DETECTION OF CLEAN SAMPLES IN NOISY LABELLED DATASETS VIA ANALYSIS OF ARTIFICIALLY CORRUPTED SAMPLES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

BOTAN YILDIRIM

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

AUGUST 2022

Approval of the thesis:

DETECTION OF CLEAN SAMPLES IN NOISY LABELLED DATASETS VIA ANALYSIS OF ARTIFICIALLY CORRUPTED SAMPLES

submitted by **BOTAN YILDIRIM** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. İlkay Ulusoy
Head of Department, **Electrical and Electronics Engineering** _____

Prof. Dr. İlkay Ulusoy
Supervisor, **Electrical and Electronics Engineering, METU** _____

Examining Committee Members:

Prof. Dr. Ece Güran Schmidt
Electrical and Electronics Engineering, METU _____

Prof. Dr. İlkay Ulusoy
Electrical and Electronics Engineering, METU _____

Prof. Dr. Alptekin Temizel
Multimedia Informatics, METU _____

Prof. Dr. Ziya Telatar
Biomedical Engineering, BAŞKENT UNIVERSITY _____

Assist. Prof. Dr. Ahmed Hareedy
Electrical and Electronics Engineering, METU _____

Date: 22.08.2022

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Botan Yıldırım

Signature :

ABSTRACT

DETECTION OF CLEAN SAMPLES IN NOISY LABELLED DATASETS VIA ANALYSIS OF ARTIFICIALLY CORRUPTED SAMPLES

Yıldırım, Botan
M.S., Department of Electrical and Electronics Engineering
Supervisor: Prof. Dr. İlkay Ulusoy

August 2022, 77 pages

Recent advances in supervised deep learning methods have shown great successes in image classification but these methods are known to owe their success to massive amount of data with reliable labels. However, constructing large-scale datasets inevitably results with varying levels of label noise which degrades performance of the supervised deep learning based classifiers. In this thesis, we make an analysis of sample selection based label noise robust approaches by providing extensive experimental evaluation. First, adverse effects of memorization of the noisy samples are investigated over results of a base model. Second, importance of knowledge of noise rate is analyzed for approaches utilizing a prior about noise rate. Third, superiority of recent semi-supervised based robust approaches over supervised ones is proved. Additionally, synthetically corrupted controlled datasets are used to show effects of the noise rate over training performance. Finally, a new framework is proposed to classify samples as clean or noisy by investigating train loss dynamics. To avoid heavily tuned parameters during clean sample detection, proposed framework artificially corrupts a noisy dataset and utilizes these artificially corrupted samples in a clean/noisy voting process. Moreover, following recent semi-supervised learning based label noise ro-

bust methods, framework applies semi-supervised and contrastive learning after classification of samples as clean-noisy. Also, effect of the co-training approach during semi-supervised learning is investigated and its effectiveness is proved.

Keywords: noisy labelled classification dataset, clean labelled sample extraction, classifier neural networks, deep learning, semi-supervised learning, contrastive learning, co-training

ÖZ

SANAL OLARAK KİRLETİLMİŞ ÖRNEKLEMLERİN ANALİZİ ARACILIĞIYLA GÜRÜLTÜLÜ ETİKETLENMİŞ VERİ SETLERİNDE TEMİZ ÖRNEKLEM TESPİTİ

Yıldırım, Botan
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. İlkay Ulusoy

Ağustos 2022 , 77 sayfa

Güdümlü derin öğrenme metodlarındaki son gelişmeler görüntü sınıflandırmada büyük başarılar sergilemiştir fakat bu methodlar başarılarını çok miktarda güvenilir etiketli veriye borçludur. Ancak büyük boyutlu veri setleri oluşturmak, kaçınılamaz olarak değişken seviyelerde gürültülü etiketlerle sonuçlanmaktadır ve bu durum, güdümlü derin öğrenme tabanlı sınıflandırıcıların performansını bozmaktadır. Bu tezde, etiket gürültüsüne karşı gürbüz olan, örneklem seçme tabanlı yöntemler kapsamlı deneysel değerlendirmeler sağlanarak analiz edilmiştir. İlk olarak, gürültülü örneklemleri ezberlemenin kötü yanları temel bir methodun sonuçlarına bakılarak incelenmiştir. İkinci olarak, gürültü bilgisinden faydalanan yöntemler için gürültü seviyesini bilmenin önemi analiz edilmiştir. Üçüncü olarak, yakın geçmişte önerilen, yarı güdümlü yapay öğrenme tabanlı ve gürbüz yöntemlerin güdümlü olanlara üstünlüğü kanıtlanmıştır. Ekstradan, yapay bir şekilde kirletilen kontrollü veri setleri, gürültü seviyesinin eğitim performansı üzerindeki etkisini göstermek için kullanılmışltır. Son olarak, örneklemleri temiz ya da gürültülü diye sınıflandırmak için eğitim kayıp değerlerini

inceleyen yeni bir yapı önerilmiştir. Temiz örneklem tespiti esnasında aşırı derecede hassas ayarlanmış paremetreleri önlemek amacıyla önerilen yöntem, gürültülü veri setini sanal bir şekilde ekstradan kirletmekte ve bu yeni, sanal gürültülü örneklemleri temiz/gürültülü oylama işlemi esnasında kullanmaktadır. Ekstradan, son zamanlarda önerilen, yarı güdümlü derin öğrenme tabanlı ve etiket gürültüsüne gürbüz yöntemlere benzer bir şekilde, önerdiğimiz yöntem, örneklemlerin temiz-gürültülü sınıflandırmasından sonra yarı güdümlü ve karşılaştırmalı öğrenmeden faydalanmaktadır. Ayrıca, eş eğitim yaklaşımının yarı güdümlü eğitim esnasındaki etkisi incelenmiş ve yararlılığı kanıtlanmıştır.

Anahtar Kelimeler: etiket gürültülü sınıflandırma veri setleri, temiz etiketli örneklem ayıklama, sınıflandırıc sinir ağları, derin öğrenme, yarı güdümlü öğrenme, karşılatırmalı öğrenme, eş eğitim

To my beloved family

ACKNOWLEDGMENTS

First of all, I would like to thank my supervisor Prof. Dr. İlkay Ulusoy for her supervision, efforts, encouragement, and incredible support throughout my thesis work. I developed my knowledge a lot during my thesis with her help. I learnt how to approach a scientific problem, how to divide a problem into pieces, and how to make scientific analyses under her supervision. I know her teachings will be helpful throughout my life. I would like to express my gratitude to Prof. Dr. İlkay Ulusoy for accepting me as her student and believing in me.

Secondly, I would like to thank my great family, who is always very close to me and supported me during not only my thesis but also my whole life. Especially, I would like to thank my mom and father, who raised me, always cared about me, listened to me without getting bored, and helped me throughout my thesis. Also, I would like to thank my brothers, who have helped me during my problems, and critical decisions and also guided me throughout my life.

I also would like to thank Alp Eren Sarı, Ahmet Aksakal, Esat Kalfaoğlu, Konstantin Tsoi, who are both my great friends and excellent colleagues. I seek their help whenever I need and also, they always listen to me without getting bored no matter what I say.

This work is supported by TÜBİTAK within the scope of 2210/A scholarship.

The experiments reported in this paper were performed at TÜBİTAK ULAKBIM, High Performance and Grid Computing Center (TRUBA resources).

I would like to thank TÜBİTAK for its supports to scientific researches.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

DL	Deep Learning
GMM	Gaussian Mixture Model
KL	Kullback Leibler divergence
JSD	Jensen Shannon Divergence
SSL	Semi-Supervised Learning
CL	Contrastive Learning
AUM	Area Under Margin

CHAPTER 1

INTRODUCTION

1.1 Motivation and Problem Definition

Deep neural networks are exhibiting state of the art results in countless computer vision problems for a while and one of the most studied problem in deep neural network context is image classification [4], [5], [6], [7]. Despite the impressive learning ability of deep learning-based classifiers, they mainly owe their success to the massive amount of collected classification data with reliable labels such as ImageNet[8], MS-COCO[9], etc. The term reliable labels refer to clean(correct) labels while incorrectly labeled samples are called noisy labeled samples which means that the given label of the sample is not the actual one. Some examples of noisy labeled samples from Clothing1M[2] dataset are given in FIG. 1.1.

Figure 1.1: Some examples of noisy labelled samples. Labels written as red: given labels, Labels written as green: correct labels.

Constructing large-scale classification datasets with clean labels is a challenging problem since the annotation process is expensive and time-consuming. Moreover, expert knowledge is a must for some datasets such as CUB-200[10], which requires to be labelled by ornithologists, and ROP[11], which is annotated by three different medical

doctors. Despite the spent time and expert's knowledge, ROP[11] states %27 disagreement among labels of three experts which indicates the difficulty of obtaining medical datasets with clean labels.

Some popular methods to handle labelling problems are searching images with tags from social media[12], querying commercial search engines[13] and using meta-data such as texts that are linked to an image downloaded from a webpage. These appealing and uncostly solutions uncontrollably result with noisy labelled samples. An example of a noisy labelled sample can be observed in Figure 1.2. Both images given in Figure 1.2 are searched with "apple" tag in a search engine and, 1.2a is obtained as a noisy sample which contains an image of a phone from the Apple brand while 1.2b is an example of a clean sample since it is a real apple image. The example given in 1.2 shows how samples obtained from search engines can result with noisy labels.

(a) An Apple brand phone image

(b) A real apple image

Figure 1.2: An example case of noisy labeled sample obtained from a search engine. The images are taken from store.storeimages.cdn-apple.com and media.istockphoto.

