\varepsilon = 0.4$		$\varepsilon = 0.6$	
					$\sigma = 0.5$	$\sigma = 5$	$\sigma = 0.5$	$\sigma = 5$
TrustNet	77.03 ± 0.32	61.22 ± 0.66	**72.67 ± 0.33**	**42.18 ± 0.61**	74.21 ± 0.69	73.88 ± 0.78	66.48 ± 0.61	67.23 ± 0.57
SCL	**81.50 ± 0.22**	**73.13 ± 0.12**	69.07 ± 1.17	15.00 ± 0.67	**80.93 ± 0.50**	**80.90 ± 0.14**	**68.67 ± 0.96**	**70.90 ± 0.67**
D2L	75.87 ± 0.33	60.54 ± 0.44	70.59. ± 0.11	34.67 ± 0.36	70.01 ± 0.21	71.22 ± 0.57	59.62 ± 0.13	62.35 ± 0.43
Forward	68.40 ± 0.36	51.27 ± 1.11	61.03 ± 0.61	33.27 ± 0.53	67.83 ± 0.86	68.63 ± 0.65	50.90 ± 0.99	51.53 ± 0.74
Bootstrap	71.03 ± 0.85	56.47 ± 1.18	61.10 ± 0.54	31.17 ± 0.59	70.80 ± 0.78	71.07 ± 0.78	54.87 ± 0.50	55.80 ± 1.23
Co-teaching+	72.44 ± 0.37	60.08 ± 0.48	55.33 ± 0.19	38.37 ± 0.77	57.02 ± 0.45	59.81 ± 0.72	41.11 ± 0.36	43.16 ± 0.29
Co-teaching	72.04 ± 0.61	58.78 ± 0.32	53.89 ± 0.25	37.51 ± 0.18	55.41 ± 0.19	58.31 ± 0.41	40.06 ± 0.69	41.95 ± 0.61

Table 2.7: Test accuracy (%) of TrustNet compared to baselines for CIFAR-100 under 0.4, and 0.6 noise rates and patterns: i) symmetric, ii) bimodal with $\mu_1 = 20$, $\sigma_1 = 10$, $\mu_2 = 70$, $\sigma_2 = 5$, and iii) truncated normal with $\mu = 10$, $\sigma = [1, 10]$. The best results are in bold.

CIFAR-100								
Methods	Symmetric		Bimodal Asymmetric		Truncated Normal Asymmetric			
	$\varepsilon = 0.4$	$\varepsilon = 0.6$	$\varepsilon = 0.4$	$\varepsilon = 0.6$	$\varepsilon = 0.4$		$\varepsilon = 0.6$	
					$\sigma = 1$	$\sigma = 10$	$\sigma = 1$	$\sigma = 10$
TrustNet	41.23 ± 0.43	**29.11 ± 0.12**	**45.01 ± 0.14**	**32.32 ± 0.30**	**37.66 ± 0.36**	**44.56 ± 0.42**	**23.96 ± 0.38**	**33.29 ± 0.41**
SCL	**42.30 ± 0.36**	28.43 ± 0.69	43.57 ± 0.42	30.70 ± 0.88	37.63 ± 0.62	43.50 ± 0.45	19.20 ± 0.57	31.93 ± 0.39
D2L	41.01 ± 0.21	21.41 ± 0.12	32.47 ± 0.43	10.55 ± 0.19	10.66 ± 0.16	10.32 ± 0.21	10.11 ± 0.38	10.05 ± 0.14
Forward	36.40 ± 0.37	16.00 ± 0.80	38.80 ± 0.28	19.03 ± 0.69	34.03 ± 0.33	39.80 ± 0.33	10.27 ± 0.47	22.90 ± 0.00
Bootstrap	28.40 ± 0.16	6.70 ± 0.59	32.17 ± 0.62	10.10 ± 0.94	27.23 ± 0.71	34.17 ± 0.96	6.10 ± 0.16	12.53 ± 1.84
Co-teaching+	39.35 ± 0.35	26.32 ± 0.54	34.64 ± 0.59	26.52 ± 0.58	34.17 ± 0.24	36.59 ± 0.32	18.24 ± 0.71	26.61 ± 0.33
Co-teaching	37.82 ± 0.22	25.44 ± 0.71	33.76 ± 0.54	26.12 ± 0.33	32.02 ± 0.56	33.85 ± 0.62	16.99 ± 0.32	25.33 ± 0.12

CIFAR-100

Table 2.7 summarizes the CIFAR-100 results over three runs. CIFAR-100 is more challenging than CIFAR-10 because it increases tenfold the number of classes while keeping the same amount of training data. This is clearly reflected in the accuracy results across all methods, but TrustNet overall seems to be more resilient. Here, TrustNet achieves the highest accuracy for both asymmetric noise patterns under all considered noise ratios. On average, the accuracy of TrustNet is higher than SCL, the second best solution, by 2%. The improvement is higher for higher noise ratios and lower variation, i.e., $\sigma = 1$. SCL outperforms TrustNet on symmetric noise of low and middle intensity, i.e., $\varepsilon = [0.4, 0.5]$, but the difference diminishes with increasing noise, and at 60% TrustNet performs better. Different from CIFAR-10, test accuracy variations become noticeable for truncated normal noise with increasing σ values producing a positive effect across most baselines. All other baselines perform worse.

2.9.3. REAL-WORLD NOISY DATA: CLOTHING1M

We use the noise pattern observed in real world data from the Clothing1M dataset to demonstrate the effectiveness and importance of estimating the noise transition matrix in TrustNet. Table 2.8 summarizes the results on the testing accuracy for TrustNet and the six baselines. The measured average noise ratio across all classes is 41%. Here, TrustNet achieves the highest accuracy, followed by SCL and Forward. Forward is another approach trying to estimate the noise transition matrix. The better accuracy of TrustNet

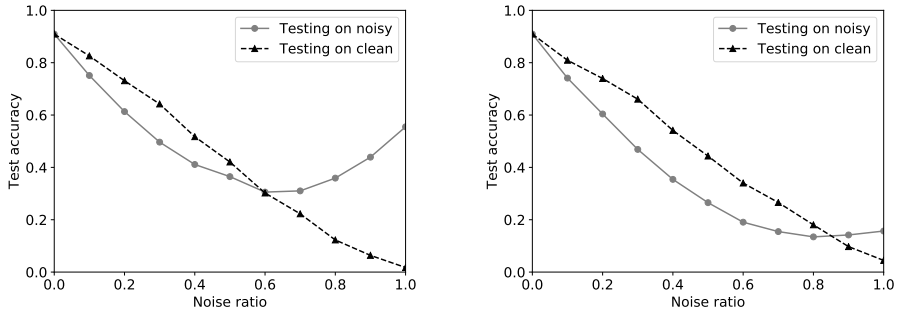
Table 2.8: Test accuracy (%) of TrustNet compared to baselines on clean testing set of real-world noisy Clothing1M.

Methods	TrustNet	SCL	D2L	Forward	Bootstrap	Co-teaching+	Co-teaching
Accuracy(%)	**73.06**	70.78	69.43	70.04	68.77	70.33	70.10

is attributed to the additional label estimation from LABELNET learned via the trusted data and dynamically weighting the loss functions from given and inferred labels. The promising results here confirm that the novel learning algorithm of TrustNet can tackle challenging label noise patterns appearing in real-world datasets.

2.9.4. DISCUSSION ON TRUSTNET

In this section, we discuss testing accuracy on clean and noisy samples. The analysis derived in § 2.5 consider testing on labels affected by the same noise as training data. This is due to the fact that the ground truth of labels is usually assumed unknown and not even available in the typical learning scenarios. However, the accuracy measured from the noisy testing data provides no information about how effective resilient networks defend the training process against the noisy data. Hence, related work on noisy label learning tests on clean samples, which show different trends as hinted in the evaluation section. Figure. 2.7 compares the two approaches across different noise patterns empirically. In general, in the case of clean test labels, the testing accuracy decreases with increasing noise ratios almost linearly. As for noisy labels, testing accuracy shows a clear quadratic trend, first decreasing before increasing again. Specifically, the lowest accuracy happens at noise ratio of 0.6 and 0.8 in the case of the truncated normal noise example with $\mu = 1$ and $\sigma = 0.5$ (Figure. 2.7(a)), and the bimodal noise example with $\mu_1 = 2, \sigma_1 = 0.5, \mu_2 = 7, \sigma_2 = 5$ (Figure. 2.7(b)), respectively. The reason is that specific class examples with erroneous labels become more numerous than examples with the true class, e.g., more truck images are labelled as an automobile than automobile images. Such an effect is missing when testing on clean labels.



(a) Truncated normal (b) Bimodal

Figure 2.7: Empirical testing accuracy of DNNs on CIFAR-10, comparing clean and noisy labeled test data under different noise ratios for both truncated normal (a) and bimodal (b) noise patterns.

2.10. CONCLUSION

In this work, we have addressed the challenge of label noise in the context of big data systems through the presentation of two novel and effective learning algorithms: LABELNET and TrustNet. These approaches aim to improve the robustness of deep learning models against noisy labels and enhance classification performance by leveraging auxiliary information of noisy labels.

The first approach, LABELNET, is motivated by the prevalence of noisy labels in publicly available images. It utilizes two networks, Amateur and Expert, to infer images by incorporating both inputs of images and noisy labels. Amateur, a deep CNN, is trained alongside Expert, which simulates the process of human experts correcting Amateur's output using ground truth labels. Through this training process, LABELNET effectively relabels and classifies images, transforming noisy labels into a learning advantage. Experimental results demonstrate the generalizability and effectiveness of LABELNET in extensive and challenging scenarios, outperforming existing robust network classifiers on MNIST, CIFAR benchmarks, and real-world datasets.

Building upon the insights gained from analyzing synthetic and real-world noise patterns, the second approach, TrustNet, presents a learning framework for noise-resilient classification. TrustNet first learns a noise transition matrix using a small set of trusted data and LABELNET. By combining the estimated labels inferred from LABELNET, TrustNet computes a robust loss function from both given and inferred labels, incorporating dynamic weights based on learning confidence, measured by the entropy. Evaluations on diverse datasets, including CIFAR-10, CIFAR-100, and Clothing1M, demonstrate that TrustNet effectively learns the noise transition and enhances the robustness of the loss function against noisy labels, achieving higher testing accuracy compared to state-of-the-art resilient networks.

Despite these contributions, the main limitation of LABELNET is the need to have both noisy labels and corresponding clean data for training, which is very expensive. While transfer learning can alleviate this, obtaining paired clean and noisy labels remains challenging. Additionally, TrustNet's performance is directly tied to LABELNET, as the latter is used within the TrustNet architecture. This dependency means that any issues faced by LABELNET in handling noisy labels can negatively impact TrustNet's overall accuracy and robustness.

In conclusion, the two presented approaches, LABELNET and TrustNet, contribute to the advancement of handling label noise in big data systems. By effectively leveraging noisy labels as learning features and employing robust loss functions, these approaches offer promising solutions for enhancing the reliability and accuracy of deep learning models in the presence of label noise. The empirical evaluations across various datasets underscore the effectiveness and potential of LABELNET and TrustNet in improving classification performance and handling real-world noise patterns, furthering the understanding and applicability of robust deep learning models in challenging big data scenarios.

3

LEARNING FROM NOISY LABELS WITH PARTIAL CLEANSED DATA SUPERVISION

*Learning robust deep models against noisy labels becomes ever critical when today's data is commonly collected from open platforms and subject to adversarial corruption. The information on the label corruption process, i.e., corruption matrix, can greatly enhance the robustness of deep models but still fall behind in combating hard classes. In this paper, we propose to construct a golden symmetric loss (*GSL*) based on the estimated corruption matrix as to avoid overfitting to noisy labels and learn effectively from hard classes.* GSL *is the weighted sum of the corrected regular cross entropy and reverse cross entropy. By leveraging a small fraction of trusted clean data, we estimate the corruption matrix and use it to correct the loss as well as to determine the weights of* GSL*. We theoretically prove the robustness of the proposed loss function in the presence of dirty labels. We provide a heuristics to adaptively tune the loss weights of* GSL *according to the noise rate and diversity measured from the dataset. We evaluate our proposed golden symmetric loss on both vision and natural language deep models subject to different types of label noise patterns. Empirical results show that* GSL *can significantly outperform the existing robust training methods on different noise patterns, showing accuracy improvement up to 18% on CIFAR-100 and 1% on real world noisy dataset of Clothing1M.*

3.1. INTRODUCTION

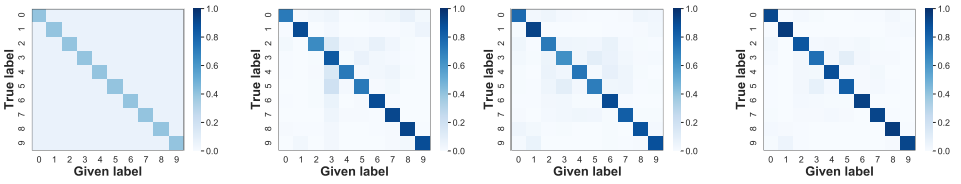
Diverse datasets collected from the public domain which power up deep learning models present new challenges – highly noisy labels. It is not only time consuming to collect labels but also difficult to ensure a consistent label quality due to various annotation errors [70] and adversarial attacks [111]. The large capacity of deep learning models enables effective learning from complex datasets but also suffers from overfitting to the noise structure in the dataset. The curse of memorization effect [85] can degrade the accuracy of deep learning models in the presence of highly noisy labels. For example, in [44] the accuracy of AlexNet to classify CIFAR-10 images drops from 77% to 10%, when there are randomly flipped labels.

Designing learning models that can robustly train on noisy labels is thus imperative. To distill the impact of noisy labels, the related work either filters out suspiciously noisy data, derives robust loss functions or tries to proactively correct labels. Symmetric Cross entropy Loss (SCL) is shown effective in combating label noise especially for hard classes by combing the regular with the reverse cross entropy. The former avoids overfitting and the latter is resilient to label noise. Given its promising results, there is yet to have a clear principle on how to weight the regular and reverse cross entropy terms, e.g., at different noise rates and patterns. In contrast, Distilling [116] and Golden Loss Correction (GLC) [49] advocate to use a small clean data to improve the estimated corruption matrix. Specifically, GLC trains the deep model on both a clean and noisy set, whose loss is corrected through the corruption matrix. While the clean set is evenly chosen from all classes, the corrupted labels may appear unevenly across classes depending on the noise pattern [25], [117]. As the corrected loss of GLC does not differentiate the difficulty of classes, it may not learn those hard classes effectively.

We propose GSL to construct the golden symmetric loss that dynamically weights regular/reverse cross entropy and corrects the label prediction based on the estimated corruption matrix. Similar to GLC, GSL leverages clean data to estimate the corruption matrix which is used to correct labels and decide the weights of the golden symmetric loss. As such, GSL can effectively differentiate the difficulty level of classes by adjusting the weights and mitigate the impact of noise overfitting via the golden symmetric cross entropy. Specifically, we use the noise rate and noise diversity to adaptively tune the weights of modified cross entropy and reverse cross entropy. We prove that modified cross entropy by using corruption matrix is noise tolerant same as the reverse cross entropy. Empirical evaluation on vision and text datasets shows that GSL outperforms the state-of-the-art methods under tested noise ratios from 0% to 100% for text datasets and noise ratios 30% and 60% for vision datasets. In addition, we illustrate that combining symmetric loss function and the corruption matrix estimation with correct dynamic weighting function is the best combination of robust methods against noisy label data.

The contributions of this chapter are summarized as follows:

- We design a noise resilient method that estimates the corruption matrix using a small proportion of trusted data and then corrects the wrong prediction into the symmetric cross-entropy loss function.
- Using noise properties, including rate and diversity, we design a weighting function for the symmetric loss function to adjust the weights of improved cross-entropy and reverse cross-entropy adaptively.



(a) Corruption matrix (b) GLC (c) SCL (d) GSL

Figure 3.1: Analysis of Label Corruption Matrix on CIFAR-10 Dataset. (a) displays the original corruption matrix with 0.6 symmetric label noise, while (b), (c), and (d) depict the estimated matrices using Golden Loss Correction (GLC), Symmetric Correction Loss (SCL), and Golden Symmetric Loss (GSL) methods, respectively.

- We compare GSL against state-of-the-art methods under noisy labels on the real-world dataset and synthetic vision and text datasets.

3.1.1. COMPARATIVE ANALYSIS OF NOISE-HANDLING STRATEGIES: A CIFAR-10 CASE STUDY

We demonstrate the advantages and disadvantages of GLC and SCL, and the their combination (the proposed GSL) through the example of learning convolution networks on CIFAR-10 injected with 60% symmetric noise. The experimental setup is detailed in §3.6. Figure 3.1 shows the corruption matrix of the injected noise and the confusion matrices from the predictions of SCL, GLC, and GSL. Even if the injected noise is symmetric across all classes (see Figure 3.1(a)), prediction errors are distributed asymmetrically across the classes (see Figure 3.1(b), Figure 3.1(c) and Figure 3.1(d)). Though GLC can achieve a lower average error rate than SCL (reflected in darker diagonal elements on average), it performs worse in hard classes, e.g., class 4 (cat) and class 6 (dog) (difference in blue shades across the diagonal elements). By setting up proper weights for two types of cross entropy, GSL is able to achieve both superior average and per class accuracy.

3.2. RELATED WORK

Enhancing the robustness of deep models against noisy labels is an active research area. The massive datasets needed to train deep models are commonly found corrupted, [112], severely degrading the achievable accuracy, [44]. The impact of label noise on deep neural networks is first characterized by the theoretical testing accuracy over a limited set of noise patterns [74]. [118] suggest an undirected graph model for modeling label noise in deep neural networks and indicate symmetric noise to be more challenging than asymmetric. Current solutions can be categorized into three directions: (i) filtering out noisy labels: [61], [119]; (ii) correcting noisy labels: [49], [70], [82], [93], [116]; and (iii) deriving noise resilient loss functions: [102], [120], [121].

3.2.1. NOISE RESILIENT LOSS FUNCTION

The loss function is modified to enhance the robustness to label noise by introducing new loss functions, [63], [75], or adjusting the weights of noisy data instances, [34], [102], [122]. Mean Absolute Error (MAE) [75], [123] and General Cross Entropy loss [123] are proposed

as a noise resilient alternative but at the cost of slow convergence. To avoid overfitting to noise, D2L [34] uses the subspace dimensionality to assign weights to each data point, whereas Konstantinov [102] determines the loss weights based on the trustworthiness level of data sources. [63] propose symmetric cross-entropy loss that combines a new term of reverse cross entropy with traditional cross entropy via constant weights on both terms. Meta-Weight-Net [57] re-weights samples during optimizing loss function in the training process by using a multi-layer perceptron to predict the weight of each sample. With the same perspective, [108] uses the similarity of samples to the clean instances in the validation set for re-weighting them in loss function.

3.2.2. LABEL CORRECTION

To avoid the data reduction caused by filtering, label correction methods adjust the predicted/given labels by using only noisy labels [70], [124], [125] or jointly with a small fraction of trusted data [49], [116], [126], [127]. [94] train the classifier by the "new" labels combining the raw and predicted labels without access to label ground truth. [70] estimate the noise corruption matrix by first training a classifier on the noisy labels and then using the softmax probabilities. [126] acquire human-verified labels to train a cleaning network for correcting noisy labels of multi-label classification problems. [127] estimate the noise transition probability by incorporating human assistance. [116] and [49] leverage a small set of clean data to estimate noise corruption matrix from the clean and noisy sets, respectively. DivideMix [128] is a semi-supervised method, including two networks and Gaussian Mixture Model for sample selection.

The proposed GSL combines resilient loss function and label correction by curating a small fraction of trusted data. We solicit a subset of informative data instances to estimate the corruption matrix and provide a minimum supervision on noisy labels. We also provide a heuristic to adaptively tune the weights of golden symmetric loss according to the noise characteristics of the dataset.

3.3. GOLDEN SYMMETRIC LOSS

Consider the classification problem having dataset $\tilde{\mathscr{D}} = \{(\boldsymbol{x}_n, \tilde{y}_n)\}_{n=1}^{N}$ where $\boldsymbol{x}_n \in \mathscr{X} \subset \mathbb{R}^d$ denotes the n^{th} observed sample, and $\tilde{y}_n \in \mathscr{Y} := \{1, ..., K\}$ the corresponding given label over K classes. Hereon n is ignored for the simplicity. $\tilde{y}$ is affected by label noise. The label corruption process is characterised by a corruption matrix $C_{ij} = P(\tilde{y} = j | y = i)$ for $i = 1, \ldots, K$ and $j = 1, \ldots, K$ where y is the true label. Synthetic noise patterns are expressed as a label corruption probability ε plus a noise label distribution. Let $g(\cdot, \boldsymbol{\theta})$ denote a neural network-based classifier parameterized by $\boldsymbol{\theta}$. For each data point $\boldsymbol{x}$, $f(\cdot, \boldsymbol{\theta})$ predicts the probability for each class label k: $p(k|\boldsymbol{x}) = \frac{e^{z_k}}{\sum_{j=1}^{K} e^{z_j}}$ where z_j are the logits.

3.3.1. SYMMETRIC CROSS ENTROPY

Let $q(k|\boldsymbol{x})$ denote the ground truth probability distribution over the K class labels where $q(k|\boldsymbol{x}) = 1$ for k equal to the true class y and $q(k|\boldsymbol{x}) = 0$ for all $k \neq y$. The cross entropy

loss (ℓ_{ce}) and reverse cross entropy loss[1] (ℓ_{rce}) for $\boldsymbol{x}$ are:

$$\ell_{ce} = -\sum_{k=1}^{K} q(k|\boldsymbol{x}) \log p(k|\boldsymbol{x}), \tag{3.1}$$

$$\ell_{rce} = -\sum_{k=1}^{K} p(k|\boldsymbol{x}) \log q(k|\boldsymbol{x}). \tag{3.2}$$

[63] combine cross entropy and reverse cross entropy into the symmetric cross entropy:

$$l_{sl} = \alpha\, \ell_{ce} + \beta\, \ell_{rce}. \tag{3.3}$$

where α and β are hyperparameters. On the one hand cross entropy loss is not robust to noise [75] but achieves good convergence [123]. On the other hand reverse cross entropy is tolerant to noise [63].

3.3.2. ESTIMATING NOISE CORRUPTION MATRIX

We estimate the noise corruption matrix as in [49]. The method fosters training a first classifier $g(\cdot,\theta)$ on noisy data to approximate the elements $\boldsymbol{C}_{ij}$ of the noise corruption matrix via a small fraction of trusted data $\mathscr{D}$ with known true label y. Practically given A_i the subset of trusted data with label of class i $\{A_i \subset \mathscr{D} : y = i\}$, the elements of $\boldsymbol{C}$ can be approximated by:

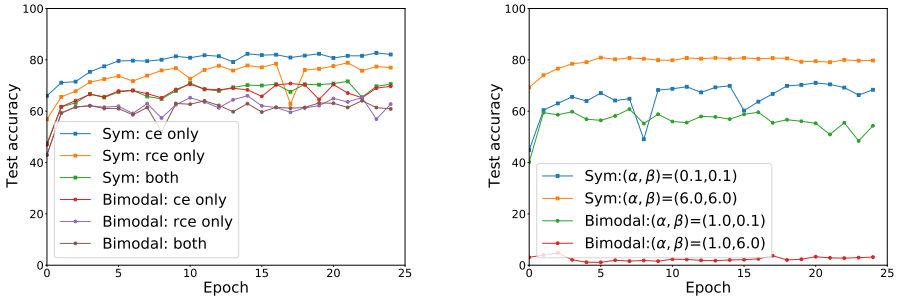
$$\hat{\boldsymbol{C}}_{ij} = P(\tilde{y} = j | y = i) \approx \frac{1}{|A_i|} \sum_{x \in A_i} g(\tilde{y} = j | \boldsymbol{x}, \Theta) \tag{3.4}$$

where $g(\tilde{y} = j|\boldsymbol{x},\Theta)$ denotes predicted probability of $\boldsymbol{x}$ having class label j. That is $\hat{\boldsymbol{C}}_{ij}$ is computed as the mean predicted probability of class j for all trusted data points having true label of class i.

3.3.3. TRAINING WITH CORRECTED LABELS

Let $\hat{\boldsymbol{C}}$ be the estimated noise corruption matrix. Using the method in [70], we increase the noise resilience by correcting the predictions of the classifier using $\hat{\boldsymbol{C}}$. Let $\hat{p}$ be the corrected predicted probabilities $\hat{p} = \hat{\boldsymbol{C}}^T p$, i.e. for data point $\boldsymbol{x}$: $\hat{p}(k|\boldsymbol{x}) = \sum_{i=1}^{K} \hat{\boldsymbol{C}}_{ik} p(i|\boldsymbol{x})$ for $k = 1,\ldots,K$. We enhance the regular cross entropy term. Applying the prediction correction to both terms holds lower benefits. We evaluate this empirically with extensive experiments on datasets of text, i.e. Twitter in Figure 3.2(a), and images, i.e. CIFAR-100 and CIFAR-10 in §3.8. Experiment details can be found in §3.7. We consider different datasets, noise rates, noise types and fractions of trusted data. We see that in all cases, except one with a difference < 0.3%, correcting only the cross entropy (*ce-only*) holds better results than correcting only the reverse cross entropy (*rce-only*) or correcting *both*. Focusing on Figure 3.2(a), *ce-only* improves accuracy by up to 5% and 8% for bimodal and symmetric noise, respectively. In case of CIFAR-10 and CIFAR-100 datasets the improvements are more pronounced with up to 11% and 50% respectively.

[1] To avoid problems with the logarithm, zero values of q are replaced by a small positive value, i.e. 10^{-4}.



(a) Different loss corrections ($\varepsilon = 0.7$)

(b) α, β-tuning: best/worst ($\varepsilon = 0.8$)

Figure 3.2: Test accuracy (%) of a 2-layer FC network with different loss corrections (a) and loss weights (b) across epochs on Twitter data.

3.3.4. GOLDEN SYMMETRIC LOSS

Towards a more effective and robust learning we propose to leverage the estimated noise corruption matrix $\hat{\boldsymbol{C}}$ to tune the two loss terms based on the observed noise pattern. α and β can significantly impact the final model accuracy. Tuning these parameters is essential since various datasets affected by different noise patterns require different optimal values [63]. Again we show this behavior by training a 2-layer FC neural network on the Twitter dataset under eleven different (α, β) combinations and two noise patterns with 80% noise. Figure 3.2(b) reports for each noise pattern the evolution over the training epochs of the test accuracy for the best and worst (α, β)-pair. For bimodal noise even with a small number of trials, the impact of (α, β) ranges from an accuracy close to 60% all the way down to almost 0%. Moreover only few (two out of eleven) (α, β)-pairs hold accuracy close to 60%. For symmetric noise the tuning impact is lower (limited between 70% and 80%) but the best and worst (α, β)-pair differ from the bimodal noise case. This underlines both the importance and difficulty of tuning (α, β). Motivated by the high impact of α and β, we propose to dynamically weight the regular and reverse cross entropy terms. Let $A(\cdot)$ and $B(\cdot)$ be weighting functions mapping $\hat{\boldsymbol{C}} \rightarrow \mathbb{R}$ we define a new loss function:

$$\ell_{GSL} = A(\hat{\boldsymbol{C}})\, \ell_{ce} + B(\hat{\boldsymbol{C}})\, \ell_{rce} \tag{3.5}$$

We call this new loss function golden symmetric loss. $A(\cdot)$ and $B(\cdot)$ should capture not only the intensity of the noise pattern, but also the diversity of the noise pattern (see Figure 3.2(b)).

3.3.5. DETERMINING WEIGHTS OF GOLDEN SYMMETRIC LOSS ($A(\cdot)$, $B(\cdot)$)

In general the more intense and asymmetric the noise pattern, the lower the weight values should be. Since the final loss function learns from both dirty and clean data (see the next paragraph), lower values of α and β reduce the influence of dirty data over that of clean data. Hence, we design $A(\cdot)$ and $B(\cdot)$ to capture both noise intensity and diversity. The intensity is given by the noise rate $\varepsilon \in [0, \ldots, 1]$, i.e. one minus the

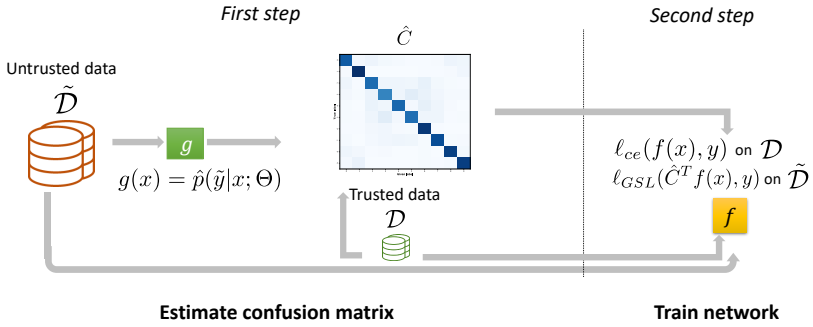


Figure 3.3: Overview of the training process for the Golden Symmetric Loss (GSL) divided into two steps: 1) estimating the confusion matrix and 2) training a robust network.

average of the diagonal elements of $\hat{\boldsymbol{C}}$. The diversity is measured via Jain's fairness index $J(x_1, x_2, \ldots, x_n) \triangleq (\Sigma_{i=n}^{n} x_i)^2 / n \Sigma_{i=n}^{n} x_i^2$. We choose J because it bounds the diversity on a similar scale as ε between 1 (all equal, full symmetry) down to $1/n$ (highest asymmetry). We apply J on all the $K(K-1)$ noise, i.e. off the diagonal, elements of $\hat{\boldsymbol{C}}$:

$$J = \frac{(\sum_{i=1}^{K} \sum_{j=1, j\neq i}^{K} \hat{\boldsymbol{C}}_{ij})^2}{K(K-1) \sum_{i=1}^{K} \sum_{j=1, j\neq i}^{K} \hat{\boldsymbol{C}}_{ij}^2} \tag{3.6}$$

For symmetric noise $J = 1$, the more asymmetric the smaller J. Final weights proportional to J, ε.

3.3.6. PUTTING IT ALL TOGETHER

As a final step, to maximize the utility of the trusted data, we foster $\mathscr{D}$ as additional trusted training data for $f(\cdot)$. Since $\mathscr{D}$ contains the true labels y no prediction correction is applied. Hence, the overall loss function for data points from both $\mathscr{D}$ and $\tilde{\mathscr{D}}$ is:

for $x \in \tilde{\mathscr{D}}$,

$$\begin{aligned} l = &- A(\hat{\boldsymbol{C}}) \sum_{k=1}^{K} q(k|\boldsymbol{x}) \log(\sum_{i=1}^{K} \hat{\boldsymbol{C}}_{ik} p(i|\boldsymbol{x})) \\ &- B(\hat{\boldsymbol{C}}) \sum_{k=1}^{K} p(k|\boldsymbol{x}) \log q(k|\boldsymbol{x}) \end{aligned} \tag{3.7}$$

and for $x \in \mathscr{D}$,

$$l = - \sum_{k=1}^{K} q(k|\boldsymbol{x}) \log p(k|\boldsymbol{x}). \tag{3.8}$$

Figure 3.3 summarises visually the training process divided into two main steps: (i) estimating noise corruption matrix through the first network g trained on untrusted dataset $\tilde{\mathscr{D}}$ and (ii) training classifier f on both untrusted $\tilde{\mathscr{D}}$ and trusted $\mathscr{D}$ through the golden symmetric loss.

3.4. GSL ALGORITHM

The concrete procedure of the GSL framework is described in Algorithm 3. During the algorithm, we train two DNNs $g(\cdot;\theta)$ and $f(\cdot,\theta)$ as the corruption matrix estimator and a noise resilient network, respectively. First, $g(\cdot,\theta)$ is trained by untrusted data $\mathscr{D}$ to be used to approximate $p(\tilde{y}|y)$ [49] (line 1). According to GLC [49], the corruption matrix $\hat{\boldsymbol{C}}$ is estimated by using the trusted data $\mathscr{D}$ and the trained network $g(\cdot)$ (line 3-10). Now with $\hat{\boldsymbol{C}}$, we try to train a new DNN $f(\cdot;\theta)$ via untrusted data $\mathscr{D}$ to construct a robust symmetric loss function against noisy label data. As we mentioned before, the Golden symmetric Loss includes cross-entropy (ℓ_{ce}) and reverse cross-entropy (ℓ_{rce}) with their corresponding weights $A(\cdot)$ and $B(\cdot)$, respectively. To adjust the weights of cross-entropy and reverse cross-entropy, we design a weighting function based on Jain's fairness index. The weighting function determines the weights of ℓ_{ce} and ℓ_{rce} by considering the noise intensity and diversity represented in the corruption matrix $\hat{\boldsymbol{C}}$ (line 12). $\hat{\varepsilon}$ denotes the noise intensity estimation of the training data ($\hat{\varepsilon}$) via taking average of diagonal elements of $\hat{\boldsymbol{C}}$ (line 13). By considering the noise intensity ($\hat{\varepsilon}$) and the value of Jain's fairness index, the weights of $A(\cdot)$ and $B(\cdot)$ are calculated (line 14-20). Finally, the weighted symmetric loss function with label correction on cross-entropy is used for training DNN $f(\cdot;\theta)$ on untrusted data $\mathscr{D}$ (line 22). For the label correction and weight calculation, the estimated corruption matrix plays an important role. In addition to use untrusted data, $f(\cdot;\theta)$ is trained directly with the available trusted data (line 23).

3.5. THEORETICAL ANALYSIS

We prove that the cross entropy loss with label correction is noise tolerant under the definition put forth by [75], [76], extending prior results.

Let the risk of classifier f and loss function ℓ_{ce} under clean labels be $R(f) = \mathbb{E}_{\boldsymbol{x},y}[\ell_{ce}(f(\boldsymbol{x}),y)]$ and the *risk* under noise rate ε be $R_\varepsilon(f) = \mathbb{E}_{\boldsymbol{x},\tilde{y}}[\ell_{ce}(f(\boldsymbol{x}),\tilde{y})]$. $\mathbb{E}$ indicates the expectation taken over the random variables indicated as its subscripts. With prediction correction via $\boldsymbol{C}$, the risk becomes $R_\varepsilon(f,\boldsymbol{C}) = \mathbb{E}_{\boldsymbol{x},\tilde{y}}[\ell_{ce}(\boldsymbol{C}^T f(\boldsymbol{x}),\tilde{y})]$. Let f^* and f_ε^* be the global minimizers of $R(f)$ and $R_\varepsilon(f)$, respectively, and $\boldsymbol{C}^* = p(\tilde{y}|y)$ and $\hat{\boldsymbol{C}}$ be the true and estimated noise corruption matrices, respectively.

Theorem 3.5.1 *In a multi-class classification problem, the cross-entropy loss with prediction correction (ℓ_{ce}) is noise-tolerant:*

- *Under symmetric label noise, if the noise rate ε satisfies:*

$$\varepsilon < \frac{K-1}{K-\frac{\Delta\mathscr{A}}{\Delta R}},$$

  where $\Delta\mathscr{A} = \sum_{k=1}^{K} \ell_{ce}(\boldsymbol{C}^{*T} f(\boldsymbol{x}),k) - \sum_{k=1}^{K} \ell_{ce}(\hat{\boldsymbol{C}}^T f(\boldsymbol{x}),k)$, and ΔR is the difference in risk minimization between the optimal classifier and f.
- *Under flip noise if the noise rate ε_{yk} for each class y satisfies:*

$$\varepsilon_{yk} \leq (1+\frac{\Delta\mathscr{W}_y}{\Delta\mathscr{W}_k}) - \varepsilon_y(1+\frac{\Delta\mathscr{W}_y}{\Delta\mathscr{W}_k}),$$

  where $1-\varepsilon_y$ and ε_{yk} are the correct and flipped class probabilities, respectively.

Algorithm 3: Golden Symmetric Loss

Input : Trusted data $\mathscr{D} = \{\boldsymbol{x}, y\}$, Untrusted data $\tilde{\mathscr{D}} = \{\boldsymbol{x}, \tilde{y}\}$, Epoch E_{max}, loss ℓ_{ce}, loss ℓ_{rce}

Output : Trained network $f(\boldsymbol{x}, \theta)$

Train network $g(\boldsymbol{x}) = \hat{p}(\tilde{y}|\boldsymbol{x}; \Theta) \in \mathbb{R}^K$ on $\tilde{\mathscr{D}}$
Fill $\hat{\boldsymbol{C}} \in \mathbb{R}^{K \times K}$ with zeros
for $k = 1, 2, \ldots, K$ **do**
    $num\_trust\_samples = 0$
    for $(\boldsymbol{x}_i, y_i) \in \mathscr{D}$ *such that* $y_i = k$ **do**
        $num\_trust\_samples$ += 1
        $\hat{\boldsymbol{C}}_{k\bullet}$ += $g(\boldsymbol{x}_i)$ {add $g(\boldsymbol{x}_i)$ to kth row}
    end
    $\hat{\boldsymbol{C}}_{k\bullet}$ /= $num\_trust\_samples$
end
Initialize new model $f(\boldsymbol{x}) = \hat{p}(y|\boldsymbol{x}; \Theta)$
$J(\hat{\boldsymbol{C}}) = \frac{(\sum_{i=1}^{K} \sum_{j=1, j\neq i}^{K} \hat{\boldsymbol{C}}_{ij})^2}{K(K-1) \sum_{i=1}^{K} \sum_{j=1, j\neq i}^{K} \hat{\boldsymbol{C}}_{ij}^2}$
$\hat{\varepsilon}$ = 1 - $AVG(diag(\hat{\boldsymbol{C}}))$; `// noise ratio estimation based on corruption matrix diagonal`
if $\hat{\varepsilon} \leq 0.5$ **then**
    $A(\hat{\boldsymbol{C}}) = J(\hat{\boldsymbol{C}})\hat{\varepsilon}$
    $B(\hat{\boldsymbol{C}}) = J(\hat{\boldsymbol{C}})(1 - \hat{\varepsilon})$
end
else
    $A(\hat{\boldsymbol{C}}) = J(\hat{\boldsymbol{C}})(1 - \hat{\varepsilon})$
    $B(\hat{\boldsymbol{C}}) = J(\hat{\boldsymbol{C}})\hat{\varepsilon}$
end
Train with $A(\hat{\boldsymbol{C}})\ell_{ce}(\hat{\boldsymbol{C}} f(\boldsymbol{x}), \tilde{y}) + B(\hat{\boldsymbol{C}})\ell_{rce}(f(\boldsymbol{x}), \tilde{y})$ on $\tilde{\mathscr{D}}$
Train with $\ell_{ce}(f(\boldsymbol{x}), y)$ on $\mathscr{D}$

The proof is based on the risk minimization framework aiming to show under which condition $R_\varepsilon(f^*, \boldsymbol{C}^*) - R_\varepsilon(f, \hat{\boldsymbol{C}}) \leq 0$, i.e. the loss function is robust to noise. The condition $\varepsilon < \frac{K-1}{K - \frac{\Delta\mathscr{A}}{\Delta R}}$ is a generalization of the previous bound $\varepsilon < \frac{K-1}{K}$ by [75]. Without label correction $\Delta\mathscr{A} = 0$ which corresponds to the previous result. Label correction improves the robustness by allowing higher noise rates, i.e. with label correction $\frac{\Delta\mathscr{A}}{\Delta R} \geq 0$ and hence $\frac{K-1}{K} \leq \frac{K-1}{K - \frac{\Delta\mathscr{A}}{\Delta R}}$. Similar observations hold for flip noise bound.