As stated, noisy labels can be a natural outcome of the difficult labeling process as in the case of ROP[11] or outcome of the proposed solutions to construct large-scale datasets in a fast and cheap way. Therefore, learning from noisy labeled datasets becomes an important area of research to find answers to how noisy labels affect deep neural networks based classifiers and how one can train a classifier neural network to be robust to label noise. Once, these mentioned questions are answered correctly, the power of easy data collection processes and label noise robust deep neural networks can be combined to solve various classification problems easily and in a fast way.

There have been various attempts in the literature to understand the effects of noisy labels. Although deep neural networks are powerful tools that can generalize very well despite training with noisy samples[14], they still have a tendency to overfit noisy labels[15], [16] because of their large number of parameters. Overfitting can be observed even over complete random noise[17] although there is not any underlying logical structure. Therefore, handling memorization problem is becoming more crucial to avoid performance degradation caused by noisy labelled datasets and so, several approaches are proposed in the literature nowadays [18], [19], [20], [3], [21], [22], [23], [24] by utilizing different strategies.

1.2 Scope and Contributions of the Thesis

This study mainly focused on sample selection-based label noise robust classifiers. Most of the sample selection-based methods report their performances over specific datasets with some tuned parameter sets. This thesis prepares a common test setup using multiple datasets with varying conditions to provide a fair comparison of sample selection-based methods. A detailed analysis of these methods over multiple cases is done to show their weak and effective sides.

The second focus of this thesis is decreasing the number of parameters that need to be tuned and avoiding any need for a clean validation set. Some approaches make assumptions about the noise rate of the datasets or utilize probabilistic models with heavily tuned parameters during the detection of the clean samples by claiming that noise rate estimation or other parameters can be tuned via a small clean validation set. However, a clean validation set may not be available in most real-world scenarios and so, performing validation to tune required parameters may not be possible. Therefore, this study focused on a solution to avoid any requirement of a clean validation set and decrease the number of parameters to be tuned during the clean-noisy classification of samples. The proposed work further corrupts the given noisy dataset artificially to obtain surely noisy samples and classify the remaining samples as clean-noisy utilizing surely noisy samples in an offline voting pipeline. The proposed solution follows a different way from most of the methods that exist in the literature and so, brings a

novelty to the literature. Additionally, an analysis related to the differences between synthetic and real-world noises is provided in our proposed method.

Recent works proved the effectiveness of the semi-supervised and contrastive learning approaches within the context of learning in the existence of label noise. Similarly, the proposed method utilizes a semi-supervised learning strategy, which is well used in recent years in label noise literature, after offline noisy-clean classification of samples. This thesis also investigates the effect of the co-training approach on performance during the semi-supervised learning part of our proposed method by training neural networks with and without co-training.

A summary of the contributions of this thesis can be given as:

- Extensive experimentation over a common setup to provide a fair comparison of the samples selection-based label noise-robust methods.
- A novel approach corrupts a noisy dataset artificially to obtain surely noisy samples and utilizes surely noisy samples in an offline voting pipeline to detect clean samples without using a clean validation set or heavy parameter tuning.
- An analysis related to differences between synthetic and real-world noise.
- An analysis of the co-training approach during utilizing semi-supervised learning over samples classified as noisy and clean in the offline voting pipeline.

1.3 The Outline of the Thesis

The work presented in this thesis consists of five chapters. In Chapter 1, the problem of interest, our motivation, the scope of the work, and contributions are introduced.

In Chapter 2, a detailed formulation of the problem, required background information, and proposed solutions of the described problem from the literature are given.

In Chapter 3, the proposed method by this work to solve the problem of interest is explained in detail.

In Chapter 4, an extensive experimental analysis of the literature and the proposed method of this thesis is presented for a better understanding of the described problem and proposed solutions. In this chapter, detailed information about datasets utilized in experiments, experimental setups and detailed experimental results with findings are given in Sections 4.1, 4.2, 4.3 respectively. Explanation of the experimental setup given in Section 4.2 includes information related to implemented algorithms, their network structures, and strategies applied during trainings. Also, an analysis of sample losses and clean sample extraction performance of the proposed method of this thesis are presented in Section 4.4 and Section 4.5 respectively.

In Chapter 5, an overall summary of this thesis and some conclusions based on experiments are given.

CHAPTER 2

BACKGROUND AND RELATED WORK

For a better understanding of the study presented in this thesis, required background information and related references from the literature are given in this chapter. Firstly, a detailed formulation of the problem of interest is presented in section 2.1. Secondly, some label noise models utilized during synthetic noise generation processes to simulate real-world label noise are presented in section 2.2. Finally, some notable solutions proposed in the literature to handle label noise are given in section 2.3.

2.1 Formulation of Classification Problem in Existence of Noisy Labels

A traditional supervised classification problem is formulated with a training dataset, a mapping function, parameters of the mapping function, a quality measure for predictions(generally a loss function), and an objective function to be minimized during parameter optimization.

Training dataset is represented as $D^{train} = \{(x_1, y_1), ..., (x_N, y_N)\}$ where $(x_i, y_i) \in (X, Y)^N$ and sampled according to a distribution P over (X, Y). Here, x_i, y_i, N represent i^{th} image, label of i^{th} image and number of samples in the train dataset respectively. y_i is C dimensional one hot vector where C is the number of total classes in the dataset.

The supervised classification problem's goal is to learn a function $f : X- > Y$ parametrized with θ that maps the input image to a class label. The quality of predictions/mappings is generally measured with the use of a loss function $L(f_\theta(x), y)$.

Optimal θ parameters of the function are obtained by an optimization process whose

objective function is set to the expected risk and formulated as follows:

$$R_{L,P}(f_\theta) = E_P[L(f_\theta(x), y)] \tag{2.1}$$

where E_p denotes the expectation over distribution P.

The distribution P is generally unknown, so computing the expected risk is unfeasible. Instead of minimizing expected risk, a general approach is minimizing empirical risk, which is defined as follows:

$$\hat{R}_{L,D^{train}}(f_\theta) = \frac{1}{N}\sum_{i=1}^{N} L(f_\theta(x), y) \tag{2.2}$$

Then, the exact solution of the traditional classification problem is formulated as follows:

$$\theta^\star = \arg\min_\theta \hat{R}_{L,D^{train}}(f_\theta) \tag{2.3}$$

On the other hand, the classification problem forms a new shape in a noisy label setup compared to the traditional one. Noisy labelled training dataset is represented as $D_n^{train} = \{(x_1, \tilde{y}_1), ..., (x_N, \tilde{y}_N)\}$ where $(x_N, \tilde{y}_N)$ is sampled according to a noisy distribution P_n. In this case, the solution of the classification problem is formulated as follows:

$$\theta_n^\star = \arg\min_\theta \hat{R}_{L,D_n^{train}}(f_\theta) \tag{2.4}$$

Optimizing the mapping function parameters θ in noisy label setup results with a parameter set diverged from the optimal one.

$$\theta_n^\star \neq \theta^\star \tag{2.5}$$

The goal in classification problem with noisy label setup is obtaining optimal parameter set $\theta^\star$ while only D_n^{train} is available instead of D^{train}.

As an obvious conclusion, applying the optimization process to classifier parameters θ according to traditional supervised classification approaches is not enough to converge to the optimal parameter set. Therefore, various training strategies are proposed in the literature to find the best classifier parameters in the existence of label noise.

2.2 An Overview of Noise Types

Developing and testing noise robust deep neural networks in a controlled manner are challenging problems since the lack of reliable labels makes it impossible to construct desired setups with specific noise rates. Thus, various methods proposed in the literature follow a prevalent approach, such as corrupting the desired amount of samples of a clean dataset to create a noisy labeled dataset synthetically. These synthetically generated noisy labeled datasets can be viewed as a simulation of the noisy real-world datasets.

Various label noise models are proposed and utilized in the literature to construct noisy labeled datasets. The most known types are uniform noise, class-dependent noise, and feature-dependent noise.

2.2.1 Uniform Label Noise

In this noise model, the probability of misclassifying a sample from its true class to any other class is uniformly distributed. Many proposed methods in the literature basically choose desired amount of samples in a clean dataset and flip their labels following a uniform distribution[20], [3]. This process can be represented by a noise transition matrix whose entries are formulated as follows:

$$L_{mn} = \begin{cases} p & \text{if } m = n \\ \dfrac{1-p}{N-1} & \text{if } m \neq n \end{cases} \tag{2.6}$$

where L_{mn} represents the probability of a sample from class m to be misclassified with class n, $1-p$ is the noise rate of the corrupted dataset, and N is the number of total classes in the dataset.

2.2.2 Class Dependent Label Noise

In this noise type, the misclassification probability of a sample depends on its true class. This can be achieved either by assigning random probabilities to inter-class

transitions [25] or taking class similarities into consideration during probability assignment. One work[26] applies corruption uniformly between similar classes such as "bed" and "couch" but not "bed" and "beaver" in CIFAR100[27]. Some other works [28], [3] follow a special corruption strategy named as "pair flipping" where misclassification is possible only from one class to its neighbor classes. One another recent work[29] follows a more structured way by utilizing a deep neural network's predictions over a test set to construct class dependent noise transition matrix.

2.2.3 Feature Dependent Label Noise

In this case, the misclassification probability of a sample depends on the similarity between its feature and the features of the other samples. Feature-dependent noise is a more complicated noise model compared to uniform and class-dependent label noise models since its corruption process requires extracting features of each sample and finding similarities between them in the feature domain; hence it is rarely referred in the literature. One recent work[29] applies knowledge distillation[30] idea to corrupt a clean dataset with feature dependent noise. In this work, a teacher network is trained over clean samples, and then a weighted sum of teacher network predictions and original hard labels are used to create soft labels for each instance. This approach claims that training a student network(a second network) with soft labels, which are generated by the teacher network, results with a more sparse distribution in data feature space compared to direct training over hard labels. After training the student network, the softmax probability of each instance is checked to find instances that have similar features, and the labels of these instances are corrupted according to similarities.

2.3 Related Work on Learning Classifiers in Existence of Label Noise

Training robust classifier models over noisy labeled datasets goes back a long way [31], and numerous methods have been presented in the literature to avoid adverse effects of label noise. The works presented in the literature are grouped under various groups in different works [1], [32], [33], [34]. Following a similar way with literature,

we collect these proposed methods under five main groups, which are "Noise Transition Matrix Estimation", "Sample Importance Weighting", "Robust Losses", "Meta Learning" and "Sample Selection" in this section. The main focus of the work presented in this thesis falls into "Sample Selection" group, and so the proposed methods of this group are given in a more detailed way compared to other ones, and an experimental evaluation of these works is presented in Chapter 3.

2.3.1 Noise Transition Matrix Estimation

Noise Transition Matrix Estimation approaches mainly analyze training datasets to discover an underlying structure between instances and their given classes. As an output of this analysis, they construct a matrix named as noise transition matrix to represent noise that exists in the dataset. FIGURE 2.1 presents two examples of the described matrix for two different noise types, and the probability given in each entry of this matrix represents the probability of labeling an instance from True Label as Noisy Label.

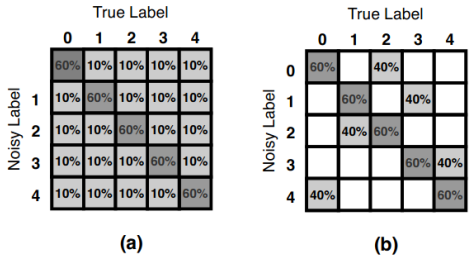

Figure 2.1: Two examples of noise transition matrix: (a) symmetric noise, (b) asymmetric noise, for a dataset with %40 noise rate and 5 classes. Image is taken from [1].

There are some works such as [35], [36], [37] which are utilizing points named as anchor points for estimating noise transition matrix. Anchor point term is used for instances with almost surely clean labels. In a more formal way, a sample $x^i \in X$ is described as an anchor point of the i^{th} class if $P(Y = i|x^i) = 1$ [35]. For these anchor points, the transition matrix can be estimated as follows by assuming that the noise transition matrix is independent of instances and noisy class posterior probabilities are known:

$$\hat{P}(\tilde{Y}=j|x^i)=\sum_{k=1}^{C}\hat{P}(\tilde{Y}=j|Y=k,x^i)P(Y=k|x^i)=\hat{P}(\tilde{Y}=j|Y=i,x)=\hat{P}_{ij} \tag{2.7}$$

Here, $\hat{P}_{ij}$ represents entry of the transition matrix in j^{th} row, i^{th} column. $P(Y=k|x^i)$ is non-zero for only $k=i$ since it is an anchor point.
Noisy class posterior($\hat{P}(\tilde{Y}=j|Y=i,x)$) used in the formula can be estimated by training a deep neural network and using its class probability estimations for anchor points.

Once an estimation for the transition matrix is obtained, a popular approach is utilizing the matrix in the construction of a noise-aware-corrected loss. [38] formulates a new loss $l_{corrected}$ using noise transition matrix instead of original loss l such as minimizing $l_{corrected}$ over noisy data corresponds to minimizing l over clean data. [39] adds an additional layer at the top of the softmax layer using transition matrix to convert clean class probability into noisy class probability, and then this additional layer is removed at the end of training to provide clean class estimation for instances. [40] further works on idea of [39] and names the approach of multiplying network predictions with noise transition matrix as "Forward Loss Correction".

2.3.2 Sample Importance Weighting

Sample Importance Weighting approaches introduce a weighting scheme over instances of the training dataset to decrease contribution from possible noisy samples and avoid their adverse effects. Weighting is applied at loss calculation step such as:

$$\hat{R}_{l,D_n}=\frac{1}{N}\sum_{k=1}^{N}w(x_k)l(f_\theta(x_k),\tilde{y}_i) \tag{2.8}$$

where D_n is noisy labelled dataset, $\hat{R}_{l,D_n}$ is empirical loss over noisy dataset, x_k is the k^{th} instance of dataset, f_θ is neural network, $\tilde{y}_i$ is noisy label and $w(x_k)$ is the weight of the k^{th} instance decided by Sample Importance Weighting algorithm.

[41] assigns a weight of 1 to detected possible clean samples on softmax loss calculation, while a weight smaller than 1 is assigned to detected possible noisy samples,

and this smaller weight is adjusted according to the probability of being noisy. [42] is another sample weighting method based on meta-learning. This approach decides the weights of the training samples by utilizing another small clean dataset as a validation dataset. Loss over the clean validation dataset is formulated as an objective function of train sample weights, and the objective function is minimized to find the optimum weights of each training sample. [43] combines meta-learning idea with a multi-layer perceptron to regress weights of each sample instead of direct assignment. Here, MLP regresses a weight for each sample by taking loss coming from the main network as input, and regression is again done by minimizing classification loss over a clean validation subset. [44] follows a different approach based on class prototype extraction using autoencoders. Class prototypes are constructed with a clean reference subset which is thought to be representative enough for each class. During training, each sample is weighted in loss calculation according to the similarity between its feature vector and the prototype of its given class. [45] proposes weighting samples in loss calculation according to their gradient value. During training, a low gradient of a sample indicates that the sample is easy to learn and has a small loss in loss calculation. These easy-to-learn samples are most likely to be correctly labeled; therefore, samples with low gradient values are emphasized with a large weight in loss calculation.

2.3.3 Robust Losses

Robust Loss approaches propose modifying loss function to prevent adverse effects of noisy labeled instances during training. [46] provides an analysis to understand what makes a loss function robust against label noise. In their analysis, they compared mean absolute error(MAE), mean square error(MSE), and cross-entropy(CE) losses. The compared MAE, MSE and CE losses are given in Equations 2.9, 2.10, 2.11 respectively.

$$L_{MAE} = \frac{1}{C}\sum_{i=0}^{C} |p(y=i|x) - \hat{p}(y=i|x)| \tag{2.9}$$

$$L_{MSE} = \frac{1}{C}\sum_{i=0}^{C} |p(y=i|x) - \hat{p}(y=i|x)|^2 \tag{2.10}$$

$$L_{CE} = \frac{1}{C} \sum_{i=0}^{C} p(y = i|x) \log(\hat{p}(y = i|x)) \tag{2.11}$$

According to the claim of this work, MAE is simple yet robust to label noise, while MSE and CE are not. CE loss is known for its tendency to learn from hard samples more compared to easy ones, while MAE is affected by all samples more uniformly. This property of the CE loss is useful during training with a clean dataset, but during training with noisy samples, hard samples are more likely to be noisy, so CE is less robust than MAE. Additionally, experimental works of [46] show that MSE has a robustness between MAE and CE. [47] further analyses the robustness of MAE to noisy labels and claims that MAE's robustness comes from its ability of emphasising uncertain instances and not from treating all samples uniformly as claimed in previous works. Also, they mentioned a problem of MAE, like its tendency to underfit data due to its small weight variance over samples. Small weight variance over samples prevents learning from even far different and rare instances. They propose a new robust loss named as IMAE, which is a modified version of MAE in order to solve the underfitting problem of MAE by keeping its robustness against to label noise. IMAE solves the underfitting problem by appropriately adjusting the weight variance of MAE. On the other hand, [48] proposes a study to solve the robustness problem of CE loss. They showed that DNNs tend to overfit to noisy samples and underfit to remaining ones, even to hard ones with CE loss, since hard samples are easy to learn compared to noisy ones during training. They tried to modify CE loss with an additional term to make learning from hard samples possible while providing noise tolerance. They are inspired by symmetric KL divergence and strengthen CE with an additional symmetric, noise-robust counterpart term named as Reverse Cross Entropy. Also, both theoretical and empirical evidence of the robustness of the modified CE loss function is provided.

2.3.4 Meta Learning

Deep learning approaches eliminate the need for hand-crafted feature extraction in most machine learning problems. Still, these approaches need many parameter-tool tunings, such as optimizer type, learning rate, loss functions, sample weights, network design, etc. Meta-learning is a way of learning to learn itself by avoiding these

mentioned parameter-tool tuning processes [49], [50]. Meta-learning approaches describe a meta objective according to desired parameter-tool tuning task and optimize its original model and its training procedure by minimizing meta objective function. [42] utilizes meta-learning to reweight its samples during training, and so, it falls into both Sample Importance Weighting and Meta-Learning groups. The meta objective function of the work is defined as a loss over a small clean validation dataset, and variables of the objective function are defined as weights of the noisy samples used during training. [51] follows a different approach based on the teacher-student network strategy. Here, the teacher network is trained over an available clean dataset to transfer its knowledge to the student network during training over the main noisy dataset, and the process is called as knowledge distillation [30]. Teacher network trained over clean dataset acts as a source of variance for noisy dataset to eliminate default variance of dataset caused by noisy labels. This elimination process is done by defining the loss function of the student network as a weighted sum of the losses between student network predictions and noisy labels, and losses between student network predictions and teacher network predictions. [52] utilizes two networks such as student and teacher networks, and proposes a meta objective function to be used before traditional backward propagation of student network. Their meta-learning algorithm generates multiple clones of the original batch with noisy labels during each mini-batch. For each generated batch with noisy labels, a student network update is done, and then a consistency loss is calculated between predictions of the student and teacher networks. Here, the proposed meta objective of the algorithm is consistency loss between teacher and student networks. In this way, the student network is forced to provide consistent predictions with the teacher network, so the student network does not overfit to generated noisy labels. On the other hand, the teacher network is updated as an exponential moving average of the student network, which makes it reliable. After minimization of the meta objective, the student network follows a traditional parameter update over original labels.

2.3.5 Sample Selection

The sample selection strategy mainly groups instances in the training dataset as clean and noisy to avoid any weight update coming from noisy instances during training.