Proof: For symmetric noise:

$$
\begin{aligned}
R_\varepsilon(f, \hat{\boldsymbol{C}}) &= \mathbb{E}_{\boldsymbol{x}, \tilde{y}}[\ell_{ce}(\hat{\boldsymbol{C}}^T f(\boldsymbol{x}), \tilde{y})] = \mathbb{E}_{\boldsymbol{x}} \mathbb{E}_{y|\boldsymbol{x}} \mathbb{E}_{\tilde{y}|\boldsymbol{x}, y}[\ell_{ce}(\hat{\boldsymbol{C}}^T f(\boldsymbol{x}), y)] \\
&= \mathbb{E}_{\boldsymbol{x}, y}[(1 - \varepsilon)\ell_{ce}(\hat{\boldsymbol{C}}^T f(\boldsymbol{x}), y)] + \mathbb{E}_{\boldsymbol{x}, y}[\frac{\varepsilon}{K-1}(\sum_{k \neq y} \ell_{ce}(\hat{\boldsymbol{C}}^T f(\boldsymbol{x}), k))] \qquad (3.9)
\end{aligned}
$$

$$
\begin{aligned}
&= \mathbb{E}_{\boldsymbol{x},y}[(1-\varepsilon)\ell_{ce}(\hat{\boldsymbol{C}}^T f(\boldsymbol{x}), y)] + \mathbb{E}_{\boldsymbol{x},y}[\frac{\varepsilon}{K-1}(\sum_{k=1}^{K}\ell_{ce}(\hat{\boldsymbol{C}}^T f(\boldsymbol{x}), k))] \\
&- \mathbb{E}_{\boldsymbol{x},y}[\frac{\varepsilon}{K-1}(\ell_{ce}(\hat{\boldsymbol{C}}^T f(\boldsymbol{x}), y))] = (1-\varepsilon)R(f, \hat{\boldsymbol{C}}) \\
&+ \frac{\varepsilon}{K-1}(\sum_{k=1}^{K}\ell_{ce}(\hat{\boldsymbol{C}}^T f(\boldsymbol{x}), k) - R(f, \hat{\boldsymbol{C}})) \\
&= R(f, \hat{\boldsymbol{C}})(1 - \frac{\varepsilon K}{K-1}) + \frac{\varepsilon}{K-1}(\sum_{k=1}^{K}\ell_{ce}(\hat{\boldsymbol{C}}^T f(\boldsymbol{x}), k))
\end{aligned}
$$

Let $\mathscr{A}(\hat{\boldsymbol{C}}^T f(\boldsymbol{x}), y) = \sum_{k=1}^{K}\ell_{ce}(\hat{\boldsymbol{C}}^T f(\boldsymbol{x}), k)$. Then we can rewrite (3.9) as $R_\varepsilon(f, \boldsymbol{C}) = (1 - \frac{\varepsilon K}{K-1})R(f, \boldsymbol{C}) + \frac{\varepsilon}{K-1}\mathscr{A}(\boldsymbol{C}^T f(\boldsymbol{x}), y)$, thus:

$$
\begin{aligned}
R_\varepsilon(f^*, \boldsymbol{C}^*) - R_\varepsilon(f, \hat{\boldsymbol{C}}) = (1 - \frac{\varepsilon K}{K-1}) \underbrace{(R(f^*, \boldsymbol{C}^*) - R(f, \hat{\boldsymbol{C}}))}_{\Delta R} \\
+ \frac{\varepsilon}{K-1} \underbrace{(\mathscr{A}({\boldsymbol{C}^*}^T f^*(\boldsymbol{x}), y) - \mathscr{A}(\hat{\boldsymbol{C}}^T f(\boldsymbol{x}), y))}_{\Delta\mathscr{A}}
\end{aligned}
\tag{3.10}
$$

where $\Delta R \leq 0$, because f^* is the global minimizer for R and $\boldsymbol{C}^*$ the optimal noise corruption matrix. Similarly, $\Delta\mathscr{A} \leq 0$ because for the optimal case we can say $\mathscr{A}({\boldsymbol{C}^*}^T f^*(\boldsymbol{x}), y) \approx 0$. ℓ_{ce} with label correction is robust to noise when $R_\varepsilon(f^*, \boldsymbol{C}^*) - R_\varepsilon(f, \hat{\boldsymbol{C}}) \leq 0$. This is true when:

$$
\begin{aligned}
R_\varepsilon(f^*, \boldsymbol{C}^*) - R_\varepsilon(f, \hat{\boldsymbol{C}}) &= (1 - \frac{\varepsilon K}{K-1})\Delta R + \frac{\varepsilon}{K-1}\Delta\mathscr{A} \\
&= \Delta R - \frac{\varepsilon K}{K-1}\Delta R + \frac{\varepsilon}{K-1}\Delta\mathscr{A} \leq 0 \xrightarrow{\Delta R \leq 0} \\
1 - \frac{\varepsilon K}{K-1} + \frac{\varepsilon}{K-1}\frac{\Delta\mathscr{A}}{\Delta R} &\geq 0 \rightarrow \\
\frac{\varepsilon}{K-1}(K - \frac{\Delta\mathscr{A}}{\Delta R}) \leq 1 &\rightarrow \varepsilon \leq \frac{K-1}{K - \frac{\Delta\mathscr{A}}{\Delta R}}
\end{aligned}
\tag{3.11}
$$

With no label correction, $\boldsymbol{C}$ is missing in $\Delta\mathscr{A}$ and the two terms become equal, i.e. $\Delta\mathscr{A} = 0$. In this condition the bound becomes $\varepsilon < \frac{K-1}{K}$ as found by [75] for cross-entropy without label correction. ΔR is non-positive because it is defined as the risk difference with respect to an optimal solution minimizing the risk. Similarly, $\Delta\mathscr{A}$ is non-positive because defined as the difference with respect to an optimal solution and an optimal corruption matrix. Hence, the ratio of $\frac{\Delta\mathscr{A}}{\Delta R}$ must be non-negative. Since $\frac{\Delta\mathscr{A}}{\Delta R} \geq 0$, $\varepsilon < \frac{K-1}{K} \leq \frac{K-1}{K - \frac{\Delta\mathscr{A}}{\Delta R}}$ the new bound can be seen as generalization of the previous bound. $\frac{\Delta\mathscr{A}}{\Delta R}$ should also be less than one to ensure a meaningful bound on ε, avoiding scenarios of noise rate greater than 1.

For asymmetric flip noise, $1 - \varepsilon_y$ is the probability of a label being correct (i.e., $k = y$), and the noise condition $\varepsilon_{yk} < 1 - \varepsilon_y$ generally states that a sample $\boldsymbol{x}$ has a higher

probability $(1-\varepsilon_y)$ of being classified correctly as class y, rather than the probability (ε_{yk}) of being classified incorrectly as class $k \neq y$.

$$
\begin{aligned}
R_\varepsilon(f, \boldsymbol{C}) &= \mathbb{E}_{\boldsymbol{x},\tilde{y}}[\ell_{ce}(\boldsymbol{C}^T f(\boldsymbol{x}), \tilde{y})] = \mathbb{E}_{\boldsymbol{x}}\mathbb{E}_{y|\boldsymbol{x}}\mathbb{E}_{\tilde{y}|\boldsymbol{x},y}[\ell_{ce}(\boldsymbol{C}^T f(\boldsymbol{x}), y)] \\
&= \mathbb{E}_{\boldsymbol{x},y}[(1-\varepsilon_y)\ell_{ce}(\boldsymbol{C}^T f(\boldsymbol{x}), y)] + \mathbb{E}_{\boldsymbol{x},y}[\sum_{k\neq y} \varepsilon_{yk}\ell_{ce}(\boldsymbol{C}^T f(\boldsymbol{x}), k)] \\
&= \mathbb{E}_{\boldsymbol{x},y}[(1-\varepsilon_y)\sum_{k=1}^{K} \ell_{ce}(\boldsymbol{C}^T f(\boldsymbol{x}), k)] - \mathbb{E}_{\boldsymbol{x},y}[(1-\varepsilon_y)\sum_{k\neq y} \ell_{ce}(\boldsymbol{C}^T f(\boldsymbol{x}), y)] \\
&+ \mathbb{E}_{\boldsymbol{x},y}[\sum_{k\neq y} \varepsilon_{yk}\ell_{ce}(\boldsymbol{C}^T f(\boldsymbol{x}), k)] \\
&= \mathbb{E}_{\boldsymbol{x},y}[(1-\varepsilon_y)\sum_{k=1}^{K} \ell_{ce}(\boldsymbol{C}^T f(\boldsymbol{x}), k)] + \mathbb{E}_{\boldsymbol{x},y}[\sum_{k\neq y} (1-\varepsilon_y-\varepsilon_{yk})\ell_{ce}(\boldsymbol{C}^T f(\boldsymbol{x}), k)]
\end{aligned}
\tag{3.12}
$$

Similar to the symmetric case we require that $R_\varepsilon(f^*, \boldsymbol{C}^*) - R_\varepsilon(f, \hat{\boldsymbol{C}}) \leq 0$ for the loss to be robust to noise:

$$
\begin{aligned}
R_\varepsilon(f^*, \boldsymbol{C}^*) - R_\varepsilon(f, \hat{\boldsymbol{C}}) = \mathbb{E}_{\boldsymbol{x},y}[(1-\varepsilon_y)(\sum_{k=1}^{K} \underbrace{\ell_{ce}(\boldsymbol{C}^{*T} f^*(\boldsymbol{x}), k) - \ell_{ce}(\hat{\boldsymbol{C}}^T f(\boldsymbol{x}), k))}_{\Delta\mathscr{W}_y} \\
+ \sum_{k\neq y} (1-\varepsilon_y-\varepsilon_{yk}) \underbrace{\ell_{ce}(\boldsymbol{C}^{*T} f^*(\boldsymbol{x}), k) - \ell_{ce}(\hat{\boldsymbol{C}}^T f(\boldsymbol{x}), k)}_{\Delta\mathscr{W}_k}] \leq 0
\end{aligned}
\tag{3.13}
$$

where $\Delta\mathscr{W}_y \leq 0$ and $\Delta\mathscr{W}_k \leq 0$ because $\boldsymbol{C}^*$ is the optimal noise corruption matrix. Rewriting (3.13):

$$
\begin{aligned}
&\mathbb{E}_{\boldsymbol{x},y}[\sum_{k=1}^{K}(1-\varepsilon_y)\Delta\mathscr{W}_y + \sum_{k\neq y}\varepsilon_{yk}\Delta\mathscr{W}_k] \leq 0 \rightarrow \\
&\sum_{k=1}^{K}(1-\varepsilon_y)\Delta\mathscr{W}_y + \sum_{k\neq y}(1-\varepsilon_y-\varepsilon_{yk})\Delta\mathscr{W}_k \leq 0 \rightarrow \\
&\Delta\mathscr{W}_y - \varepsilon_y\Delta\mathscr{W}_y \leq -\Delta\mathscr{W}_k + \varepsilon_y\Delta\mathscr{W}_k + \varepsilon_{yk}\Delta\mathscr{W}_k \xrightarrow{\Delta\mathscr{W}_k \leq 0} \\
&\frac{\Delta\mathscr{W}_y}{\Delta\mathscr{W}_k} - \varepsilon_y\frac{\Delta\mathscr{W}_y}{\Delta\mathscr{W}_k} \geq -1 + \varepsilon_y + \varepsilon_{yk} \rightarrow \frac{\Delta\mathscr{W}_y}{\Delta\mathscr{W}_k} - \varepsilon_y(\frac{\Delta\mathscr{W}_y}{\Delta\mathscr{W}_k} + 1) \geq \varepsilon_{yk} - 1
\end{aligned}
\tag{3.14}
$$

According to (3.14), the bound is $\varepsilon_{yk} \leq (1 + \frac{\Delta\mathscr{W}_y}{\Delta\mathscr{W}_k}) - \varepsilon_y(1 + \frac{\Delta\mathscr{W}_y}{\Delta\mathscr{W}_k})$. With no label correction $\Delta\mathscr{W}_y = 0$ and the bound becomes $\varepsilon_{yk} < 1 - \varepsilon_y$ as found by prior art.

□

3.6. EXPERIMENTAL SETUP

3.6.1. DATASET, ARCHITECTURE AND PARAMETERS

We consider two types of datasets: vision and text analysis. For vision, we use convolutional neural networks (CNN) to classify CIFAR-10 and CIFAR-100 with injected label

noise and Clothing1M as real world noisy dataset. For text, we use fully connected neural networks to classify noisy Twitter and Stanford Sentiment Treebank (SST). In principle, we use the same network architecture on all comparative approaches across different noise resilience techniques. In addition, we test the original network from the respective papers too and report the best results among the two.

- **CIFAR-10 [129]:** It contains 60K images classified into 10 classes: 50K as a training set and 10K as validation set. We use the architecture of Wide-ResNet by [130] with depth 28 and a widening factor 10 and train it with SGD with Nesterov momentum and a cosine learning rate schedule [131]. For GSL, we first train g for 75 epochs to obtain the noise corruption matrix. Then we train f for 120 epochs.
- **CIFAR-100 [129]:** It contains 60K images classified into 100 classes: 50K as training set and 10K as the validation set. We use the same Wide-ResNet architecture used for CIFAR-10. For GSL, we train the g and f networks for 75 and 200 epochs, respectively.
- **Clothing1M [25]:** This is a real world dataset with label noise. It includes clothing images scrapped from the Internet classified into 14 categories. We resize and crop each image to 224×224 pixels. This dataset contains 1 million noisy labeled samples that we use for training as our untrusted data. Besides, it consists of 57K human-annotated images, which we take 47K images as our trusted examples and 10K images for testing. These two sets have both given (scrapped) and true (human-checked) labels. We use ResNet-50 pretrained with ImageNet and further train for 10 epochs with batch size 32, SGD optimizer, momentum 0.9, weight decay 10^{-3}, and learning rate 10^{-3} which is divided by 10 after 5 epochs.
- **Twitter [132]:** The Twitter dataset includes 1,827 tweets annotated with 25 POS tags split in 1000 tweets as training set, 327 tweets as development set and 500 tweets as test set. We add development set to training set, and consider it as a training set. We use a 2-layer fully connected network with 256 hidden neurons each and GELU nonlinearity as activation function. We train g with Adam for 15 epochs with batch size 64 and learning rate of 0.001. We train f for 25 epochs. To regularize all linear output layer, we use ℓ_2 weight decay with $\lambda = 5 \times 10^{-5}$.
- **Stanford Sentiment Treebank [133]:** The SST dataset includes single sentence movie reviews. We use the 2-class version, including 6911 reviews in the training set, a development set with 872 reviews, and 1821 reviews in the test set. We augment the training set by using development set. We learn 100-dimensional word vectors from scratch for a vocab size of 10000. We train a word-averaging model with an affine output layer using Adam optimizer for 5 epochs for network g and 10 epochs for network f. The batch size and learning rate are 50 and 0.001, respectively. To regularize all linear output layer, we use ℓ_2 weight decay with $\lambda = 1 \times 10^{-4}$.

3.6.2. NOISE CORRUPTION

We consider symmetric noise and two different asymmetric noises, namely flip and bimodal. Symmetric noise corrupts the true label into a random other labels with equal probability based on the noise rate. The flip noise is generated by flipping the original label to a paired other class with a specific probability. The bimodal noise imitates targeted adversarial attacks [111]. Specifically, the true labels are corrupted into two

neighborhoods centered on two targeted classes, each of which follows truncated normal distribution, $\mathcal{N}^T(\mu,\sigma,a,b)$. μ specifies the target and σ controls the spread. a and b simply define the class label boundaries. For CIFAR-10 we target class 3 and 7, for CIFAR-100 class 30 and 70, for Twitter class 6 and 18, and for SST class 0 and 1. Instead, Clothing1M is already affected by real world label noise and left untouched.

3.7. EVALUATION

In this section, we empirically compare GSL against state of the art noise resilient networks on noisy vision and text data. We aim to show the effectiveness of GSL via testing accuracy on diverse and challenging noise patterns. Our target evaluation metric is the accuracy achieved on the clean testing set, i.e. not affected by noise.

3.7.1. VISION ANALYSIS

We compare GSL against ten noise resilient networks from the state of the art: GLC [49], SCL [63], FORWARD [70], BOOTSTRAP [94], CO-TEACHING+ [119], DIVIDEMIX [128], GFORWARD, SGFORWARD, TMATRIX and STMATRIX. As the proposed loss of golden symmetric cross entropy is general and can be combined with different resilient networks, we hence use following four variations of loss correction and symmetric cross entropy on the existing work:

- Forward gold (GFORWARD): we replace the estimation of the corruption matrix by the identity matrix on trusted samples and apply loss correction through the matrix.
- True corruption matrix (TMATRIX): we use the true corruption matrix and apply loss correction through it.
- Forward gold with symmetric cross entropy (SGFORWARD): we extend the corrected loss of GFORWARD to the corrected symmetric cross entropy as in the GSL.
- True corruption with symmetric cross entropy (STMATRIX): we apply golden symmetric cross entropy and the true corruption matrix instead of the estimated matrix.

For training GSL, CO-TEACHING+, SGFORWARD, GFORWARD, TMATRIX, STMATRIX, DIVIDEMIX and GLC, we use PyTorch v1.4.0. For all other methods, we use Keras v2.2.4 and Tensorflow v1.13.0. All experiments run on Alienware Aurora R11 equipped with an NVIDIA GeForce RTX 2080 Ti, 32 GB RAM, and Core i9 CPU @ 3.70 GHz.

We assume 10% of trusted data is available for GSL, GLC, GFORWARD and SGFORWARD. Table 3.1 summarizes the testing accuracy for all combinations of noise patterns and comparative approaches.

For CIFAR-10, we report the average and standard deviation across three runs in Table 3.1. GSL achieves the highest accuracy among all resilient networks except for flip noise with 30% noise rate. DIVIDEMIX, STMATRIX and SGFORWARD are the closest rivals to GSL. GSL and SGFORWARD both use the same mechanism in the loss function. Besides, GSL has 2 to 8% higher accuracy than GLC, demonstrating the benefit of introducing symmetric cross-entropy, especially in high noise rates. In terms of comparison between GSL and SCL, the accuracy difference is even more visible, implying the benefit of using corruption matrix to assign weights on two terms in symmetric cross-entropy. We note that SCL uses an 8-layer CNN with 6 convolutional layers followed by 2 fully connected layers instead of a Wide ResNet because of the superior results. SCL performs particularly

Table 3.1: The average test accuracy (%) of GSL on CIFAR10/CIFAR100 and real-world noisy Clothing1M compared to the baselines corrupted with 30% and 60% noise with three different patterns symmetric, bimodal, and flip (Best results in bold).

CIFAR-10

Noise Rate	Noise Pattern	GSL	GLC	SCL	FORWARD	BOOTSTRAP	SGFORWARD	CO-TEACHING+	DIVIDEMIX	GFORWARD	TMATRIX	STMATRIX
30%	Sym.	**92.90** ± 0.24	89.94 ± 0.36	83.50 ± 0.28	74.28 ± 0.20	75.62 ± 0.15	90.81 ± 0.22	76.70 ± 0.72	91.68 ± 0.28	79.76 ± 0.92	90.66 ± 0.19	91.09 ± 0.36
30%	Bimodal	**92.81** ± 0.18	90.18 ± 0.91	83.06 ± 0.21	73.42 ± 0.54	75.69 ± 0.12	91.10 ± 0.45	75.21 ± 0.54	84.13 ± 0.18	78.45 ± 0.49	89.95 ± 0.31	90.59 ± 0.24
30%	Flip	90.30 ± 0.36	**91.15** ± 0.17	81.53 ± 0.33	78.72 ± 0.19	78.51 ± 0.38	90.54 ± 0.42	79.83 ± 0.78	86.01 ± 0.43	81.18 ± 0.51	88.79 ± 0.71	89.29 ± 0.73
60%	Sym.	**89.10** ± 0.19	82.24 ± 0.59	72.71 ± 0.86	53.48 ± 0.79	57.56 ± 1.86	88.02 ± 0.56	63.33 ± 0.93	88.27 ± 0.76	60.62 ± 0.61	87.31 ± 0.14	88.07 ± 0.53
60%	Bimodal	**87.75** ± 0.24	84.98 ± 0.16	60.76 ± 0.82	47.49 ± 0.69	48.18 ± 1.01	86.29 ± 0.52	57.82 ± 0.73	81.45 ± 0.37	58.93 ± 0.46	84.33 ± 0.62	86.79 ± 0.21
60%	Flip	**86.23** ± 1.10	80.40 ± 0.32	55.84 ± 0.70	59.99 ± 0.47	59.66 ± 0.45	82.19 ± 0.43	65.31 ± 0.36	79.76 ± 0.67	62.04 ± 0.63	81.88 ± 0.37	84.91 ± 0.37

CIFAR-100

Noise Rate	Noise Pattern	GSL	GLC	SCL	FORWARD	BOOTSTRAP	SGFORWARD	CO-TEACHING+	DIVIDEMIX	GFORWARD	TMATRIX	STMATRIX
30%	Sym.	**75.80** ± 0.12	61.81 ± 1.19	58.01 ± 0.71	42.33 ± 1.34	41.51 ± 1.54	72.31 ± 0.77	54.04 ± 0.33	72.73 ± 0.22	52.64 ± 0.73	70.42 ± 0.71	73.04 ± 0.69
30%	Bimodal	**76.25** ± 0.35	61.77 ± 0.91	46.88 ± 0.63	45.22 ± 0.13	42.14 ± 0.38	73.65 ± 0.29	55.42 ± 0.65	74.21 ± 0.18	54.69 ± 0.82	72.07 ± 0.32	74.41 ± 0.22
30%	Flip	75.80 ± 0.21	66.55 ± 0.52	55.46 ± 0.47	54.92 ± 0.25	54.44 ± 0.59	**75.83** ± 0.42	58.46 ± 0.61	74.16 ± 0.39	58.32 ± 0.20	73.11 ± 0.14	75.15 ± 0.43
60%	Sym.	**68.49** ± 0.16	52.23 ± 0.85	29.00 ± 0.54	18.56 ± 1.11	16.22 ± 0.81	66.32 ± 0.79	38.15 ± 0.94	66.81 ± 0.66	39.32 ± 0.33	63.48 ± 0.22	66.87 ± 0.44
60%	Bimodal	**65.39** ± 0.48	50.33 ± 1.05	29.12 ± 0.77	18.79 ± 0.82	10.32 ± 0.63	63.03 ± 0.66	34.09 ± 0.15	64.88 ± 0.38	41.65 ± 0.79	63.84 ± 0.53	64.29 ± 0.23
60%	Flip	**69.60** ± 0.42	66.58 ± 0.43	41.37 ± 0.66	40.18 ± 1.34	37.27 ± 0.75	67.42 ± 0.38	40.68 ± 0.36	65.09 ± 0.86	42.77 ± 0.14	65.21 ± 0.66	67.38 ± 0.44

Clothing1M

Noise	GSL	GLC	SCL	FORWARD	BOOTSTRAP	SGFORWARD	CO-TEACHING+	DIVIDEMIX	GFORWARD	TMATRIX	STMATRIX
Real World	**74.86**	73.91	70.78	70.04	67.87	73.96	70.33	74.29	70.95	72.04	72.41

worse in 60% bimodal noise because this is a more challenging pattern and has no access to the corruption matrix. Also, we achieve higher accuracy than DIVIDEMIX which is one of the accurate state-of-the-art. Moreover, our method can still obtain 11 to 30% higher test accuracy than CO-TEACHING+ that uses two deep networks concurrently.

CIFAR-100 is more challenging than CIFAR-10 due to the larger number of classes and results are summarized over three runs in Table 3.1. GSL achieves the highest accuracy except for flip noise with 30% rate, and same as CIFAR-10, STMATRIX, DIVIDEMIX, and SGFORWARD are the closest competitors. Although for flip noise with 30% rate SGFORWARD performs better than GSL, the improvement of GSL is more significant than SGFORWARD compared to the CIFAR-10 dataset. The largest difference (more than 2%) in accuracy between the GSL and SGFORWARD methods is with bimodal noise, and between GSL and STMATRIX is with flip noise. In case of 60% symmetric noise, GSL achieves the accuracy of 68%, whereas GLC and SCL trail far behind. Moreover, given the difficulty of training a robust classifier for CIFAR-100 with 60% label noise, it is worth mentioning that SCL can achieve similar performance as GLC that is given 10% of trusted data in case of 30% symmetric noise. This also indicates the effectiveness of symmetric cross entropy in learning hard classes even without trusted data. However, when facing extremely noisy labels and patterns, the small amount of trusted data can greatly improve the robustness of the classifier but not necessarily the symmetric cross entropy.

Seen from the high accuracy compared to GLC, SCL, GFORWARD and SGFORWARD, GSL effectively uses the trusted data to correct symmetric cross entropy loss and improve the learning on the hard classes. GSL performs slightly better with symmetric noise than with bimodal and flip noise that is more challenging for CIFAR-10. In the CIFAR-100, GSL works better on the asymmetric noise rather than symmetric.

For Clothing1M dataset, as shown in Table 3.1, GSL obtains the highest test accuracy compared to other methods. Same as CIFAR-10 and CIFAR-100, DIVIDEMIX achieves a relatively good performance. The difference between GSL and SCL comes from the effectiveness of corruption matrix that makes the regular cross entropy robust.

3.7.2. TEXT ANALYSIS

We evaluate GSL on text datasets of Twitter and SST, against resilient networks that leverage corruption matrix, namely GLC and FORWARD. Both GSL and GLC use the trusted data for estimating the corruption matrix, wheres the original FORWARD [70] relies solely on the noisy data.

We extensively evaluate GSL, GLC GFORWARD, TMATRIX, SGFORWARD, SCL, FORWARD, BOOTSTRAP, CO-TEACHING+, DIVIDEMIX and STMATRIX on Twitter and SST, with label corruption ranging from 0% to 100%. We also vary the percentage of trusted data among 1% and 5%. We summarize the average accuracy across 11 noise rates and three runs in Table 3.2.

Table 3.2: The average test accuracy (%) across the entire range of noise rates [0, 100] of GSL compared to the baselines on the Twitter and SST datasets for the noise patterns symmetric, bimodal and flip (Best accuracy in bold).

	Noise Pattern	Percent Trusted	GSL	GLC	GFORWARD	TMATRIX	SGFORWARD	STMATRIX	FORWARD	SCL	BOOTSTRAP	CO-TEACHING+	DIVIDEMIX
Twitter	Sym.	1	**79.30** ± 0.11	65.41 ± 0.90	53.21 ± 0.17	76.61 ± 0.37	78.39 ± 0.18	78.24 ± 0.24	52.25 ± 0.25	62.77 ± 0.63	50.59 ± 0.12	65.79 ± 0.22	76.95 ± 0.76
	Sym.	5	**81.94** ± 0.29	77.20 ± 0.17	59.61 ± 0.44	79.63 ± 0.33	81.20 ± 0.16	81.33 ± 0.17	59.07 ± 0.54	63.53 ± 0.31	52.04 ± 0.30	67.67 ± 0.34	78.73 ± 0.42
	Bimodal	1	75.92 ± 0.29	67.15 ± 0.28	52.53 ± 0.19	**77.64** ± 0.56	75.49 ± 0.34	76.73 ± 0.39	50.13 ± 0.40	62.31 ± 0.24	49.11 ± 0.25	63.89 ± 0.54	75.06 ± 0.27
	Bimodal	5	**84.35** ± 0.39	78.45 ± 0.37	60.63 ± 0.28	80.58 ± 0.32	80.41 ± 0.88	80.73 ± 0.19	54.64 ± 0.72	66.87 ± 0.31	53.87 ± 0.38	68.94 ± 0.32	81.95 ± 0.62
	Flip	1	82.75 ± 0.63	83.13 ± 0.24	39.52 ± 0.22	**86.13** ± 0.31	73.89 ± 0.41	73.28 ± 0.16	48.21 ± 0.31	60.63 ± 0.15	48.87 ± 0.28	62.66 ± 0.29	84.66 ± 0.26
	Flip	5	84.75 ± 0.31	85.49 ± 0.38	48.42 ± 0.61	**87.04** ± 0.21	79.48 ± 0.36	80.20 ± 0.18	53.87 ± 0.65	64.74 ± 0.39	51.88 ± 0.18	66.19 ± 0.29	86.29 ± 0.42
SST	Sym.	0.1	**75.18** ± 0.55	73.47 ± 0.28	72.15 ± 0.29	73.55 ± 0.29	72.22 ± 0.09	73.66 ± 0.49	70.13 ± 0.31	71.36 ± 0.26	70.03 ± 0.42	71.84 ± 0.33	74.07 ± 0.17
	Sym.	1	**75.96** ± 0.46	72.62 ± 0.28	73.47 ± 0.25	75.48 ± 0.36	72.93 ± 0.19	75.42 ± 0.34	72.52 ± 0.27	72.86 ± 0.39	71.31 ± 0.22	72.13 ± 0.29	74.93 ± 0.29
	Bimodal	0.1	**74.97** ± 0.24	74.70 ± 0.31	72.75 ± 0.14	74.19 ± 0.44	72.63 ± 0.51	74.16 ± 0.36	70.02 ± 0.18	71.22 ± 0.16	70.21 ± 0.21	72.25 ± 0.15	73.93 ± 0.31
	Bimodal	1	**74.88** ± 0.41	74.53 ± 0.32	72.10 ± 0.13	74.34 ± 0.29	71.79 ± 0.42	73.60 ± 0.30	72.67 ± 0.29	72.13 ± 0.41	71.38 ± 0.19	72.93 ± 0.31	74.01 ± 0.16
	Flip	0.1	**75.38** ± 0.29	74.07 ± 0.25	49.40 ± 0.51	74.83 ± 0.22	49.50 ± 0.34	74.81 ± 0.16	70.79 ± 0.54	72.52 ± 0.12	70.79 ± 0.41	72.14 ± 0.27	74.54 ± 0.30
	Flip	1	**76.59** ± 0.14	74.51 ± 0.32	50.21 ± 0.43	76.33 ± 0.13	49.81 ± 0.23	75.49 ± 0.43	73.04 ± 0.17	73.76 ± 0.51	71.78 ± 0.14	72.83 ± 0.10	74.99 ± 0.25

TWITTER

As shown in Table 3.2, GSL consistently achieves the highest average accuracy in most cases. Compared to GLC, GSL has significant higher accuracy for Twitter corrupted with symmetric and bimodal noises, but the difference diminishes with increasing amounts of trusted data. When the percent of trusted data is low, say, 1%, GLC is unable to estimate the corruption matrix accurately nor to correct the loss, seen by the difference between GLC and TMATRIX.

SST

Here, the classification involves only two classes and turns out to be less challenging than the Twitter case. The results in Table 3.2 show that the difference among the different comparative approaches considered is smaller than for Twitter. For instance, though GSL consistently achieves the best average accuracy in almost all cases, the difference between GSL and GLC is around 1-3%. Again, we see that GSL visibly outperforms GLC on low amounts of trusted data because of using cross entropy and the difference among them becomes limited. We note that TMATRIX and GFORWARD collapse under Flip noise. We plot an extension of Table 3.2 for analysis on text datasets for varying noise rates in the next part.

TEXT ANALYSIS ON TWITTER AND SST DATASETS WITH VARYING NOISE RATES

Figure 3.4 and Figure 3.5 show how the accuracy changes with respect to different noise rates on the Twitter and SST datasets. The noise pattern is symmetric. GSL and GLC are provided with one percent trusted data. In contrast, GSL can effectively use the

symmetric cross entropy to overcome the limitation of low trusted data. This also explains why STMATRIX always trails closely behind GSL by using the true corruption matrix and symmetric entropy loss. One may further improve STMATRIX by using the optimal weights of A and B according to the true corruption matrix, instead of estimated corruption matrix of GSL. The Twitter dataset highlights well the differences (see Figure 3.4). The SST dataset is an easier problem with only two classes and all methods are able to perform equally well (see Figure 3.5).

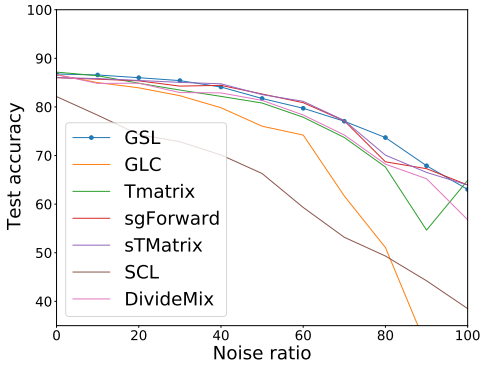


Figure 3.4: Test accuracy (%) of GSL compared to six baselines on the Twitter dataset with noise rates [0, 100] using 1% trusted Data.

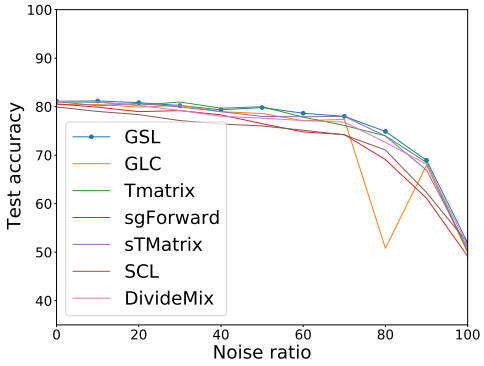


Figure 3.5: Test accuracy (%) of GSL compared to six baselines on SST dataset with noise rates [0, 100] using 1% trusted data.

3.8. DISCUSSION

Here we present the extensive results of our empirical evaluation on training with corrected labels for the vision datasets in Table 3.3 and Table 3.4, respectively. This com-

plements the results presented in §3. We compare the impact of correcting labels only on the cross entropy term (*ce only*), only on the reverse cross entropy term (*rce only*), or *both*. Table 3.3 shows the achieved accuracy for CIFAR-100, under two noise rates, 30% and 60%, three different noise types, symmetric, bimodal and flip, and three fractions of trusted data, 5%, 10% and 15%. For each noise scenario the best case is highlighted in bold.

Table 3.3: Accuracy (%) of GSL with different label correction into the loss under noise rates of 30% and 60% for three noise patterns (bimodal, symmetric, and flip) of varying gold fractions (5-15%) on CIFAR-100.

Noise rate = 30%									
Label correction	Bimodal			Symmetric			Flip		
	5%	10%	15%	5%	10%	15%	5%	10%	15%
ce only	**71.66**	**73.90**	**75.20**	**71.69**	**74.24**	**75.11**	**74.78**	**75.46**	**77.01**
rce only	25.90	61.44	67.15	26.15	61.19	67.14	23.75	59.68	65.33
both	23.37	57.52	64.30	23.58	57.36	63.76	19.74	54.50	61.20
Noise rate = 60%									
Label correction	Bimodal			Symmetric			Flip		
	5%	10%	15%	5%	10%	15%	5%	10%	15%
ce only	**58.42**	66.96	**69.41**	**55.22**	**66.87**	**69.46**	**65.20**	**68.33**	**70.41**
rce only	54.35	**67.24**	69.41	24.90	58.72	64.39	14.43	46.09	68.75
both	24.35	55.18	61.19	25.62	55.16	61.46	12.46	36.76	50.83

ce only achieves the highest accuracy in all cases except one. Under 60% bimodal noise on CIFAR-100 with 10% trusted data *rce only* is slightly better by 0.28 percent points. More in general, *rce only* typically performs second best and *both* achieves the worst accuracy. Focusing on *ce only* over the other two, the gain tends to increase with the difficulty of the noise scenarios, i.e. with higher number of classes, higher noise rates and less trusted data. *ce only* outperforms the other two by up to 51.03 percent points for CIFAR-100. Same as CIFAR-100 results, *ce only* outperforms the other two by up to 11.74 percent points for CIFAR-10 in Table 3.4.

Table 3.4: Accuracy (%) of GSL with different label correction into the loss under noise rates of 30% and 60% for three noise patterns (bimodal, symmetric, and flip) of varying gold fractions (5-15%) on CIFAR-10.