Decoupling[18] states that wrong predictions result with a weight update which makes the classifier better. Since predictions are most likely to be wrong at initial iterations compared to the last ones, there are fewer updates at the last iterations compared to initial ones. On the other hand, a neural network memorizes clean samples at initial iterations and then memorizes noisy samples as training progresses[16]. Both these statements mean that updates are mainly done by noisy samples at the last iterations of training since predictions of noisy samples are likely to be wrong because of their incorrect labels. Decoupling proposes checking whether a sample is worthy to update or not using an update criterion to decrease the effect of noisy samples at the last iterations. For this purpose, Decoupling trains two randomly initialized networks simultaneously and updates both network parameters using samples that fall into the disagreement region, which naturally decreases the number of samples in weight update at the last iterations.

MentorNet[19] introduces "learning to teach" approach to learning with noisy labels task. "learning to teach" approach consists of two networks named as teacher and student networks, where the pre-trained teacher network leads to the student network during training for useful sample extraction. MentorNet uses an LSTM model named as MentorNet(teacher network) to supply a curriculum to the student network. The curriculum is a weighting scheme such as provided weights by MentorNet are used as coefficients of training samples in loss calculation, and these provided weights increase the importance of probably clean samples in student network train loss. One another approach, which MentorNet proposes, is using a predefined curriculum instead of using a teacher network. This predefined curriculum is based on choosing small loss instances as probably correct samples and ignoring the remaining ones in loss calculations. This predefined curriculum-based MentorNet version is known as self-paced MentorNet.

Co-teaching[20] utilizes small loss observation[16], which proves that clean samples are most likely to have small losses compared to noisy samples. Co-teaching trains two networks simultaneously, and during each iteration, a desired percentage of samples are selected from small loss instances, possible clean samples, for each network independently. Here, the desired percentage of small loss instances is found by a

variable named as remember rate, which is formulated as follows:

$$R(T) = 1 - min(\frac{T}{T_k}\tau, \tau) \tag{2.12}$$

Here, τ depends on the noise rate of the dataset and ideally should be set to $1 - \epsilon$ if the noise rate is ϵ. T_k is an epoch threshold used for warm-up, such as all samples are used at the first epoch, and the rate of used samples is decreased to τ until T_k.
After sampling small loss instances, each network is updated by its peer network's small loss instances, which can be viewed as each network teaches to the other by sharing its knowledge. The main motivation behind using two networks is utilizing different learning abilities for filtering different types of errors during training.

Co-teaching+[3] states that networks used in Co-teaching[20] converge to a consensus as training progresses, and so, the Co-teaching algorithm turns into self-paced MentorNet[19] which suffers from accumulated error caused by sample selection bias. To prevent convergence of Co-teaching to self-paced MentorNet, networks used in the Co-teaching algorithm should be kept diverged during training. Co-teaching+ observed that the "Disagreement" strategy introduced by Decoupling[18] is a useful tool for keeping two networks diverged from each other during training, while the "Disagreement" strategy can not directly handle noisy label problem effectively. Co-teaching+ unifies Co-teaching and Decoupling algorithms to obtain a more powerful tool for combating noisy labels. Co-teaching+ mainly proposed using two networks during training, getting samples in the disagreement region, choosing small loss instances in the disagreement region for both networks, and updating the weights of a network using its peer network's small loss instances.

JoCoR[21] suspects necessity of "Disagreement" approach used by both Decoupling[18] and Co-teaching+[3] for training two networks to handle noisy labels. Noisy samples also exist in the disagreement region, and so, Decoupling is not successful enough to handle the noisy label problem. On the other hand, Co-teaching+ suffers from the insufficient number of samples at high noise rates since it selects samples utilizing a strategy such as taking the desired percentage of small loss instances in the disagreement region, which filters most of the samples. These observations related to the "Disagreement" strategy indicate that it is not a good approach for handling the noisy label problem. JoCoR also trains two networks simultaneously but this time us-

ing a single joint loss function which also includes a regularization term to decrease divergence between networks. This joint loss term is the summation of standard classification losses from both networks and an additional regularization loss named as contrastive loss. Contrastive loss is defined as symmetric KL divergence between predictions of a sample coming from two networks which helps networks to reach a consensus in predictions. JoCoR claims that both networks are more likely to agree on clean samples, while a disagreement exists for samples with incorrect labels, and agreement over a sample results with a small joint loss. For this reason, after calculating joint loss for each sample, JoCoR chooses the desired percentage of small loss instances for weight update.

Nested Co-teaching[22] is another promising work which combines a compression regularization method named as Nested Dropout[53] with Co-teaching[20] algorithm. Nested Dropout[53] provides importance ordered outputs at the end of a network, and it was originally proposed for adaptive data compression. Since Nested Dropout provides importance ordered outputs, one can easily remove meaningless parts of the output by cutting it from the desired dimension. This nice property of Nested Dropout can be easily utilized during learning with noisy labels since representations learned by noisy data are also meaningless. Nested Co-teaching states a weakness of the Co-teaching algorithm related to the small loss selection strategy. They claim that a network should be reliable enough for the small loss selection strategy. Otherwise, an unreliable network can provide a small loss for a noisy sample while a large loss for a clean sample, resulting in incorrect selections. For this reason, they came up with a two stages strategy where two networks are trained with Nested Dropout approach to obtain two reliable base networks in the first stage, and two base networks are further fine-tuned using the Co-teaching algorithm.

SELF[23] is a recent work that utilizes the mean teacher network concept in noisy labeled training problem. The mean teacher approach utilizes a student and teacher network during training. The student network is trained over provided batches by method, while the teacher network is set to the moving average of the student network's weights. Each batch consists of possible clean and noisy samples detected by the algorithm. The clean-noisy detection process is also based on the moving average approach. All samples are assigned as clean at the initial iteration. As training

progresses, network prediction of each sample is used in moving average calculations to find a representative prediction for each sample. For the current iteration, a sample is assigned as possible noisy if its moving average prediction differs from its ground truth label. The student network utilizes all samples during loss calculation by applying supervised learning to possible clean samples and unsupervised learning to possible noisy samples. Clean samples are used in traditional classification loss calculations, while noisy samples are used in a consistency loss between student and teacher networks.

The work named Area Under the Margin Ranking[24] provides a novel approach based on the analysis of clean and noisy samples through an artificial noisy set. They mainly propose creating an artificial class that does not originally exist in the training dataset and assigning a desired percentage of samples to the created artificial class to obtain undoubtedly noisy samples. The purpose of creating surely noisy samples is to investigate their train dynamics to classify the remaining samples as clean or noisy. Train dynamics investigation is based on a metric named as Area Under Margin(AUM). AUM is the average of Margin values of a sample calculated at each iteration, and Margin is defined as the difference between the largest class score and the second largest class score for a sample. Their claim is that noisy samples have a tendency to have a small AUM value compared to clean samples, and one can apply a threshold to AUM values in the training dataset to classify samples as noisy or clean. The created artificial samples are used for setting the AUM threshold.

DIVIDEMIX[54] is another sample selection-based approach that utilizes multiple powerful tools from different deep learning domains. To achieve robust training, they use Co-training[55] idea by training two networks together. At the beginning of the algorithm, they start with warm-up training using cross-entropy loss with an additional penalty term which is simply negative cross entropy and avoids overconfident predictions. Penalty term slows down memorization of the noisy sample and provides a more even loss distribution during training. Then, they follow a similar way with [56] to separate samples as clean and noisy by fitting a Mixture of Gaussians to samples' loss values during training. Each network chooses clean samples for its peer network to avoid self-bias during the clean sample selection process. After the clean-noisy classification of the samples, clean samples are treated as labeled samples

while noisy ones as unlabelled samples. In this way, one set with labeled samples and one set with unlabeled samples are obtained, and these sets are utilized with a semi-supervised learning approach. During each iteration, one batch is taken from each set and passed through an augmentation process separately. Augmented versions of the samples from both sets are passed through networks to obtain label predictions. Then, the labels of the labeled set are updated by taking the weighted sum of the original label and current network predictions. As a next step, Mixup[57] augmentation is applied between the labeled set with updated labels and the unlabelled set with network label predictions. Mixup outputs are divided into two parts. In one part, samples from the labeled set are dominant, while in the other one, samples from the unlabeled set are dominant. Cross entropy loss is calculated for the part where labeled samples are dominant, and mean square error is calculated for the part where unlabelled samples are dominant. Also, the previously mentioned penalty loss is calculated for each mixup output. Finally, the weighted sum of cross-entropy loss, mean square error, and penalty loss is calculated for optimization.

UNICON[58] utilizes Co-training[55] idea with two networks for increasing robustness. Two networks start learning with warm-up training, and then each network classifies samples as clean or noisy using Jensen Shannon Divergence(JSD) for its peer network. JSD is a tool of probability theory that calculates divergence between two distributions, and so it actually shows the similarity between two distributions. UNICON calculates divergence between y_i and p_i where i is sample index, y_i is given label of sample, p_i is average of two networks' predictions(p_i^1, p_i^2) for sample of interest. This calculation is formulated as given in Eq. 2.13. The function KL given in the equation is the well-known Kullback–Leibler divergence.

$$\begin{aligned} p_i &= \tfrac{p_i^1+p_i^2}{2} \\ d_i &= JSD(y_i, p_i) = \tfrac{KL(y_i,m_i)+KL(p_i,m_i)}{2} \\ m_i &= \tfrac{yi+p_i}{2} \end{aligned} \tag{2.13}$$

UNICON classify a sample as noisy if calculated divergence(d_i in Eq. 2.13)is high. After the classification of samples, clean samples are treated as labeled samples, while noisy ones are treated as unlabeled samples. Similar to DIVIDEMIX, all samples are passed through an augmentation process, and augmented images are given to networks to predict their labels. Labels of the labeled set are updated by taking the

weighted sum of the original labels and network predictions. On the other hand, label predictions obtained from networks for the unlabeled set are used as sample labels in that iteration. Labeled set with updated labels and unlabeled set with network predictions are combined with Mixup[57] augmentation. Similar to DIVIDEMIX, Mixup outputs are divided into two parts, and in one of them, samples of the labeled set are dominant, while in the other one, samples of the unlabeled set are dominant. Cross entropy loss is calculated for the labeled set dominant part, and mean square error is calculated for the unlabeled set dominant part. Also, UNICON utilizes augmented images of the unlabeled set via contrastive learning and calculates a contrastive loss. As a final step, UNICON calculates a penalty term for each sample to avoid overconfident predictions. To find the final total loss, the weighted sum of the cross entropy loss, mean square error, contrastive loss, and penalty loss is calculated, and the total loss is used for optimization purpose.