Noise rate = 30%									
Label correction	Bimodal			Symmetric			Flip		
	5%	10%	15%	5%	10%	15%	5%	10%	15%
ce only	**90.06**	**91.50**	**92.53**	**90.17**	**91.70**	**92.50**	**89.96**	**91.27**	**92.58**
rce only	84.67	88.46	89.30	84.82	88.11	89.98	81.70	86.79	88.40
both	83.25	87.45	89.30	83.43	87.70	89.46	78.00	85.42	88.21
Noise rate = 60%									
Label correction	Bimodal			Symmetric			Flip		
	5%	10%	15%	5%	10%	15%	5%	10%	15%
ce only	**83.06**	**88.43**	**90.52**	**85.79**	**89.19**	**90.19**	**80.03**	**82.64**	**85.80**
rce only	81.73	86.86	89.07	81.54	86.63	88.98	68.29	80.22	84.35
both	79.83	85.79	88.63	80.27	86.22	88.53	63.63	82.19	83.18

3.9. CONCLUSION

To enhance the robustness of deep models against by label noise, we propose GSL that features on correcting the symmetric cross entropy loss by the noise corruption matrix.

GSL uses a small fraction of trusted data to accurately estimate the corruption matrix, and further determine the weights applied on regular and reverse cross entropy. GSL learns deep networks from trusted samples through regular cross entropy and from untrusted noisy samples through golden symmetric cross entropy. We prove that the cross entropy corrected by the corruption matrix is noise robust. To adapt to noise patterns of dataset, we heuristically set the weights of golden symmetric loss based on the corruption matrix. We extensively evaluate GSL on vision and text analysis under diversified noise rates and patterns. Evaluation results show that GSL can achieve a remarkable accuracy improvement, i.e., from 2 to 18% on CIFAR benchmarks and real world noisy data, compared to methods that either correct loss or leverage symmetric cross entropy.

Despite the effectiveness of this approach, one limitation arises in the weighting mechanism for cross-entropy and reverse cross-entropy terms. We employ the Jain fairness index to assign these weights based on the noise pattern, and although this yields strong results with GSL outperforming competitors, other potential functions could be studied to optimize this weighting process. Further research could focus on finding an optimal function for this mechanism and developing mathematical proofs to better understand the properties of these weight assignments for achieving optimal results.

4

LEARNING FROM NOISY LABELS WITH ORACLE SUPERVISION

Noisy labeled data is more the norm than the exception for self-generated content continuously published on the web and social media by non-experts. Active querying experts are conventionally adopted to provide labels for the informative samples which don't have labels, instead of possibly incorrect labels. The new challenge that arises here is how to discern the informative and noisy labels which benefit from expert cleaning. In this paper, we aim to leverage the stringent oracle budget to robustly maximize learning accuracy. We propose a noise-aware active learning framework, QActor, and a novel measure CENT, *which considers both cross-entropy and entropy to select informative and noisy labels for an expert cleansing. QActor iteratively cleans samples via quality models and actively querying an expert on those noisy yet informative samples. To adapt to learning capacity per iteration, QActor dynamically adjusts the query limit according to the learning loss for each learning iteration. We extensively evaluate different image datasets with noise label ratios ranging between 30% and 60%. Our results show that QActor can nearly match the optimal accuracy achieved using only clean data at the cost of only an additional 10% of ground truth data from the oracle.*

4.1. INTRODUCTION

We are in the era of big data, which are continuously generated on different web platforms, e.g., social media, and disseminated via search engines often in a casual and unstructured way. Consequently, such a big data analysis suffers from diversified quality issues, e.g., images tagged with incorrect labels, so-called noisy labels. According to [134], noisy data costs the US industry more than $3 trillion per year to cleanse or to mitigate the impact of derived incorrect analyses. While the learning models conveniently leverage such a free source of data, its quality greatly undermines the learning efficiencies and their associate utilities [135]. For instance in [44], using an image classifier trained from data with highly noisy labels can significantly degrade classification accuracy and hinder its applicability on different domains.

Noisy label issue has been a long-standing challenge [93], from standard machine learning (ML) models to deep neural networks (DNN), whose large learning capacities can have detrimental memorization effects on dirty labels [44]. The central theme here is to filter out the suspicious data which might have corrupted labels via quality estimates. The drawback of filtering approaches is the risk of dropping informative data points which can be influential for the underlying learning models. It might be worthwhile to actively cleanse such data due to its high potential in improving the learning tasks, even at a certain expense.

Active learning (AL) techniques [136] are designed to query extra information from an oracle for the data whose (true) labels are not readily available. Such an oracle is assumed to know the ground truth, but at high costs, e.g. a human expert. Hence, only the informative/uncertain data is queried within a certain query budget. The efficacy of active learning relies on uncertainty measurements of learning tasks, e.g., class probability [137], entropy value [138] or posterior predictive densities [139].

As the majority of active learning approaches focus on the unsupervised scenarios and constant budget, it is not clear how the active query approach can be adopted when encountering noisy data - a kind of noisy supervision.In the noisy labeled data scenario different from traditional active learning, the sample selection process ought to identify both informative and noisy samples. Using a limited query budget to cleanse clean samples leads to the waste of queries. Such a noisy supervision calls for a new measure that asks for the expert query on highly informative and noisy samples and retain the clean samples.

In this chapter, we focus on a challenging multi-class learning problem whose labels are extremely noisy. Our objective is to enhance the noise-resiliency of the underlying classifier by selectively learning from good data as well as noisy labels that are critical for training the classifier. In order to turn the noisy labels into a learning advantage, we resort to the oracle for recovering their label ground truth under a given query budget. Ultimately, we aim to optimize classification accuracy with a minimum number of oracle queries.

To such an end, we design an active learning framework termed Quality-driven Active Learning (QActor), which marries quality models with active learning. Upon receiving new data instances, QActor first filters it via the *quality model* into “clean” and “noisy” categories. Second, we propose a novel measure noise-aware informative measure, *CENT*, combining cross-entropy and regular entropy, to identify erroneous and informative

samples and send them for oracle cleansing. Another unique feature of QActor is that the overall query budget is fixed but the number of queries per batch is dynamically adjusted based on the current training loss value. Our results show that in the presence of very large label noise, i.e., up to 60% corrupted labels, QActor can achieve remarkable accuracy, i.e., almost match the optimal accuracy obtained excluding all noisy labels, at the cost of just a small fraction of oracle information, i.e., up to 10% oracle queried labels. Moreover, compared to state of the art on noise resilient DNN, QActor achieves higher accuracy by 15% and 8% for CIFAR-10 and CIFAR-100, respectively.

Our contributions are threefold. We design a novel and efficient learning framework, termed QActor, whose core combines a quality model with active learning. Secondly, we propose a novel noise-aware informative measure, *CENT*. Thirdly we propose a dynamic learning strategy that can adapt to the dynamic nature of iterative active learning and achieve better results than the static one. To the best of our knowledge, this is the first study on the dynamic allocation of an active learning budget.

4.2. RELATED WORK

Human error and careless annotators result in unreliable datasets with mistakes in labels available in public domains [140], [141]. Adversaries are another source of label noise attacking the performance of (deep) learning systems [142]. Corresponding to the contribution of QActor, we categorize the related work of learning from noisy supervision into two categories: (i) noise resilient models that filter the noise or alter the loss function without ground truth, (ii) active learning from the oracle supervision.

Noise resilient model. Learning with noise in the labels with no quality filtering shows the effect of noise in the degradation of the classification accuracy of deep neural networks [82]. As mentioned in [44], the accuracy of using trained AlexNet to classify CIFAR-10 images with random label assignment drops from 77% to 10% due to network memorization of the noisy samples. Co-teaching [61] trains two neural networks simultaneously on two different data and exchanges the model information trained by the data causing the lowest loss. RAD and its extensions [143], [144] cascadically train two models to find out "clean" data. They also use the help of external experts to verify the labels and optimize the cost with a limited budget. On the other hand, the study in [70], Forward, assumes there is a noise transition matrix to cleanse the noisy labels for a deep neural network. Furthermore, D2L [34] uses the Local Intrinsic Dimension (LID) as a measure to filter the noisy labeled instances during training. Re-weighting samples based on their similarity to a clean set to increase the robustness against label noise has been studied in [145]. Noise confusion matrix estimation is another method to improve learning on corrupted data [117].

Active learning. Active learning has been employed at a growing rate in recent studies with deep networks due to the expenses of large dataset collection. Various studies focus on the identification of informative data instances. The studies in [146], [147] consider geometrical approaches to select the data instances, i.e. the core-set, that is the representative of the data space. Meanwhile, [148] uses deep Bayesian neural networks with monte-carlo dropout to identify the most uncertain samples for labeling. Following the same framework, [149] argues the effectiveness of batch labeling via a expert in a deep neural network. Furthermore, [150] uses the probability output of the convolutional

neural network to label the instances based on discrete entropy and best-vs-second-best. A relevant line of research in active learning is to deal with noisy oracles which can not accurately provide the labels. For instance, [151] benefits from the disagreement of a committee of models with the given label. In [152] the goal of the active learner is to identify the informative noisy instances and ask the oracle their true label. [153] assumes that a strong labeler is sided by a weak labeler, termed quality model which is cheaper than an expert, and only queries the oracle when the two labelers disagree. However, these studies fail to consider the noisy data characteristic in their query selection strategies.

4.3. QUALITY-AWARE ACTIVE LEARNING QACTOR

4.3.1. PROBLEM STATEMENT

We consider multi-class classification problems that map data inputs $\mathbf{x}$ of K features into labels y of C classes, $\mathbf{x} \in \mathbf{X}^{N\times K}$ into $y \in \mathbf{C} = \{1,\dots,C\}$. We assume that given dataset $\mathbf{D} = \{(\mathbf{x}_j, \hat{y}_j), j = 1,...,N\}$ has noisy labels, i.e. the label of a fraction of the data is altered from its true label. A small set of initial data instances with clean labels used as the initial seed is given, together with a testing set for evaluation. A clean data instance refers to a sample whose given label is properly annotated, without any alteration. In this paper, η shows the noise rate which indicates the ratio of the noisy label data to the entire dataset size.

The goal is to identify the noisy labeled data samples and clean their labels by an expert labeler, i.e the oracle[1], within a limited budget. Since expert labeling is expensive, we aim to identify informative noisy labeled data and it to send the oracle for relabeling. In the end, the evaluation is done by training a classifier on the filtered and cleansed data. In the following section, we demonstrate the procedure of the introduced method QActor.

4.3.2. ARCHITECTURE AND METHODOLOGY OF QACTOR

Figure 4.1 depicts the architecture. The main components are the following: 1)quality model $\mathfrak{Q} : \mathbf{x} \rightarrow \tilde{y} \in \mathbf{C}$ where $\tilde{y}$ is the predicted label by the quality model, 2) the label comparator which discerns noisy from clean labels, 3) the active leaner which determines which and how many data instances to send to the oracle, and 4) the classifier $\mathfrak{C} : \tilde{\mathbf{x}} \rightarrow \tilde{y} \in \mathbf{C}$. $\tilde{X} = \{\tilde{\mathbf{x}}\}$ is a subset of X defined below.

We use Deep Neural Networks (DNN) as the classifier since they have shown extremely promising results in classifying complex image datasets [142]. Due to the high training costs of deep neural networks, to reduce the computational burden, instead of having two different models for $\mathfrak{Q}$ and $\mathfrak{C}$, we leverage the time difference between data arrivals to use the previously trained $\mathfrak{C}$ as $\mathfrak{Q}$ for the following time period, i.e., $\mathfrak{Q}(t) = \mathfrak{C}(t-1)$. This optimization allows us to train only one model per time period.

At each iteration t the data x is first sent to the quality model from the previous time instance $\mathfrak{Q}(t-1)$ which predicts their labels $\tilde{y}$. If the predicted labels are the same as the given ones, i.e. $\tilde{y} = \hat{y}$, the label comparator marks them as clean, denoted as $\mathbf{x}^c(t) \in X^c(t)$; otherwise as suspicious, $\mathbf{x}^s(t) \in X^s(t)$. After this filtering step, the goal is to efficiently clean the suspicious data. Since single sample relabeling is very inefficient for the training process of deep neural networks, we relabel a batch of informative samples at each

[1] In this paper, we interchangeably use terms of expert and oracle.

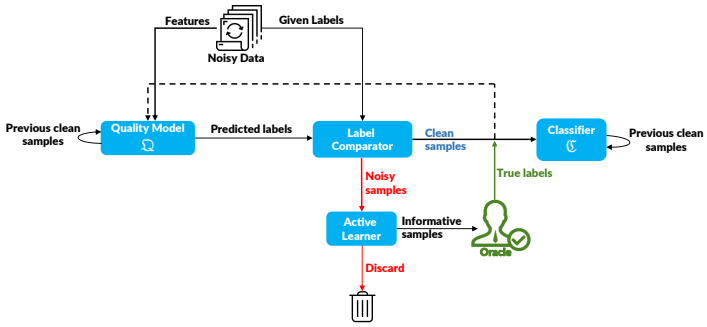


Figure 4.1: Overview of QActor: workflow of quality model, active learner, and classifier.

4

iteration as [149]. To this end, the suspicious data samples are sent to the active learner to rank them based on their informativeness and select a batch of $o(t)$ data instances $X^o(t)$ to send to the oracle to query their true label. Note that $o(t)$ is limited by the given available budget B, i.e., $\sum_t o(t) \leq B$. Since $o(t)$ is typically much smaller than the number of noisy data instances, i.e. $o(t) \ll |X^s(t)|$, instances are ranked and selected based on an uncertainty metric. Highly uncertain samples indicate high informativeness and thus we argue that re-labeling these samples would increase the performance of the classifier.

The clean and relabeled instances are denoted by $\tilde{X}(t) = (\cup_{\tau=1}^{t} X^o(\tau)) \cup X^c(t)$ and used to re-train $\mathfrak{C}(t)$ and $\mathfrak{Q}(t)$. We repeat this procedure again for a few iterations until the budget is exceeded or the performance reaches to a desired level. To avoid pitfalls in learning we monitor the accuracy on a small hold-out of the initial set. If performance drops by more than a we roll back the model before processing the next data batch.

In the following section, we explain our proposed noise-aware informativeness measure that identifies the useful samples to be relabeled by the oracle.

4.3.3. NOISE-AWARE INFORMATIVE MEASURE: *CENT*

Since relabeling all the data in the suspicious set is an expensive and time-consuming task, we aim to identify the most informative and useful samples to relabel by the oracle. We argue that in noisy labeled data settings different from traditional active learning where no label information is available, relabeling is more effective if the budget is spent on informative and noisy data. To overcome this issue, we introduce a novel noise-aware active learning measure, *CENT*, which consists of two parts, informative data identification, and clean/noisy separation. *CENT* queries the oracle to relabel an informative noisy set, and furthermore, identifies the informative clean set within the suspicious set and keeps their own label and then adds to clean set $X^c(t)$. The motivation behind this strategy is to leverage the clean data that is mistakenly categorized as noisy and is then discarded by the traditional AL methods for their lack of informativeness.

To measure the informativeness of the data, first, we calculate the entropy of the

suspicious set:

$$\mathscr{L}_E(\mathbf{x}_i^s) = -\sum_{c=1}^{C} p(y=c|\mathbf{x}_i^s)\log p(y=c|\mathbf{x}_i^s) \tag{4.1}$$

where p is the neural network's softmax prediction output for each class label c: $p(c|\mathbf{x}) = \frac{e^{z_c}}{\sum_{j=1}^{C} e^{z_j}}$ where z_j are the logits.

Entropy is an information-theoretic measure and the higher this value, the higher in information in the data sample is. We pick $2 \cdot o(t)$ samples with the highest $\mathscr{L}_E$ value and put them in the informative set $X^I(t)$. Although entropy is a popular informativeness measure in active learning literature where the data is unlabeled, in noisy labeled data problems it doesn't necessarily identify the noisy informative data. Therefore, we employ another metric, cross-entropy, which can distinguish between noisy and clean data:

$$\mathscr{L}_{CE}(\mathbf{x}_i^I, y_i) = -\sum_{c=1}^{C} q(c|\mathbf{x}_i^I)\log p(y=c|\mathbf{x}_i^I) \tag{4.2}$$

where $q(c|\mathbf{x}_i)$ denotes the given label probability distribution over the C class labels where $q(c|\mathbf{x}_i) = 1$ for c equal to the given class $\hat{y}_i$ and $q(c|\mathbf{x}_i) = 0$ for all $c \neq \hat{y}_i$. We show in Theorem 4.3.1 that the higher values of CE are the indication of noisy labeled data. Therefore, we pick the $o(t)$ data with the highest CE values among $X^I(t)$ as $X^o(t)$ and send them to the oracle to relabel. Moreover, since smaller values of CE represent the clean data among the informative set, we pick the other half of $X^I(t)$ that have the lowest CE values and call them the *semi-clean* data $X^{semi-c}(t)$ to add to the clean set, keeping their own labels. Algorithm 4 shows the overview of our proposed method.

Theorem 4.3.1 *With uniform label noise with the rate η and the accuracy A of the classifier in a C class classification task, the average of $\mathscr{L}_{CE}(\mathbf{x}, \hat{y})$ for noisy samples is higher than $\mathscr{L}_{CE}(\mathbf{x}, \hat{y})$ for clean samples with the given label $\hat{y}$, if $\frac{(1-A)}{A+(1-A)\gamma}\frac{\eta}{(1-\eta)} < C-1$ where $\gamma << 1$ is a positive number.*

Proof: Let $\hat{y}$ and $\tilde{y}$ be the given (noisy) and the predicted label for the true label y. We show the clean and noisy set as $\Omega = \{(\mathbf{x}_j, \hat{y}_j)|\hat{y}_j = y_j\}$ with $|\Omega|$ samples and $\Phi = \{(\mathbf{x}_i, \hat{y}_i)|\hat{y}_i \neq y_i\}$ with $|\Phi|$ samples respectively. We argue that the average $\mathscr{L}_{CE}$ value for the noisy samples is higher than the clean samples:

$$\mathbb{E}_{\mathbf{x}\sim\Omega(\mathbf{x},\hat{y})}\mathscr{L}_{CE}(\mathbf{x}, \hat{y}) < \mathbb{E}_{\mathbf{x}\sim\Phi(\mathbf{x},\hat{y})}\mathscr{L}_{CE}(\mathbf{x}, \hat{y}) \tag{4.3}$$

According to equation 4.2, since q is a one-hot vector of the labels, $\mathscr{L}_{CE}(\mathbf{x}_i, \hat{y}_i) = -\log p(\hat{y}_i|\mathbf{x}_i)$. By removing logarithm from both sides, we have:

$$\mathbb{E}_{\mathbf{x}\sim\Phi(\mathbf{x},\hat{y})}\{p(\hat{y}|\mathbf{x})\} < \mathbb{E}_{\mathbf{x}\sim\Omega(\mathbf{x},\hat{y})}\{p(\hat{y}|\mathbf{x})\} \tag{4.4}$$

where p is the neural network's softmax prediction output. Consider Figure 4.2 which categorizes the clean and noisy data based on the noise ratio and the classifier's accuracy. With the uniform noise patter, the probability of the noisy label being equal to the predicted label is $\frac{1}{C-1}$, and $\frac{C-2}{C-1}$ otherwise. Therefore, the inequality above is equivalent to the following:

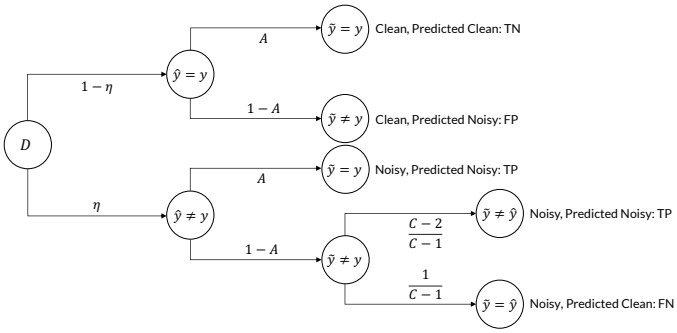


Figure 4.2: Categorization of the data D based on the true, given and predicted labels. TP and FP indicate true and false positives and, TN and FN indicate true and false negatives respectively, where noisy and clean data are considered positive and negative respectively.

4

$$
\begin{aligned}
&A\eta\mathbb{E}_{\mathbf{x}\sim\Phi(\mathbf{x},\hat{y})}\{p(\hat{y}|\mathbf{x},\tilde{y}=y)\}+\frac{(1-A)\eta}{C-1}\mathbb{E}_{\mathbf{x}\sim\Phi(\mathbf{x},\hat{y})}\{p_{max}(\mathbf{x})\}+\\
&\frac{(1-A)(C-2)\eta}{C-1}\mathbb{E}_{\mathbf{x}\sim\Phi(\mathbf{x},\hat{y})}\{p(\hat{y}|\mathbf{x},y\neq\tilde{y}\neq\hat{y})\}<\\
&A(1-\eta)\mathbb{E}_{\mathbf{x}\sim\Omega(\mathbf{x},\hat{y})}\{p_{max}(\mathbf{x})\}+(1-A)(1-\eta)\mathbb{E}_{\mathbf{x}\sim\Omega(\mathbf{x},\hat{y})}\{p(\hat{y}|\mathbf{x},\tilde{y}\neq\hat{y}=y)\}
\end{aligned}
\tag{4.5}
$$

where p_{max} for the data $\mathbf{x}_j$ is the maximum value of prediction vector for $\mathbf{x}_j$, associated with $\tilde{y}$. The inequality will hold by omitting the rest of the p values in the left hand side which are significantly smaller than p_{max}. Moreover, on the right had side, we can use $\mathbb{E}_{\mathbf{x}\sim\Omega(\mathbf{x},\hat{y})}\{p(\hat{y}|\mathbf{x},\tilde{y}\neq\hat{y}=y)\}=\gamma\mathbb{E}_{\mathbf{x}\sim\Omega(\mathbf{x},\hat{y})}\{p_{max}(\mathbf{x})\}$ where $\gamma<<1$. Therefore we will have:

$$
\frac{(1-A)\eta}{C-1}\mathbb{E}_{\mathbf{x}\sim\Phi(\mathbf{x},\hat{y})}\{p_{max}(\mathbf{x})\}<(1-\eta)(A+(1-A)\gamma)\mathbb{E}_{\mathbf{x}\sim\Omega(\mathbf{x},\hat{y})}\{p_{max}(\mathbf{x})\} \tag{4.6}
$$

Since $\mathbb{E}_{\mathbf{x}\sim\Phi(\mathbf{x},\hat{y})}\{p_{max}(\mathbf{x})\}\leq\mathbb{E}_{\mathbf{x}\sim\Omega(\mathbf{x},\hat{y})}\{p_{max}(\mathbf{x})\}$, therefore, the inequality holds when:

$$
\frac{(1-A)}{A+(1-A)\gamma}\frac{\eta}{(1-\eta)}<C-1 \tag{4.7}
$$

□

4.3.4. ACTIVE LEARNER QUERY POLICIES

The aforementioned uncertainty measures are used by the active learner in combination with two different policies on how to deplete the query budget over time:

Static policy. The active learner asks a constant number $o(t)=M,\forall t$ of queries at every iteration of learning. Essentially, for each iteration, the active learner queries the most uncertain M data instances that are considered noisy by the quality model.

Dynamic policy. The active learner dynamically adjusts $o(t)$ based on the value of the loss function of the quality model. The rationale behind this is to increase the number of queries when the quality model has a low learning capacity, reflected by high loss function values, and to decrease the number of queries when the loss function converges to lower values. Specifically, we propose to adjust $o(t)$ as following:

$$
o(t)=o(t-1)(1-\frac{L^{\Omega}(t-2)-L^{\Omega}(t-1)}{L^{\Omega}(t-1)}) \tag{4.8}
$$

Algorithm 4: Quality Driven Active Learning.

Input : Initial Dataset D^I, Noisy Labeled Dataset $D = \{(\mathbf{x}_j, \hat{y}_j)\}$, Budget B, Total Number of Iterations T

Output : Quality model $\mathfrak{Q}$, Classifier $\mathfrak{C}$

Train $\mathfrak{Q}$ and $\mathfrak{C}$ with D^I
while *iteration* $t < T$ **do**
  $\tilde{y}_j :=$ Predicted label by Quality model $\mathfrak{Q}$ for $\mathbf{x}_j$
  $X^c(t) = \{\forall \mathbf{x}_j \in D, \tilde{y}_j = \hat{y}_j\}$
  $X^s(t) = \{\forall \mathbf{x}_j \in D, \tilde{y}_j \neq \hat{y}_j\}$
  $o(t) =$ Query size according to Section 4.3.4
  if *Informativeness Measure is CENT* **then**
    $X^I(t) =$ The first $2o(t)$ high entropy samples selected from $X^s(t)$
    Sort $X^I(t)$ based on their CE value
    $X^o(t) =$ The $o(t)$ samples from $X^I(t)$ with the highest CE
    $X^{semi-c}(t) =$ The $o(t)$ samples from $X^I(t)$ with the lowest CE
  else
    $X^o(t) =$ The $o(t)$ samples with the highest informativeness from $X^s(t)$
  end
  Send $X^o(t)$ to the oracle to relabel
  Train $\mathfrak{Q}$ and $\mathfrak{C}$ with $\tilde{X}(t) = (\cup_{\tau=1}^{t} X^o(\tau)) \cup X^c(t) \cup X^{semi-c}(t)$
end

where

$$L^{\mathfrak{Q}}(t) = \frac{-1}{|\tilde{\mathbf{x}}(t)|} \sum_{x_i \in \tilde{\mathbf{x}}} \sum_{c=1}^{C} p(y=c|\mathbf{x}_i) \log p(y=c|\mathbf{x}_i) \tag{4.9}$$

is the average entropy loss across all training samples used at period t. Since the retraining of the model(s) happens after the oracle querying, we use the loss from the periods $t-1$ and $t-2$. Finally, we note that the number of active queries is capped by the given budget B, i.e. the number of active queries used is $\min(B - \sum_{\tau=1}^{t-1} o(\tau), o(t))$.

Standard active learning studies query one instance at a time and train the model by adding that instance to the training set. Then the learner queries the next instance based on the retrained model and repeats the procedures recursively until all the budget is spent or the desired performance is achieved. However, it is computationally too expensive to retrain the model after each oracle query and repeat for the next one. Therefore, we decide to query $o(t)$ instances per round. This applies to both policies: static and dynamic.

4.4. EXPERIMENTAL SETUP

Here we describe the datasets, the model parameters and the baselines used for comparison.

Datasets. We consider image datasets using the pixel values as inputs. In particular we use the well-known CIFAR-10 and CIFAR-100 datasets [129]. These datasets try to classify colored 32×32-pixel images into ten and hundred classes, respectively. CIFAR-100

is more complex due to both the higher number of classes and lower number of data per class. For both dataset we use 50000 samples for training and the rest 10000 samples for testing.

Label Noise. We inject label noise into the training set by corrupting the label of randomly sampled data instances. We term the sampling probability as noise rate. Corrupted samples are subject to symmetric noise, e.g., the true label is exchanged with a random different label with uniform probability. We consider noise rates of 30% and 60%. Test data is not subject to label noise.

Baselines. To better show the overall effectiveness of our proposed QActor method we compare it two sets of baselines. First we compare against different active query selection baselines:

- **No-Sel**: uses all samples that arrive in the batch to train the classifier without filtering.
- **Q-only**: in this case the quality model filters the suspicious samples but there is no active learner to relabel the informative noisy instances. Therefore the classifier will train only on the clean data instances identified by the quality model.
- **Opt-Sel**: which assumes a perfect quality model able to identify all the true clean and noisy samples and uses all the clean samples for training the classifier without active learning.
- **Entropy (ENT)**: ranks the data based on their entropy value, i.e. $\sum_{c=1}^{C} p(y = c|\mathbf{x}_i^s) \log p(y = c|\mathbf{x}_i^s)$. Entropy is an information theoretic measure and the higher this value, the higher in information in the data sample is.
- **Re-Active**: based on the idea discussed by [152], where a weighted average of entropy and cross-entropy values are use for informative sample selection, i.e. $(1-\alpha)\mathscr{L}_E + \alpha\mathscr{L}_{CE}$.
- **Random**: among the suspicious set, randomly selects $o(t)$ samples in each iteration to be relabeled by the oracle.
- **Semi-Random**: among the suspicious set, randomly selects $2o(t)$ samples in each iteration, one half to be relabeled by the oracle and another half to be added directly to the clean set keeping their own label.

Second we put QActor in the context of other noise-resistant techniques drawn from the related work on learning with noise. For a fair comparison we have used extra clean data also when training the noise resilient baselines, so that they have the same number of initial clean data during their training processes.

- **D2L** [112]: estimates the dimensionality of sub-spaces during training to adapt the loss function. This approach allows for a more tailored loss landscape, which can enhance model performance in the presence of noisy labels.
- **Forward** [70]: corrects the loss function based on the noise transition matrix. This method effectively maps the probabilities of true labels to observed labels, providing a more accurate representation of the training signal.
- **Bootstrap** [154]: using convex combination of the given and predicted labels for training. This strategy helps in smoothing the label noise by leveraging the model's predictions as a secondary source of information.
- **Co-teaching** [61]: exchanges mini-batches between two networks trained in parallel. This technique allows each network to learn from the other, thereby reducing the

impact of noisy labels by emphasizing the most confident predictions.
- **Re-weighting** [145]: Trains the neural network with a weighted loss function per sample, where the weights are learned based on the similarity of the data to a clean set. This adaptive weighting mechanism enables the model to focus more on high-quality examples while down-weighting the influence of noisy labels.

4.4.1. MODEL PARAMETERS

As QActor classifier for CIFAR-10 and CIFAR-100 we use the Convolutional Neural Network (CNN) architectures defined in [112] with ReLU and softmax activation functions as image classifier and cross-entropy as loss function. We train the models by using stochastic gradient descent with momentum 0.9, learning rate 0.01, and weight decay 10^{-4}. QActor and all baselines are implemented using Keras v2.2.4 and Tensorflow v1.12, except Co-teaching and Reweight. They are based on PyTorch v1.1.0, and we reproduce the same CNN structure in PyTorch as we use for other baselines implemented with Keras.

With CIFAR-10 QActor is trained initially with clean 1000 instances and 40 epochs. We inject noise for the remaining 49000 instances with 30% (60%) rate. Under static policy, in each iteration we query $o(t) = \{100, 300\}$ samples actively from the oracle and retrain the model for 10 epochs for 50 iterations which adds up to 540 epochs overall. Similarly, the dynamic policy uses $B = 5000$ which is equal to the budget used in the static policy with $o(t) = 100$. At the end of each batch, we test the model with the test set of 10000 instances. Rollback uses $a = 20\%$. For CIFAR-100 we increase the initial set size to size to 5000 and 100 epochs per batch to cope with the higher complexity. We also lower the total training iterations for CIFAR-100 to 30, but the epochs per iteration and $o(t)$ stay the same, which would be 400 epochs overall. For fair comparison, baselines are also trained under the same initial set and the same CNN structure. Therefore, we change *Re-weighting* neural network from Wide-ResNet to our own network structure. All baselines use the same parameters as from their papers except for *D2L*. Here we reduce the dimensionality estimation interval to 40 and 10 for the initial and subsequent batches, respectively. This keeps roughly the original ratio against the overall training period. We repeat each experiment for 3 times and for each experiment we report the average accuracy of the last 5 iterations. It is worth mentioning that for hyperparameter tuning, we employed grid search to optimize the model performance.

4.5. RESULTS

Here we present the accuracy achieved by QActor on the CIFAR-10 and CIFAR-100 datasets. We first compare QActor against six noise-resistant models (without using active learning strategies) from four state-of-the-art related papers, followed by the analysis over the uncertainty metrics. Finally, we reveal our model sensitivity analysis and eventually show the effect of the dynamic policy.

4.5.1. NOISE-RESISTANT MODELS

We compared our proposed QActor with the measure *CENT*, with the noise-resistant described in Section 4.4. Table 4.1 summarizes the results for different noise-resistant prior arts and QActor. For a fair comparison, we use the same initial clean set and neural

Table 4.1: Accuracy (%) of different noise resilient networks under 60% label noise.

Methods	Baselines						Our		
	D2L	Forward	Co-teaching	Bootstrap soft	Bootstrap hard	Re-weighting	QActor(100)	QActor(300)	QActor ^{D}(100)
CIFAR-10	69.33	62.89	35.45	64.46	69.20	46.36	76.94	81.26	77.83
CIFAR-100	37.35	39.52	6.92	38.51	29.99	8.54	47.63	50.57	48.40

network architecture to train all models. As the results illustrate, QActor is outperforming all the prior arts significantly by relabeling only 10% of the data. Although these state-of-the-art models are successful in classification tasks of samples affected by label noise, they don't benefit from an expert labeler during their training procedure. The best performance is achieved by *D2L* and *Bootstrap hard* which however are still 15 percent points lower than our QActor with 100 active queries per iteration for CIFAR-10. Increasing the active queries to 300 per iteration increases the gap by another 6 percent point. The other models, i.e. *Co-teaching*, *Forward*, *Bootstrap soft*, and *Re-weighting* only achieve at best about 46% accuracy. This underlines how our method copes very well with noisy labeled data. For CIFAR-100, *Forward* and *Bootstrap soft* are best performing models while being almost 9% and 12% below QActor with only 100 and 300 queries per iteration respectively. As the last row of the table demonstrates, dynamic allocation of the budget boosts the accuracy of QActor with 100 up to one more percent. We will analyze this dynamic policy with more details in section 4.5.4.

4.5.2. INFORMATION METRICS

Here we compare the effect on the accuracy obtained on the test set when changing the underlying uncertainty measure.

In particular, we consider the uncertainty measures *CENT*, highest Entropy (*ENT*), and random baselines *Random* and *Semi-Random* to select samples from the noisy set to be queried for their labels. Moreover, we also analyze the methodology discussed by [152] and we set $\alpha = 0.8$. Figure 4.3 summarises the results on CIFAR-10 for 60% noise and $o(t) = 100$ over 50 iterations. As the figure shows, *CENT* is outperforming all the baselines over the iteration particularly in the earlier stages. As the number of queried samples grows, the performance of all the methods converges close to each other except for *Semi-Random*. The reason is that in the final iterations a large number of cleaned samples weigh more than the informativeness of them which is observed in all AL studies. Comparing *ENT* and *CENT* overtime shows the that leveraging the ability of CE function to detect the clean samples in the suspicious set and training on them is beneficial especially with small query number in the early stages. Moreover, results show the weighted average of entropy and cross-entropy values in *Re-Active* is not a suitable measure for informativeness. Analyzing the performance of Random, however, shows that random selection of samples to be cleaned by the oracle is less effective than smartly querying the informative samples.

The decline in the performance of *Semi-Random* over time is due to the selection of the noisy samples as the *semi-clean* data and adding them to the training pool. The reason is that *Semi-Random* randomly chooses samples from the informative suspicious set and directly sends them to the training set without cleaning, while *CENT* chooses these samples based on their low CE value. This comparison shows the effectiveness of

Table 4.2: Accuracy (%), clean sample percentage in active set, and False Discovery Rate (FDR) (%) of our informative measure (*CENT*) compared to baselines on CIFAR-10 and CIFAR-100 with 60% noise and 100 queries over 50 and 30 iterations, respectively. FP and TP represent false and true positives in the suspicious set where positive denotes noisy data.

Dataset	Method	Accuracy(%)	Clean(%) in Active Set	FDR = FP/(FP+TP)(%)
CIFAR-10	CENT	**76.94**	9.58	11.62
	ENT	75.77	39.92	11.27
	Re-Active	73.62	1.16	12.67
	Random	75.37	12.56	13.01
	Semi-Random	71.52	12.04	11.55
CIFAR-100	CENT	**47.63**	9.06	27.34
	ENT	45.80	38.07	26.98
	Re-Active	47.00	1.97	26.75
	Random	47.59	25.24	26.50
	Semi-Random	46.76	26.99	26.41

using CE values to distinguish between clean and noisy samples.

Table 4.2 shows the detailed statistics of clean samples in the active for *CENT* and the sample selection baselines for both CIFAR-10 and CIFAR-100. Observing the last column that indicates the ratio of the FP (samples that are clean but have been considered noisy in the suspicious set by the quality model) to the suspicious set size, shows the total ratio of the clean samples in the suspicious set. This number is close to the *Semi-Random* clean ratio which shows the *Semi-Random* fails smartly select clean samples for the *semi-clean* set. Moreover, to see the effect of the *semi-clean* set in *CENT*, we compare the percentage of the clean samples in that set with the same value with *Semi-Random* strategy, where the data in the *semi-clean* set is selected randomly. For CIFAR-10 this number is 63.24% while it is only 11.98% for *Semi-Random*. The numbers are 73.86% and 26.24% for CIFAR-100 respectively. This illustrates how the cross-entropy value helps *CENT* to select mostly clean samples as the *semi-clean* set.

Furthermore, comparing the number of clean samples in the active set for *ENT* and *CENT* validates our argument over the ability of CE value to recognize noisy samples. It should be noted that although *Re-Active* is successful in including mostly noisy samples in the active set, the selected samples are the least helpful for the performance. Our detailed analysis indicates that this measure results in selecting the data mainly from a few classes instead of a more homogeneous selection. The high value of clean samples in the active set for *ENT* indicates that over half of the budget is being wasted on the samples that, although informative, were already clean. As mentioned earlier, our motivation to introduce *CENT*, was to overcome this waste and have a measure that focuses on both informativeness and noisiness of the samples that are being selected to be relabeled. As the experimental results confirm, we believe having such a measure is essential in active learning applications on noisy labeled data.

4.5.3. QACTOR MODEL SENSITIVITY ANALYSIS

Here we present the results using the static policy termed QActor(100) and QActor(300) with constant 100 and 300 queries per iteration, respectively. We compare our QActor

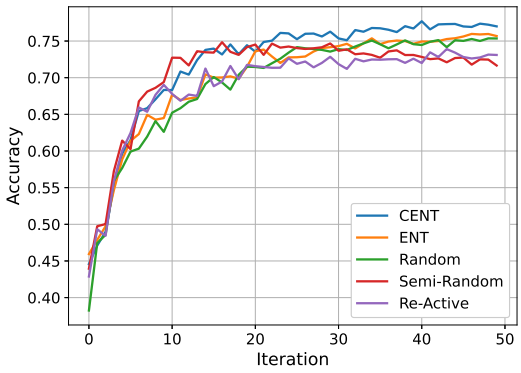
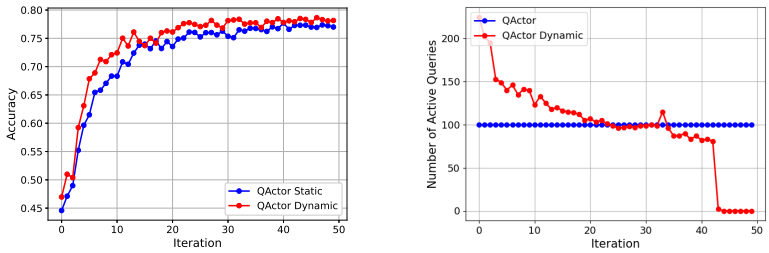


Figure 4.3: QActor accuracy under five different noise-aware informative measures on CIFAR-10 with 60% noise where 100 queries are labeled by the oracle at each iteration.



a) Accuracy b) Number of Queries

Figure 4.4: Accuracy (a) and the number of queries (b) of QActor with static (blue) and dynamic (red) querying strategies on CIFAR-10 with 60% noise.

using *CENT* with the selection baselines from Section 4.4 with the different numbers of active queries and noise rates. Table 4.3 summarises the results. As expected, *No-Sel* which directly learns from the data has the worst results in the presence of noise, achieving at most 68.11% accuracy. Note that the accuracy of *No-Sel* declines over time and we reported the highest accuracy that the model achieved over epochs. However, our static QActor with *CENT* with only 100 active querying per iteration from the oracle comes remarkably close to *Opt-Sel*, where the model is trained with only the clean samples.

Applying either sample selection or active querying alone achieves intermediate results compared to *No-Sel*. *Q-only* alone is more efficient driving up the accuracy to approximately 78% for CIFAR-10. However, *Q-only* has very poor performance for CIFAR-100. This comparison shows the effectiveness of both the quality model and active learning since neglecting any of both would result in a decrease in the accuracy. Moreover, we can observe with the increase in the number of the active query per iteration the performance increases over time.

Table 4.3: Accuracy (%) of QActor using a 100-query budget compared to three alternative approaches on CIFAR-10 and CIFAR-100 under 30% and 60% noise ratios.

Dataset	Noise	Opt-sel	No-sel	Q-only	QActor(100)	QActor^D(100)
CIFAR-10	30%	88.52	68.11	78.74	86.24	86.31
	60%	85.19	64.86	75.04	76.94	77.83
CIFAR-100	30%	59.29	50.50	36.52	54.28	54.74
	60%	52.67	33.71	35.75	47.63	48.40

4.5.4. DYNAMIC QACTOR

We compare our dynamic policy QActor^D using a given budget of $B = 5000$ with our static policy QActor (100) that queries over the whole time horizon the same number of instances. Figure 4.4(a) and Table 4.3 summarize the accuracy results for CIFAR-10. Using dynamic query allocation policy (red line) of the budget across the batches leads to a better performance of 77.83% than the static policy (blue line) 76.94%. This increase in performance is more visible in higher noise rate as it is more crucial to clean more samples by the oracle. Looking at the evolution, we observe a higher performance compared to the static policy particularly in the first iterations.

This result stems from the fact that the dynamic model queries more in the earlier batches when the model is less accurate and confident. Figure 4.4(b) shows the evolution of the number of active queries used in QActor^D across the time periods. We see that indeed in the beginning the number of queries increases goes above the static assignment (100). In later batches, the number of queries goes then near and lower than the static case. The fluctuation of the query number per iteration shows how this number imitates the model's overall performance. Eventually all the budget is used over the whole time.

4.6. CONCLUSION

In this paper, we consider the challenging problem of image classification with corrupted labels. We propose QActor, a learning algorithm for very noisy label datasets, introducing an active learning methodology suited for noisy labeled data called *CENT*. The core of QActor is composed of a quality model that filters out noisy labels and an active learner that smartly selects informative noisy instances to be relabeled by an oracle. The unique feature of QActor is its noise-awareness while selecting informative data and its dynamic query allocation over training iterations based on the training loss. The flexible design enables QActor to be generalized on both standard and deep learning models that have limited clean data labels. Our extensive evaluation on CIFAR-10 and CIFAR-100 shows that QActor can effectively combine the merits of the quality model and active learning when encountering extremely noisy labels, i.e., up to 60%. Compared to the state-of-the-art addressing noisy labels, QActor achieves higher accuracy by at least 8% at the cost of querying the oracle to cleanse 10% of suspicious images.

Despite these contributions, several limitations must be acknowledged. First, while future work could explore different types of deep neural network models, such as Bayesian neural networks, to enhance prediction accuracy, this aspect remains unaddressed in the current study. Additionally, we assume that the oracle provides perfect labels, which

may not reflect real-world scenarios. A more general approach would consider the oracle providing labels with associated confidence levels and probabilities of correctness. Furthermore, the proposed method and its informativeness strategy are designed for single-label learning, which may limit its applicability in multi-label learning contexts. These aspects represent opportunities for future research to refine and broaden the effectiveness of QActor.

5

LEARNING FROM NOISY LABELS WITH LABEL AGGREGATION APPROACH

Today, to label the massive datasets needed to train Deep Neural Networks (DNNs), cheap and error-prone methods such as crowdsourcing are used. Label aggregation methods aim to infer the true labels from noisy labels annotated by crowdsourcing workers via labels statistics features. Aggregated labels are the main data source to train deep neural networks, and their accuracy directly affects the deep neural network performance. In this paper, we argue that training DNN and aggregating labels are not two separate tasks. Incorporation between DNN training and label aggregation connects data features, noisy labels, and aggregated labels. Since each image contains valuable knowledge about its label, the data features help aggregation methods enhance their performance. We propose LABNET *an iterative two-step method. Step one: the label aggregation algorithm provides labels to train the DNN. Step two: the DNN shares a representation of the data features with the label aggregation algorithm. These steps are repeated until the converging label aggregation error rate. To evaluate* LABNET *we conduct an extensive empirical comparison on CIFAR-10 and CIFAR-100 under different noise and worker statistics. Our evaluation results show that* LABNET *achieves the highest mean accuracy with an increase of at least 8% to 0.6% and lowest error rate with a reduction of 7.5% to 0.25% against existing aggregation and training methods in most cases.*

5.1. INTRODUCTION

The remarkable success of Deep Neural Networks (DNNs) in supervised learning models, e.g., in computer vision, natural language processing, and recommender systems, is mainly dependent on data annotated by human knowledge [155]. It is drastically time-consuming and expensive to label each instance by a human expert for massive datasets.

Crowdsourcing platforms have emerged as an efficient and inexpensive solution for the label annotation task. Since workers in the crowd might have different expertise levels, the quality of annotated labels is not remarkable [140], [156]. Because the data labeled by crowdsourcing is used to train DNNs, the label quality has a direct impact on the performance and accuracy of the trained models. Label aggregation methods aim to infer the correct labels from noisy annotated data such as stemming from crowdsourcing. In other words, label aggregation methods try to provide high quality labels for training DNNs. We need to mention here, our focus is on image classification tasks, but our method has the ability to apply to all datasets that contain data features. The current solutions consider label aggregation and image classification as two separate tasks. At the first step, the label aggregation algorithm performs data labeling to enhance the quality of the labels generated by crowdsourcing workers. Next, the provided dataset, including sample and annotated label by label aggregation method, is used to train DNN.

The critical point in the process of labeling, then learning, is the lack of connection between label aggregation and the DNN training. In addition, the current label aggregation, as its name says, only uses label information to find the correct labels without considering the data features and samples themselves.

In contrast to the current solutions we consider the label aggregation and DNN training in a collaborative and interrelated manner. In our proposed framework named LABNET, label aggregation algorithms and classification tasks work together iteratively and exchange useful knowledge. As mentioned before, label aggregation algorithms work based on label information without taking into account data features. The data features are used to train DNNs, but their information can be used to aggregate labels. Since our method is a collaborative process between training DNN and labels aggregation, DNN extracts a representation of data features to share with the label aggregation algorithm as additional information for increasing the quality of annotated labels. Aggregated labels are used to train DNN as training data, and DNN performs the classification task besides features extraction for aggregation. In other words, our proposed method interleaves label aggregation and classification tasks such that the feature extraction part can be useful to both simultaneously.

The majority of label aggregation methods are iterative-based algorithms. One of the most well-known method is Expectation–Maximization [157]. The Expectation–Maximization (EM) algorithm is commonly used in aggregation via maximizing the likelihood of the data. Prior probabilities are an essential part of calculating the data likelihood. Numerous probabilistic and statistics models [158] have studied the impact of different priors on maximum likelihood and, consequently, EM algorithms. The error rate of aggregated labels directly depends on prior probabilities. Hence choosing the wrong prior probability leads to poor performance of the EM algorithm, and as a result, the number of incorrectly aggregated labels increases.

In LABNET, DNN has two tasks: 1) the traditional classification and, 2) features extrac-

tion for aggregation algorithm. We use the output of the last layer of the DNN, which is a softmax layer, to map data features into a probability vector. In other words, LABNET links data features and labels via the DNN softmax output and aggregated labels. Hence LABNET consists of a classifier and label aggregator. The label aggregator works based on the EM algorithm, and the classifier is a DNN, e.g., for image classification. At each iteration, the aggregator provides labels for input images to train the image classifier. In addition to learning the image classifier, the classifier represents each data sample by a softmax vector, which is considered as the prior probability in the label aggregation algorithm.

Since training the classifier at each iteration of the EM algorithm would be extremely time-consuming, we design an algorithm for deciding when to train the DNN with the latest aggregated labels. After each iteration of the EM algorithm, we use cross-entropy function to evaluate the difference between aggregated labels and the classifier prediction at each iteration. At each iteration, the value of cross-entropy is compared to cross-entropy in the previous iteration. After comparing the cross-entropy of two consecutive iterations, the training classifier and label aggregation algorithm will be performed for the next iteration if the difference is ascending. Otherwise, only label aggregation will be performed.

We evaluate LABNET on two popular image datasets [129] with synthetic label noise for each worker. We consider three noise patterns including uniform, bimodal, and flip noise under different miss rates for workers. We compare LABNET performance including classifier accuracy and aggregated labels error rate against three standard methods for aggregation, i.e., EM algorithm without using softmax as the prior, Minimax Entropy and Majority Voting. Our results show that LABNET achieves both high classification accuracy and low label aggregation error rate against baselines methods under various scenarios with different worker numbers, noise patterns, and noise ratios. LABNET outperforms other baselines accuracy 2% and 3% on average for CIFAR-10 and CIFAR-100, respectively. In terms of label aggregation error rate, LABNET reduces error rate by 1% and 3% on average against the baselines for CIFAR-10 and CIFAR-100, respectively.

The contributions of this chapter are summarized as follows.

- LABNET leverages an interactive method for training DNN and aggregating labels. LABNET considers the two as collaborative task. Each step sends its feedback to each other one to improve the performance of the whole framework, i.e., high DNN accuracy and low aggregation error rate.
- We estimate the prior probability to use in the EM algorithm via the softmax output of the DNN which benefits of using features and labels together.
- We design an algorithm to decide when to train the DNN based on the cross-entropy between the aggregated labels and the labels predicted by the DNN in the previous training round.

5.2. RELATED WORK

Robust training DNNs and label aggregation are well-studied subject areas. Most of the existing robust learning methods do not take into account collaboration with label aggregation algorithms.

Robust DNNs consider three different approaches to distill the impact of wrong labels,

including filtering wrong labels [61], [119], estimation of noise confusion matrix [49], [70] and noise tolerant loss function [57], [63]. Robust training methods [57], [82], [85], [93], [128] mainly focus on the learning task considering robust architecture and loss adjustment. In robust training, it does not matter how the data is labeled and by what process. Hence, in robust training label aggregation is considered as an entirely separate task.

Label aggregation methods mostly use unsupervised solutions. As an example, [159] introduces an unsupervised method using two neural networks including a classifier and re-constructor for label aggregation. We can categorize the aggregation methods into two groups. The first group relies on probabilistic inference models [160], [161], which study the impact of latent variables on the likelihood of noisy label samples. [162]–[165] develop a probabilistic graphical model for label aggregation, and a confusion matrix is considered for evaluating each worker. [166] works based on prior approximation via variational inference.

The second group encompasses confusion matrix based methods. The focus is to find how correct labels are corrupted to noisy labels by each worker [157]. GLAD [167] is a binary labeling method that infers the true label and difficulty of each sample at the same time. Also, Zhou et al. [168], [169] design a framework that uses minmax entropy estimator and assigns a different probabilistic distribution to each sample-worker pair.

Another aspect of label aggregation is to use a deep neural network for aggregating crowdsourced responses. As opposed to the existing unsupervised method, DeepAgg [170] is a supervised model. DeepAgg, instead of using Bayesian algorithms, the label aggregation model relies on DNNs to encode required information for statistical models. Another aggregation method is based on a disagreement between the provided labels by workers and predictions of learning algorithm [171].

The aforementioned studies are limited to label aggregation without taking into account feature space. In this paper, we show that the use of data representation in the label aggregation algorithm reduces the error and increases the quality of the output labels. Besides, prior arts do not consider the training process and relation with the aggregation algorithm. In other words, there is useful knowledge that can be shared between the label aggregation and the training processes. LABNET is the first study that focuses on model interaction between label aggregation and training a deep neural network to the best of our knowledge.

5.3. METHODOLOGY

Consider the classification and label aggregation problem having training set with N samples $\mathscr{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N}$, where $\boldsymbol{x}_i$ is the feature vector of the i^{th} sample and $\boldsymbol{y}_i = \{y_i^1, y_i^2, \ldots, y_i^w, \ldots, y_i^K\}$ is the set of labels from different workers $w = \{1, 2, \ldots, K\}$ for the i^{th} instance. $y_i^w \in \{0,1\}^C$ is the corresponding label vector generated by worker w in the crowdsourcing setting and C denotes the number of classes.

To label data, it is common practice to use crowdsourcing. Crowdsourcing is a cheap and fast method to label massive data sets compared to labeling by human experts, but it is less accurate. The accuracy of a crowd system depends on the ability of inexperienced workers to identify the correct label. Label aggregation algorithms are proposed

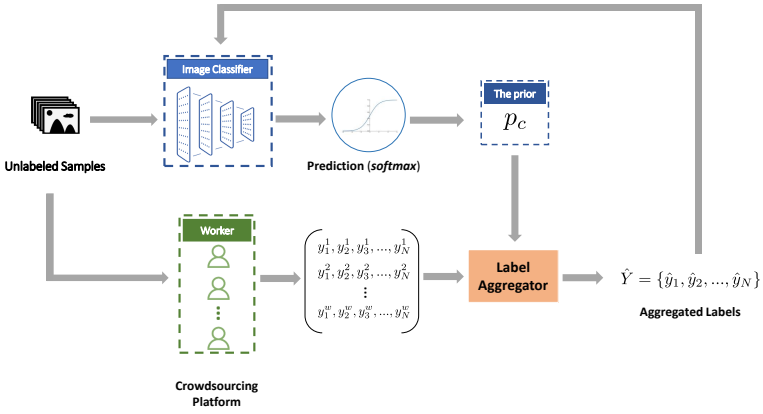


Figure 5.1: The label aggregation and training DNN scenario in LABNET.

to increase the accuracy of crowd labels. Thus the aggregated labels are more accurate than the label sets provided by single workers. After aggregation process, the aggregated labels $\hat{Y} = \{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_N\}$ are used to train a classifier network. We denote the classifier network prediction as $f(\boldsymbol{x};\boldsymbol{\theta})$ where $\boldsymbol{\theta}$ are the weights of the classifier network. Since the aggregated labels together with the samples constitute the training data for the neural network, the accuracy of the trained neural network predictions depends directly on the accuracy of the aggregated labels.

We propose a method that leverages a direct cooperative approach between the classifier network and the label aggregator. The method first aggregates labels using the estimated probabilities by the classifier via the softmax layer. After label aggregation using the softmax output as the prior knowledge in the aggregation algorithm, the network is trained via the aggregated labels. In other words, the label aggregation algorithm and the classifier exchange helpful knowledge iteratively. As a result, both the accuracy of the classifier network and the accuracy of the aggregated labels increase.

We illustrate the procedure of aggregation and training the classifier in Figure 5.1. At the beginning, each worker assigns a label to each instance based on its expertise level. In the next step, the label aggregator performs the aggregation task using the EM algorithm. In the first iteration since we have no prior estimated by the classifier, we use majority voting algorithm to aggregate the labels with which we train the classifier. After training the classifier, we use it to infer the softmax vector output for each sample, to be used as prior by the EM-based aggregation algorithm. EM-based algorithms are commonly used for label aggregation. EM needs knowledge of the prior data distribution to estimate the probability of correctness of a label generated by a worker. p_c denotes the prior probability of class label c. In LABNET, we estimate p_c via the predicted class probability by the softmax layer of the neural network. Hence the inferred softmax probability vector from each instance is useful knowledge for the EM algorithm. Vice versa, to train the DNN, we use the labels provided by the EM algorithm, which selects the labels having maximum likelihood among the labels provided by the workers.

5.3.1. LABEL AGGREGATION

We consider the scenario shown in Figure 5.1. To each data sample $\boldsymbol{x}_i$ corresponds a set of labels $\boldsymbol{y}_i = \{y_i^1, y_i^2, \ldots, y_i^K\}$ from K different crowd workers. The label aggregator uses the EM algorithm to estimate the correct labels. The EM algorithm is an iterative algorithm, including an E- and an M- step. In the E-step for each instance i, the algorithm first calculates the probability that the aggregated label $\hat{y}_i$ equals t given that the label y_i^w generated by worker w equals l. $P_w(\hat{y}_i \mid y_i^w)$ denotes this probability. Therefore we have the following expression

$$P_w(\hat{y}_i = t \mid y_i^w = l) = \frac{\sum_{j=1}^{N} \mathbb{1}(\hat{y}_i = t \,\&\, y_j^w = l)}{\sum_{c=1}^{C} \mathbb{1}(\hat{y}_i = t \,\&\, y_i^w = c)} \tag{5.1}$$

where $\mathbb{1}(.)$ is the indicator function. In Equation (5.1), We count the number of times that worker k labels item i to l when the aggregated label is t over the number of classes are labeled by worker k when the aggregated label is t.

The next stage of the EM algorithm needs the prior probability of each class. In the classic EM algorithm these probabilities are calculated as $p_c = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}(\hat{y}_i = c)$ where $\hat{y}_i$ is the aggregated label and c is the class label. Thus $\mathscr{P} = \{p_1, p_2, \ldots, p_c, \ldots, p_C\}$ is the set of prior probabilities for each class.

In our proposed method, we replace the prior class probabilities to make them dependent on the features extracted by the classifier. As anticipated in Figure 5.1, we use the softmax output of the classifier prediction $f(\boldsymbol{x};\boldsymbol{\theta})$ as set of prior probabilities. The softmax vector is computed based on image features. Hence, the prior probabilities from the softmax vector contain information on the features of the sample data because the softmax vector is a representation of the input data in the form of a probability vector. For label aggregation,we use as prior the predicted class probability across all samples:

$$p_c = \frac{1}{N}\sum_{i=1}^{N} f_c(\boldsymbol{x}_i;\boldsymbol{\theta}). \tag{5.2}$$

where $f_c(\boldsymbol{x};\boldsymbol{\theta})$ denotes the softmax value of class c by the classifier $f(\boldsymbol{x};\boldsymbol{\theta})$. After finding the prior probability based on the proposed method, the EM algorithm begins maximizing the data likelihood. In the M-step, the likelihood $q_{i,c}$ of each instance i being of class c is computed as:

$$q_{i,c} = p_c \prod_{w=1}^{K} P_w(\hat{y}_i = c \mid y_i^w). \tag{5.3}$$

which leverages the Bayes's theorem. $\mathscr{Q}_i = \{q_{i,1}, q_{i,2}, \ldots, q_{i,C}\}$ denotes the set of likelihoods for each data instance i for all C classes. The aggregated label is the class c having maximum likelihood:

$$\hat{y}_i = \underset{c}{\operatorname{argmax}} \;\; \mathscr{Q}_i. \tag{5.4}$$

After the M-step the label aggregator passes the set of aggregated labels $\hat{Y} = \{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_N\}$ to the training of the classifier network.

5.3.2. CLASSIFIER TRAINING

The data to train the classifier $f(\boldsymbol{x};\boldsymbol{\theta})$ includes features vector $\boldsymbol{x}_i$ and aggregated labels vector $\hat{y}_i$ from the aggregator. Hence the loss functions for training can be written as follows:

$$\ell = \min_{\boldsymbol{\theta}} \frac{1}{N} \sum_{i=1}^{N} \mathscr{L}_{CE}(f(\boldsymbol{x}_i;\boldsymbol{\theta}), \hat{y}_i) \tag{5.5}$$

where $\mathscr{L}_{CE}(.,.)$ is the cross entropy loss function:

$$\mathscr{L}_{CE} = -\sum_{j=1}^{C} \hat{\boldsymbol{y}}_j \log f_j(\boldsymbol{x_i}) \tag{5.6}$$

where $\hat{\boldsymbol{y}}_j$ is the aggregated label and $f_j(\boldsymbol{x_i})$ is the softmax probability for the j^{th} class. These probabilities represent the data features and are fed back to EM algorithm via Equation (5.2).

5.3.3. MAKING DECISION TO TRAIN DNN

Training the DNN at each iteration of the EM algorithm would be extremely computation intensive and unpractical. Therefore, we need to carefully decide when to train to reduce the computational load while guaranteeing the accuracy of the classifier and correctness of the aggregated labels. At the end of each round of the EM algorithm we measure the (dis)agreement between the DNN predicted labels and the aggregated labels. If the disagreement increases with respect to the previous iteration, we triggers the training of the DNN to prevent further drifts between the predicted and aggregated labels.

More in detail at each iteration we calculate the disagreement using the cross-entropy function as :

$$\mathscr{H}_t = -\frac{1}{N} \sum_{i=1}^{N} \tilde{y}_i \log f(\boldsymbol{x_i};\boldsymbol{\theta}) \tag{5.7}$$

Equation (5.7) measures the distance between DNN predicted and the aggregated labels. Therefore, if the value of $\mathscr{H}$ increases between iteration rounds the predicetd and aggregated labels are diverging. Hence, we trigger training of the DNN with the new aggregated labels. Likewise, if the value of $\mathscr{H}$ does not increase between iteration rounds, there is no need to train DNN at the next iteration.

5.3.4. END-TO-END TRAINING PROCEDURE

Our framework aims to jointly aggregate labels with low label error rate and train a DNN with high accuracy. We describe the whole LABNET and collaboration between DNN and label aggregator in Algorithm 5. In this scenario, we have multiple workers which provide labels for each data instance, thus the input dataset is $\mathscr{D} = \{\boldsymbol{x}, Y\}$ (line 1). To initialize the algorithm we need some aggregated labels to start with. We derive these via simple Majority Voting (line 2). Before label aggregation, we calculate the number of labels obtained from each class after aggregation from each worker named $L_{C\times K\times C}$ (line 4-6) and force training of the classifier $f(\boldsymbol{x},\boldsymbol{\theta})$ (line 7). At the beginning of each iteration if the *train* flag is true, the classifier $f(\boldsymbol{x},\boldsymbol{\theta})$ is trained with the aggregated labels $\hat{Y}$ (line

10-12), then we use the softmax output of $f(\boldsymbol{x}, \boldsymbol{\theta})$ as the prior (line 13-14). In case of *train* is false, the prior probability for c^{th} class in $\mathscr{P}$ vector equals to the average number of occurrence of class c into aggregated label set (line 16-17). In each iteration, we calculate the need to train based on Equation (5.7) (line 18-23). $\tilde{Y}$ denotes the one-hot vectors of aggregated labels $\hat{Y}$, and $\tilde{y}_i \in \tilde{Y}$ is the one-hot vector for aggregated label of sample i. The label aggregator method works based on an EM algorithm that starts with E-step via finding the conditional probability of aggregated labels given each worker's generated label (line 24-26). The M-step includes finding the likelihood of each instance and class label (line 27-29). Finally, the aggregated label is the class with maximum probability among all classes (line 30). This procedure repeats for each iteration. The aggregated labels are used to train the DNN and the DNN softmax output used as prior.

5.4. EVALUATION

5.4.1. EXPERIMENT SETUP

Dataset. We consider two different vision datasets in our experiments to evaluate the performance of our algorithm against other methods.

- **CIFAR-10** [129]: contains 60K 32×32 pixels samples. The labels are classified into 10 categories. Here we use 50K samples as training data and 10K for testing data.
- **CIFAR-100** [129]: is similar to CIFAR-10 except that its labels are grouped into 100 categories.

Noise and workers. To simulate workers with different level of expertise for annotating images, we use three different noise patterns in our experiments.

- **Uniform:** This noise corrupts the true label into another random label with equal probability.
- **Bimodal:** This noise corrupts the original class label around two other targeted classes, each following truncated normal distribution. The $\mathscr{N}^T(\mu, \sigma, a, b)$ includes μ that specifies the target and σ which controls the spread. a and b denote the class label boundaries.
- **Flip:** This noise is generated by flipping the original class label to another class with a specific probability.

Baselines. We consider three different baselines for comparison.

- **Majority Voting (MV):** is a basic label aggregation method which chooses the label with the highest consensus.
- **Expectation Maximization (EM)** [157]: is an iterative method to estimate the confusion matrix of workers by maximizing the likelihood of observed labels. The diagonal elements show the probability of aggregated label.
- **Minimax Entropy (ME)** [168]: This method considers a confusion matrix for workers and encodes their labeling expertise. In addition, the ME assigns a vector to items and encodes their labeling difficulty. It uses a minimax entropy approach to estimate the confusion matrix and vector together.

We implement all algorithms in Python programming language using Keras version 2.2.4 and TensorFlow version 1.12.

Parameters. To conduct our experiments on CIFAR-10 and CFAR-100, we use an 8-layer CNN with 6 convolutional layers followed by 2 fully connected layers, and ResNet-44,

Algorithm 5: LABNET

Input: training set $\mathscr{D} = \{\boldsymbol{x}, Y\}$, Epoch E_{max}, Iteration I_{max} Initialize randomly $\boldsymbol{\theta}$
$\hat{Y} = \text{MajorityVoting}(Y)$ `/* ` $Y_{i,j}$ `denotes` i^{th} `instance from` j^{th} `worker */`
Output: The aggregated labels $\hat{Y}$, Trained network $f(\boldsymbol{x}, \boldsymbol{\theta})$
for $i = 1, 2, \ldots, N$ **do**
  for $w = 1, 2, \ldots, K$ **do**
    $L[\hat{y}_i][w][y_i] += 1$
$train$ = **True**
$\mathscr{H}_0 = 0$
for *each iteration* $t = 1, 2, \ldots, I_{max}$ **do**
  if $train$ **then**
    for *each* $e = 1, 2, \ldots, E_{max}$ **do**
      Train $f(\boldsymbol{x}, \boldsymbol{\theta})$ with $(\boldsymbol{x}, \hat{Y})$;
    for *each* $c = 1, 2, \ldots, C$ **do**
      $\mathscr{P}_c = \frac{1}{N} \sum_{i=1}^{N} f_c(\boldsymbol{x}_i; \boldsymbol{\theta})$;
  else
    for *each* $c = 1, 2, \ldots, C$ **do**
      $\mathscr{P}_c = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}(\hat{y}_i = c)$
  $\tilde{Y} = \text{ONE-HOT}(\hat{Y})$
  $\mathscr{H}_t = -\frac{1}{N} \sum_{i=1}^{N} \tilde{y}_i \log f(\boldsymbol{x_i}; \boldsymbol{\theta})$
  if $\mathscr{H}_t - \mathscr{H}_{t-1} > 0$ **then**
    $train$ = **True**
  else
    $train$ = **False**
  for $i = 1, 2, \ldots, N$ **do**
    for $w = 1, 2, \ldots, K$ **do**
      $P_w(\hat{y}_i \mid y_i^w) = \frac{\sum_{j=1}^{N} L[\hat{y}_i^w][w][y_j^w]}{\sum_{c=1}^{C} L[\hat{y}_i^w][w][c]}$
  for $i = 1, 2, \ldots, N$ **do**
    for $c = 1, 2, \ldots, C$ **do**
      $\mathscr{Q}_{i,c} = \mathscr{P}_c \times \prod_{w=1}^{K} P_w(\hat{y}_i = c \mid y_i^w)$
    $\hat{Y} = \underset{c}{\text{argmax}} \ \mathscr{Q}$

respectively. All networks train with SGD with momentum 0.9, weight decay 5×10^3 and an initial learning rate of 0.1. For CIFAR-10 in each iteration, we train the DNN for 120 epochs, and the learning rate is divided by 10 after 40 and 80 epochs. For CIFAR-100, the total number of epochs is 150, and the learning rate is divided by 10 after 80 and 120 epochs. For the EM algorithm, the maximum number of iterations is 10. For bimodal noise, the center of the distribution around classes $\mu_1 = 3.0$, $\mu_2 = 7.0$ with variance $\sigma_1 = 1.0$, $\sigma_2 = 0.5$. For flip noise in CIFAR-10, we flip similar classes including **bird** → **airplane**, **truck** → **automobile**, **deer** → **horse** and **dog** ↔ **cat**. For CIFAR-100, the 100 classes are categorized into 20 super-classes. Each super-class consists of 5 sub-classes. We randomly select two sub-classes in each super-class and flip labels between sub-classes. The results presented are based on five independent runs to ensure the reliability and robustness of the findings.

Evaluation Metrics. To evaluate the performance of our proposed model against the baselines, we use as metric the label aggregation error rate and accuracy of the trained neural network.

- **Aggregation Error Rate:** is the percentage of inferred labels which differ from the true labels. The true labels are used only for evaluation not for training.
- **Accuracy:** Test accuracy is the percentage of correct predictions by the DNN on the testing data.

5

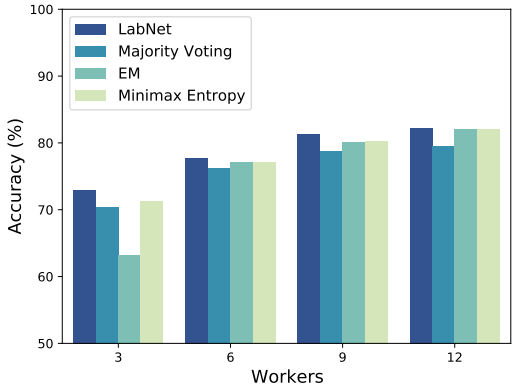
5.4.2. NUMBER OF WORKERS IMPACT

CIFAR-10

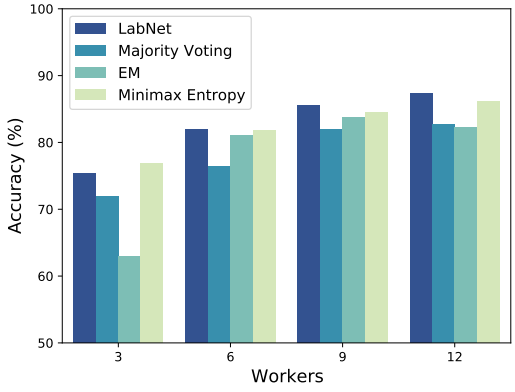
We summarize the results of CIFAR-10 in terms of DNN's accuracy and aggregation error rate for a different number of workers in Figure 5.2 and Figure 5.3, respectively. According to Figure 5.2(a), our method outperforms all competitors through various numbers of workers for the uniform noise pattern. When the noise pattern is bimodal in Figure 5.2(b), LABNET is the most accurate method against MV, EM, and ME except for the case of 3 workers that Minimax Entropy is the best one and our method is the second best result. Also, for the flip noise in Figure 5.2(c), LABNET achieves the highest accuracy by 65.59% and 70.12% when numbers of workers are 9 and 12, respectively. In the case of 3 and 6 workers, Minimax Entropy achieves the best accuracy by 38.88% and 64.24% test accuracy. Another observation worth mentioning is that the test accuracy increases with the number of workers for all cases. In general for LABNET, increasing the number of workers has the highest and lowest impacts on the classifier accuracy for the flip and uniform noise patterns, respectively.

We compare aggregation label error rate in Figure 5.3 for three different noise patterns over various numbers of workers. For uniform noise pattern shown in Figure 5.3(a), LABNET achieves the lowest error rate against other rivals by 58.93%, 36.87%, 27.74% and 19.53% for 3, 6, 9 and 12 workers, respectively.

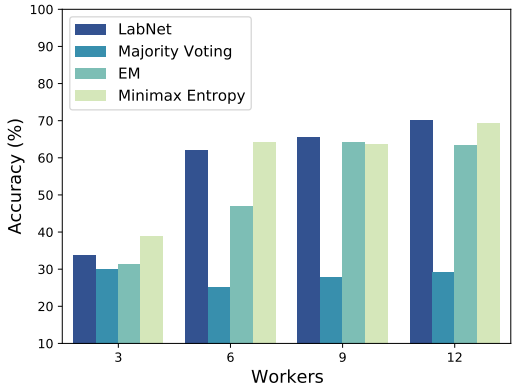
The direct impact of aggregation error rate on DNN accuracy is shown in Figure 5.3(b) and Figure 5.3(c). The Minimax Entropy has lowest error rate with 3 workers in bimodal noise pattern and the highest DNN accuracy in Figure 5.3(b) and Figure 5.2(b), respectively. Also LABNET achieves the best error rate results for 6, 9 and 12 workers against the baselines. Furthermore, we see the same pattern for flip noise in Figure 5.2(c) and Figure 5.3(c) which in Minimax Entropy performs better than other baselines when the



(a) Uniform

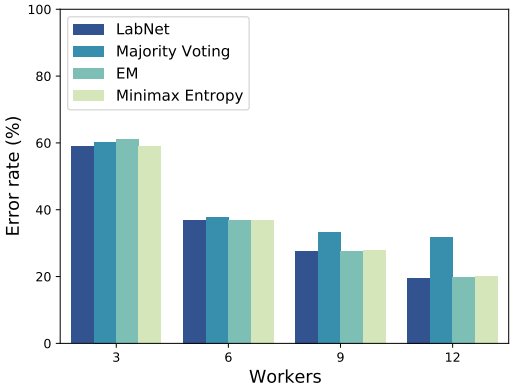


(b) Bimodal

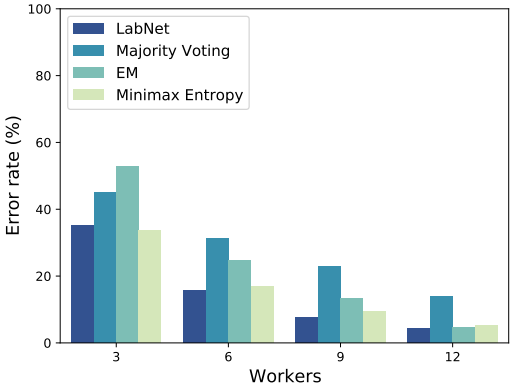


(c) Flip

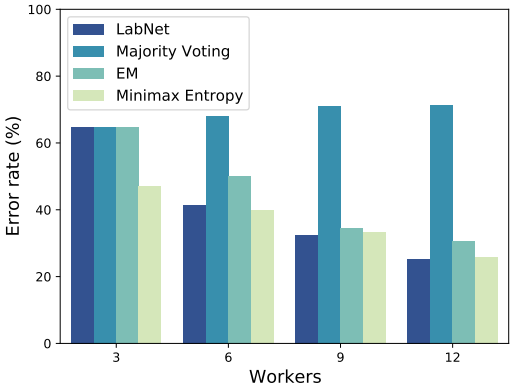
Figure 5.2: Accuracy (%) for different numbers of workers in LABNET compared to three baselines on CIFAR-10 under 60% noise ratio for three noise patterns: a) uniform, b) bimodal, and c) flip and missing ratio of 30%.



(a) Uniform



(b) Bimodal



(c) Flip

Figure 5.3: Label aggregation error rates for different numbers of workers in the LABNET, compared to three baselines, on CIFAR-10 under a 60% noise ratio for three noise patterns: a) uniform, b) bimodal, and c) flip, with a missing ratio of 30%.

number of workers equals 3 and 6. In addition, the worst results belong to MV that is the least accurate trained DNN among all baselines. For the case of 9 and 12 workers, LABNET achieves 32.45% and 25.20% aggregation error rate, respectively, which is the best method compared to MV, EM, and Minimax Entropy. The highest impact of increasing workers on error rate belongs to the flip noise pattern with a reduction of 39.59% for LABNET.

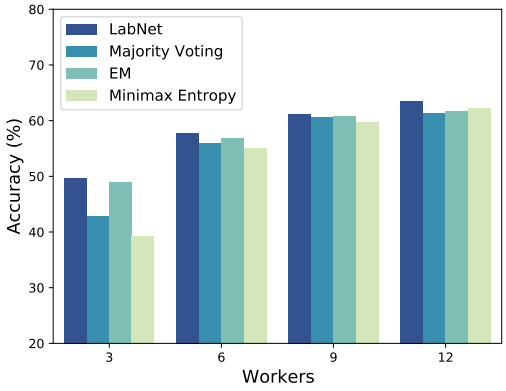
CIFAR-100

CIFAR-100 is more challenging than CIFAR-10 due to the higher number of classes. The accuracy of DNN and label aggregation error rate results are shown in Figure 5.4 and Figure 5.5, respectively. The first important observation through the results is the poor performance of Minimax Entropy for CIFAR-100 because, for a large number of classes, this method easily converges to the wrong local optimum. According to Figure 5.4, the bimodal noise pattern is the most straightforward pattern, and flip noise is the most complex one for classification and labels aggregation. As shown in Figure 5.4(a), LABNET achieves the best accuracy among all methods. The highest difference between the accuracy of LABNET and second best method is 1.5% for 12 workers and the least one is 0.4% for 9 workers. For bimodal noise in Figure 5.4(b), LABNET achieves the highest accuracy for all the scenarios with different numbers of workers. The best accuracy of LABNET equals 64.28% with 12 workers. In addition, the test accuracy for 3, 6, and 9 workers are 17.80%, 58.90%, and 63.98%, respectively. Since the flip noise is the most difficult noise pattern, the best accuracy of LABNET is 24.73% with 12 workers. Based on the Figure 5.4(c), LABNET outperforms all the baselines. In LABNET, increasing the number of workers from 3 to 12 improves the classifier accuracy 46.48% for the bimodal noise significantly rather than 13.81% and 14.56% for the uniform and flip noise patterns, respectively.