To summarize all explained sample selection methods, we collect all of them by giving their some key features in Table 2.1. Explanation of these key features is given below.

- **Single / Two Network(s):** Some methods are utilizing two networks simultaneously during training to increase robustness. "Two" and "Single" keys correspond to two networks and single network usage respectively.
- **Clean Sample Selection Method:** Each method utilizes a different approach to filter clean samples. "SL" key means that approach utilizes small loss selection technique. "GT-P-C" corresponds to checking ground truth and network prediction consistency for clean sample detection. "AUM" refers to a special metric named as area under margin which is thresholded for each sample to classify it as clean or not. "GMM" corresponds to fitting Gaussian Mixture Models to sample losses to divide them as clean or noisy in a unsupervised manner. "JSD" refers to Jensen Shannon Divergence which is thresholded to detect clean samples.
- **Networks Agreement / Disagreement:** Some approaches with two networks utilize agreement or disagreement of the network predictions during training. Here, "D" and "A" corresponds to utilizing disagreement and agreement recpec-

tively.

- **Clean Data:** Some methods may utilize a clean validation dataset during training. Here, "+" means that method uses a clean dataset.
- **Noise Rate Info.:** Some methods utilize a prior information about noise rate during clean sample detection. Here, "+" means that method uses noise rate information.
- **Online Selection:** Clean-noisy classification of samples can be done during main training or as a pre-processing before main training. Here, online selection means that classification is done during main training.
- **Semi Supervised Learning:** Detected noisy samples can be utilized during training using Semi-Supervised Learning. Here, "+" means that method utilizes Semi-Supervised Learning.

Table 2.1: Properties of Sample Selection Algorithms

Algorithm Name	Single / Two Network(s)	Clean Sample Selection Method	Networks Agreement / Disagreement	Clean Data	Noise Rate Info.	Online Selection	Semi Supervised Learning
Decoupling	Two	-	D	-	-	+	-
SP MentorNet	Single	SL	-	-	+	+	-
Co-teaching	Two	SL	-	-	+	+	-
Co-teaching+	Two	SL	D	-	+	+	-
JoCoR	Two	SL	A	-	+	+	-
Nested Co-teaching	Two	SL	-	+	+	+	-
SELF	Two	GT-P-C	-	-	-	+	-
AUM Ranking	Single	AUM	-	-	-	-	-
DIVIDEMIX	Two	GMM	-	-	-	+	+
UNICON	Two	JSD	-	-	-	+	+

CHAPTER 3

PROPOSED METHOD: OFFLINE CLEAN-NOISY CLASSIFICATION OF SAMPLES VIA CONTROLLED ARTIFICIALLY NOISY SUBSET

Train loss dynamic of samples contains useful hints about the correctness of their labels. Multiple studies[19], [20], [3], [22] utilize small loss selection trick by claiming that a neural network learns from clean samples first and then, overfits to noisy samples. Consequently, clean samples' loss are small compared to noisy ones until overfitting starts. Power of this simple approach has been extensively investigated and proved in [19], [20], [3] and [22]. Some important weaknesses of these studies are predefined hyperparameters and assumptions related to the noise rate of the dataset. The amount of small loss samples to be chosen as clean depends on the real noise rate of the dataset, and performance degrades if noise rate estimation differs from the exact rate. Noise rate can be estimated via validation if a small clean validation set exists, but this is not possible for most real-world cases.

In this chapter, we propose a pipeline, which is based on train loss dynamic investigation utilizing artificially corrupted samples of a noisy dataset, for classifying samples as clean or noisy without any assumption related to noise rate. Further, we utilize samples via Semi-Supervised Learning after classifying them as clean or noisy.

3.1 Methodology

3.1.1 Overview

An overall diagram of our proposed approach is given in Figure 3.1. Our approach can mainly be divided into three parts such as artificially corruption of a noisy dataset to

obtain a controlled setup with surely noisy samples, classification of samples as clean-noisy utilizing artificially corrupted samples, and a final semi-supervised learning-based training process using predicted clean and noisy subsets.

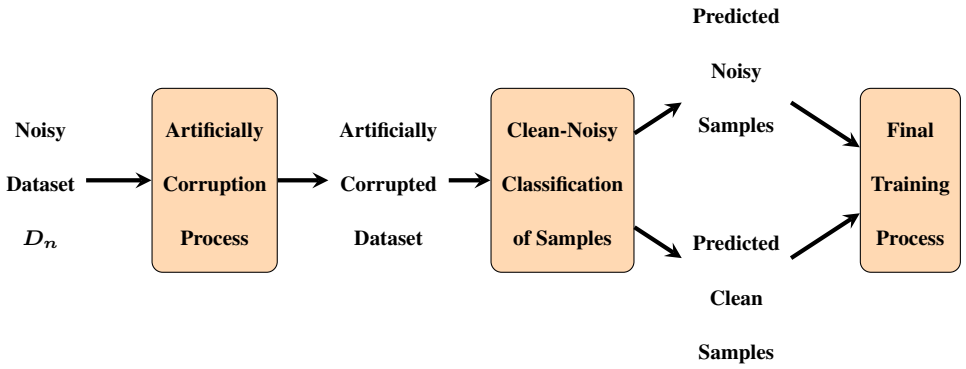

Figure 3.1: Overall Pipeline

3.1.2 Details of the Proposed Method

3.1.2.1 Artificial Corruption Process of Noisy Dataset

The first step of our approach is artificially corrupting noisy dataset D_n to obtain a controlled dataset $\tilde{D}_n$ and it is depicted in Figure 3.2. Here, the main purpose is to obtain surely noisy samples and observe their train losses to classify the remaining samples as clean or noisy.

The input of the corruption pipeline is noisy dataset D_n with K samples, and each sample is from one of the C possible classes $\{0, 1, ..., C-1\}$. $\%s$ of K samples are chosen for the corruption process and form a new set D_{n1} while the remaining ones form another set D_{n2}. Total class number is increased from C to $C+1$ for both subsets D_{n1} and D_{n2} by modifying their one-hot ground truth label vectors and new possible classes are $\{0, 1, ..., C\}$. After increasing the number of classes, the original ground truth class of samples of the subset D_{n2} is preserved, and D_{n2} is converted into a new set D_{n4}. On the other hand, all samples of D_{n1} are assigned to artificially created C^{th} class, and in this way, all samples of set D_{n1} are guaranteed to be noisy. After increasing the number of classes and assigning samples to the new class, subset

D_{n1} is converted into a new set D_{n3}. As a final step of the corruption process, subsets D_{n3} and D_{n4} are combined by preserving their indices to obtain a final dataset $\tilde{D}_n$.

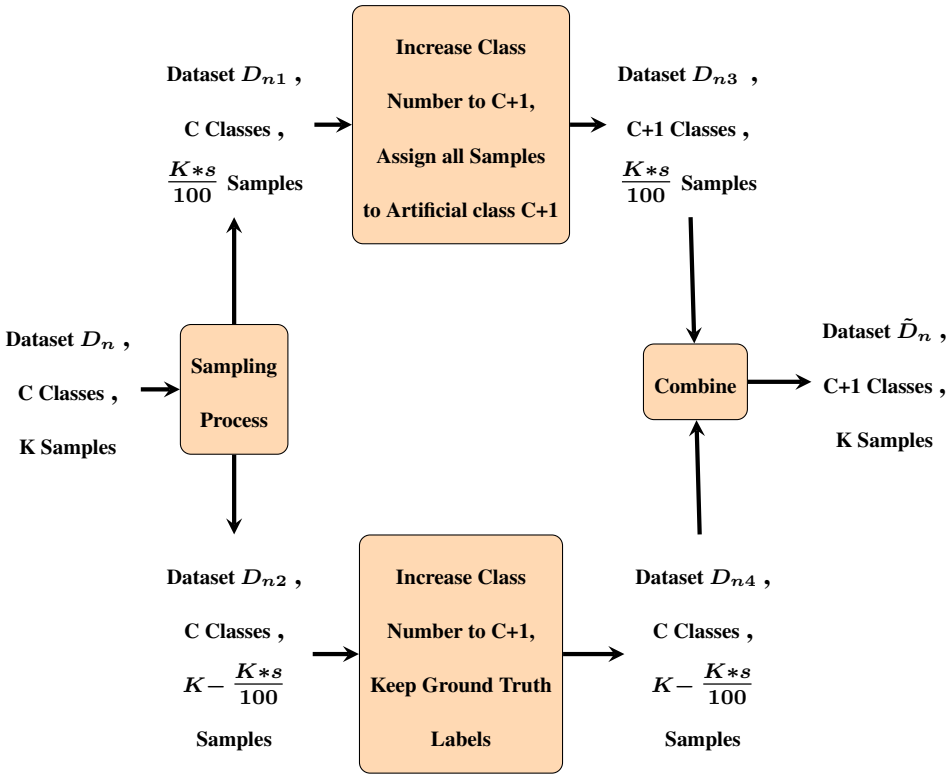

Figure 3.2: Artificially Corruption Process of Noisy Dataset

3.1.2.2 Classification of Samples as Clean-Noisy

Once the artificially corrupted dataset $\tilde{D}_n$ is obtained, a neural network-based classifier is trained over the new dataset. During this training, the loss value of each sample at each iteration is stored for the next steps. Here, the aim is to find some clues about noisy samples based on losses of artificially corrupted samples.