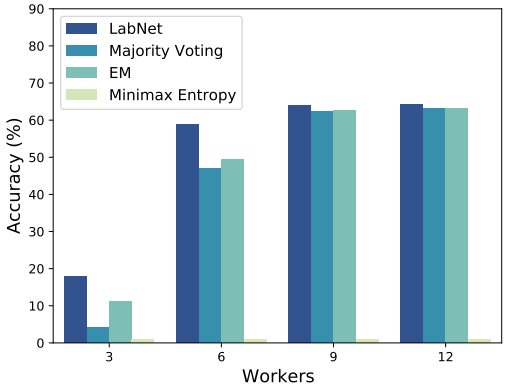
Figure 5.5 summarizes the aggregation error rate of our proposed method and other baselines over different numbers of workers. With respect to the test accuracy results in Figure 5.4, the most accurate method has the lowest label aggregation error rate in all the cases with different noise patterns and numbers of workers. For uniform noise, LABNET achieves 50.01% error rate in the case of 3 workers, which is the lowest among all methods, although performance of each method is close to each other. As we mentioned before, Minimax Entropy suffers from poor performance on dataset with large number of classes under bimodal and flip noise patterns which is shown in Figure 5.5(b) and Figure 5.5(c). Through all the noise patterns and different number of workers, LABNET performs as the best method against baselines with gaps of 7.33%, 3.23% and 54.21% error rate under uniform, bimodal and flip noise pattern with 12 workers, respectively. According to the results of LABNET in Figure 5.5, increasing the number of workers has the most effect on reducing the error for bimodal noise pattern.

5.4.3. IMPACT OF MISSING RATE ON DNN ACCURACY AND LABEL AGGREGATION ERROR RATE

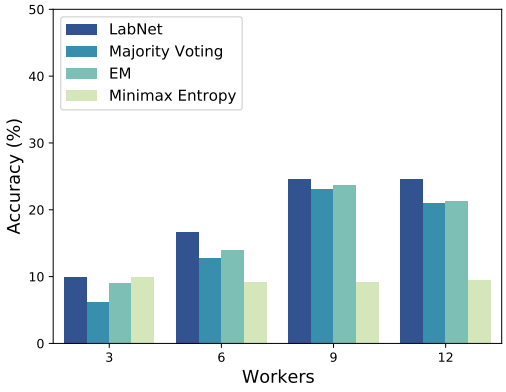
We extensively evaluate LABNET, MV, EM, and Minimax Entropy on CIFAR-10 and CIFAR-100, with label missing rates ranging from 0.0 to 0.3. The missing rate is the probability that a worker does not assign a label to a sample, resulting in the missing label. We also consider two different numbers of workers: 3 and 9. We summarize the average test



(a) Uniform

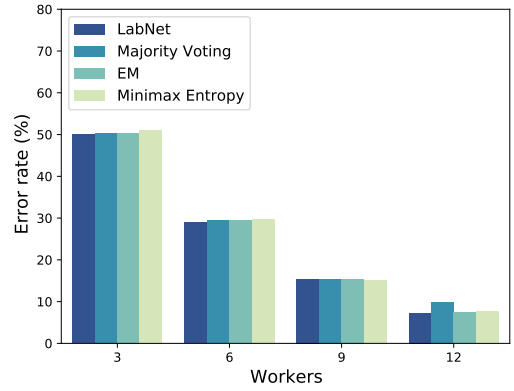


(b) Bimodal

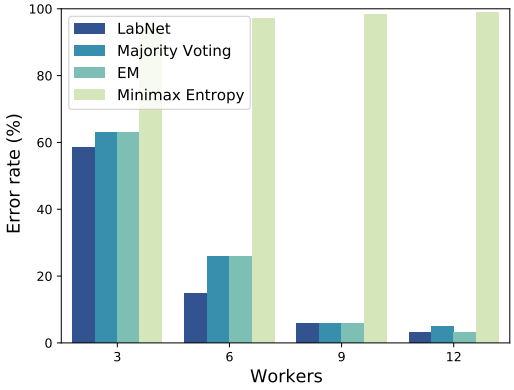


(c) Flip

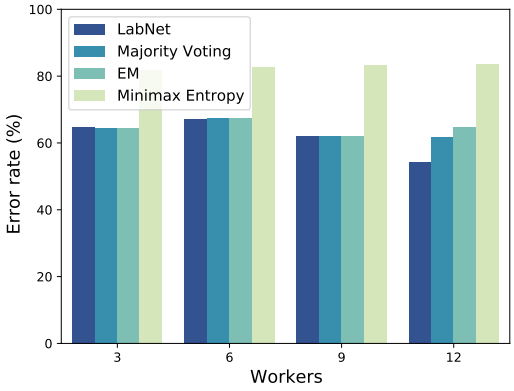
Figure 5.4: Accuracy (%) for different numbers of workers in LABNET compared to three baselines on CIFAR-100 under 60% noise ratio for three noise patterns: a) uniform, b) bimodal, and c) flip and missing ratio of 30%.



(a) Uniform



(b) Bimodal



(c) Flip

Figure 5.5: Label aggregation error rate for different numbers of workers in LABNET compared to three baselines on CIFAR-100 under 60% noise ratio for three noise patterns: a) uniform, b) bimodal, and c) flip and missing ratio of 30%.

accuracy and label aggregation error rate across three runs for CIFAR-10 and CIFAR-100 in Table. 5.1 and Table. 5.2, respectively.

Table 5.1: Accuracy and error rate of LABNET compared to MV, EM, and Minimax Entropy for three noise patterns (uniform, bimodal, and flip) with different numbers of workers and missing rates on CIFAR-10 with a 60% noise ratio.

# of Workers	Missing rate	LABNET		MV		EM		Minimax Entropy	
		Accuracy (%)	*Error rate (%)*	*Accuracy (%)*	*Error rate (%)*	*Accuracy (%)*	*Error rate (%)*	*Accuracy (%)*	*Error rate (%)*
Noise pattern = Uniform									
3	0.0	74.62 ± 0.32	54.51 ± 0.10	71.48 ± 0.29	55.93 ± 0.0	66.17 ± 0.33	59.07 ± 0.12	72.25 ± 0.31	54.87 ± 0.16
	0.1	74.11 ± 0.40	56.59 ± 0.05	71.31 ± 0.22	56.75 ± 0.0	70.15 ± 0.38	57.58 ± 0.05	74.17 ± 0.52	56.28 ± 0.08
	0.3	73.02 ± 0.25	58.91 ± 0.03	70.44 ± 0.19	60.28 ± 0.0	63.26 ± 0.28	61.17 ± 0.08	71.26 ± 0.19	58.99 ± 0.10
9	0.0	83.59 ± 0.52	20.75 ± 0.04	81.75 ± 0.38	21.26 ± 0.0	82.12 ± 0.42	20.88 ± 0.05	81.70 ± 0.62	21.15 ± 0.04
	0.1	83.31 ± 0.15	20.84 ± 0.03	81.07 ± 0.22	20.96 ± 0.0	82.51 ± 0.13	21.16 ± 0.02	70.85 ± 0.66	21.47 ± 0.09
	0.3	81.35 ± 0.19	27.71 ± 0.04	78.72 ± 0.21	33.31 ± 0.0	80.05 ± 0.31	27.71 ± 0.11	80.21 ± 0.27	27.98 ± 0.02
Noise pattern = Bimodal									
3	0.0	76.69 ± 0.31	34.09 ± 0.06	72.44 ± 0.42	44.63 ± 0.0	52.18 ± 0.26	50.71 ± 0.08	78.05 ± 0.38	31.76 ± 0.05
	0.1	76.24 ± 0.15	34.49 ± 0.02	73.40 ± 0.37	43.77 ± 0.0	47.03 ± 0.43	54.14 ± 0.07	77.22 ± 0.27	33.79 ± 0.04
	0.3	75.39 ± 0.37	35.15 ± 0.03	71.83 ± 0.17	44.92 ± 0.0	48.29 ± 0.36	52.81 ± 0.03	76.44 ± 0.56	33.94 ± 0.11
9	0.0	87.75 ± 0.21	4.19 ± 0.02	84.43 ± 0.33	11.56 ± 0.0	85.05 ± 0.20	8.51 ± 0.05	86.05 ± 0.18	5.43 ± 0.02
	0.1	87.44 ± 0.11	4.2 ± 0.02	83.84 ± 0.34	11.76 ± 0.0	84.55 ± 0.31	8.72 ± 0.04	85.42 ± 0.29	6.37 ± 0.02
	0.3	85.52 ± 0.38	7.53 ± 0.01	81.83 ± 0.46	22.86 ± 0.0	84.88 ± 0.25	7.56 ± 0.03	84.47 ± 0.30	9.41 ± 0.04
Noise pattern = Flip									
3	0.0	40.47 ± 0.34	59.17 ± 0.05	30.75 ± 0.29	64.79 ± 0.0	34.73 ± 0.25	64.78 ± 0.02	51.34 ± 0.49	50.39 ± 0.09
	0.1	39.89 ± 0.16	59.32 ± 0.03	31.07 ± 0.39	64.79 ± 0.0	32.19 ± 0.10	64.78 ± 0.09	45.76 ± 0.67	55.89 ± 0.08
	0.3	38.92 ± 0.37	60.93 ± 0.01	25.15 ± 0.66	68.03 ± 0.0	31.34 ± 0.26	64.80 ± 0.07	38.82 ± 0.41	60.95 ± 0.04
9	0.0	64.65 ± 0.10	28.56 ± 0.03	18.20 ± 0.43	73.37 ± 0.0	64.39 ± 0.23	30.62 ± 0.02	64.78 ± 0.36	28.43 ± 0.09
	0.1	64.64 ± 0.31	29.56 ± 0.02	21.93 ± 0.32	73.37 ± 0.0	64.19 ± 0.53	30.64 ± 0.08	64.21 ± 0.22	29.95 ± 0.12
	0.3	64.89 ± 0.69	31.45 ± 0.08	27.94 ± 0.19	70.96 ± 0.0	63.01 ± 0.40	32.33 ± 0.04	63.54 ± 0.37	32.29 ± 0.08

CIFAR-10. As shown in Table. 5.1, in the case of uniform noise pattern, LABNET achieves the highest accuracy in all cases. As we expect, variation of label missing rate affects DNN accuracy and label aggregation error rate. Increasing the missing rate reduces the accuracy of DNN and increases the label aggregation error rate. In addition, more workers enhance the DNN accuracy. According to the results, we observe an 8.97%, 9.2%, and 8.33% increase in accuracy by adding six workers to our method for the missing rate 0.0, 0.1 and 0.3 under uniform noise pattern, respectively. For the case of bimodal noise, Minimax Entropy is the best method when three workers are available. However, with nine workers, LABNET outperforms all baselines in terms of accuracy and error rate. According to the results in Table. 5.1, LABNET performs better aggregation on higher missing rate under flip noise pattern. Consequently, the accuracy of DNN is higher than other methods in the higher missing rate.

CIFAR-100. Since CIFAR-100 contains a more significant number of classes than CIFAR-10, label aggregation is a challenging task. According to the results in Table. 5.2, LABNET achieves the highest DNN accuracy and the lowest aggregation error rate against other baselines under uniform, bimodal, and flip noise patterns. The impact of missing rate variation on accuracy and error rate for CIFAR-100 is the same as the impact of missing rate on CIFAR-10. Another observation worth mentioning is the significant enhancement in Bimodal noise accuracy when the number of workers changes from 3 to 9. Also, the difference between LABNET accuracy and other baselines for CIFAR-100 is significantly higher than the results on CIFAR-10. There exist the same observations for labels aggregation error rate. In other words, our proposed model is significantly more accurate on a more complex dataset under bimodal and flip noise. Furthermore, Minimax Entropy performs label aggregation poorly for various missing rates equal to 0.0, 0.1, and

Table 5.2: Accuracy and error rate of LABNET compared to MV, EM, and Minimax Entropy for three noise patterns (uniform, bimodal, and flip) with different numbers of workers and missing rates on CIFAR-100 with a 60% noise ratio.

# of Workers	Missing rate	LABNET		MV		EM		Minimax Entropy	
		Accuracy (%)	*Error rate (%)*	*Accuracy (%)*	*Error rate (%)*	*Accuracy (%)*	*Error rate (%)*	*Accuracy (%)*	*Error rate (%)*
Noise pattern = Uniform									
3	0.0	48.83 ± 0.28	49.53 ± 0.05	42.89 ± 0.53	50.48 ± 0.0	48.21 ± 0.35	50.37 ± 0.04	39.64 ± 0.28	51.12 ± 0.02
	0.1	48.62 ± 0.18	49.55 ± 0.03	42.50 ± 0.29	50.58 ± 0.0	48.03 ± 0.42	50.41 ± 0.06	39.85 ± 0.35	50.92 ± 0.03
	0.3	49.65 ± 0.18	50.02 ± 0.03	42.48 ± 0.47	50.64 ± 0.02	48.05 ± 0.59	50.39 ± 0.04	39.31 ± 0.33	51.09 ± 0.03
9	0.0	64.20 ± 0.16	8.17 ± 0.02	62.14 ± 0.38	8.35 ± 0.0	63.43 ± 0.27	8.33 ± 0.03	62.19 ± 0.14	8.34 ± 0.02
	0.1	63.35 ± 0.31	8.19 ± 0.01	61.72 ± 0.41	8.38 ± 0.0	63.21 ± 0.28	8.36 ± 0.02	62.15 ± 0.25	8.38 ± 0.05
	0.3	61.16 ± 0.22	15.34 ± 0.02	60.53 ± 0.25	15.42 ± 0.0	60.73 ± 0.39	15.40 ± 0.03	59.78 ± 0.46	15.52 ± 0.05
Noise pattern = Bimodal									
3	0.0	19.34 ± 0.21	54.72 ± 0.03	11.21 ± 0.16	62.86 ± 0.0	11.19 ± 0.36	62.86 ± 0.03	1.05 ± 0.12	92.73 ± 0.04
	0.1	18.93 ± 0.27	55.18 ± 0.02	11.85 ± 0.48	62.89 ± 0.0	11.91 ± 0.31	62.87 ± 0.06	1.09 ± 0.17	95.37 ± 0.03
	0.3	17.82 ± 0.24	58.49 ± 0.03	4.12 ± 0.31	62.88 ± 0.0	11.32 ± 0.42	62.89 ± 0.04	1.04 ± 0.29	95.38 ± 0.03
9	0.0	64.89 ± 0.43	3.15 ± 0.03	64.11 ± 0.26	3.19 ± 0.0	64.15 ± 0.28	3.16 ± 0.04	1.03 ± 0.10	98.86 ± 0.02
	0.1	64.53 ± 0.14	3.15 ± 0.02	63.20 ± 0.43	3.18 ± 0.0	63.82 ± 0.37	3.17 ± 0.02	1.04 ± 0.09	98.91 ± 0.03
	0.3	63.98 ± 0.29	5.76 ± 0.02	62.27 ± 0.36	5.89 ± 0.0	62.48 ± 0.42	5.87 ± 0.02	1.03 ± 0.11	98.39 ± 0.02
Noise pattern = Flip									
3	0.0	11.43 ± 0.28	64.53 ± 0.03	8.88 ± 0.16	64.54 ± 0.0	8.95 ± 0.43	64.54 ± 0.02	10.02 ± 0.33	81.84 ± 0.02
	0.1	10.65 ± 0.42	64.53 ± 0.03	9.12 ± 0.28	64.55 ± 0.0	9.14 ± 0.37	64.54 ± 0.01	9.92 ± 0.10	81.84 ± 0.03
	0.3	9.98 ± 0.25	64.83 ± 0.02	6.14 ± 0.31	64.91 ± 0.0	8.99 ± 0.20	64.90 ± 0.01	9.76 ± 0.41	81.85 ± 0.09
9	0.0	25.68 ± 0.46	61.87 ± 0.02	22.06 ± 0.24	63.64 ± 0.0	22.63 ± 0.16	63.62 ± 0.05	9.38 ± 0.57	84.06 ± 0.04
	0.1	25.05 ± 0.17	61.95 ± 0.02	21.49 ± 0.40	63.65 ± 0.0	22.58 ± 0.39	63.63 ± 0.02	9.18 ± 0.21	84.07 ± 0.03
	0.3	24.65 ± 0.26	62.11 ± 0.01	21.11 ± 0.33	63.77 ± 0.0	22.38 ± 0.14	63.72 ± 0.04	9.17 ± 0.52	83.27 ± 0.03

0.3 because of getting stuck in the local optimum for the case of the biased dataset with a large number of classes to a specific class. In case of bimodal and flip noise patterns, EM achieves second best results in terms of accuracy and error rate.

5.5. CONCLUSION

Motivated by the need for accurate data labeling of crowd workers and using the provided labels for training DNNs, we design an iterative method for label aggregation and training DNN together. The prior art performs label aggregation and training classifier in two separate processes. We propose LABNET that considers aggregation and training in contact with each other. In our model, the classifier extracts the prior knowledge for passing to the aggregation algorithm. Also, the estimated correct labels by aggregation algorithm are used to train the classifier. In addition, we design an algorithm to decide when DNN needs to be trained through the aggregation algorithm iteration. Compared to the baselines, LABNET outperforms in most scenarios, especially for in challenging scenarios with large number of classes.

Despite its strengths, the proposed method has some limitations. One limitation is that we only consider one type of DNN for image classification, leaving the impact of other network architectures unexplored. Furthermore, the classifier does not incorporate any robust mechanisms, highlighting a potential area for future research to explore the integration of robust DNNs and aggregation algorithms capable of addressing label noise, as well as their collaborative effects. In LABNET, we demonstrate that this collaborative approach can significantly enhance the robustness of both the DNN and the label aggregator. Therefore, adding additional robustness mechanisms to both components could further improve overall robustness.

6

ROBUST MULTI-LABEL LEARNING

*In this chapter, we tackle the challenging and realistic problem of noisy multi-label classification, where each data sample is associated with a set of labels that may suffer from errors or missing annotations. The goal is to enhance the accuracy of multi-label classification models in the presence of noisy labels. To achieve this objective, we propose two innovative methods, Trusted Loss Correction for Multi-Label Learning (*TLCM*), and Multi-Label Loss Correction against Missing and Corrupted Labels (*MLLSC*), that aim to improve the performance of multi-label classifiers when dealing with label noise.*

TLCM *leverages partial data supervision to estimate the noise corruption matrix and address the challenges posed by noisy and imbalanced label distributions in multi-label learning. By utilizing a small fraction of trusted data, we estimate the extent of label corruption and effectively handle label imbalance, resulting in enhanced accuracy for multi-label classifiers.*

On the other hand, MLLSC *introduces a robust loss function specifically designed to mitigate the negative impact of noisy labels in multi-label classification. Unlike traditional loss functions, MLLSC does not rely on additional data or expert supervision. Instead, it incorporates robustness directly into the loss function, enhancing the model's resilience to label noise and improving performance even when clean or expert-labeled data is unavailable.*

By designing TLCM *and* MLLSC*, we provide comprehensive solutions for effectively addressing noisy labels in multi-label classification. These approaches leverage partial data supervision and a robust loss function to handle label corruption and missing annotations, resulting in significantly improved accuracy and reliability for multi-label classifiers.*

Empirical evaluations on real-world vision and object detection datasets, including MS-COCO, NUS-WIDE, and MIRFLICKR, demonstrate the effectiveness of our methods under medium and high corruption levels, outperforming state-of-the-art multi-label classifiers, ASL, and MPVAE, by a substantial margin in terms of mean average precision (mAP) points.

6.1. INTRODUCTION

This chapter of the thesis delves into the challenging and realistic problem of noisy multi-label classification, where each data sample is associated with a set of labels that may contain errors or missing annotations. The main goal is to extend the robustness of multi-class deep neural network (DNN) methods to multi-label classifiers, with the ultimate objective of enhancing the accuracy of multi-label classification in the presence of noisy labels.

Real-world datasets, often collected from diverse resources like social media and web search engines, are prone to label noise issues due to crowd-based annotation processes [172], [173]. In multi-label learning, where each sample can have multiple objects and complex object-label mappings, label noise becomes an even more significant obstacle to learning accurate classifiers [174]. Such datasets may contain both false-positive and false-negative labels, making the problem even more intricate [175].

Label noise and its impact on the performance of DNNs have been widely studied in single-label classification [172], [176], [177]. It is shown that DNNs can overfit to noise degrading their performance significantly [178] due to the memorization effect [179]. In multi-label learning, where each sample may include several corrupted labels as shown in Figure 6.1(a), label noise becomes a more consequential obstacle to learn accurate classifiers [174]. We empirically evaluate the effect of label corruption by injecting different noise levels on the MS-COCO dataset[1] used to train a state-of-the-art multi-label classifier, i.e., ASL [38]. In Figure 6.1(b), one can clearly see that the mean Average Precision (mAP) significantly degrades with increasing noise levels.

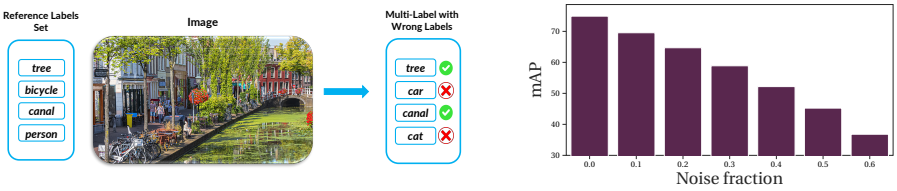
Also, from another perspective, the focus of prior art in multi-label learning is on missing labels [180], [181] foregoing the effect of wrong labels. In reality each image can contain some correct labels (true-positives), miss some others (false-negatives), and the annotator can add some wrong labels (false-positives). Hence, different from previous work, we consider annotators that can simultaneously produce missing labels (marked red Figure 6.2(a)) and wrong labels (marked orange in Figure 6.2(a)) for each image.

We empirically evaluate the separate and combined impact of false-positives and false-negatives on a multi-label classifier using binary cross-entropy loss by injecting missing and corrupted labels in the MIR-FLICKR dataset. In Figure 6.2(b), one can clearly see that the mean Average Precision (mAP) significantly degrades in the case of false-positives only and false-negatives only, but even more in the presence of both. Due to the memorization effect [182], [183], all curves in Figure 6.2(b) first raise (learn correct labels) then degrade (overfit to wrong labels) in terms of mAP.

With both missing and corrupted labels the mAP of Binary Cross Entropy (BCE) is reduced from 75.73% (highly curated dataset) to 54.91%. For real world applications it is thus vital to have loss functions robust to concurrent false-positive and false-negative labels.

Previous research in multi-label learning has focused on addressing missing labels, neglecting the simultaneous presence of both false-positive and false-negative labels [180], [181]. To tackle these challenges, we propose two innovative methods, namely Trusted Loss Correction for Multi-Label Learning (TLCM) and Multi-Label Loss Correction against

[1] The details of noise injection process can be found in Section 6.5.1



(a) Example of image with wrong labels

(b) Impact of increasing wrong label ratios

Figure 6.1: Wrong labels and their impact in multi-label classification.

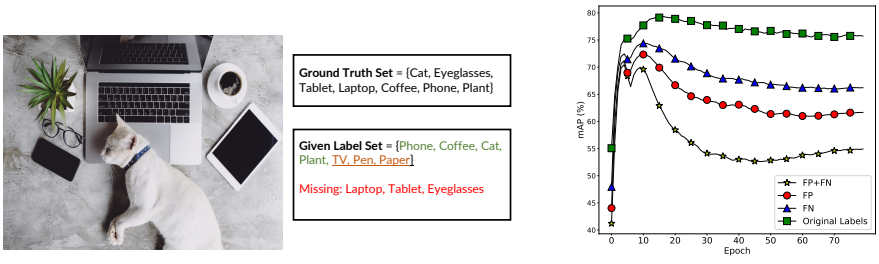
Missing and Corrupted Labels (MLLSC).

TLCM leverages partial data supervision and involves estimating the noise corruption matrix, addressing label imbalance issues commonly encountered in multi-label learning scenarios. On the other hand, MLLSC introduces a robust loss function that directly incorporates resilience to label noise, making it suitable for both missing and corrupted labels without requiring additional expert supervision.

To cope with label corruption in multi-label learning, we first propose a novel approach (TLCM) which uses gold loss correction leveraging a small fraction of trusted (gold) data for estimating the label corruption matrix. Although the gold loss correction has been used effectively against single-label corruption [172], noisy multi-label data needs an accurate method to estimate the corruption matrix handling the label mapping complexity and the label unbalance difficulty. We overcome these challenges by taking advantage of trusted single-labels to regulate the estimation of the multi-label corruption matrix. In other words, we introduce single-label regulators to correct the captured matrix on the miss classification behavior of the trained model on noisy multi-label samples. It is worth mentioning, having a small amount of trusted data, i.e. clean single-label and multi-label data, is common practice because we use clean label data in the validation and testing process. Therefore, it is possible to curate a small amount of training data. To the best of our knowledge, our method is the first that estimates the label corruption matrix in multi-label classification considering the labels dependence and unbalance.

Furthermore, to cope with simultaneous missing and corrupted labels, we design MLLSC that distinguishes the false-positive (false-negative) from true-positive (true-negative) by using the knowledge of a multi-label classifier. We use the predicted probability for each label of a DNN as a confidence value and as an indicator for false or true positive (negative) detection. After that, we compute the suitable loss based on being true/false positive (negative) since the loss value calculation for positive labels differs from negative labels in multi-label learning to counter label imbalance. Unlike methods that require a small amount of correct labels [49], [82], [117], this efficient and effective loss correction works properly without using any ground truth labels against both missing and corrupted labels. Furthermore, our proposed method can be applied to different kinds of multi-label loss functions, e.g. BCE, Focal [37] and ASL [38] to make them robust against corrupted and missing labels. MLLSC protects the underlying loss function from the negative impact of missing and corrupted labels with only a slight modification.

The combination of TLCM and MLLSC provides comprehensive solutions for effec-



(a) Example of image with corrupted and missing labels

(b) Impact of false-positive (corrupted) and false-negative (missing) labels on DNN performance with BCE loss

Figure 6.2: Wrong labels and their impact in multi-label classification.

tively addressing noisy labels in multi-label classification. TLCM leverages partial data supervision through the estimation of the label corruption matrix, while MLLSC incorporates robustness directly into the loss function, enabling enhanced performance even in the absence of clean or expert-labeled data. To demonstrate the effectiveness of our proposed methods, we conduct extensive empirical evaluations on various multi-label datasets, such as MS-COCO, NUS-WIDE, and MIRFLICKR, under different noise levels. We evaluate the performance of TLCM and MLLSC against state-of-the-art classifiers like ASL and MPVAE [38], [184]. The contributions of this chapter can be summarized as follows:

- We propose two innovative methods, TLCM and MLLSC, to address noisy labels in multi-label classification, leveraging partial data supervision and a robust loss function, respectively.
- The combination of TLCM and MLLSC provides a comprehensive solution for effectively handling label corruption and missing annotations, enhancing the accuracy and resilience of multi-label classifiers.
- Through extensive empirical evaluation, we demonstrate the superior performance of our approach compared to state-of-the-art methods on various multi-label datasets under different noise levels, establishing its effectiveness in real-world applications.

6.2. RELATED WORK

Multi-label classification is a well-studied problem [38], [185], [186] across a wide range of learning applications, e.g., object detection [187], [188], speech recognition [189], [190], natural language processing [191], and image classification [37], [38]. However, all of these methods perform well under the assumption of clean and complete labels for each training sample. Multi-label classifiers robust against noisy labels have recently received attention from literature compared to robust single-label classifiers. We first investigate learning methods addressing the multi-label classification task, and then study the methods robust against missing and noisy label data.

6.2.1. MULTI-LABEL LOSS FUNCTIONS

In multi-label learning, designing a loss function demands to take into account the label imbalanced label issue because, in practice, the impact of negative (missing) labels is higher than positive (present) ones [38]. The classic loss function commonly used in multi-label learning is Binary Cross-Entropy (BCE). BCE is agnostic to the imbalance issue and weighs negative and positive labels equally. Focal loss [37] tries to solve the imbalance problem by using different weights for negative and positive labels in the loss function. ASL [38] computes the weights asymmetrically by shifting the label probability to ensure no loss for negative labels with very low probability. Query2Label [185] is a transformer based multi-label classifier that leverages decoder structures to query the presence of certain labels.

6.2.2. ROBUST MULTI-LABEL LOSS FUNCTIONS

The standard multi-label methods only consider the label imbalance issue foregoing any corruption in the training data. Hence, they cannot perform well in the presence of data with label noise, i.e. missing or corrupted labels [180]. The label corruption can be categorized into three different scenarios. First, false-negative labels that represent the missing label problem [181], [192], [193]. Second, false positive labels which represent wrongly added labels leading to partial label learning [194]–[196]. Finally, the last and most complex scenario considers the coincident presence of both false negative and false positive labels [184].

[180] proposes a loss re-weighting method for negative labels to solve the missing label problem termed Hill. This method does not take into account the false positive labels. In addition [180] introduces a self-paced loss correction (SPLC) method using the confidence value of the model under training to regularize the negative and positive labels. Again, SPLC cannot handle false positive labels. Role [181] considers the scenario in which only one positive label is provided. Role uses a combination of BCE loss and loss regularization based on the expected positive labels for each sample. The drawback of Role is that the single positive label must be correct, and the average number of positive labels for each data sample must be known. The Multivariate Probit Variational AutoEncoder (MPVAE) [184] has been observed to remain robust against a median level of false positive and false negative labels. It is worth mentioning that the primary goal of MPVAE is not robustness against noisy labels.

In contrast to the above, TLCM aims to leverage single-label regulators together with a small fraction of trusted data to avoid overfitting to noisy labels in multi-label classification. also, MLLSC is a robust multi-label learning method that leverages the knowledge of the trained model during the learning process to avoid overfitting the loss to false-positive and false-negative labels.

6.3. TLCM METHODOLOGY

Consider the multi-label dataset $\mathscr{D}$ comprising N tuples $(\boldsymbol{x}, \tilde{\boldsymbol{y}})$ where $\boldsymbol{x} \subset \mathbb{R}^d$ denotes one sample with d features and $\tilde{\boldsymbol{y}} = [\tilde{y}_1, ..., \tilde{y}_k, ..., \tilde{y}_K]$ denotes the corresponding label vector over K classes with binary elements $\tilde{y}_k \in [0, 1]$. The label vectors are affected by noise, hence a $\tilde{y}_k$ can be the true (y_k) or a corrupted $(\bar{y}_k)$ label. We assume that a subset of

the data, i.e., the *gold* dataset $\mathscr{G} \subset \mathscr{D}$, can be trusted having no (or for practical purposes negligible) corrupted labels. We refer to the rest of the data with potentially corrupted labels as *silver* dataset $\mathscr{S} = \mathscr{D} - \mathscr{G}$. We define the *trusted fraction* as the ratio $\eta = \frac{|\mathscr{G}|}{|\mathscr{G}|+|\mathscr{S}|}$. Furthermore, we assume that a small dataset of clean single-label images $\mathscr{G}_S$ is available. We use $\mathscr{G}$ and $\mathscr{G}_S$ to estimate a $K \times K$ noise corruption matrix $\widehat{\boldsymbol{C}}$ characterizing the label noise. Each element $\hat{c}_{ij}$ of noise corruption matrix $\widehat{\boldsymbol{C}}$ represents the probability of a label of class i to be flipped into a label of class j, formally:

$$\hat{c}_{ij} = p(\tilde{y}_j = 1 \wedge \tilde{y}_i = 0 | y_j = 0 \wedge y_i = 1)$$

Finally, similar to related work, e.g. [38], we treat the multi-label classification problem as a series of binary classification tasks, one for each label class.

6.3.1. OVERVIEW OF TLCM

We propose a novel *Trusted Loss Correction for Noisy Multi-Label Learning* (TLCM) approach for noise resilient multi-label training. To address the challenge of noisy multi-labels, TLCM uses a gold loss correction which uses a small set of trusted (gold) samples to estimate the noise corruption matrix used to correct the model predictions during training. The gold loss correction has been shown to be highly effective against single-label noise. However, multi-label noise exacerbates the difficulty to estimate the noise corruption matrix. The fact that each sample can have an arbitrary number of labels leads to two main challenges. First, single-label classification assumes conditional independence of $\boldsymbol{y}$ given $\boldsymbol{x}$. In the single-label setting this assumption holds since $\boldsymbol{y}$ is deterministic in $\boldsymbol{x}$. Due to having an arbitrary number of labels and possible correlation between labels, this assumption does not hold in the multi-label setting which makes it hard to separate out which label or labels lead to a specific corrupted label. Second, the frequency distribution of labels across samples is harder to control. This leads to intrinsically higher label unbalances in multi-label datasets. For example, the popular and commonly used single-label CIFAR datasets have equal numbers of samples for each class. On the contrary, the ratio between the most (person) and least (toaster) frequent label in the MS-COCO dataset is 1197.4. This unbalance influences the noise. The more common a label, the higher the impact of that label on the noise characteristics.

Both the lack of conditional label independence and the label unbalance lead to poor estimates of multi-label noise corruption matrices. To overcome this and estimate the true multi-label noise corruption matrix, TLCM introduces the use of single-label regulators. These capture the miss classification behavior of a model trained on noisy data for specific labels and are used to correct the estimation of the multi-label noise corruption matrix.

Figure 6.3 presents an overview of the TLCM method for noise resilient multi-label classification. It includes three main steps. Step 1: we train a *silver* classifier $f(\cdot;\theta)$ on the noisy samples in $\mathscr{S}$. Step 2, the heart of TLCM, consists of two substeps. First, we capture the single-label noise characteristics in $\widetilde{\boldsymbol{C}}$ using f on the trusted single-label samples in $\mathscr{G}_S$. Second, we use f on the trusted samples in $\mathscr{G}$ to compute a multi-label noise corruption matrix $\widehat{\boldsymbol{C}}$ regularized by $\widetilde{\boldsymbol{C}}$ to counter the detrimental effect of multi labels on noise estimation. Step 3: we train the *gold* classifier $g(\cdot;\phi)$ on the samples from $\mathscr{S}$ with label predictions corrected via the noise corruption matrix $\widehat{\boldsymbol{C}}$, plus the samples from $\mathscr{G}$, to

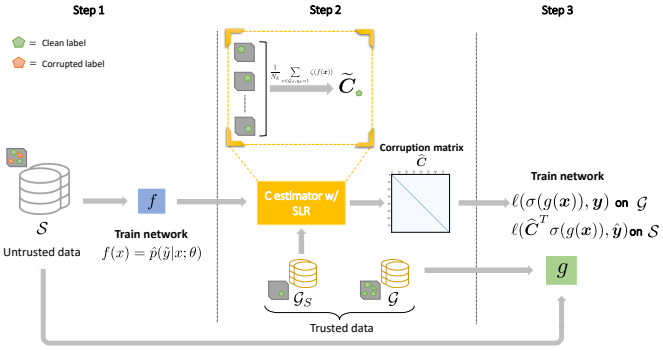


Figure 6.3: Overview of the training process for the Trusted Loss Correction for Noisy Multi-Label Learning (TLCM) mechanism.

maximize the impact of the trusted data.

6.3.2. NOISE CORRUPTION MATRIX ESTIMATION

***Silver* classifier**. Before starting the estimation of the multi-label corruption matrix, we need to train a *silver* classifier $f(\boldsymbol{x};\theta) = \widehat{p}(\tilde{\boldsymbol{y}}|\boldsymbol{x})$ on $\mathscr{S}$. Given the labels in $\mathscr{S}$ are potentially corrupted, f is not a reliable classifier for our final predictions. However, we can use f for our multi-label noise corruption matrix estimation.

It is worth mentioning that our estimation method for multi-label noise corruption matrix is independent of the loss and network types used to train $f(\cdot;\theta)$. Here, we consider asymmetric loss [38] due to its superior performance to counter the detrimental effect of irrelevant class labels in multi-label datasets. For each class (we omit the class index k for brevity) the loss is:

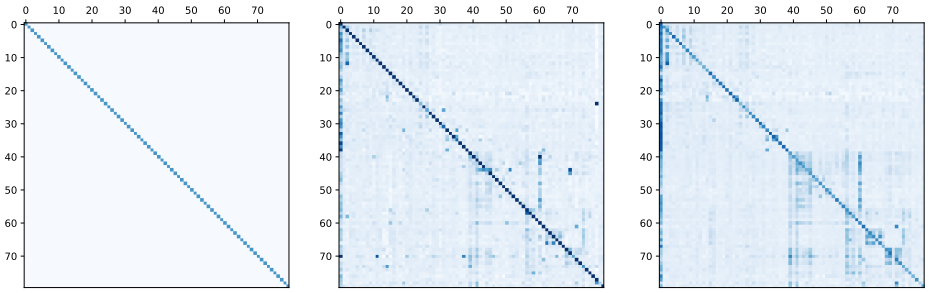
$$\mathscr{L}_{ASL} = \begin{cases} L_+ = (1-p)^{\gamma_+}\log(p), & y_k = 1 \\ L_- = (p_m)^{\gamma_-}\log(1-p_m), & y_k = 0 \end{cases}$$

where L_+ and L_- are the positive and negative loss parts used for relevant and irrelevant labels, respectively, p is the network output probability and γ_+, γ_- are the focusing parameters. The focusing parameters control the weights of positive and negative labels. Finally, $p_m = \max(p-m, 0)$ denotes the shifted probability where the margin m is a tunable hyper-parameter controlling the contribution of irrelevant labels introduced by [38].