Following the first training phase, our approach continues with a voting process. An overall diagram of the voting pipeline is given in Figure 3.3. Losses obtained after first training phase are divided into two loss sets L_{ai} and L_{ri} where L_{ai} and L_{ri} are losses of samples corresponded to D_{n3} and D_{n4} at i^{th} epoch respectively. Here, L_{ai} is losses of surely noisy samples and is used as an indicator of label noise. For each epoch i from 0 to E_l, mean value of the L_{ai} is obtained and represented as u_{ai}. Then,

each loss value L_{rij}, which is loss value of j^{th} sample at i^{th} epoch, is compared with u_{ai}. If L_{rij} is larger than u_{ai}, NV_{rj} is increased by one, otherwise CV_{rj} is increased. Here, NV_{rj} and CV_{rj} are noisy and clean votes of j^{th} sample respectively. After voting each sample for all first E_l epoch, the voting phase is ended. At the end, possible minimum and maximum values of NV_{rj} and CV_{rj} are 0 and E_l respectively.

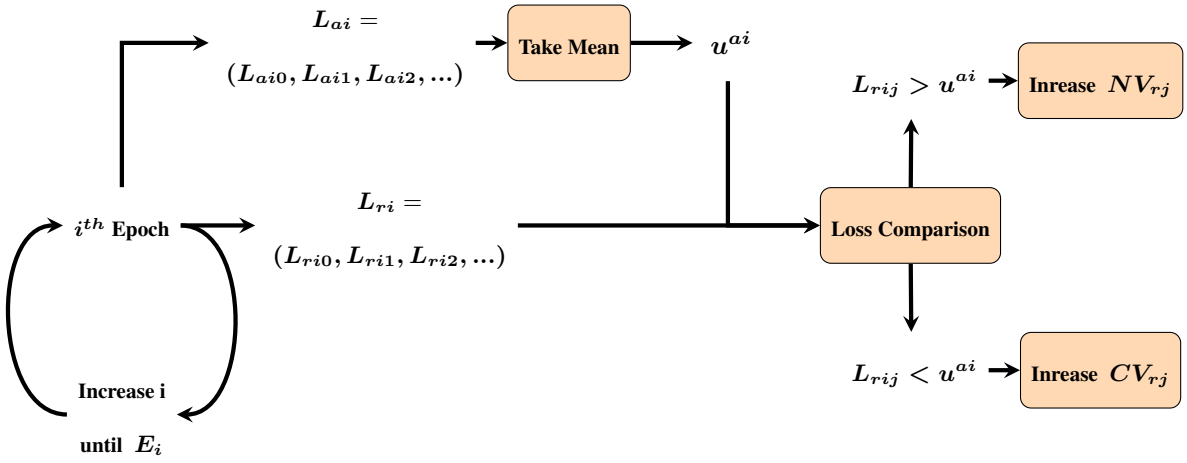

Figure 3.3: Clean-Noisy Voting Strategy

A thresholding-based noisy-clean classification is followed as the next step of the voting process. Each CV_{rj} value is compared with V_t, which is the voting threshold. If CV_{rj} is larger than V_t, j^{th} sample is classified as clean, otherwise it is classified as noisy.

As an output of classification process, we obtain a clean and a noisy set which are represented as $X = \{(x_0, y_0), ..., (x_n, y_n)\}$ and $U = \{u_0, ..., u_m\}$ respectively. Here, x_i, y_i are i^{th} image of clean set and its corresponding label respectively. u_i is the i^{th} image of the noisy set, and its label is ignored since it is classified as noisy. These two sets are provided to Semi-Supervised Learning step to train our final classifier.

3.1.2.3 Semi-Supervised Learning Based Final Training

The final step of our approach is training a neural network-based classifier over the extracted clean(X) and noisy(U) subsets. To be able to utilize noisy subset without

degrading network performance, a semi-supervised learning-based approach is followed similar to recent works [54], [58]. Additionally, two classifier networks($g_1(.)$, $g_2(.)$) are trained together to increase the robustness of the approach against possible samples misclassified as clean.

The input of the training process is noisy and clean subsets, where the clean subset is used as a labeled set while the noisy one is used as an unlabeled set. During each iteration, one of the networks is frozen, and just the other free one is trained for the current iteration, and two networks are alternately trained in this way.

To train the free network, labeled set X is passed through a strong augmentation process twice, which results with X_{s1} and X_{s2}. On the other hand, unlabelled set U is passed through a strong augmentation twice to obtain U_{s1}, U_{s2} and passed through a weak augmentation twice to obtain U_{w1} and U_{w2}.

For utilizing unlabelled set in semi-supervised learning, its weak augmented versions U_{w1} and U_{w2} are passed through both networks($g_1(.)$, $g_2(.)$) and predictions of the both networks are averaged to create label predictions for unlabelled set as given in Equation 3.1.

$$\begin{aligned} W_1 &= g_1(U_{w1}) \\ W_2 &= g_1(U_{w2}) \\ W_3 &= g_2(U_{w1}) \\ W_4 &= g_2(U_{w2}) \\ W &= \tfrac{W_1+W_2+W_3+W_4}{4} \end{aligned} \tag{3.1}$$

Then, strongly augmented versions of both labelled(X_{s1}, X_{s2}) and unlabelled(U_{s1}, U_{s2}) sets are given to MixUp[57] augmentation and during this augmentation averaged network predictions w are used as ground truth of U_{s1}, U_{s2}. Details related to the MixUp augmentation algorithm are given below.

MixUp Augmentation: MixUp[57] is a simple but powerful augmentation technique that can be done easily just by mixing both input data features and their labels. A basic formulation of the MixUp augmentation is given in Equation 3.2. x_i, u_i are input data features of two samples from (X_{s1}, X_{s2}) and (U_{s1}, U_{s2}) respectively. y_i, w_i are labels

of x_i, u_i respectively where $y_i \in Y$ and $w_i \in W$.
α value is sampled from beta distribution. Both feature and label vectors are mixed using sampled α value.

$$
\begin{aligned}
&\hat{x}_i = \alpha x_i + (1-\alpha)u_i \\
&\hat{y}_i = \alpha y_i + (1-\alpha)w_i \\
&x_i \in \{X_{s1} + X_{s2}\}, \quad u_i \in \{U_{s1} + U_{s2}\} \\
&y_i \in Y, \quad q_i \in W \\
&D_{mixup} = \{(\hat{x}_0, \hat{y}_0), (\hat{x}_1, \hat{y}_1), ...\}
\end{aligned}
\tag{3.2}
$$

A visualization of the mixing features of two samples can be observed in Figure 3.4.

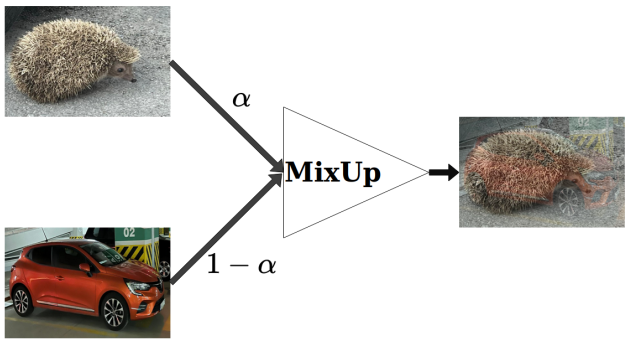

Figure 3.4: An example of MixUp Augmentation

Mixup augmentation increases samples number synthetically in a fast way. Also, Mixup provides a way of label smoothing since the final label is the weighted sum of the input labels. In this way, the individual effect of each label decreases during training, so if there is any mislabelled ground truth, its effect will decrease during training. For this reason, Mixup augmentation is a powerful and robust tool against noisy labeled samples.

After applying MixUp augmentation to strongly augmented versions of labelled and unlabelled sets, we obtain a new set $D_{mixup} = \{(\hat{x}_0, \hat{y}_0), (\hat{x}_1, \hat{y}_1), ...\}$ with soft labels as output of MixUp augmentation. Then, D_{mixup} is given to the free network to calculate cross-entropy loss which is given in Eq. 3.3.

$$L_X = \frac{1}{|D_{mixup}|} \sum_{i=0}^{|D_{mixup}|} \frac{1}{C} \sum_{c=0}^{C} p(\hat{y}_i = c|\hat{x}_i) \log(\hat{p}(\hat{y}_i = c|\hat{x}_i)) \tag{3.3}$$

Here, $\hat{p}$ and p correspond to predicted and true class distributions, respectively. i is index of the sample and c corresponds to c^{th} class.

On the other hand, we utilize strongly augmented versions of the unlabeled set(U_{s1}, U_{s2}) further via contrastive learning. Contrastive learning makes it possible to use samples without ground truth labels, and it can be viewed as a tool to avoid memorization of noisy samples in our approach. Details related to contrastive learning are given below.

Contrastive Learning: Contrastive learning is a machine learning algorithm utilized during learning a mapping function from the input images to a latent space where visually similar inputs are located close to each other while visually dissimilar ones are distant from each other. It is a powerful tool in self-supervised learning since the optimization of the contrastive loss function can be done without using any ground truth label.

FIGURE 3.5 presents a simple pipeline for contrastive learning. For a given unlabelled train set $U = \{u_i\}_{i=1}^{N}$, each image is passed through an augmentation algorithm twice to generate two different images with the same class label. In our case, these augmented sets are $U_{s1} = \{u_{i,s1}\}_{i=1}^{N}$ and $U_{s2} = \{u_{i,s2}\}_{i=1}^{N}$ which are already generated at the beginning of the SSL training pipeline.

Augmented images $u_{i,s1}$ and $u_{i,s2}$ are passed through the feature extractor part of our free network, which is a fully convolutional network. Feature extractor provides middle representations h_{i1} and h_{i2} for augmented inputs. Then, middle representations are further processed via a dense layer(MLP) known as the projection head. Latent space representations z_{i1} and z_{i2} are obtained at the output of the projection head(dense layer). z_{i1} and z_{i2} are the final(latent space) representations of the inputs $u_{i,s1}$, $u_{i,s2}$ and these representations are directly used in contrastive loss function which is given in Equation 3.4.

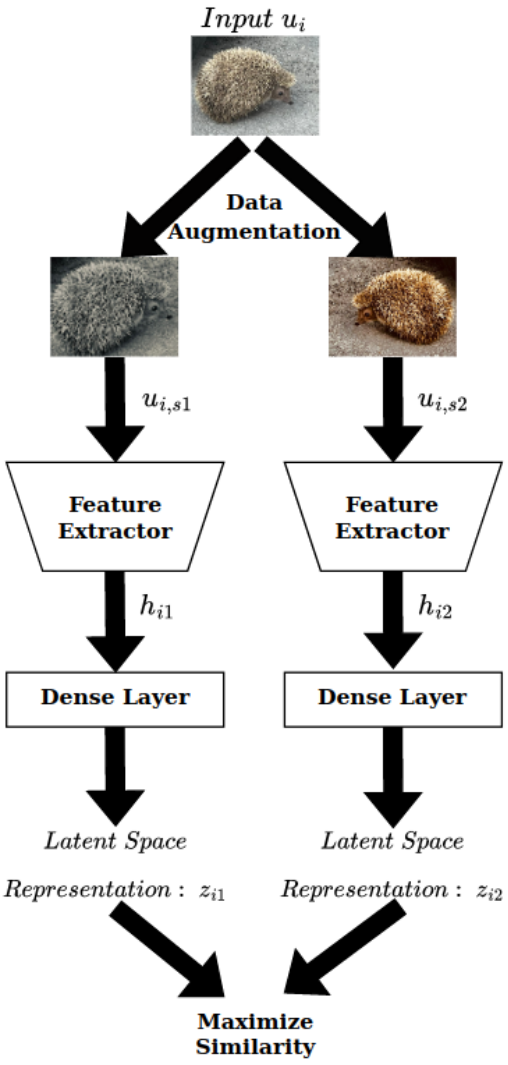

Figure 3.5: Contrastive Learning Pipeline

$$l_{i,1} = -\log \frac{\frac{exp(sim(z_{i1},z_{i2}))}{\tau}}{\sum_{k=1}^{k=N}\sum_{m=1}^{2} I_{km\neq i1}\frac{exp(sim(z_{i1},z_{km}))}{\tau}} \tag{3.4}$$

N is the number of images used in contrastive loss calculation, τ is temperature sharpening constant, $I_{km\neq i1}$ is an indicator function that gives 0 if $km = i1$ and 1 in all other cases. $sim(.)$ is a similarity function that can be defined as cosine similarity between inputs. When contrastive loss is calculated for all augmented images according to Eq. 3.4, overall contrastive loss can be calculated via Equation 3.5.

$$L_C = \frac{1}{2N} \sum_{i=1}^{N} (l_{i,1} + l_{i,2}) \tag{3.5}$$

By minimizing overall contrastive loss, the network learns to map two augmented versions of a single image to close region in latent space, and so, the network learns underlying visual similarities in the augmented images.

By utilizing explained contrastive learning algorithm, a contrastive loss L_C is calculated for U_{s1}, U_{s2} as given in Equation 3.5. Then, final total loss is calculated as given in Equation 3.6.

$$L_T = L_X + \lambda_C * L_C \tag{3.6}$$

Here, λ_C is coefficient of the contrastive loss which controls effect of the contrastive learning in overall train.

CHAPTER 4

EXPERIMENTAL EVALUATION OF PROPOSED METHOD AND LITERATURE

This chapter aims to analyze strong and negative sides of the sample selection based label noise robust algorithms proposed in the literature and proposed method of this thesis(see Chapter 3) through extensive experimentation. In the beginning of the chapter, widely used datasets in label noise robust training literature are presented. Then, experimental setup is introduced by giving details related to dataset preparation, tested algorithms, their network structures and training strategies. As a next step, detailed results of each algorithm over prepared datasets are presented and findings obtained from test results are given. Thereafter, an analysis of sample losses is provided to see differences of clean samples, noisy samples and artificially corrupted samples by the proposed method of this thesis. At final, clean sample extraction performance of the proposed method of this thesis is presented.

4.1 Datasets Used in Experiments

In label noise robust training literature, widely used datasets are MNIST[59], MNIST-Fashion[60], CIFAR10[27], CIFAR100[27], Clothing1M[2] and Animal10N[61]. In fact, MNIST, MNIST-Fashion, CIFAR10 and CIFAR100 are clean datasets but they are frequently used in label noise robust training literature after they are corrupted by utilizing synthetic noise generation methods(see Section 2.2). On the other hand, Animal10N and Clothing1M are real world noisy datasets.

4.1.1 MNIST

MNIST[59] dataset is a handwritten digit dataset which contains 60000 train and 10000 test images. Each image in MNIST is a gray-level image with size of 28x28. There are 10 different digit classes in MNIST dataset.

4.1.2 MNIST-Fashion

MNIST-Fashion[60] is article image dataset which contains 60000 train and 10000 test images similar to MNIST[59]. Each image is a gray-level image with size of 28x28 and dataset contains 10 different classes.

4.1.3 CIFAR10

Cifar10[27] is a toy dataset which contains 10 different object classes inside. There are 50000 train and 10000 test images in dataset. Each image is RGB image with size of 32x32.

4.1.4 CIFAR100

Cifar100[27] contains 100 different object classes inside. It is a more complicated dataset compared to Cifar10[27] because of its large number of classes. There are 50000 train and 10000 test images and each image is RGB image with size of 32x32 similar to Cifar10 dataset.

4.1.5 Animal10N

Animal10N[61] dataset is a real world noisy dataset which is constructed by collecting animal images using search engines and then, annotating these images by 15 annotators. The noise rate of Animal10N is estimated to be $\%8$ and the reason of noisy samples is confusing animal classes. Animal10N contains 10 different animal

classes inside. There are 50000 train and 5000 test images in dataset and each image is RGB image with size of 64x64.

4.1.6 Clothing1M

Clothing1M[2] dataset is a real world noisy dataset which is collected from online shopping websites. There are 14 different clothing classes in dataset. Clothing1M contains two sets inside which are clean set and noisy set. Clean set contains 74000 images which are manually labelled by annotators. Clean set is divided into train data, validation data and testing data with 50000, 14000 and 10000 images respectively. These clean train, validation and test sets are shown in orange circle, green circle and red circle in Figure 4.1 respectively. Also, Clothing1M provides original initial labels of some images of the clean set and these labels belong to before of the annotation process and so, these images also can be used as noisy labelled samples using their noisy labels. These images are shown as overlap of orange and blue circles, green and blue circles, red and blue circles in Figure 4.1. On the other hand, noisy part of Clothing1M contains 1 million images which are not annotated manually and so, these 1 million images are noisy labelled directly. These images are shown in the non-overlapping part of blue circle in Figure 4.1.

To summarize Clothing1M distribution with Figure 4.1, non-overlapping part of orange circle contains 22933 images with only clean labels, overlapping part contains 24637 images with both initial noisy labels and clean labels. Non-overlapping part of green circle contains 6848 images with only clean labels, overlapping part contains 7465 images with both initial noisy labels and clean labels. Non-overlapping part of red circle contains 5131 images with only clean labels, overlapping part contains 5395 images with both initial noisy labels and clean labels. Non-overlapping part of blue circle contains 1 million images with only noisy labels.

Each image of Clothing1M is RGB image with varying size. There is not a constant size for all images.

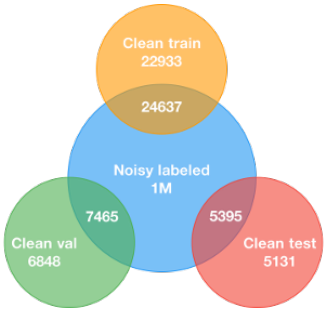

Figure 4.1: Clothinh1M dataset image distribution. Image is taken from [2].

4.2 Experimental Setup

For analysing literature, some popular sample selection based algorithms are implemented and tested over different datasets under varying noise levels. For a fair comparison between algorithms, similar networks and training strategies are applied in these implementations. Now, detailed information related to preparation of datasets, algorithms, their network structures and training strategies will be provided in the following subsections.

4.2.1 Preparation of Datasets

As mentioned in Section 4.1, MNIST, MNIST-Fashion, CIFAR10 and CIFAR100 are originally clean datasets. During experiments, these clean datasets are corrupted by uniform synthetic noise(see Section 2.2.1) under varying noise levels such as $\%20$, $\%30$, $\%40$ and $\%50$.

As a real world noisy dataset, we use Animal10N. We do not apply additional processing for Animal10N and it is used directly.

Clothing1M is also a real world noisy dataset and it is utilized in three different ways during experiments for a detailed analysis of the sample selection methods.

The first version of Clothing1M is constructed with clean part of Clothing1M. Noise rate of the noisy part of Clothing1M is constant and it is not exactly known. Therefore,

controlled experiments with noisy part of Clothing1M are not possible. To achieve a controlled environment with a realistic and complex dataset, clean part of Clothing1M is corrupted with uniform synthetic noise under varying noise rates similar to corruption of our toy datasets and this version of Clothing1M is called as Clothing1M V1. Available noise rates in Clothing1M V1 are $\%10$, $\%20$, $\%30$, $\%40$ and $\%50$.