Noise corruption matrix. Once we trained the *silver* classifier $f(\cdot;\theta)$ on $\mathscr{S}$ we can start the multi-label noise corruption matrix estimation. First, we use f on the samples in $\mathscr{G}_S$ to capture the single-label noise characteristics via the single-label noise corruption matrix $\widetilde{\boldsymbol{C}}$. For each label $k \in K$ the corresponding row of $\widetilde{\boldsymbol{C}}$ is given by the averaged softmax output of f for all samples $\boldsymbol{x} \in \mathscr{G}_S$ with $y_k = 1$:

$$\widetilde{\boldsymbol{C}}_{k\cdot} = \frac{1}{N_k} \sum_{x \in \mathscr{G}_S, y_k = 1} \zeta(f(\boldsymbol{x}))$$

where $\widetilde{\boldsymbol{C}} \subset \mathbb{R}^{K \times K}$, $\widetilde{\boldsymbol{C}}_{k\cdot}$ denotes the k^{th} matrix row, N_k the number of samples with $y_k = 1$,



(a) True (b) TLCM (c) GLC

Figure 6.4: The multi-label corruption matrix of the MS-COCO dataset with 40% symmetric noise (a), along with the estimated corruption matrices by TLCM (b) and GLC (c).

and $\zeta(\cdot)$ the softmax function. We scale the output via softmax to enhance the training since here we only consider single-label samples.

Next we use $\widetilde{\boldsymbol{C}}$ to regulate the label correlations from the multi-label samples in $\mathscr{G}$. To do this we compute the multi-label noise corruption matrix using f on the samples in $\mathscr{G}$ while regulating each time the output of f via $\widetilde{\boldsymbol{C}}$. For each label $k \in K$ the corresponding row of $\widehat{\boldsymbol{C}}$ is given by the averaged sigmoid output of f for all samples $\boldsymbol{x} \in \mathscr{G}$ having $y_k = 1$ regulated by the difference in the single-label corruption probabilities of label k and the other labels present in $\boldsymbol{x}$:

$$\widehat{\boldsymbol{C}}_{k\cdot} = \frac{1}{N_k} \sum_{(\boldsymbol{x},\boldsymbol{y})\in\mathscr{G}, y_k=1} \left(\sigma(f(\boldsymbol{x})) + \sum_{\forall l \neq k, y_l=1} \left(\widetilde{\boldsymbol{C}}_{k\cdot} - \widetilde{\boldsymbol{C}}_{l\cdot} \right) \right) \tag{6.1}$$

where $\sigma(\cdot)$ denotes the sigmoid function.

Figure 6.4 compares the multi-label corruption matrix estimated by TLCM for 40% symmetric label noise against the one injected –ground truth–, and the one estimated without regularization. The latter corresponds to equation (6.1) without the internal summation term for rows of $\widetilde{\boldsymbol{C}}$. The comparison highlights the benefits of single-label regularization. This can be seen from the darker, closer to the truth, diagonal values and the more pronounced difference with respect to the off-diagonal values. Algorithm 6 depicts our novel noise corruption matrix estimation method.

***Gold* classifier**. With the estimated multi-label noise corruption matrix $\widehat{\boldsymbol{C}}$, we finally train the robust *gold* classifier $g(\cdot;\phi)$. We correct labels of the samples in $\mathscr{S}$ via $\widehat{\boldsymbol{C}}$ while leveraging samples in $\mathscr{G}$ as is. The loss function follows as:

$$\begin{aligned} \ell &= \mathscr{L}_{ASL}(\widehat{\boldsymbol{C}}^T \sigma(g(\boldsymbol{x})), \hat{\boldsymbol{y}}), \quad && \forall x \in \mathscr{S} \\ \ell &= \mathscr{L}_{ASL}(\sigma(g(\boldsymbol{x})), \boldsymbol{y}) && \forall x \in \mathscr{G}. \end{aligned}$$

Algorithm 7 depicts our TLCM algorithm.

Algorithm 6: TLCM $\widehat{\boldsymbol{C}}$ estimation

Input : Noisy data $\mathscr{S}$, trusted single-label data $\mathscr{G}_S$, silver classifier f, Number of epoch E_{max}
Output : Estimated $\widehat{\boldsymbol{C}}$

for $e = 1, ..., E_{max}$ **do**
  for $(\boldsymbol{x}, \hat{\boldsymbol{y}}) \in \mathscr{S}$ **do**
    Train $f(\cdot;\theta)$ with $\ell = \mathscr{L}_{ASL}(f(\boldsymbol{x}), \hat{\boldsymbol{y}})$
  end
end
`/* Single-label noise matrix */`
Fill $\widetilde{\boldsymbol{C}} \in \mathbb{R}^{K \times K}$ with zeros
for $k = 1, ..., K$ **do**
  $N_k = 0$
  for $(\boldsymbol{x}, \boldsymbol{y}) \in \mathscr{G}_S, y_k = 1$ **do**
    $N_k \mathrel{+}= 1$
    $\widetilde{\boldsymbol{C}}_{k\cdot} \mathrel{+}= f_{soft}(\boldsymbol{x})$
  end
  $\widetilde{\boldsymbol{C}}_{k\cdot} = \frac{1}{N_k} \widetilde{\boldsymbol{C}}_{k\cdot}$
end
`/* Multi-label noise matrix */`
Fill $\widehat{\boldsymbol{C}} \in \mathbb{R}^{K \times K}$ with zeros
for $k = 1, ..., K$ **do**
  $N_k = 0$
  for $(\boldsymbol{x}, \boldsymbol{y}) \in \mathscr{G}$, $y_k = 1$ **do**
    $N_k \mathrel{+}= 1$
    Fill $\boldsymbol{r} \in \mathbb{R}^K$ with zeros
    for $l = 1, ..., K$, $l \neq k, y_l = 1$ **do**
      $\boldsymbol{r} \mathrel{+}= \widetilde{\boldsymbol{C}}_{k\cdot} - \widetilde{\boldsymbol{C}}_{l\cdot}$
    end
    $\widehat{\boldsymbol{C}}_{k\cdot} \mathrel{+}= \sigma(f(\boldsymbol{x})) + \boldsymbol{r}$
  end
  $\widehat{\boldsymbol{C}}_{k\cdot} = \frac{1}{N_k} \widehat{\boldsymbol{C}}_{k\cdot}$
end

Algorithm 7: TLCM

Input : Multi-label corruption matrix $\widehat{\boldsymbol{C}}$ (Algorithm 6), Untrained classifier g, Noisy data $\mathscr{S}$, trusted multi-label data $\mathscr{G}$, Number of epochs E_{max}
Output : Trained robust classifier g

```
for e = 1, ..., E_max do
    for (x, y) ∈ G do
        Train g(·; φ) with ℓ = L_ASL(σ(g(x)), y)
    end
    for (x, ŷ) ∈ S do
        Train g(·; φ) with ℓ = L_ASL(Ĉ^T σ(g(x)), ŷ)
    end
end
```

6.4. MLLSC METHODOLOGY

In this section, we first discuss standard losses in multi-label learning that are not equipped with any mechanism to deal with false-positive and false-negative labels. Next, we introduce our method called MLLSC, which is a robust loss function against missing and corrupted labels in multi-label learning. Furthermore, we show that our technique can be applied to all multi-label losses and improve their performances compared to the original ones.

6.4.1. PRELIMINARY

Let $\mathscr{D}$ be a multi-label dataset of pairs of features $\boldsymbol{x} \in \mathbb{R}^d$ and corresponding labels $\tilde{\boldsymbol{y}} \in [0,1]^K$ where K is the number of classes. The presence or absence of label of class k for instance i is represented by $\tilde{y}_k^i = 1$ and $\tilde{y}_k^i = 0$, respectively. Since we consider false-positive and false-negative labels in our problem, we use $\boldsymbol{y}^i$ to represent the ground truth, i.e., the correct label vector for instance i. $\tilde{\boldsymbol{y}}^i$ can contain both correct and corrupted class labels.

The aim is to classify an input instance using a deep neural network $f : \mathbb{R}^d \rightarrow [0,1]^K$ which is trained through the learning iteration. We define f to be a classifier with parameters $\boldsymbol{\theta}$ and sigmoid activation function at the last layer which denotes the probability of each label for input instance $\boldsymbol{x}_i$ as $\boldsymbol{p}_i = f(\boldsymbol{x}^i) = \frac{1}{1+e^{-\boldsymbol{x}^i}}$. The common loss function for multi-label classification is binary cross-entropy which is defined as

$$\ell = -\frac{1}{N}\sum_{i=1}^{N}[\sum_{k=1}^{K}(\tilde{y}_k^i \log\, p_k^i + (1-\tilde{y}_k^i)\log\,(1-p_k^i))] \tag{6.2}$$

where N denotes the number of samples in the training set. According to equation. 6.2, the positive and negative labels losses equal to $\log p_k^i$ and $\log(1-p_k^i)$, respectively. In the rest of the paper for simplicity we ignore the instance superscript i and consider the loss of each instance. Hence we have:

$$\mathscr{L} = -\sum_{k=1}^{K} (\tilde{y}_k \underbrace{\log p_k}_{\mathscr{L}_k^+} + (1-\tilde{y}_k) \underbrace{\log (1-p_k)}_{\mathscr{L}_k^-}) \tag{6.3}$$

The BCE loss does not provide any mechanism to handle the imbalance issue of multi-label data. Hence, Focal loss [37] is proposed to cope with the imbalance issue of negative and positive labels by weighting differently positive and negative losses. The Focal loss is defined as:

$$\mathscr{L}_{Focal} = -\sum_{k=1}^{K} \tilde{y}_k \underbrace{(\alpha_+ (1-p_k)^\gamma \log p_k)}_{\mathscr{L}_k^+} + (1-\tilde{y}_k) \underbrace{(\alpha_- p_k^\gamma \log (1-p_k)))}_{\mathscr{L}_k^-} \tag{6.4}$$

where $\alpha_-, \alpha_+ \in [0,1]$ are weighting factors for balancing the impact of positive and negative labels on the training loss. γ represents the focusing parameter that controls the model focus on hard and easy instances during training. For instance, by setting $\gamma > 0$, samples with $p_k << 0.5$ are the easy negative labels, and their loss contributions are reduced.

To weight positive and negative labels differently, ASL [38] recently proposed asymmetric multi-label loss to diminish the positive-negative imbalance issue, which is introduced as:

$$\mathscr{L}_{ASL} = -\sum_{k=1}^{K} \tilde{y}_k \underbrace{((1-\mathscr{P}'_{k,m})^{\gamma_+} \log \mathscr{P}'_{k,m})}_{\mathscr{L}_k^+} + (1-\tilde{y}_k) \underbrace{(\mathscr{P}'^{\gamma_-}_{k,m} \log (1-\mathscr{P}'_{k,m}))}_{\mathscr{L}_k^-} \tag{6.5}$$

where γ^- and γ^+ are focus parameters for negative and positive labels, respectively, which control the weights of negative and positive labels and $\gamma^- > \gamma^+$. Also, $\mathscr{P}'_{k,m} = \max(p_k - m, 0)$ denotes the shifted probability where m is the probability margin.

6.4.2. LEARNING FROM NOISY LABELS

A multi-label classifier when trained with corrupted and missing labels under one of the mentioned losses, e.g., BCE, Focal, and ASL, tends to overfit the false-positive and false-negative labels and model accuracy drops significantly. As shown in $\mathscr{L}_k^+$ and $\mathscr{L}_k^-$ (see equation. 6.4), there is no loss correction mechanism to alleviate the impact of false-positive and false-negative labels. We propose MLLSC as a robust multi-label loss correction method that can be applied to all kinds of multi-label loss functions to provide a shield against corrupted and missing labels. MLLSC is inspired by the design of SPLC [180], which is a loss correction method only for the problem of missing labels. First, we consider the BCE loss as a Maximum Likelihood (ML) estimation problem. The idea of using BCE for multi-label learning (without any corruption and missing labels) can be seen as an ML approximation problem under the Bernoulli distribution:

$$P(\boldsymbol{y}) = \prod_{k=1}^{K} p_k^{y_k} (1-p_k)^{1-y_k} \tag{6.6}$$

where P is the likelihood for each instance i. By taking log and finding the optimal value of equation. 6.6, the BCE loss is inferred. Now, we consider equation. 6.6 in presence of false-positive and false-negative labels. Let $q_k \in [0,1]$ be the probability of the corresponding class k being truly positive, and $q'_k \in [0,1]$ is the probability of the corresponding class k being truly negative. In other words, q_k denotes the probability that the ground truth and given labels both assign the same positive labels and q'_k is the probability of a true negative means ground truth and given labels are both negative labels. Hence the probability of being false-positive and false-negative are $1-q'_k$ and $1-q_k$, respectively. Then, we can propose the likelihood of each instance for the new setting as the following:

$$P(\tilde{\boldsymbol{y}}, \boldsymbol{s}, \boldsymbol{t}) = \prod_{k=1}^{K} \{(q_k p_k)^{s_k}((1-q'_k)(1-p_k))^{(1-s_k)}\}^{\tilde{y}_k} \times \{(q'_k(1-p_k))^{t_k}((1-q_k)p_k)^{(1-t_k)}\}^{1-\tilde{y}_k} \tag{6.7}$$

where $s_k \in \{0,1\}$ is the indicating variables of true and false positives, and $t_k \in \{0,1\}$ is the indicating variables of true and false negatives. Besides, $\boldsymbol{s}$ and $\boldsymbol{t}$ are the vectors of indicating variables, i.e., s_k and t_k. Hence we can derive the optimal loss from the likelihood in equation. 6.7:

$$\begin{cases} \mathscr{L}_k^+ = s_k \; log(p_k) + (1-s_k) \; log(1-p_k) \\ \mathscr{L}_k^- = t_k \; log(1-p_k) + (1-t_k) \; log(p_k) \end{cases} \tag{6.8}$$

In equation. 6.8, in the case of positive label loss $\mathscr{L}_k^+$, if the label is a true positive $s_k = 1$ and the loss function behaves this sample same as for positive labels. If the label is a false positive, i.e., $s_k = 0$, the loss only considers the negative label term $log\,(1-p_k)$ because this label is corrupted thus the original label was negative. The same scenario happens the other way around for the case of negative labels loss $\mathscr{L}_k^-$. When the label is true negative $t_k = 1$, the negative BCE loss is calculated, while if the label is false negative $t_k = 0$, then the positive label BCE loss is computed.

To compute our proposed loss in equation. 6.8, we need to determine $\boldsymbol{s}$ and $\boldsymbol{t}$ for distinguishing false and true positive/negative labels. Since the given label set $\tilde{\boldsymbol{y}}$ is the only available information, and the value of $\boldsymbol{s}$ and $\boldsymbol{t}$ are not known, we leverage the knowledge of the deep neural network itself. We use the predicted label probabilities as a proxy of the model's certainty of each label. Then, we define two thresholds $\tau, \tau' \in (0,1)$ to detect whether a label should be considered a true positive or false positive and a true negative or false negative, respectively. After training $f(\cdot, \boldsymbol{\theta})$ at each step, we use the model prediction probability $p_k = f_k(\cdot, \boldsymbol{\theta})$ as the confidence value of the DNN for each specific class k based on the input instance. For a given positive label $y_k = 1$, if $p_k > \tau$ it would be a true positive label, otherwise it is a false positive. Also, for the case of $y_k = 0$, the true negative and false negative can be determined by $p_k < \tau'$ and $p_k > \tau'$, respectively. Hence we can write the new robust losses for positive and negative labels based on new thresholds τ, τ' as follows

$$\begin{cases} \mathscr{L}_k^+ = \mathbb{1}(p_k > \tau) \; log(p_k) + (1 - \mathbb{1}(p_k > \tau)) \; log(1-p_k) \\ \mathscr{L}_k^- = \mathbb{1}(p_k < \tau') \; log(1-p_k) + (1 - \mathbb{1}(p_k < \tau')) \; log(p_k) \end{cases} \tag{6.9}$$

where $\mathbb{1}(\cdot)$ denotes the indicator function. equation. 6.9 shows the general form of

positive and negative losses when we use BCE as the base multi-label loss function.The complete form of MLLSC for multi-label classification is

$$\mathscr{L}_{\text{MLLSC}} = -\sum_{k=1}^{K} \tilde{y}_k \mathscr{L}_k^+ + (1-\tilde{y}_k)\mathscr{L}_k^- \tag{6.10}$$

where $\mathscr{L}_k^+$ and $\mathscr{L}_k^-$ are the proposed losses in equation. 6.9 which are equipped with false-negative and false-positive detection mechanisms to select the proper loss term and alleviate the impact of coincident missing and corrupted labels.

The strength of MLLSC is the easy applicability to different multi-label loss functions. equation. 6.10 and equation. 6.9 are based on the BCE loss. Applying our method to Focal loss gives:

$$\begin{cases} \mathscr{L}_k^+ = \mathbb{1}(p_k > \tau)\ \alpha_+ (1-p_k)^\gamma log(p_k) + (1-\mathbb{1}(p > \tau))\ \alpha_- p_k^\gamma log(1-p_k) \\ \mathscr{L}_k^- = \mathbb{1}(p_k < \tau')\ \alpha_- p_k^\gamma log(1-p_k) + (1-\mathbb{1}(p_k < \tau'))\ \alpha_+ (1-p_k)^\gamma log(p_k) \end{cases} \tag{6.11}$$

with the same hyper-parameters defined in equation. 6.4 and equation. 6.9.

Inspired by ASL [38] and SPLC [180], to emphasize more the uncertain classes, i.e., $0.4 < p_k < 0.6$, we make use of Focal margin loss for positive labels instead of Focal loss due to its superior performance in multi-label classification [37], [180]. More specifically, we define MLLSC loss based on Focal margin loss as

$$\begin{cases} \mathscr{L}_k^+ = \mathbb{1}(p_k > \tau)\ (1-\mathscr{P}_{k,m})^\gamma log(\mathscr{P}_{k,m}) + (1-\mathbb{1}(p_k > \tau))\ \mathscr{P}_{k,m}^\gamma log(1-\mathscr{P}_{k,m}) \\ \mathscr{L}_k^- = \mathbb{1}(p_k < \tau')\ p_k^\gamma log(1-p_k) + (1-\mathbb{1}(p_k < \tau'))\ (1-p_k)^\gamma log(p_k) \end{cases} \tag{6.12}$$

where $\mathscr{P}_{k,m} = f_k(\boldsymbol{x} - m)$ and m is the margin parameter. For the case of $m = 0$, the Focal margin loss collapses into the Focal loss. It is worth mentioning that throughout this paper we have used Focal margin loss for all experiments. To initialize the knowledge of the DNN, we initially train it for two epochs with uncorrected Focal loss, before switching to the MLLSC corrected loss (equation. 6.11). This is because DNNs learn easy (correct) labels first, before overfitting to missing and corrupted labels due to the memorization effect [49], [63], [179]. Switching losses allow us to leverage the easy labels for the initial epochs and afterward actively cope with false-negative and false-positive labels.

6.5. TLCM EVALUATION

6.5.1. EXPERIMENT SETUP

Datasets. We evaluate TLCM using three popular multi-label vision and object detection datasets, i.e. MS-COCO [197], NUS-WIDE [198] and MIRFLICKR [199].

- **MS-COCO** is a popular real-world dataset of common objects in context widely used for evaluation of multi-label classification. The training and validation datasets contain 82K and 40K images, respectively. Each image is tagged on average with 2.9 labels belonging to 80 classes.

- **NUS-WIDE** is a web image dataset with 269.6K manually annotated samples in 81 visual concepts. In our experiments, since some URLs have been eliminated, we were able to download 218K images and split into 186K as training and 32K testing samples. Images have 2.9 labels on average.
- **MIRFLICKR** is an open image collection with $25K$ images annotated in 38 categories. The training and testing sets include 23,300 and 1,700 images, respectively. Here, the average labels per image is 8.9.

To explicitly test TLCM in the more challenging multi-label setting, we remove all single label images from both the training and validation datasets. This does not only elicit a more reliable evaluation, but it also allows to collect single-label samples to construct $\mathscr{G}_S$. After filtering the number of images per class label varies from 1, for unpopular classes, to 1,234, for the most popular class with an average of 210.2 images per class for MS-COCO and from 4 to 40,026 and from 112 to 9,613 for NUS-WIDE and MIRFLICKR, respectively. More single-label samples allow to estimate more accurate regulators which in turn leads to a more robust classifier. Finally, we split the training data into a *gold* ($\mathscr{G}$) and a *silver* ($\mathscr{S}$) datasets. As the base, we use 10% as gold data, leading to 6.5K clean samples and 58.7K samples injected with noisy labels for MS-COCO. Similarly we have 7.5K clean and 67.5K noisy samples for NUS-WIDE and 2K clean and 18K noisy samples for MIRFLICKR.

Label Noise. Label noise in multi-label data is more complex than in a single-label context since each sample has an arbitrary number of labels. We follow previous works [200], [201] and inject symmetric noise, but with an extra step. Specifically, we select a fraction η, i.e. the noise ratio, of labels and flip them to another class with uniform probability. This corresponds to a noise corruption matrix $\boldsymbol{C}$ having elements c_{ij} as follows:

$$c_{ij} = \begin{cases} 1-\eta & \text{if } i = j \\ \dfrac{\eta}{K-1} & \text{if } i \neq j \end{cases}$$

In order to ensure wrong label injection, we test whether or not the new label is already associated with the image. If it does, we repeatedly elect a new label until we select one which is not yet present. In order to evaluate how robust TLCM is to noise, we test our method against multiple noise ratios –from 0% to 60%.

Evaluation Metrics. For a comprehensive and reliable evaluation, we follow conventional settings and report the following metrics: mean average precision (mAP), average per-class F1 (CF1) and average overall F1 (OF1). These metrics have been widely used in literature to evaluate multi-label classification [38], [202], [203] and have been shown to dramatically decrease with label noise [174]. Note that only the training set is affected by noise, whereas the evaluation metrics are computed on the clean testing set.

DNN Architecture. As base architecture for the DNN, we use TResNet [204]. TResNet network is a high performance GPU-dedicated architecture based on ResNet50 designed to increase the model prediction performance without increasing training or inference time. The TResNet network is pre-trained on the ImageNet-21K dataset for better generalizability and increased prediction accuracy [205]. In particular we use the TResNet-M version with input resolution 224 for MS-COCO and MIRFLICKR datasets. We use the TResNet-L version with input resolution 448 for NUS-WIDE dataset. The encoder and decoder use the structure from [184], i.e, 3-layer fully connected neural networks with

ReLU activation function. Moreover, we set the hyper-parameters to the default values provided in [184] for each dataset.

Baseline. We compare the performance of TLCM against MPVAE [184] which is a noise resilient multi-label model, and ASL [38] which is one of the state-of-the-art multi-label learning algorithms.

- **MPVAE** [184] proposes a variational autoencoder based method that encodes the features and labels to Gaussian subspaces and then decodes the samples from the subspaces into multivariate probit to predict the image labels.
- **ASL** [38] introduces new asymmetric loss for multi-label classification to reduce the impact of irrelevant labels in focal loss [206] and balance the weights of relevant and irrelevant labels.

For a fair comparison, we test ASL, MPVAE and TLCM on the same datasets with the same label noise. TLCM assumes access to a small subset of clean samples. Thus the only additional knowledge of our method is which labels are trusted, i.e., belonging to the small golden dataset $\mathscr{G}$, and which labels are potentially corrupted.

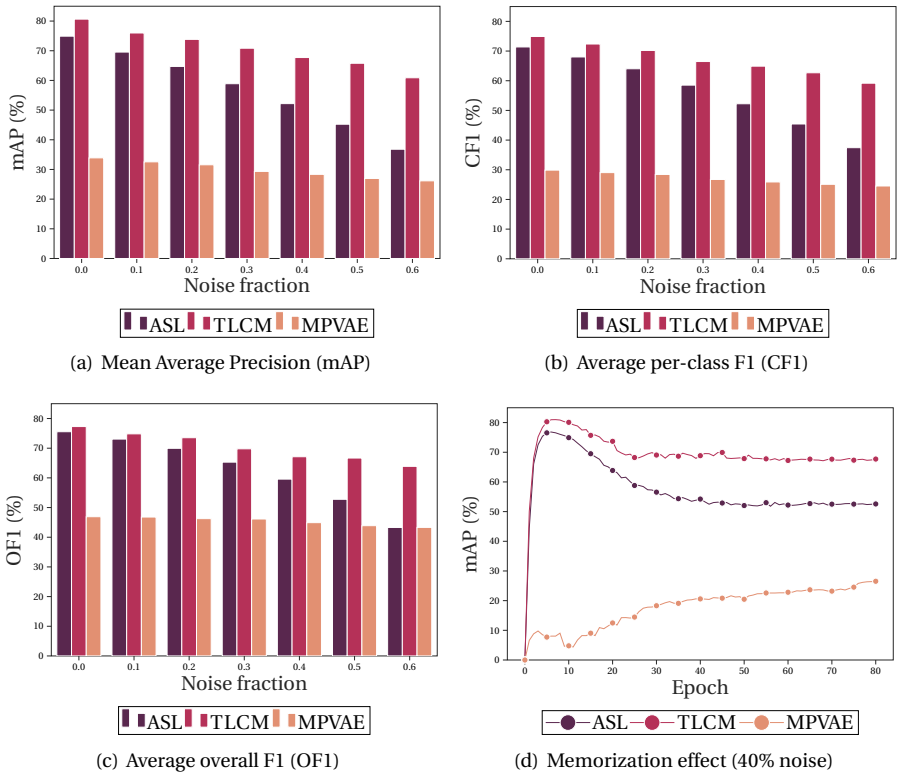
Implementation Details. We use PyTorch v1.9.0 for all the methods, and the default parameters provided in [38], [184] except that we always take the last trained model due to the *memorization effect*. The number of training epochs is an important parameter for a reliable evaluation, especially in a noisy setting. DNNs are shown to present the so-called *memorization effect* [179], [207], [208] benefiting in general from this factor to achieve a better prediction performance in atypical samples. However, [209] suggests that with noisy data, DNNs prioritize learning simple patterns first. From preliminary experiments we see that 80 epochs are enough for the learning to stabilize.

6.5.2. RESULTS

In this subsection we empirically compare the performance of TLCM to two rivals, i.e., ASL and MPVAE under 0.0 to 0.6 symmetric noise ratios. We aim to show the effectiveness of our TLCM in robustly learning from noisy data.

Figure 6.5 shows the comparison results. The performance of all systems decreases under increasing noise levels, but TLCM is significantly more robust. In terms of mAP TLCM consistently outperforms ASL and MPVAE for all noise ratios (see Figure 6.5(a)). After TLCM the best mAP performance is achieved by ASL followed finally by MPVAE, under all noise levels. Without noise, i.e. $\eta = 0.0$, the mAP of TLCM and ASL are extremely close, but MPVAE performance is still significantly lower. Since MPVAE uses a Variational Auto Encoder based method and the network architecture is less deep and complex than ASL and TLCM, the MPVAE can not perform well. It is worth mentioning that the network architecture is taken from [184]. ASL's performance drops an average of 5.34% points with each 10% noise, while TLCM's performance decreases by only 3.29% points. Also the mAP degradation for MPVAE is 1.28% points. Under severe noise, i.e. $\eta = 0.6$, the gap between TLCM and ASL is more than 24% points and only 14.01% points worse than without noise. In comparison ASL drops by 38.1% points from 0.0 to 0.6 noise ratio. This shows that TLCM is robust even to high noise levels. Similar results apply for both CF1 and OF1, see Figure 6.5(b) and Figure 6.5(c), respectively. Even if ASL is slightly close to TLCM in the no noise case, the performance quickly degrades with additional noise. At 60% noise TLCM is better than ASL by 21.73% and 20.59% points for CF1 and OF1, respectively. Moreover,

the CF1 and OF1 for MPVAE slightly degrade from 29.0% and 46.8% to 24.5% and 43.3%, respectively, over increasing noise ratios.



(a) Mean Average Precision (mAP)

(b) Average per-class F1 (CF1)

(c) Average overall F1 (OF1)

(d) Memorization effect (40% noise)

Figure 6.5: Mean Average Precision (mAP) (a), Average per-class F1 (CF1) (b), and Average Overall F1 (OF1) (c) vs. noise fraction of TLCM, ASL, and MPVAE on MS-COCO with symmetric label noise using 10% of trusted data. (d) Specifically explores the mAP (%) to underscore the Memorization effect under 40% symmetric label noise.

To reliably assess the correctness of TLCM, we also investigate the observed memorization effect for TLCM (depicted in Figure 6.5(d)). Both TLCM and ASL follow the same trend. First they learn the easy patterns, achieving a high accuracy after just a few epochs. However, afterwards the performance slowly degrades over training effort and finally stabilizes after approximately 60 epochs. But the mAP trend for MPVAE is different. It slightly increases over the training epoch and then reaches a steady-state which however is significantly lower than TLCM and ASL. The figure clearly shows the advantage of TLCM over ASL and MPVAE in the different levels at which they plateau. Moreover, one can observe that TLCM has a slight delay in learning at the beginning of the training, i.e. TLCM peaks at epoch 10, while ASL at epoch 6. This observation indicates that TLCM does not help in terms of learning speed nor in reaching a higher performance during training, but by preventing overfit to the noisy labels. This makes the DNN more resistant to wrong label information. Furthermore, this suggests that our

method can also be applied to other existing classifiers and domains.

Table 6.1 summarizes the mAP of TLCM, ASL and MPVAE for the NUS-WIDE and MIRFLICKR datasets under 0.0, 0.3, and 0.6 label noise ratios. As illustrated in Table 6.1, the mAP of TLCM, ASL and MPVAE for NUS-WIDE under all noise ratios are significantly lower than for MS-COCO (as well as MIRFLICKR, see Table 6.1). NUS-WIDE includes images with bigger sizes, one additional class and higher label imbalance. This makes NUS-WIDE a more complex dataset for multi-label classification. TLCM is still able to provide robustness for training against label noise and beats both baselines. ASL and TLCM perform similarly at 0.0 noise ratio but ASL degrades faster under increasing noise. MPVAE is less affected by increasing noise but plateaus at a significantly lower level. By using only 10% trusted samples, TLCM obtains 38.7% mAP under severe label corruption, i.e., $\eta = 0.6$, while ASL and MPVAE only achieve 22.6% and 14.2% mAP, respectively. Across the different noise ratios TLCM achieves on average 7% and 16% points higher mAP compared to ASL and MPVAE, respectively. Overall we conclude that while NUS-WIDE is a more challenging dataset the trends are similar to MS-COCO.

Table 6.1: Mean Average Precision (mAP) (%) of TLCM compared to two baselines on noisy NUS-WIDE and MIRFLICKR corrupted with 0.3 and 0.6 noise ratios (η)

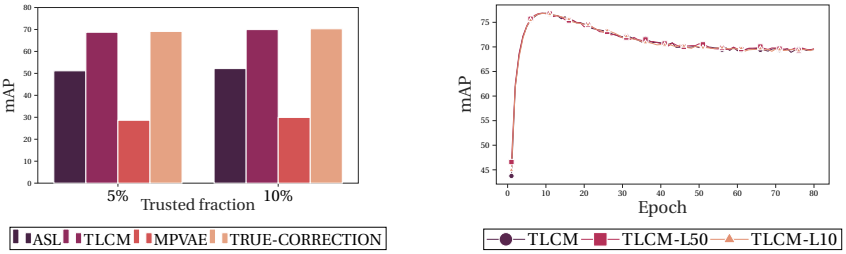
Method	***Percent Trusted***	***NUS-WIDE***			***MIR-FLICKR***		
		$\eta = 0.0$	$\eta = 0.3$	$\eta = 0.6$	$\eta = 0.0$	$\eta = 0.3$	$\eta = 0.6$
ASL	-	60.15	43.59	22.64	80.85	63.06	35.26
MPVAE	-	17.55	15.67	14.21	41.01	39.03	35.68
TLCM	5	59.91	46.62	34.11	80.90	65.72	46.37
	10	60.23	48.56	38.68	80.88	67.34	53.86
	15	60.66	48.79	38.89	80.91	68.83	54.05

Furthermore, Table 6.1 analyzes the performance of TLCM compared to ASL and MPVAE on MIRFLICKR. This dataset is less challenging than MS-COCO and NUS-WIDE because it only uses about half, i.e. 38, the number of label classes. As a consequence all three multi-label classification methods achieve higher mAP compared to two other datasets. Again TLCM is robust to noise and the best of the three methods achieving on average across all noise levels 7.6% and 28.8% points higher mAP than ASL and MPVAE, respectively, using 10% trusted samples. While the general trends are similar to the previous two datasets, interestingly, for $\eta = 0.6$ MPVAE performs slightly better than ASL. This emphasizes that MPVAE is less sensitive to noisy labels. Still MPVAE's mAP is 18.2% points lower than TLCM.

6.5.3. ABLATION STUDY

To better understand the performance of TLCM, we perform extra ablation studies to investigate the effects of: i) errors in the noise corruption matrix estimation; ii) impact of the gold dataset size (both studied in experiment I); and iii) impact of number of single-label images (studied in experiment II). The base setup of the experiments is the same as in Section 6.5.1 with only the changes specifically mentioned.

Experiment I: Figure 6.4 shows visually the difference between the true and our estimated noise corruption matrix. To assess also quantitatively how good our estimation method works, we train classifier g with the true corruption matrix. Figure 6.6(a) compares the achieved mAP results on MS-COCO under 40% noise. g trained with the true



(a) Impact of trusted data (b) Impact of single-label data

Figure 6.6: Mean Average Precision (mAP) of TLCM compared to three baselines with different trusted data fractions (a) and mAP of TLCM with different numbers of single-label images per class (10 and 50 images) (b) on MS-COCO.

corruption matrix represents the upper performance bound achievable by noise corruption matrix estimators. Since TLCM uses trusted data to estimate the noise corruption matrix and train the robust classifier g, we expect the size of $\mathscr{G}$ to have an impact on the estimation accuracy and consequently on model performance. We investigate this effect by repeating the previous experiments with halve the fraction of trusted data, i.e. 5%. This corresponds to 3,263 clean samples and 62,005 samples injected with noisy labels. Figure 6.6(a) shows these results via the two different bar plot groups.

Estimating the noise corruption matrix with TLCM, we reach 68.7% and 69.9% mAP under 5% and 10% of trusted data, respectively. Using the true noise correction matrix increases these numbers to 69.1% and 70.3%. The difference between TLCM and the upper bound (True-Correction) is below 0.5% points in both cases. This shows that our noise estimation can capture almost perfectly the impact of the noise corruption matrix, and that it works even with a reduced amount of trusted data. The difference of 1.2% points when using the true noise correction matrix stems from including $\mathscr{G}$ to train the gold classifier $g(\cdot,\phi)$.

Similar trends hold also in the other two datasets (see Table 6.1). Increasing the size of the trusted data achieves higher mAP scores. This gain increases with the amount of label noise but decreases with the amount of trusted data. For 30% label noise, moving from 5% trusted data to 10% increases the mAP for NUS-WIDE/MIRFLICKR by 1.9%/1.6% points, but from 10% to 15% only by 0.2%/1.5% points. Instead for 60% label noise, the respective gains are 4.6%/7.5% points and 0.2%/1.1% points. Also gains are generally higher for MIRFLICKR than for NUS-WIDE. MIRFLICKR has fewer classes which eases the noise corruption matrix estimation. Even on different datasets, TLCM is able to provide good performance gains even with little trusted data.

Experiment II: To assess the impact of trusted single-label images on the estimation of the corruption matrix, we conduct two extra experiments with varying images per class for MS-COCO. In addition to the previous case using all single-label images, we limit the number of single-label images per class to at most 50 and 10, referred to as TLCM-L50 and TLCM-L10, respectively. This results in a total of 2824/721 single-label images used for TLCM-L50/TLCM-L10. Figure 6.6(b) shows the impact on the mAP over training epochs. One can observe that limiting the number of single-label images has only a minor impact

on the performance of TLCM. Hence our proposed method is not only robust to wrong labels in multi label learning but it also can estimate an accurate noise corruption matrix by using only a small proportion of trusted single-label data. In other words, TLCM has a limited dependency on the amount of clean single-label data.

6.6. MLLSC EVALUATION

In this section, we first explain the details of the experiment setup, evaluation metrics, and baselines. Then present the evaluation results of the performance of our proposed MLLSC on two well-known datasets: MS-COCO [197] and MIR-FLICKR [199].

6.6.1. EXPERIMENTAL SETTINGS

DATASETS

We use MS-COCO, and MIR-FLICKR as datasets for evaluating the performance of MLLSC (more detailed in §. 6.5.1).

FALSE NEGATIVE AND FALSE POSITIVE LABELS

Both MS-COCO and MIR-FLICKR are highly curated datasets. We use the original labels as ground truth and synthetically inject corrupted and missing labels. We follow previous work [201] with minor modifications to adapt it to the multi-label scenario and inject both false-positive and false-negatives labels into the training data. We flip one positive class to other classes with uniform probability η. It is worth mentioning that changing to a new label is acceptable only when the new label is not a member of the original label set. Under such noise, we change a positive label to a negative label (missing label/false negative) and create one false positive label (corrupted label). Thus, the corresponding transition matrix $\boldsymbol{C}$ having $c_{i,j}$ elements for the mentioned missing and corrupt labels injection is as follows:

$$c_{ij} = \begin{cases} 1-\eta & \text{if } i = j \\ \dfrac{\eta}{K-1} & \text{if } i \neq j \end{cases}$$

In order to evaluate the performance of MLLSC, we test our method against multiple missing and corruption probabilities, i.e., in the range $[0.0, 0.6]$.

BASELINES

We compare the performance of MLLSC against BCE, Focal [37], ASL [38], Hill [180], SPLC [180], and MPVAE [184].

- **Binary Cross Entropy (BCE)** is a standard simple loss function for multi-label classification introduced in equation. 6.2.
- **Focal** re-weights the positive and negative terms using the model output probability and weighting factor in the loss to reduce the impact of label imbalance (See equation. 6.4).
- **ASL** introduces an asymmetric loss to better address the label imbalance issue for multi-label classification (See equation. 6.5).
- **Hill** is mainly designed to address the missing label problem (false-negatives). It re-weights the Mean Squared Error (MSE) loss term for negative labels. The loss term for positive labels remains the same as BCE.

- **SPLC** is a loss correction method for false-negative labels which uses the output probability of the multi-label classifier to distinguish false-negative from true-negative labels.
- **MPVAE** is a multi-label classification model that works based on variational autoencoder. This method encodes the features and labels to Gaussian subspaces and then decodes the samples from the subspaces into a multivariate probit to predict the image labels.

EVALUATION METRICS

To evaluate the performance of our proposed method under all aspects against the baselines, we consider and report the following metrics: mean average precision (mAP), average per-class F1 (CF1), and average overall F1 (OF1). These metrics have been commonly used in the related art [38], [180] to evaluate the performance of multi-label classification models. We report the average and standard deviation across three runs for Table 6.2 and Table 6.3.

DNN CONFIGURATIONS

As a base model we consider a ResNet50 pre-trained on the ImageNet-21K dataset [205] which has been widely used in vision classification problems. The DNN architecture is the same for all the baselines according to default values provided in [37], [38], [180] except MPVAE, which is an auto-encoder based method. The encoder and decoder use the structure from [184], i.e., 3-layer fully connected neural networks with ReLU activation function. Moreover, we set the hyper-parameters to the default values provided in [184] for each dataset. We perform all the experiments using PyTorch v1.9.0, and we train all the methods for 60 epochs except MPVAE which trains for 200 epochs. To train with MLLSC, we set the batch size, learning rate, and weight decay to 32, 0.0001, and 10^{-4}, respectively.

6.6.2. COMPARISON WITH BASELINES

In this section, we evaluate the performance of our proposed method against baselines on the MIR-FLICKR and MS-COCO datasets under three different ratios of η, i.e., {0.0, 0.3, 0.6}.

For MIR-FLICKR, we report the mAP, CF1, and OF1 of MLLSC against baselines in Table 6.2. MLLSC achieves the highest mAP among all the methods except for $\eta = 0.0$. In this case, MLLSC is the second best with only Focal reaching a higher score of 75.50. SPLC is the closest rival to MLLSC with 1.28 and 2.02 percents mAP difference under $\eta = 0.3$ and $\eta = 0.6$, respectively. Besides, the MLLSC achieves the highest CF1 compared to all the baselines for all cases. Since MPVAE uses a Variational Auto Encoder-based method and the network architecture is less deep and complex than other baselines, the MPVAE can not perform well for the case of $\eta = 0.0$ and $\eta = 0.3$. However, it is also a robust method against false-positive and false-negative labels, because the difference between the mAP at $\eta = 0.0$ and $\eta = 0.6$ is the lowest compared to other methods. Our method significantly improves robustness with respect to Focal. Under a severe ratio of missing and corrupted labels, i.e., $\eta = 0.6$, its mAP score is 11.46% points higher.

MS-COCO is more challenging than MIR-FLICKR due to the larger number of classes and higher label imbalance. The results are summarized in Table 6.3. Here MLLSC obtains

Table 6.2: Mean Average Precision (mAP) (%), average per-class F1 (CF1) (%), and average overall F1 (OF1) (%) of MLLSC compared to six baselines on MIR-FLICKR under ratios η of false-negative and false-positive labels.

Method	$\eta = 0.0$			$\eta = 0.3$			$\eta = 0.6$		
	mAP	***CF1***	***OF1***	***mAP***	***CF1***	***OF1***	***mAP***	***CF1***	***OF1***
BCE	74.79 ± 0.14	74.85 ± 0.07	**78.20** ± 0.15	60.56 ± 0.52	62.76 ± 0.20	68.07 ± 0.16	35.43 ± 0.39	33.46 ± 0.07	38.41 ± 0.18
Focal	**75.50** ± 0.26	74.34 ± 0.18	77.39 ± 0.13	60.73 ± 0.09	61.53 ± 0.12	68.50 ± 0.32	33.87 ± 0.06	31.34 ± 0.26	36.76 ± 0.23
ASL	74.86 ± 0.56	71.80 ± 0.11	74.99 ± 0.17	51.90 ± 2.02	26.83 ± 1.84	26.78 ± 0.89	34.24 ± 0.29	40.40 ± 0.22	42.68 ± 0.33
Hill	74.79 ± 0.53	74.43 ± 0.06	77.68 ± 0.16	59.23 ± 0.42	62.48 ± 0.18	67.81 ± 0.14	33.04 ± 0.38	35.19 ± 0.21	39.47 ± 0.19
SPLC	73.72 ± 0.04	72.35 ± 0.24	75.98 ± 0.14	63.81 ± 0.05	68.50 ± 0.18	71.55 ± 0.33	43.31 ± 0.23	55.23 ± 0.16	59.95 ± 0.08
MPVAE	41.32 ± 0.27	37.64 ± 0.16	48.85 ± 0.15	39.23 ± 0.19	36.26 ± 0.21	47.39 ± 0.27	35.37 ± 0.29	31.87 ± 0.11	41.72 ± 0.14
MLLSC	75.38 ± 0.24	**75.20** ± 0.19	77.72 ± 0.19	**65.09** ± 0.19	**68.98** ± 0.05	**71.88** ± 0.12	**45.33** ± 0.25	**56.35** ± 0.15	**60.48** ± 0.24

Table 6.3: Mean Average Precision (mAP) (%), average per-class F1 (CF1) (%), and average overall F1 (OF1) (%) of MLLSC compared to six baselines on MS-COCO under ratios η of false-negative and false-positive labels.

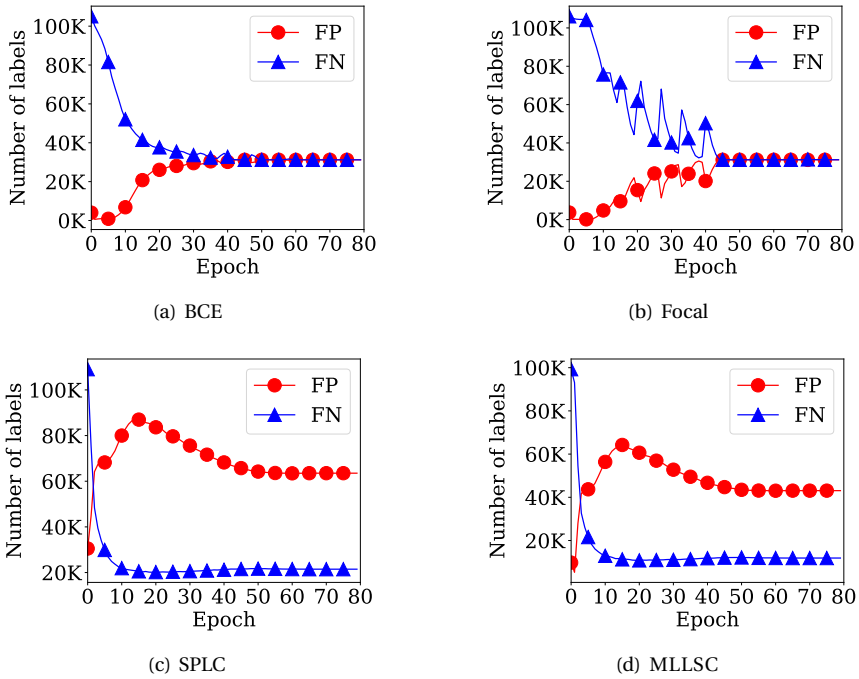
Method	$\eta = 0.0$			$\eta = 0.3$			$\eta = 0.6$		
	mAP	***CF1***	***OF1***	***mAP***	***CF1***	***OF1***	***mAP***	***CF1***	***OF1***
BCE	65.72 ± 0.15	**64.23** ± 0.27	66.29 ± 0.14	43.66 ± 0.24	41.26 ± 0.25	47.31 ± 0.28	19.40 ± 0.31	18.07 ± 0.21	23.41 ± 0.28
Focal	66.77 ± 0.24	63.91 ± 0.17	65.92 ± 0.21	42.73 ± 0.21	36.54 ± 0.29	42.46 ± 0.18	20.67 ± 0.23	20.58 ± 0.14	26.29 ± 0.27
ASL	68.52 ± 0.23	46.72 ± 0.14	60.36 ± 0.27	50.53 ± 0.38	22.74 ± 0.32	51.49 ± 0.38	22.66 ± 0.21	16.70 ± 0.26	40.66 ± 0.22
Hill	63.76 ± 0.14	61.50 ± 0.31	65.78 ± 0.23	48.61 ± 0.24	51.89 ± 0.14	56.50 ± 0.34	27.79 ± 0.20	32.50 ± 0.24	36.42 ± 0.37
SPLC	64.77 ± 0.28	61.71 ± 0.16	65.51 ± 0.33	60.65 ± 0.23	56.68 ± 0.14	61.36 ± 0.42	49.39 ± 0.21	**56.87** ± 0.15	57.67 ± 0.16
MPVAE	39.98 ± 0.24	29.71 ± 0.15	46.82 ± 0.23	29.83 ± 0.42	26.58 ± 0.32	46.20 ± 0.16	26.33 ± 0.24	24.72 ± 0.19	43.51 ± 0.24
MLLSC	**68.83** ± 0.28	63.58 ± 0.32	**69.83** ± 0.14	**65.69** ± 0.28	**61.89** ± 0.18	**68.75** ± 0.13	**51.68** ± 0.16	55.57 ± 0.46	**58.60** ± 0.25

the highest mAP among all baselines under all three ratios of missing and corrupted labels. For the case of $\eta = 0.0$, our proposed method outperforms all the rivals achieving 68.83% mAP. ASL is the second best with a 0.31% points lower mAP. Under $\eta > 0$, i.e., 0.3 and 0.6, SPLC is the closest competitor since it alleviates the impact of missing labels using a self-paced loss correction method for negative labels. The most considerable difference in mAP between MLLSC and SPLC methods is 5.04% with $\eta = 0.3$. Here MLLSC and SPLC reach 65.69% and 60.65% mAP, respectively. In the case of $\eta = 0.6$, MLLSC achieves the highest mAP, and OF1, whereas the baselines trail far behind. According to the results, not only can the MLLSC withstand missing and corrupted labels, but it also mitigates the impact of imbalance labels even for MS-COCO, which contains a high variation of classes [210]. Besides, the performance of all the multi-label losses, i.e., BCE, Focal, and ASL, significantly drops with increasing ratios of false-negative and false-positive labels in the training set.

6.6.3. INVESTIGATING FALSE-POSITIVE AND FALSE-NEGATIVE LABELS IN TRAINING WITH MLLSC

To provide insights on MLLSC and evaluate the ability of our proposed method to distinguish false-positive (FP) and false-negative (FN) labels, we monitor the number of FN and FP during the training process. We compute FP and FN labels by considering the whole label vector of each predicted image and comparing it to the ground truth. Note that the ground truth is only used to compute these statistics but not for training. Figure 6.7 plots the number of FP and FN labels during training for BCE, Focal, SPLC, and MLLSC over 80 epochs on MIR-FLICKR under $\eta = 0.3$. At the beginning of training, the number of FN labels is high because there is no knowledge to detect and correct labels. On the contrary, the number of FP labels is low because the model refrains from predicting any labels due to low confidence. With increasing training epochs, the number of FN labels decreases,

and the number of FP labels increases for both BCE and Focal until they reach a steady-state (see Figure 6.7(a) and Figure 6.7(b)). Since BCE and Focal are not equipped with any mechanism to make them robust against FP and FP, they overfit to missing and corrupted labels and reach a steady-state in which the ratio of FP and FP labels are both equal to the η in use. In contrast SPLC and MLLSC use resilient losses against FP and FN. Thus they reduce the impact of label errors and reach different steady-state values. Comparing SPLC and MLLSC to BCE and focal shows that the number of FN labels significantly decreases with training epochs. Moreover, the end values for MLLSC and SPLC are approximately $87.5K$ and $80K$ less, respectively. MLLSC and SPLC share the same trend for the number of FP labels, but our proposed method incurs about $43K$ FP labels which is approximately $7K$ FP labels less than SPLC (see Figure 6.7(c) and Figure 6.7(d)). The loss term for positive labels in SPLC is margin Focal loss which is not equipped with a robust mechanism to alleviate the impact of false-positive labels. Hence, the number of false-positive is higher than for MLLSC. Although the number of FP labels in BCE and Focal are slightly less than SPLC and MLLSC, handling both FP and FN labels simultaneously in MLLSC improves the mAP significantly compared to the baselines (see Table 6.3 and Table 6.2).



(a) BCE (b) Focal

(c) SPLC (d) MLLSC

Figure 6.7: Number of false-positive and false-negative labels of BCE (a), Focal (b), SPLC (c), and MLLSC (d) multi-label losses during training under a noise ratio of 0.3 on the MIR-FLICKR dataset.

Table 6.4: Mean Average Precision (mAP) (%), average per-class F1 (CF1) (%), and average overall F1 (OF1) (%) of multi-label classifiers under missing and corruption ratio of 0.4 for different loss functions on MIR-FLICKR

Method	*mAP*	*CF1*	*OF1*
SPLC + BCE	50.10	60.49	61.94
SPLC + ASL	48.60	42.91	47.59
SPLC + margin Focal	59.24	66.17	69.14
MLLSC + BCE	59.22	67.23	70.31
MLLSC + ASL	59.15	66.30	69.39
MLLSC + margin Focal	**60.61**	**66.92**	**70.87**

6.6.4. ANALYZING THE PERFORMANCE OF MLLSC WITH VARIED LOSS FUNCTIONS

In this part, we complement our method and SPLC with three different multi-label losses, i.e., ASL, Focal, and BCE, to evaluate their improvements under $\eta = 0.4$ on the MIR-FLICKR dataset. The detailed formula of MLLSC with ASL loss is given in the supplementary material due to space reasons. Focal loss with margin shows the best performance improvement combined with both SPLC and MLLSC compared to ASL and BCE (see Table 6.4). Moreover, the mAP of MLLSC outperforms SPLC when using BCE, ASL and Focal loss functions by 9.12%, 10.55% and 1.37% point difference, respectively. Although ASL can deal well with label imbalance compared to BCE, it can not handle missing and corrupted labels appropriately.

6.6.5. ABLATION STUDY

IMPACT OF THE HYPER-PARAMETERS (τ, τ')

To distinguish false-positive and false-negative labels correctly, we need to find the best threshold values τ and τ' used to compute the correct loss term. We empirically evaluate the best combination of (τ, τ'). Table 6.5 presents the mAP of MLLSC under different $\tau, \tau' \in [0.4, 0.7]$ using Focal margin loss under a ratio of missing and corrupted labels of $\eta = 0.4$ on MIR-FLICKR. Here we keep one of the two parameters fixed while changing the other one to determine the effect on mAP. All other parameters use their default values, i.e., $\tau' = 0.6$ when varying τ, and $\tau = 0.55$ when varying τ'. From Table 6.5, one can see that the mAP of MLLSC is more sensitive to τ' than τ because the difference between the highest and lowest mAPis 39.92% and 0.56%, respectively. This sensitivity is due to the predominant number of negative labels compared to positive labels in each image. A wrong threshold for negative labels, i.e., τ', creates a more significant number of false label identifications. Thus it influences the performance of MLLSC more. Higher thresholds (τ, τ') make substantial restrictions for true-positive and true-negative label detection, and consequently, in the case of uncertain labels, the number of false-negative and false-positive labels increases. Overall we identify as best values for τ and τ' to detect false-positive and false-negative labels via the model output confidence to be 0.55 and 0.6, respectively. We also consider two extremes for (τ, τ') when we change both values at the same time. We can see that the mAP degrades significantly, i.e., 35.36% points, when we increase τ from 0.4 to 0.7 and decrease τ' from 0.7 to 0.4.

Table 6.5: Mean Average Precision (mAP) (%) of MLLSC under missing and corruption ratio of 0.4 for different threshold values (τ, τ') in the loss function on MIR-FLICKR

Threshold (τ)							
τ	0.4	0.45	0.5	0.55	0.6	0.65	0.7
mAP	60.05	60.30	60.53	**60.61**	60.32	60.29	60.26
Threshold (τ')							
τ'	0.4	0.45	0.5	0.55	0.6	0.65	0.7
mAP	20.69	26.76	50.26	59.74	**60.61**	58.74	56.14
Thresholds (τ,τ')							
(τ, τ') = (0.4, 0.7) 55.91				(τ, τ') = (0.7, 0.4) 20.55			

Table 6.6: Mean Average Precision (mAP) (%) of MLLSC under missing and corruption ratio of 0.4 for different margin parameter of focal loss (m) on MIR-FLICKR

m	0.0	0.5	1.0	1.5	2.0
mAP	55.69	59.87	**60.61**	59.38	55.73

IMPACT OF m ON MLLSC WITH FOCAL MARGIN LOSS

To assess the impact of the margin parameter (m) of the Focal loss on MLLSC, we vary its value from 0.0 to 2.0 in Table 6.6. Since the margin manages the attention of the loss on positive labels, a small value of m concentrate more on hard positive labels. As shown in Table 6.6, for the case of $m = 0$, i.e., standard Focal loss, MLLSC achieves 55.69% mAP, while when increasing the margin, MLLSC can achieve 60.61% mAP when $m = 1$. Increasing the value beyond this shifts the focus of the loss function from hard positives to semi-hard positive labels, which leads again to an mAP reduction. Hence, we set $m = 1$ in all the experiments.

IMPACT OF γ ON MLLSC WITH FOCAL MARGIN LOSS

Here, we evaluate the sensitivity of MLLSC to the focus parameter (γ) of the Focal margin loss (see equation. 6.12). γ controls the weights of positive and negative labels. With γ below 2.0, the loss can not bring down the weight of easy negative labels, and this degrades the performance of MLLSC by 9.14% and 0.66% points when is $\gamma = 0.0$ and $\gamma = 1.0$, respectively (see Table 6.7). A large value of γ causes a significant weight reduction of positive labels which are rarely seen in the training data. According to the empirical study, we set $\gamma = 2.0$ through the experiments for best mAP performance.

Table 6.7: Mean Average Precision (mAP) (%) of MLLSC under missing and corruption ratio of 0.4 for different values of the focus parameter (γ) of the focal margin loss on MIR-FLICKR

γ	0.0	1.0	2.0	4.0
mAP	51.47	59.95	**60.61**	59.03

6.7. CONCLUSION

In summary, we addressed the problem of noisy multi-label classification and proposed two innovative approaches to enhance the accuracy of multi-label classifiers in the pres-

ence of label noise.

The first approach, Trusted Loss Correction for Multi-Label Learning (TLCM), effectively estimated the noise corruption matrix and improved the robustness of multi-label classifiers against noisy labels. Our evaluations on real-world datasets showed that TLCM achieved significant improvements in mean average precision (mAP) compared to state-of-the-art classifiers, ranging from 2.7% to 36.7% and 10.7% to 38.2% under different levels of label corruption.

The second approach, Multi-Label Loss Correction against Missing and Corrupted Labels (MLLSC), effectively mitigated the negative impact of both missing and corrupted labels without requiring additional ground truth labels. Our experiments demonstrated that MLLSC outperformed state-of-the-art baselines by 9.33% to 19.48% and 8.88% to 23.85% in terms of mAP under various noise ratios.

By proposing TLCM and MLLSC, we provided comprehensive solutions to effectively address noisy labels in multi-label classification, significantly improving the accuracy and resilience of multi-label classifiers. Overall, our work contributes valuable insights and innovative methods to tackle the challenges of noisy multi-label classification in real-world scenarios, offering robust and accurate solutions for practical applications.

Despite these contributions, there are some limitations worth noting. For the TLCM method, training two separate networks—one on untrusted data and another on corrected data—can be resource-intensive, making it challenging in environments with limited computational resources. Additionally, obtaining a small portion of clean single and multi-label datasets simultaneously can be difficult. Regarding MLLSC, determining the best threshold values to reduce the impact of corrupted and missing labels in the loss function presents a challenge. Moreover, we rely on the model's confidence to decide on the correctness or missingness of labels, which is not the most optimal solution. Addressing these limitations could further enhance the performance and generalizability of these approaches.

7

CONCLUSION

In this thesis, we investigate the robustness of deep neural networks (DNNs) against noisy labels by exploring three key elements: data, DNN models, and expert supervision. Through a combination of these elements, we design seven robust mechanisms. Initially, we emphasize the importance of data and DNN models, introducing two novel mechanisms: LABELNET and TrustNet. LABELNET utilizes fully supervised data to estimate noise patterns and correct noisy labels, thereby enhancing model accuracy. Conversely, TrustNet addresses the complex noise patterns found in real-world datasets by effectively estimating noise transition matrices using a small set of trusted data and dynamically adjusting loss weights to improve model performance. Furthermore, we address noisy label learning under partial data supervision, proposing the Golden Symmetric Loss mechanism to mitigate noise overfitting and outperform state-of-the-art methods. To study the impact of expert supervision, we present QActor, a mechanism that combines quality models and active learning to select informative samples and improve DNN accuracy in the presence of noisy labels. Additionally, LABNET introduces a collaborative mechanism that integrates DNNs and label aggregation to handle corrupted and missing labels in multi-label learning effectively. Finally, we explore a more challenging problem: noisy multi-label data, and we design two innovative mechanisms, TLCM and MLLSC, to enhance multi-label classifiers' performance against corrupted labels (false positives) and missing labels (false negatives).

7.1. CONCLUSIONS

The main conclusions of this thesis are as follows:

1. In Chapter 2, we introduce two novel learning algorithms, LABELNET and TrustNet, to address label noise in big data systems. LABELNET effectively leverages noisy labels as learning features, transforming them into a learning advantage by employing two networks, Amateur and Expert. The approach outperforms existing robust network classifiers across diverse datasets. LABELNET has two advantages: first, the model is robust against noisy label data, and second, it recovers correct

labels from noisy data. Additionally, TrustNet, presents a noise-resilient classification framework, achieving higher testing accuracy compared to state-of-the-art resilient networks. TrustNet is a robust mechanism that leverages LABELNET and a small proportion of noisy label data. We find that these mechanisms contribute to handling label noise, enhancing the reliability and accuracy of deep learning models in challenging big data scenarios. Empirical evaluations underscore their effectiveness in improving classification performance and handling real-world noise patterns.

2. In Chapter 3, we introduce Golden Symmetric Loss (GSL), a mechanism dedicated to enhancing the robustness of deep models against label noise using clean data and a modified loss function. GSL focuses on correcting the symmetric cross-entropy loss using the noise corruption matrix. By leveraging a small fraction of trusted data, we accurately estimate the corruption matrix and determine weights for regular and reverse cross-entropy. Our approach involves learning deep networks from trusted samples through regular cross-entropy and from untrusted noisy samples through golden symmetric cross-entropy. Our findings highlight the inherent noise robustness of the cross-entropy corrected by the corruption matrix. Additionally, we adapt to noise patterns by heuristically setting the weights of the golden symmetric loss based on the corruption matrix. Through extensive evaluations in vision and text analysis across varied noise rates and patterns, GSL consistently achieves a remarkable accuracy improvement over baselines on CIFAR benchmarks and real-world noisy data. This notable enhancement distinguishes GSL from methods focusing solely on correcting loss or leveraging symmetric cross-entropy.
3. In Chapter 4, we introduce QActor, a novel learning mechanism specifically designed for very noisy labeled datasets. QActor incorporates a dual-purpose approach: a quality model that effectively filters out the noise, and an active learning component, *CENT*, that strategically selects informative noisy instances for oracle relabeling. The distinctive characteristic of QActor lies in its noise-aware selection of informative data. This selection process dynamically adapts query allocation throughout the training iterations based on the observed training loss. This flexible design allows QActor to be seamlessly integrated with both standard and deep learning models, even in scenarios with limited access to clean labels. We find that QActor effectively handles the challenge of learning from datasets with corrupted labels using human experts in the loop of training. Notably, QActor proves advantageous in addressing highly noisy labels, achieving a significant accuracy improvement on CIFAR-10 and CIFAR-100. This straightforward yet robust approach positions QActor as an effective solution for improving classification performance in the presence of noisy label datasets.
4. In Chapter 5, we introduce LABNET, a novel iterative mechanism that addresses the inherent challenges associated with accurate data labeling by crowd workers and its subsequent utilization for training deep neural networks (DNNs). Unlike traditional approaches that perform label aggregation and DNN training in isolation, LABNET fosters a collaborative environment where these processes interact synergistically. Our framework leverages the DNN's ability to extract valuable prior knowledge, which is then fed into the label aggregation algorithm to enhance its accuracy. Con-

versely, the aggregation algorithm's estimated correct labels are utilized to refine the DNN's training process. Furthermore, LABNET incorporates a sophisticated algorithm that dynamically determines when the DNN requires retraining based on the ongoing label aggregation iterations. This iterative approach fosters mutual improvement between the DNN's training and label aggregation processes, leading to superior performance compared to existing baselines. Notably, LABNET demonstrates its greatest efficacy in challenging scenarios characterized by a large number of classes.

5. In Chapter 6, we introduce two novel and effective mechanisms to enhance the robustness of multi-label classifiers in the presence of label noise. The first mechanism, Trusted Loss Correction for Multi-Label Learning (TLCM), leverages a meticulously estimated noise corruption matrix to significantly improve the resilience of multi-label classifiers against label noise. The second mechanism, Multi-Label Loss Correction against Missing and Corrupted Labels (MLLSC), tackles the even more complex challenge of mitigating the detrimental effects of both missing and corrupted labels. This mechanism does so without requiring any additional ground truth labels. By introducing TLCM and MLLSC, we provide a comprehensive suite of solutions for effectively addressing noisy labels in multi-label classification. These mechanisms significantly enhance the accuracy and resilience of multi-label classifiers, paving the way for improved performance in real-world applications.

Following is the general conclusion of the thesis:

- Except for MLLSC, all proposed mechanisms leverage the combined effect of at least two key elements (see Figure 1.6). This synergistic approach enhances the robustness of DNNs against noisy labels compared to mechanisms that focus on a single key element.

7

7.2. FUTURE DIRECTIONS

We outline opportunities for future directions extending beyond each chapter, focusing on advancing the robustness of deep learning models and addressing challenges related to noisy labels.

1. In Chapter 2, we have proposed novel learning algorithms, LABELNET and TrustNet, to effectively handle noisy labels in image classification tasks, achieving improved accuracy by leveraging auxiliary information of noisy labels and estimating noise transition matrices from a small set of trusted data. To enhance the robustness of LABELNET in handling noisy labels, future investigations may explore more complex DNN architectures and regularization techniques. Additionally, there is potential for leveraging additional sources of auxiliary information, e.g., oracle, and aggregation algorithms, in TrustNet, leading to more accurate noise estimation and correction. Beyond image classification, the application of TrustNet and LABELNET to other domains such as natural language processing or speech recognition, can be investigated.
2. In Chapter 3, we have introduced the Golden Symmetric Loss Correction (GSL) approach, dynamically weighing regular and reverse cross-entropy based on the estimated corruption matrix to handle noisy labels. In future work, the potential of GSL can be explored in diverse domains beyond image classification, such as

audio or sensor data, to assess its generalization. Investigating hybrid methods by combining GSL with other noise-resilient approaches may further enhance accuracy in noisy label scenarios. Fine-tuning GSL's loss function with different weighting strategies based on the corruption matrix can optimize its performance. Additionally, extending GSL for semi-supervised settings could prove valuable for handling limited labeled data effectively.

3. In Chapter 4, we have designed the Quality-driven Active Learning (QActor) framework, combining quality models and active learning to effectively address noisy labels by selecting informative samples for relabeling while minimizing the reliance on oracle queries. QActor could be extended to explore its potential in semi-supervised and transfer learning contexts. Integration with advanced deep learning models like Bayesian neural networks might enhance oracle query efficiency. Evaluating QActor's performance in scenarios with high label noise levels would provide insights into its capabilities and limitations, especially when dealing with untrusted oracles that occasionally provide incorrect labels.
4. In Chapter 5, we have introduced LABNET, a collaborative mechanism that integrates DNN training and label aggregation. LABNET's innovative approach involves bidirectional interactions between the classifier and aggregation algorithm, enhancing label quality and classifier performance simultaneously. In the future, LABNET's potential can be expanded by accommodating various forms of noisy annotations beyond label noise, ensuring adaptability to real-world situations involving attribute or instance noise. Additionally, investigating LABNET's performance with restricted trusted data would shed light on its capabilities when only a small portion of accurate labels is accessible. The integration of LABNET with advanced deep learning architectures and aggregation algorithms could further enhance its effectiveness across diverse datasets and applications.
5. In Chapter 6, we have proposed two innovative mechanisms, Trusted Loss Correction for Multi-Label Learning (TLCM) and Multi-Label Loss Correction against Missing and Corrupted Labels (MLLSC), to enhance the accuracy of multi-label classifiers in the presence of label noise. TLCM effectively estimates noise corruption matrices and enhances classifier robustness, while MLLSC introduces a robust loss function to mitigate the impact of noisy labels without requiring additional ground truth. Future research directions outlined in Chapter 6 include exploring ensemble models formed by combining TLCM and MLLSC with other multi-label learning methods. The goal is to enhance the robustness of multi-label classifiers against label noise even further. Extending the application of TLCM and MLLSC to diverse multi-label learning tasks beyond image classification, including text categorization and document tagging, will provide insights into their generalization capabilities across various data domains. Furthermore, exploring the integration of TLCM and MLLSC into Graph Neural Networks (GNNs) with noisy labels can significantly enhance the performance and reliability of GNNs in tasks involving multi-label graph data.

BIBLIOGRAPHY

[1] I. Goodfellow, "Deep learning", *MIT press*, 2016.

[2] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning", *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[3] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-supervised learning*. The MIT Press, 2006, p. 103 126.

[4] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, "Deep reinforcement learning that matters", in *AAAI*, vol. 32, 2018.

[5] J. Geiping, Q. Garrido, P. Fernandez, A. Bar, H. Pirsiavash, Y. LeCun, and M. Goldblum, "A cookbook of self-supervised learning", *arXiv preprint arXiv:2304.12210*, 2023.

[6] T. Dhar, N. Dey, S. Borra, and R. S. Sherratt, "Challenges of deep learning in medical image analysis—improving explainability and trust", *IEEE Transactions on Technology and Society*, vol. 4, no. 1, pp. 68–75, 2023.

[7] D. Coelho and M. Oliveira, "A review of end-to-end autonomous driving in urban environments", *IEEE Access*, vol. 10, pp. 75 296–75 311, 2022.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition", *CVPR*, pp. 770–778, 2016.

[9] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation", *MICCAI*, pp. 234–241, 2015.

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets", *NeurIPS*, vol. 27, 2014.

[11] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models", *NeurIPS*, vol. 33, pp. 6840–6851, 2020.

[12] A. Mehrish, N. Majumder, R. Bharadwaj, R. Mihalcea, and S. Poria, "A review of deep learning techniques for speech processing", *Information Fusion*, p. 101 869, 2023.

[13] O. B. Sezer, M. U. Gudelek, and A. M. Ozbayoglu, "Financial time series forecasting with deep learning: A systematic literature review: 2005–2019", *Applied soft computing*, vol. 90, p. 106 181, 2020.

[14] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks", *ICML*, pp. 1126–1135, 2017.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks", *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[16] M.-L. Zhang and Z.-H. Zhou, "A review on multi-label learning algorithms", *IEEE Transactions on knowledge and data engineering*, vol. 26, no. 8, pp. 1819–1837, 2013.

[17] Q. Zhang, Y. Wen, X. Tian, X. Gan, and X. Wang, "Incentivize crowd labeling under budget constraint", *INFOCOM*, pp. 2812–2820, 2015.

[18] F. K. Khattak and A. Salleb-Aouissi, "Quality control of crowd labeling through expert evaluation", *NeurIPS Workshop on Computational Social Science and the Wisdom of Crowds*, vol. 2, p. 5, 2011.

[19] J. Surowiecki, *The Wisdom of Crowds: Why the Many Are Smarter Than the Few and How Collective Wisdom Shapes Business, Economies, Societies, and Nations*. Doubleday, 2004.

[20] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The Caltech-UCSD birds-200-2011 dataset", *California Institute of Technology*, 2011.

[21] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database", *CVPR*, pp. 248–255, 2009.

[22] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge", *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[23] Google Cloud, *AI platform data labeling service pricing*, https://cloud.google.com/ai-platform/data-labeling/pricing, Accessed on: 10/07/2023.

[24] B. Mozafari, P. Sarkar, M. Franklin, M. Jordan, and S. Madden, "Scaling up crowd-sourcing to very large datasets: A case for active learning", *VLDB Endowment*, vol. 8, no. 2, pp. 125–136, 2014.

[25] T. Xiao, T. Xia, Y. Yang, C. Huang, and X. Wang, "Learning from massive noisy labeled data for image classification", *CVPR*, pp. 2691–2699, 2015.

[26] C. Northcutt, L. Jiang, and I. Chuang, "Confident learning: Estimating uncertainty in dataset labels", *Journal of Artificial Intelligence Research*, vol. 70, pp. 1373–1411, 2021.

[27] C. Yu, X. Ma, and W. Liu, "Delving into noisy label detection with clean data", *ICML*, vol. 202, pp. 40 290–40 305, 2023.

[28] C. G. Northcutt, A. Athalye, and J. Mueller, "Pervasive label errors in test sets destabilize machine learning benchmarks", *NeurIPS Datasets and Benchmarks*, 2021.

[29] Harvard Business Review, *Bad data costs the U.S. $3 trillion per year*, https://hbr.org/2016/09/bad-data-costs-the-u-s-3-trillion-per-year, 2016.

[30] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, "Do imagenet classifiers generalize to imagenet?", *ICML*, pp. 5389–5400, 2019.

[31] A. Veit, N. Alldrin, G. Chechik, I. Krasin, A. Gupta, and S. Belongie, "Learning from noisy large-scale datasets with minimal supervision", *CVPR*, pp. 839–847, 2017.

[32] S. Sukhbaatar, J. Bruna, M. Paluri, L. Bourdev, and R. Fergus, "Training convolutional networks with noisy labels", *arXiv:1406.2080*, 2014.

[33] A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré, "Snorkel: Rapid training data creation with weak supervision", *VLDB Endowment*, vol. 11, no. 3, p. 269, 2017.

[34] X. Ma, Y. Wang, M. E. Houle, S. Zhou, S. Erfani, S. Xia, S. Wijewickrema, and J. Bailey, "Dimensionality-driven learning with noisy labels", *ICML*, pp. 3355–3364, 2018.

[35] B. Van Rooyen, A. Menon, and R. C. Williamson, "Learning with symmetric label noise: The importance of being unhinged", *NeurIPS*, vol. 28, 2015.

[36] Z. Zhang, H. Zhang, S. O. Arik, H. Lee, and T. Pfister, "Distilling effective supervision from severe label noise", *CVPR*, pp. 9294–9303, 2020.

[37] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 318–327, 2020.

[38] T. Ridnik, E. B. Baruch, N. Zamir, A. Noy, I. Friedman, M. Protter, and L. Zelnik-Manor, "Asymmetric loss for multi-label classification", *ICCV*, pp. 82–91, 2021.

[39] A. C. de Carvalho and A. A. Freitas, "A tutorial on multi-label classification techniques", *Foundations of Computational Intelligence Volume 5: Function Approximation and Classification*, pp. 177–195, 2009.

[40] G. Tsoumakas and I. Katakis, "Multi-label classification: An overview", *International Journal of Data Warehousing and Mining (IJDWM)*, vol. 3, no. 3, pp. 1–13, 2007.

[41] H. Song, M. Kim, D. Park, Y. Shin, and J.-G. Lee, "Learning from noisy labels with deep neural networks: A survey", *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[42] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning (still) requires rethinking generalization", *Communications of the ACM*, vol. 64, no. 3, pp. 107–115, 2021.

[43] D. Arpit, S. Jastrzebski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. C. Courville, Y. Bengio, and S. Lacoste-Julien, "A closer look at memorization in deep networks", *ICML*, vol. 70, pp. 233–242, 2017.

[44] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization", *ICLR*, 2017.

[45] G. Algan and I. Ulusoy, "Image classification with deep learning in the presence of noisy labels: A survey", *Knowledge-Based Systems*, vol. 215, p. 106 771, 2021.

[46] C. Zhang, J. Shen, and P. Awasthi, "Efficient active learning of sparse halfspaces with arbitrary bounded noise", *NeurIPS*, 2020.

[47] S. Branson, C. Wah, F. Schroff, B. Babenko, P. Welinder, P. Perona, and S. Belongie, "Visual recognition with humans in the loop", *ECCV*, pp. 438–451, 2010.

[48] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis", *ICML*, pp. 1060–1069, 2016.

[49] D. Hendrycks, M. Mazeika, D. Wilson, and K. Gimpel, "Using trusted data to train deep networks on labels corrupted by severe noise", *NeurIPS*, vol. 31, pp. 10 477–10 486, 2018.

[50] G. Zheng, A. H. Awadallah, and S. Dumais, "Meta label correction for noisy label learning", *AAAI*, vol. 35, no. 12, pp. 11 053–11 061, 2021.

[51] H. Qiu, K. Chintalapudi, and R. Govindan, "MCAL: minimum cost human-machine active labeling", *ICLR*, 2023.

[52] S. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich, "Training deep neural networks on noisy labels with bootstrapping", *ICLR*, 2014.

[53] E. Arazo, D. Ortego, P. Albert, N. O'Connor, and K. McGuinness, "Unsupervised label noise modeling and loss correction", *ICML*, pp. 312–321, 2019.

[54] L. Huang, C. Zhang, and H. Zhang, "Self-adaptive training: Beyond empirical risk minimization", *NeurIPS*, vol. 33, pp. 19 365–19 376, 2020.

[55] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, and L. Fei-Fei, "Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels", *ICML*, pp. 2304–2313, 2018.

[56] X. Xia, T. Liu, N. Wang, B. Han, C. Gong, G. Niu, and M. Sugiyama, "Are anchor points really indispensable in label-noise learning?", *NeurIPS*, vol. 32, pp. 6835–6846, 2019.

[57] J. Shu, Q. Xie, L. Yi, Q. Zhao, S. Zhou, Z. Xu, and D. Meng, "Meta-weight-net: Learning an explicit mapping for sample weighting", *NeurIPS*, vol. 32, pp. 1917–1928, 2019.

[58] D. Patel and P. S. Sastry, "Adaptive sample selection for robust learning under label noise", *WACV*, pp. 3921–3931, 2023.

[59] E. Malach and S. Shalev-Shwartz, "Decoupling" when to update" from" how to update"", *NeurIPS*, vol. 30, pp. 960–970, 2017.

[60] Y. Lyu and I. W. Tsang, "Curriculum loss: Robust learning and generalization against label corruption", *ICLR*, 2019.

[61] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama, "Co-teaching: Robust training of deep neural networks with extremely noisy labels", *NeurIPS*, vol. 31, pp. 8536–8546, 2018.

[62] Y. Li, H. Han, S. Shan, and X. Chen, "DISC: learning from noisy labels via dynamic instance-specific selection and correction", *CVPR*, pp. 24 070–24 079, 2023.

[63] Y. Wang, X. Ma, Z. Chen, Y. Luo, J. Yi, and J. Bailey, "Symmetric cross entropy for robust learning with noisy labels", *ICCV*, pp. 322–330, 2019.

[64] E. Amid, M. K. Warmuth, R. Anil, and T. Koren, "Robust bi-tempered logistic loss based on bregman divergences", *NeurIPS*, vol. 32, pp. 14 987–14 996, 2019.

[65] Z. Zhang and M. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels", *NeurIPS*, vol. 31, pp. 8792–8802, 2018.

[66] H.-S. Chang, E. Learned-Miller, and A. McCallum, "Active bias: Training more accurate neural networks by emphasizing high variance samples", *NeurIPS*, vol. 30, pp. 1002–1012, 2017.

[67] D. Patel and P. S. Sastry, "Memorization in deep neural networks: Does the loss function matter?", *PAKDD*, vol. 12713, pp. 131–142, 2021.

[68] X. Zhou, X. Liu, J. Jiang, X. Gao, and X. Ji, "Asymmetric loss functions for learning with noisy labels", *ICML*, vol. 139, pp. 12 846–12 856, 2021.

[69] M. Yang, Y. Li, Z. Huang, Z. Liu, P. Hu, and X. Peng, "Partially view-aligned representation learning with noise-robust contrastive loss", *CVPR*, pp. 1134–1143, 2021.

[70] G. Patrini, A. Rozza, A. Krishna Menon, R. Nock, and L. Qu, "Making deep neural networks robust to label noise: A loss correction approach", *CVPR*, pp. 1944–1952, 2017.

[71] I. Jindal, M. Nokleby, and X. Chen, "Learning deep networks from noisy labels with dropout regularization", *ICDM*, pp. 967–972, 2016.

[72] A. J. Bekker and J. Goldberger, "Training deep neural-networks based on unreliable labels", *ICASSP*, pp. 2682–2686, 2016.

[73] J. Goldberger and E. Ben-Reuven, "Training deep neural-networks using a noise adaptation layer", *ICLR*, 2017.

[74] P. Chen, B. Liao, G. Chen, and S. Zhang, "Understanding and utilizing deep neural networks trained with noisy labels", *ICML*, pp. 1062–1070, 2019.

[75] A. Ghosh, H. Kumar, and P. S. Sastry, "Robust loss functions under label noise for deep neural networks", *AAAI*, pp. 1919–1925, 2017.

[76] N. Manwani and P. S. Sastry, "Noise tolerance under risk minimization", *IEEE Transactions on Cybernetics*, vol. 43, no. 3, pp. 1146–1151, 2013. DOI: 10.1109/TSMCB.2012.2223460.

[77] *Dimensionality-driven learning with noisy labels*, https://github.com/xingjunm/dimensionality-driven-learning.

[78] *Gold loss correction*, https://github.com/mmazeika/glc.

[79] *Symmetric learning (sl) via symmetric cross entropy (sce) loss*, https://github.com/YisenWang/symmetric_cross_entropy_for_noisy_labels.

[80] *Asymmetric loss for multi-label classification*, https://github.com/Alibaba-MIIL/ASL.

[81] J. Prendki, *The curse of big data labeling and three ways to solve it*, Accessed on 22.08.2021, 2018. [Online]. Available: https://aws.amazon.com/blogs/apn/the-curse-of-big-data-labeling-and-three-ways-to-solve-it/.

[82] A. Ghiassi, R. Birke, R. Han, and L. Y. Chen, "LABELNET: Recovering noisy labels", *IJCNN*, pp. 1–8, 2021.

[83] K. Yi and J. Wu, "Probabilistic end-to-end noise correction for learning with noisy labels", *CVPR*, pp. 7017–7025, 2019.

[84] A. Vahdat, "Toward robustness against label noise in training deep discriminative neural networks", *NeurIPS*, pp. 5596–5605, 2017.

[85] L. Jiang, Z. Zhou, T. Leung, L. Li, and L. Fei-Fei, "Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels", *ICML*, pp. 2309–2318, 2018.

[86] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, M. Ranzato, and T. Mikolov, "DeViSE: A deep visual-semantic embedding model", *NeurIPS*, pp. 2121–2129, 2013.

[87] M. Mirza and S. Osindero, "Conditional Generative Adversarial Nets", *arXiv:1411.1784*, 2014.

[88] K. K. Thekumparampil, A. Khetan, Z. Lin, and S. Oh, "Robustness of conditional gans to noisy labels", *NeurIPS*, pp. 10 271–10 282, 2018.

[89] H. Song, M. Kim, and J.-G. Lee, "Selfie: Refurbishing unclean samples for robust deep learning", *ICML*, pp. 5907–5915, 2019.

[90] Y. Yao, T. Liu, B. Han, M. Gong, J. Deng, G. Niu, and M. Sugiyama, "Dual t: Reducing estimation error for transition matrix in label-noise learning", *NeurIPS*, vol. 33, 2020.

[91] G. Liu, I. Khalil, and A. Khreishah, "ZK-GanDef: A GAN based zero knowledge adversarial training defense for neural networks", *DSN*, pp. 64–75, 2019.

[92] T. Kaneko, Y. Ushiku, and T. Harada, "Label-noise robust generative adversarial networks", *CVPR*, pp. 2467–2476, 2019.

[93] A. Ghiassi, T. Younesian, Z. Zhao, R. Birke, V. Schiavoni, and L. Y. Chen, "Robust (deep) learning framework against dirty labels and beyond", *IEEE TPS-ISA*, pp. 236–244, 2019.

[94] S. E. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich, "Training deep neural networks on noisy labels with bootstrapping", in *ICLR Workshop*, 2015.

[95] X. Yu, B. Han, J. Yao, G. Niu, I. W. Tsang, and M. Sugiyama, "How does disagreement help generalization against label corruption?", *ICML*, pp. 7164–7173, 2019.

[96] F. R. Cordeiro and G. Carneiro, "A survey on deep learning with noisy labels: How to train your model when you cannot trust on the annotations?", *SIBGRAPI*, pp. 9–16, 2020.

[97] S. Hooker, Y. Dauphin, A. Courville, and A. Frome, "Selective brain damage: Measuring the disparate impact of model pruning", *arXiv:1911.05248*, 2019.

[98] B. Frénay and M. Verleysen, "Classification in the presence of label noise: A survey", *IEEE Transactions on neural networks and learning systems*, vol. 25, no. 5, pp. 845–869, 2013.

[99] D. Tsipras, S. Santurkar, L. Engstrom, A. Ilyas, and A. Madry, "From imagenet to image classification: Contextualizing progress on benchmarks", *ICML*, pp. 9625–9635, 2020.

[100] W. Li, L. Wang, W. Li, E. Agustsson, and L. Van Gool, "Webvision database: Visual learning and understanding from web data", *arXiv:1708.02862*, 2017.

[101] V. Shankar, R. Roelofs, H. Mania, A. Fang, B. Recht, and L. Schmidt, "Evaluating machine accuracy on imagenet", *ICML*, pp. 8634–8644, 2020.

[102] N. Konstantinov and C. Lampert, "Robust learning from untrusted sources", *ICML*, vol. 97, pp. 3488–3498, 2019.

[103] J. Han, P. Luo, and X. Wang, "Deep self-learning from noisy labels", *ICCV*, pp. 5137–5146, 2019.

[104] K. Lee, X. He, L. Zhang, and L. Yang, "Cleannet: Transfer learning for scalable image classifier training with label noise", *CVPR*, pp. 5447–5456, 2018.

[105] E. Malach and S. Shalev-Shwartz, "Decoupling" when to update" from" how to update"", *NeurIPS*, pp. 960–970, 2017.

[106] B. Han, J. Yao, G. Niu, M. Zhou, I. W. Tsang, Y. Zhang, and M. Sugiyama, "Masking: A new perspective of noisy supervision", *NeurIPS*, pp. 5841–5851, 2018.

[107] Z. Zhang and M. R. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels", *NeurIPS*, pp. 8792–8802, 2018.

[108] M. Ren, W. Zeng, B. Yang, and R. Urtasun, "Learning to reweight examples for robust deep learning", *ICML*, 2018.

[109] J. Goldberger and E. Ben-Reuven, "Training deep neural-networks using a noise adaptation layer", *ICLR*, 2017.

[110] S. Sukhbaatar, J. Bruna, M. Paluri, L. Bourdev, and R. Fergus, "Training convolutional networks with noisy labels", *arXiv:1406.2080*, 2014.

[111] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples", *ICLR*, 2015.

[112] Y. Wang, W. Liu, X. Ma, J. Bailey, H. Zha, L. Song, and S.-T. Xia, "Iterative learning with open-set noisy labels", *CVPR*, pp. 8688–8696, 2018.

[113] Y. LeCun and C. Cortes, "MNIST handwritten digit database", 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/.

[114] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-10 (canadian institute for advanced research)", 2009. [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html.

[115] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-100 (canadian institute for advanced research)", 2009. [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html.

[116] Y. Li, J. Yang, Y. Song, L. Cao, J. Luo, and L. Li, "Learning from noisy labels with distillation", *ICCV*, pp. 1928–1936, 2017.

[117] A. Ghiassi, R. Birke, and L. Y.Chen, "Trustnet: Learning from trusted data against (a)symmetric label noise", *BDCAT*, pp. 52–62, 2021.

[118] A. Vahdat, "Toward robustness against label noise in training deep discriminative neural networks", *NeurIPS*, pp. 5596–5605, 2017.

[119] X. Yu, B. Han, J. Yao, G. Niu, I. W. Tsang, and M. Sugiyama, "How does disagreement help generalization against label corruption?", *ICML*, pp. 7164–7173, 2019.

[120] Z. Jiang, K. Zhou, Z. Liu, L. Li, R. Chen, S.-H. Choi, and X. Hu, "An information fusion approach to learning with instance-dependent label noise", *ICLR*, 2021.

[121] Z. Zhu, T. Liu, and Y. Liu, "A second-order approach to learning with instance-dependent label noise", *CVPR*, pp. 10 113–10 123, 2021.

[122] M. Ren, W. Zeng, B. Yang, and R. Urtasun, "Learning to reweight examples for robust deep learning", *ICML*, pp. 4331–4340, 2018.

[123] Z. Zhang and M. R. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels", *NeurIPS*, pp. 8792–8802, 2018.

[124] D. Tanaka, D. Ikami, T. Yamasaki, and K. Aizawa, "Joint optimization framework for learning with noisy labels", *CVPR*, pp. 5552–5560, 2018.

[125] C. Hong, A. Ghiassi, Y. Zhou, R. Birke, and L. Y. Chen, "Online label aggregation: A variational bayesian approach", *WWW*, pp. 1904–1915, 2021.

[126] A. Veit, N. Alldrin, G. Chechik, I. Krasin, A. Gupta, and S. J. Belongie, "Learning from noisy large-scale datasets with minimal supervision", *CVPR*, pp. 6575–6583, 2017.

[127] B. Han, J. Yao, G. Niu, M. Zhou, I. W. Tsang, Y. Zhang, and M. Sugiyama, "Masking: A new perspective of noisy supervision", *NeurIPS*, pp. 5841–5851, 2018.

[128] J. Li, R. Socher, and S. C. Hoi, "Dividemix: Learning with noisy labels as semi-supervised learning", *ICLR*, 2020.

[129] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-10/100 (Canadian Institute for Advanced Research)", 2009. [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html.

[130] S. Zagoruyko and N. Komodakis, "Wide residual networks", *BMVC*, 2016.

[131] I. Loshchilov and F. Hutter, "SGDR: stochastic gradient descent with warm restarts", *ICLR*, 2017.

[132] K. Gimpel, N. Schneider, B. O'Connor, D. Das, D. Mills, J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan, and N. A. Smith, "Part-of-speech tagging for twitter: Annotation, features, and experiments", *ACL*, pp. 42–47, 2011.

[133] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank", *EMNLP*, pp. 1631–1642, 2013.

[134] T. C. Redman, *Bad data costs the U.S. $3 trillion per year*, https://hbr.org/2016/09/bad-data-costs-the-u-s-3-trillion-per-year, Accessed: 2020-02-20, 2016.

[135] L. Jiang, Z. Zhou, T. Leung, L. Li, and L. Fei-Fei, "Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels", *ICML*, pp. 2309–2318, 2018.

[136] B. Settles, "Active learning literature survey", University of Wisconsin-Madison Department of Computer Sciences, Tech. Rep., 2009.

[137] G. Schohn and D. Cohn, "Less is more: Active learning with support vector machines", *ICML*, pp. 839–846, 2000.

[138] A. Holub, P. Perona, and M. C. Burl, "Entropy-based active learning for object recognition", *CVPR Workshops*, pp. 1–8, 2008.

[139] M. Haußmann, F. A. Hamprecht, and M. Kandemir, "Deep active learning with adaptive acquisition", *IJCAI*, pp. 2470–2476, 2019.

[140] Y. Yan, R. Rosales, G. Fung, R. Subramanian, and J. Dy, "Learning from multiple annotators with varying expertise", *Machine learning*, vol. 95, no. 3, pp. 291–327, 2014.

[141] A. Blum, A. Kalai, and H. Wasserman, "Noise-tolerant learning, the parity problem, and the statistical query model", *Journal of the ACM*, vol. 50, no. 4, pp. 506–519, 2003.

[142] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples", *ICLR*, 2015.

[143] Z. Zhao, S. Cerf, R. Birke, B. Robu, S. Bouchenak, S. B. Mokhtar, and L. Y. Chen, "Robust anomaly detection on unreliable data", *DSN*, pp. 630–637, 2019.

[144] Z. Zhao, R. Birke, R. Han, B. Robu, S. Bouchenak, S. B. Mokhtar, and L. Y. Chen, "Enhancing robustness of on-line learning models on highly noisy data", *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 5, pp. 2177–2192, 2021.

[145] M. Ren, W. Zeng, B. Yang, and R. Urtasun, "Learning to reweight examples for robust deep learning", *ICML*, vol. 80, pp. 4331–4340, 2018.

[146] O. Sener and S. Savarese, "Active learning for convolutional neural networks: A core-set approach", *ICLR*, 2018.

[147] O. Sener and S. Savarese, "A geometric approach to active learning for convolutional neural networks", *CoRR*, vol. abs/1708.00489, 2017.

[148] Y. Gal, R. Islam, and Z. Ghahramani, "Deep bayesian active learning with image data", *ICML*, vol. 70, pp. 1183–1192, 2017.

[149] A. Kirsch, J. van Amersfoort, and Y. Gal, "Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning", *NeurIPS*, pp. 7024–7035, 2019.

[150] P. Stanitsas, A. Cherian, A. Truskinovsky, V. Morellas, and N. Papanikolopoulos, "Active convolutional neural networks for cancerous tissue recognition", *ICIP*, pp. 1367–1371, 2017.

[151] M. Bouguelia, S. Nowaczyk, K. C. Santosh, and A. Verikas, "Agreeing to disagree: Active learning with noisy labels without crowdsourcing", *Int. J. Mach. Learn. Cybern.*, vol. 9, no. 8, pp. 1307–1319, 2018.

[152] C. H. Lin, M. Mausam, and D. S. Weld, "Re-active learning: Active learning with relabeling", *AAAI*, 2016.

[153] C. Zhang and K. Chaudhuri, "Active learning from weak and strong labelers", *NeurIPS*, pp. 703–711, 2015.

[154] S. E. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich, "Training deep neural networks on noisy labels with bootstrapping", *ICLR*, 2015.

[155] M. Imran, P. Mitra, and C. Castillo, "Twitter as a lifeline: Human-annotated twitter corpora for NLP of crisis-related messages", *LREC*, 2016.

[156] Z. Xu, Y. Liu, J. Xuan, H. Chen, and L. Mei, "Crowdsourcing based social media data analysis of urban emergency events", *Multimedia Tools and Applications*, vol. 76, no. 9, pp. 11 567–11 584, 2017.

[157] A. P. Dawid and A. M. Skene, "Maximum likelihood estimation of observer error-rates using the em algorithm", *Applied statistics*, pp. 20–28, 1979.

[158] D. Cousineau and S. Helie, "Improving maximum likelihood estimation using prior probabilities: A tutorial on maximum a posteriori estimation and an examination of the weibull distribution", *Tutorials in Quantitative Methods for Psychology*, vol. 9, no. 2, pp. 61–71, 2013.

[159] L. Yin, J. Han, W. Zhang, and Y. Yu, "Aggregating crowd wisdoms with label-aware autoencoders", *IJCAI*, pp. 1325–1331, 2017.

[160] J. Yang, A. Smirnova, D. Yang, G. Demartini, Y. Lu, and P. Cudré-Mauroux, "Scalpel-cd: Leveraging crowdsourcing and deep probabilistic modeling for debugging noisy training data", *WWW*, pp. 2158–2168, 2019.

[161] J. Yang, T. Drake, A. Damianou, and Y. Maarek, "Leveraging crowdsourcing data for deep active learning an application: Learning intents in alexa", *WWW*, pp. 23–32, 2018.

[162] H.-C. Kim and Z. Ghahramani, "Bayesian classifier combination", *Artificial Intelligence and Statistics*, pp. 619–627, 2012.

[163] M. Venanzi, J. Guiver, G. Kazai, P. Kohli, and M. Shokouhi, "Community-based bayesian aggregation models for crowdsourcing", *WWW*, pp. 155–164, 2014.

[164] E. D. Simpson, M. Venanzi, S. Reece, P. Kohli, J. Guiver, S. J. Roberts, and N. R. Jennings, "Language understanding in the wild: Combining crowdsourcing and machine learning", *WWW*, pp. 992–1002, 2015.

[165] C. Hong, A. Ghiassi, Y. Zhou, R. Birke, and L. Y. Chen, "Online label aggregation: A variational bayesian approach", *WWW*, pp. 1904–1915, 2021.

[166] Q. Liu, J. Peng, and A. T. Ihler, "Variational inference for crowdsourcing", *NeurIPS*, pp. 692–700, 2012.

[167] J. Whitehill, T.-f. Wu, J. Bergsma, J. R. Movellan, and P. L. Ruvolo, "Whose vote should count more: Optimal integration of labels from labelers of unknown expertise", *NeurIPS*, pp. 2035–2043, 2009.

[168] D. Zhou, S. Basu, Y. Mao, and J. C. Platt, "Learning from the wisdom of crowds by minimax entropy", *NeurIPS*, pp. 2195–2203, 2012.

[169] D. Zhou, Q. Liu, J. Platt, and C. Meek, "Aggregating ordinal labels from crowds by minimax conditional entropy", *ICML*, pp. 262–270, 2014.

[170] A. Gaunt, D. Borsa, and Y. Bachrach, "Training deep neural nets to aggregate crowdsourced responses", *UAI*, p. 242 251, 2016.

[171] A. Khetan, Z. C. Lipton, and A. Anandkumar, "Learning from noisy singly-labeled data", *ICLR*, 2018.

[172] D. Hendrycks, M. Mazeika, D. Wilson, and K. Gimpel, "Using trusted data to train deep networks on labels corrupted by severe noise", *NeurIPS*, pp. 10 456–10 465, 2018.

[173] P. Chen, B. Liao, G. Chen, and S. Zhang, "Understanding and utilizing deep neural networks trained with noisy labels", in *ICML*, vol. 97, 2019, pp. 1062–1070.

[174] W. Zhao and C. P. Gomes, "Evaluating multi-label classifiers with noisy labels", *ArXiv*, vol. abs/2102.08427, 2021.

[175] A. Veit, N. Alldrin, G. Chechik, I. Krasin, A. Gupta, and S. J. Belongie, "Learning from noisy large-scale datasets with minimal supervision", *CVPR*, 2017.

[176] G. Algan and I. Ulusoy, "Image classification with deep learning in the presence of noisy labels: A survey", *Knowl. Based Syst.*, vol. 215, p. 106 771, 2021.

[177] J. Yao, J. Wang, I. Tsang, Y. Zhang, J. Sun, C. Zhang, and R. Zhang, "Deep learning from noisy image labels with quality embedding", *IEEE Transactions on Image Processing*, vol. 28, pp. 1909–1922, 2019.

[178] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization", *ICLR*, 2017.

[179] H. Xu, X. Liu, W. Wang, W. Ding, Z. Wu, Z. Liu, A. Jain, and J. Tang, "Towards the memorization effect of neural networks in adversarial training", *arXiv:2106.04794*, 2021.

[180] Y. Zhang, Y. Cheng, X. Huang, F. Wen, R. Feng, Y. Li, and Y. Guo, "Simple and robust loss design for multi-label learning with missing labels", *arXiv:2112.07368*, 2021.

[181] E. Cole, O. Mac Aodha, T. Lorieul, P. Perona, D. Morris, and N. Jojic, "Multi-label learning from single positive labels", *CVPR*, pp. 933–942, 2021.

[182] Q. Yao, H. Yang, B. Han, G. Niu, and J. T.-Y. Kwok, "Searching to exploit memorization effect in learning with noisy labels", *ICML*, pp. 10 789–10 798, 2020.

[183] A. Ghiassi, R. Birke, and L. Y. Chen, "LABNET: A collaborative method for dnn training and label aggregation.", *ICAART*, pp. 56–66, 2022.

[184] J. Bai, S. Kong, and C. Gomes, "Disentangled variational autoencoder based multi-label classification with covariance-aware multivariate probit model", *IJCAI*, 2020.

[185] S. Liu, L. Zhang, X. Yang, H. Su, and J. Zhu, "Query2label: A simple transformer way to multi-label classification", *arXiv:2107.10834*, 2021.

[186] V. O. Yazici, A. Gonzalez-Garcia, A. Ramisa, B. Twardowski, and J. v. d. Weijer, "Orderless recurrent models for multi-label classification", *CVPR*, 2020.

[187] Y. Zhang, Z. Wang, and Y. Mao, "Rpn prototype alignment for domain adaptive object detector", *CVPR*, pp. 12 425–12 434, 2021.

[188] Z. Zhao, Y. Guo, H. Shen, and J. Ye, "Adaptive object detection with dual multi-label prediction", *ECCV*, pp. 54–69, 2020.

[189] D. Zhang, X. Ju, W. Zhang, J. Li, S. Li, Q. Zhu, and G. Zhou, "Multi-modal multi-label emotion recognition with heterogeneous hierarchical message passing", *AAAI*, vol. 1, 2021.

[190] R. Cabral, F. Torre, J. P. Costeira, and A. Bernardino, "Matrix completion for multi-label image classification", *NeurIPS*, vol. 24, 2011.

[191] T. Ishida, G. Niu, W. Hu, and M. Sugiyama, "Learning from complementary labels", *NeurIPS*, vol. 30, 2017.

[192] H.-F. Yu, P. Jain, P. Kar, and I. Dhillon, "Large-scale multi-label learning with missing labels", *ICML*, pp. 593–601, 2014.

[193] T. Pu, T. Chen, H. Wu, and L. Lin, "Semantic-aware representation blending for multi-label image recognition with partial labels", *AAAI*, 2022.

[194] N. Xu, J. Lv, and X. Geng, "Partial label learning via label enhancement", *AAAI*, vol. 33, no. 01, pp. 5557–5564, 2019.

[195] M.-K. Xie and S.-J. Huang, "Partial multi-label learning", *AAAI*, vol. 32, no. 1, 2018.

[196] Y. Yan and Y. Guo, "Partial label learning with batch label correction", *AAAI*, vol. 34, no. 04, pp. 6575–6582, 2020.

[197] T.-Y. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context", *ECCV*, 2014.

[198] T. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y. Zheng, "NUS-WIDE: a real-world web image database from national university of singapore", *CIVR*, 2009.

[199] M. J. Huiskes and M. S. Lew, "The mir flickr retrieval evaluation", *ICMIR*, pp. 39–43, 2008.

[200] L. Jiang, Z. Zhou, T. Leung, L. Li, and L. Fei-Fei, "Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels", *ICML*, pp. 2309–2318, 2018.

[201] G. Patrini, A. Rozza, A. Menon, R. Nock, and L. Qu, "Making deep neural networks robust to label noise: A loss correction approach", *CVPR*, pp. 2233–2241, 2017.

[202] H. Song, M. Kim, D. Park, and J. Lee, "Learning from noisy labels with deep neural networks: A survey", *ArXiv*, vol. abs/2007.08199, 2020.

[203] Y. Wang, D. He, F. Li, X. Long, Z. Zhou, J. Ma, and S. Wen, "Multi-label classification with label graph superimposing", *AAAI*, 2020.

[204] T. Ridnik, H. Lawen, A. Noy, E. Ben Baruch, G. Sharir, and I. Friedman, "Tresnet: High performance gpu-dedicated architecture", *WACV*, pp. 1400–1409, 2021.

[205] T. Ridnik, E. B. Baruch, A. Noy, and L. Zelnik-Manor, "Imagenet-21k pretraining for the masses", *CoRR*, vol. abs/2104.10972, 2021.

[206] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 318–327, 2020.

[207] V. Feldman and C. Zhang, "What neural networks memorize and why: Discovering the long tail via influence estimation", *NeurIPS*, 2020.

[208] V. Feldman, "Does learning require memorization? a short tale about a long tail", *ACM SIGACT Symposium on Theory of Computing*, 2020.

[209] D. Arpit, S. Jastrzebski, N. Ballas, and et al., "A closer look at memorization in deep networks", *ICML*, vol. 70, pp. 233–242, 2017.

[210] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context", *ECCV*, pp. 740–755, 2014.

ACKNOWLEDGEMENTS

Here is the culmination of my PhD journey, a beautiful chapter that began on February 17, 2019, when I embarked on a new adventure by moving to the Netherlands and the charming city of Delft. These past four years have been filled with challenges, from adapting to a new country, environment, culture, and language, to navigating the unexpected trials brought by the COVID-19 pandemic. Despite these hurdles, I have been incredibly fortunate to have the unwavering support of my exceptional supervisors, friends, colleagues, and, most importantly, my family. Their constant encouragement, belief in my abilities, and steadfast support in every situation have been invaluable. Without their trust and support, this journey would not have been as enriching and successful as it has been.

First, I would like to extend my deepest gratitude to Lydia Chen, my daily supervisor. Lydia has been more than a supervisor; she has been a mentor and a dear friend who has taught me invaluable lessons both academically and in various aspects of life. Her guidance and support have been instrumental throughout my PhD journey. I miss our Tuesday and Friday meetings, where we delved into discussions about new research ideas and papers. Those sessions were not just about academic growth but also moments of inspiration and learning. I consider myself incredibly fortunate that you entrusted me as your first PhD student at TU Delft. This opportunity has been a profoundly valuable and enriching experience for me. Thank you, Lydia, for believing in me and for your unwavering support and encouragement.

A special thanks to my promoter, Dick Epema, for his support throughout my PhD. I have learned immensely from him, especially during the thesis writing process. His vast knowledge, experience, and critical feedback have significantly improved my writing skills. I deeply appreciate your guidance and support, which have been invaluable to my academic growth. Thank you, Dick, for your dedication and encouragement throughout this journey.

I would like to extend my deep gratitude to Robert Birke, my co-supervisor, for his invaluable support throughout my PhD. You have been incredibly kind, smart, and supportive, and I have truly enjoyed every moment of collaborating with you. Your readiness to help at any time and your constant availability for discussions have been immensely appreciated. Thank you for your guidance and friendship. I look forward to continuing our connection in the future.

I would like to thank my committee members, Matthijs Spaan, Marten van Dijk, Avishek Anand, Pin-Yu Chen, Robert Birke, and George Smaragdakis, for taking the time to review my thesis and being a part of my PhD defense.

I would also like to thank my colleagues at Shell. A special thanks to Hadi, who trusted me to be part of his team. You are not only my manager but also a mentor and good friend. Your advice on improving various aspects of research, work, and life has been invaluable. Many thanks to my colleague Ojas, who is exceptionally talented and smart, and who onboarded me when I joined Shell. I wish you great success in your PhD at TU

Delft. Desmond, you have been a fantastic friend and collaborator. Anuj, I appreciate your readiness to help with any questions and our engaging discussions on various scientific topics and the big question, "Being happy or right?". Sieger, you are a great person, and I have truly enjoyed collaborating with you. I value your critical insights and have learned a lot from them. Stelios, you are very smart and a truely researcher, still waiting for the list of the best places to visit in Greece.

Next, I had the privilege of being part of the Distributed Systems group, where I was fortunate to work alongside some truly wonderful colleagues: Johan, Jan, Stefanei, Kim, Sophie, and Jeremie. Jeremie, chatting about ideas, receiving your advice, and enjoying coffee together was always a pleasure.

I would also like to extend my gratitude to the members of the Blockchain Lab—Georgy, Martijn, Bulat, Rowdy, Vadim, and Sandip—for all the stimulating discussions and coffee break chats. Quinten, it was always fun talking with you, and I appreciated your efforts to speak Farsi. Can, you are not just a friend but like an older brother to me. I truly enjoyed every moment spent with you, from playing pool to learning from your incredible talent in playing various musical instruments. Your presence made this journey all the more enjoyable. It is my honor to have you as my friend.

Next, I would like to extend my thanks to my colleagues in the DML lab. Bart, I always enjoyed collaborating with you. You are not only a smart and dedicated person but also a good friend. A special thanks for helping me translate the summary into Dutch. Chi, you have been a great colleague, and your critical thinking and insightful questions during group meetings were always appreciated. Gill, thank you for taking over the organization of the group meetings after me. Jeroen, I always learned something new about programming from you. You are incredibly smart and a true geek in the best sense of the word. Aditya and Abel, it was my pleasure to be part of your master's supervisory team. I wish you both great success in your PhDs—you are both exceptionally talented.

I would like to thank my friends who have been a constant source of support and joy during my PhD journey. Spending time with you all has been invaluable. Ozzy, you were the best office mate—smart, funny, and always up for a good chat. Let's plan on watching the next Euro final at my place! Satwik, you are a great friend and an excellent Indian chef. I'm looking forward to learning your special recipes. Osman, "nasılsın abi?"—I always enjoyed our conversations about football, music, and everything else.

I would like to thank my Iranian friend. Ramin, you are the kind of friend everyone needs—supportive and dependable. I completely agree that we should continue our travels together. Yosra, you are a true friend who supports and helps in any situation. I'm very grateful to have such a great friend like you. Ahmad, every time we talk, you say, "I'll meet you in Tehran," but it never happens. You are a kind and true friend—come back to the Netherlands soon! Masoud, I'm so glad you finally made it to the Netherlands so we can relive the good old days. You are one of my best friends, and I truly value our friendship. Mahsa, I am truly happy to have you as one of my friends. Khatereh, it's always fun chatting with you, and your jokes never fail to make me laugh. Ali, I really miss our political discussions and deep conversations about humanity and psychology. Amir Deljouyi, you are incredibly kind and always open to discussing random topics, which I truly appreciate. Parsa, you are the only one who understands most of my 'Zard Pages' social media jokes. Mostafa, you are a great friend and the best companion for playing

games and competitions. Let's play more games together!

I would like to extend my thanks to my friends who, despite being in different countries, remain in close touch and continue to be a part of my life. My old friend Sohrab, you are like a brother to me. After 22 years of friendship, it's wonderful that we still chat, share our experiences, and support each other. Koohsa, you have always encouraged me at every stage of my life. Your support and motivation mean a lot to me. I hope we can meet soon!

It is also my honor to be friends with so many others who have enriched my life: Parviz, Zahra, Mousa, Farinaz, Mahsa, Fidel, AmirHossein, Mahtab, Khashayar, Kolsom, Sina, Saba, and Sobhan. Your friendship means a great deal to me.

Last but not least, I would like to express my deepest gratitude to the love of my life, Raha. You are the most lovely, kind, supportive, and patient person I could ask for. I am truly happy to have you in my life. Your presence is my peace and strength. My heartfelt thanks go to my parents, my mom Mahin, and my dad Reza. Your unconditional love and unwavering support have been my anchor through every challenge. Without your encouragement and backing, overcoming the difficulties of this journey would have been impossible. I also want to thank the best sister in the world, Mehrnaz, who has always been there to listen to my stories and offer support and encouragement. Your presence has been a constant source of strength for me. A special thanks to my mother-in-law, Atefeh, and my brother-in-law, Kasra, for their support and kindness throughout this journey. In memory of my aunt Shokouh, who passed away during COVID-19, I cherish the fond memories of her. Her spirit remains with me always.

SeyedAmirMasoud Ghiassi
Eindhoven, June 2024

CURRICULUM VITÆ

SeyedAmirMasoud GHIASSI

14-02-1994 Date of birth in Tehran, Iran.

EDUCATION

2012–2016 BSc. in Computer Engineering - Hardware
Thesis title: *Herding Algorithm Implementation on Khepera III*
Iran University of Science and Technology
Tehran, Iran

2016–2018 MSc. in Computer Engineering - Computer Network
Thesis title: *Energy and traffic aware workload offloading on mobile edge computing in 5G networks*
EASY Lab, Sharif University of Technology
Tehran, Iran

2019–2023 PhD Candidate
Dissertation title: *Label Alchemy: Transforming Noisy Data into Precious Insights in Deep Learning*
DIS Lab, Delft University of Technology
Delft, The Netherlands

WORK EXPERIENCE

2023-Present AI Researcher
Shell
Amsterdam, The Netherlands

LIST OF PUBLICATIONS

★ 1. **A. Ghiassi**, R. Birke, and L. Y. Chen, "Robust Learning via Golden Symmetric Loss of (un)Trusted Labels", *SIAM International Conference on Data Mining (SDM)*, pp. 568-576, 2023

★ 2. **A. Ghiassi**, R. Birke, and L. Y. Chen, "Multi Label Loss Correction against Missing and Corrupted Labels", *Asian Conference on Machine Learning (ACML)*, pp. 359-374, 2023

★ 3. **A. Ghiassi**, C. O. Pene, R. Birke, and L. Y. Chen, "Trusted Loss Correction for Noisy Multi-Label Learning", *Asian Conference on Machine Learning (ACML)*, pp. 343-358, 2023

★ 4. **A. Ghiassi**, R. Birke, and L. Y. Chen, "LABNET: A Collaborative Method for DNN Training and Label Aggregation", *International Conference on Agents and Artificial Intelligence (ICAART)*, pp. 56-66, 2022

5. C. Hong, **A. Ghiassi**, Y. Zhou, R. Birke, and L. Y. Chen, "Online Label Aggregation: A Variational Bayesian Approach", *The Web Conference (WWW)*, pp. 1904-1915, 2021

★ 6. **A. Ghiassi**, R. Birke, and L. Y. Chen, "LABELNET: Recoveries Noisy Labels", *International Joint Conference on Neural Networks (IJCNN)*, pp. 1-8, 2021

7. B. Cox, J. Galjaard, **A. Ghiassi**, R. Birke, and L. Y. Chen, "MASA: Responsive multi-DNN inference on the edge", *International Conference on Pervasive Computing and Communications (PerCom)*, pp. 1-10, 2021

8. J. Galjaard, B. Cox, **A. Ghiassi**, R. Birke, and L. Y. Chen, "MEMA: Fast inference of multiple deep models", *International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pp. 281-286, 2021

★ 9. **A. Ghiassi**, R. Birke, and L. Y. Chen, "TrustNet: Learning from Trusted Data Against (A)symmetric Label Noise", *International Conference on Big Data Computing, Applications and Technologies (BDCAT)*, pp. 52-62, 2021.

★ 10. T. Younesian, Z. Zhao, **A. Ghiassi**, R. Birke, and L. Y. Chen, "QActor: Active learning on noisy labels", *Asian Conference on Machine Learning (ACML)*, pp. 548-563, 2021

11. T. Younesian, C. Hong, **A. Ghiassi**, R. Birke, and L. Y. Chen, "End-to-End Learning from Noisy Crowd to Supervised Machine Learning Models", *International Conference on Cognitive Machine Intelligence (CogMI)*, pp. 17-26, 2020

12. **A. Ghiassi**, T. Younesian, Z. Zhao, R. Birke, V. Schiavoni, and L. Y. Chen, "Robust (Deep) Learning Framework Against Dirty Labels and Beyond", *International Conference on Trust, Privacy and Security in Intelligent Systems, and Applications (TPS)*, pp. 236-244, 2019

★ Included in this thesis.