The second version of Clothing1M is constructed with clean and clean-noisy overlapping parts of Clothing1M. To achieve a controlled environment with real world noise instead of uniform synthetic noise, surely noisy samples in the overlapping region are randomly sampled to create 4 different sets with $\%5$, $\%10$, $\%15$ and $\%20$ noise rates and these versions are called as Clothing1M V2 in our thesis. We only can achieve $\%20$ maximum noise rate in Clothing1M V2 since number of surely noise samples is limited.

The third version of Clothing1M is directly constructed with noisy part of Clothing1M. Clothing1M suffers from class imbalance problem since number of samples exist in classes varies with a large amount. One followed approach in the literature is finding class with minimum number of samples and randomly sampling from other classes with an amount of samples exist in the found class with minimum number of samples. We also follow same approach and a class balanced noisy Clothing1M subset with approximately 270000 images is obtained. This final version is called as Clothing1M V3.

4.2.2 Implemented Algorithms

For evaluation of label noise robust training literature, a base model, Self-Paced MentorNet, Co-teaching, Co-teaching+, Nested Co-teaching, DIVIDEMIX and UNICON algorithms are implemented and compared with each other.

Base model is a simple classifier model. It does not include any specific precaution to handle noisy label problem. The main purpose of training a base model is observing improvements gained by all other algorithms proposed to handle label noise problem and analyzing effect of noisy samples over train-test losses and performance.

On the other hand, Self-Paced MentorNet, Co-teaching, Co-teaching+ and Nested Co-teaching are four popular sample selection based robust methods which utilize small loss selection approach. These algorithms apply small loss selection trick during training to eliminate possible noisy samples.

Self-paced MentorNet is a special case of the MentorNet where a predefined curriculum is used during training. Predefined curriculum is choosing only samples with small loss for backpropagation calculation. Algorithm utilizes a single network and network uses small loss samples chosen by itself in its optimization process at each iteration.

Co-teaching can be seen as an updated version of the Self-Paced MentorNet. It utilizes two networks during training which are randomly initialized with different weights. Each network chooses its small loss instances and updates its parameter using its peer network small loss instances. Difference between Co-teaching and Self-Paced MentorNet can be visualized in FIGURE 4.2.

Co-teaching+ claims that networks used in Co-teaching reach a consensus as training progress and Co-teaching algorithm is simply converted into self-paced MentorNet. For this reason, Co-teaching+ tries to keep both networks divergent from each other and this is done by applying small loss selection operation on the samples in disagreement region of networks. Comparison of the Co-teaching and Co-teaching+ algorithms can also be visualized in FIGURE 4.2.

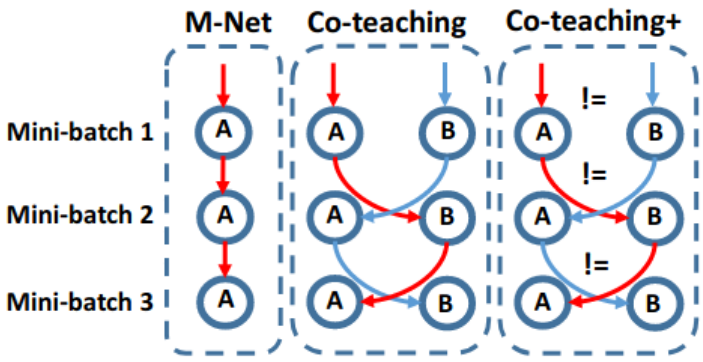

Figure 4.2: Loss Flow through Self-paced MentorNet, Co-teaching and Co-teaching+. Image is taken from [3]

Nested Co-teaching claims that networks used in both Co-teaching and Co-teaching+ algorithms should be trustable enough for selection of small loss instances at the beginning of algorithms. Therefore, it proposes a two stage based method for noise robust training where the first stage aims to provide two trustable networks for small loss instance selection and the second stage aims to fine tune trustable networks of the first stage using Co-teaching algorithm. The first stage of algorithm utilizes Nested Dropout[53] idea for training of trustable base networks. Nested Dropout is a way of importance ordered representation learning which makes possible to learn underlying structure of clean samples in the important part of output feature vector.

The other two implemented algorithms, DIVIDEMIX and UNICON are sample selection based robust algorithms which exploit semi-supervised learning. DIVIDEMIX utilizes two networks during training. During each iteration, it fits GMM to sample loss values for detection of clean samples in an unsupervised manner. Then, each network gives its clean and noisy predictions to its peer network as training data. Each network uses clean and noisy predictions of its peer network as labelled and unlabelled sets respectively. Then, labels of labelled set are refined by utilizing both networks in co-refinement. Also, labels of unlabelled set are predicted by utilizing both networks in co-guessing. Thereafter, samples of both labelled and unlabelled sets are mixed via MixUp augmentation and these are used in semi-supervised learning loss calculations.

On the other hand, UNICON also follows a very similar approach with DIVIDEMIX. UNICON also utilizes two networks during training. During each iteration, network predictions are averaged to obtain common predictions for samples and then, Jensen-Shannon Divergence(JSD) is calculated between this averaged predictions and ground truth labels. Thereafter, samples are classified as clean or noisy according to their JSD values. Possible clean samples are utilized as a labelled set while possible noisy samples are utilized as an unlabelled set. A constrastive learning loss is calculated from unlabelled set different from DIVIDEMIX. Also, similar to DIVIDEMIX, both networks are utilized to refine labels of labelled set via co-refinement and guess labels of unlabelled set via co-guessing. Then, labelled and unlabelled sets are mixed using MixUp augmentation. Additional to contrastive loss, semi-supervised learning loss is calculated using output of MixUp augmentation. At final, networks are updated using

weighted sum of contrastive and semi-supervised learning losses.

The final implemented algorithm is the method proposed of this thesis and it is described in detail in Chapter 3. Also, more detail about implemented methods of the literature can be seen in Section 2.3.5 and related references.

4.2.3 Network Structures

Mainly implementations of two networks are used during experiments according to dataset type and independent of the algorithms. MNIST, MNIST-Fashion, CIFAR10, CIFAR100 and ANIMAL10N datasets are simpler datasets compared to Clothing1M in terms of both image size and number of images. For this reason a simpler custom network used for these datasets while a more complicated network is utilized for Clothing1M experiments.

This custom CNN used for simple datasets is taken from original Co-teaching algorithm implementation[20] and will be named as CoTeachingCNN throughout our thesis. CoTeachingCNN is constructed with 9 convolutional layers which are followed by a batch normalization and leaky ReLU activation function. Negative slope of the each leaky ReLU is set to 0.01. A 2D max pooling operation is applied to donwsample feature maps after leaky ReLUs coming after 3^{rd} and 6^{th} convolutional layers. A 2D average pooling layer is used after leaky ReLU coming after 9^{th} convolutional layer. Dropout is applied after 2D max pooling layers. Both dropout rate is set to 0.25. At the end, a single fully connected layer is applied after 2D average pooling layer to obtain class scores.

The complicated network used during experiments over Clothing1M is ResNet50[62]. Network structure of ResNet50 besides of last fully connected layer is preserved during experiments. The last layer is modified according the number of classes exist in Clothing1M.

For the methods utilizing contrastive learning(UNICON and the proposed method of this thesis), both network types(CoTeachingCNN, ResNet50) are further modified by adding a projection head in parallel to fully connected layer at the end of network. This projection head is also a fully connected layer followed by 1D batch normal-

ization layer. Output of the projection head is a vector with 128 entries. Projection head is only used during training for loss calculations and it can be removed during inference time. Further detail about projection head is given in Section 3.1.2.3.

4.2.4 Training Strategies

Base model, SP-MentorNet, Co-teaching and Co-teaching+ are trained for 200 epochs over MNIST, MNIST-Fashion, CIFAR10, CIFAR100 and ANIMAL10N with ADAM optimizer. Initial learning rate of ADAM optimizer is set to 0.001 and then, it is linearly decreased to 0 after 80^{th} epoch for MNIST, MNIST-Fashion, CIFAR10, ANIMAL10N and after 100^{th} epoch for CIFAR100. β_1 value of optimizer is set to 0.9 until 80^{th} epoch for MNIST, MNIST-Fashion, CIFAR10, ANIMAL10N and until 100^{th} epoch for CIFAR100 and then, it is set to 0.1. β_2 value is set to 0.999 throughout training. As a loss function, cross entropy loss is used for all methods and datasets.

On the other hand, base model, SP-MentorNet, Co-teaching and Co-teaching+ are trained for 30 epochs with SGD optimizer over Clothing1M V1 and Clothing1M V3. Initial learning rate of SGD optimizer is set to 0.01 for Clothing1M V1 experiments while it is set to 0.02 for experiments of Clothing1M V3. Learning rate is decayed by 0.1 after 5^{th} epoch for both Clothing1M V1 and Clothing1M V3 experiments. β_1 and β_2 values of optimizer are set to 0.9 and 0.999 respectively throughout training for both Clothing1M experiments. Again, cross entropy loss is used over both version of Clothing1M for loss calculation.

Nested Dropout contains two stages and different strategies applied at each stage. In the first stage, networks are trained for 350 epochs with batch size of 320 over MNIST, MNIST-Fashion, CIFAR10, CIFAR100, and trained for 100 epochs with batch size of 128 over ANIMAL10N, and trained 30 epochs with batch size of 448 in Clothing1M V1 and Clothing1M V3 experiments. In all experiments SGD optimizer is utilized. Before starting training for 350 epochs, 100 epochs and 30 epochs, a warm-up training is applied for warming learning rate which continues 6000 iterations. Initial learning rate of SGD optimizer is set to 0.1 and then, decayed by 0.1 after 200^{th} and 300^{th} epochs for MNIST, MNIST-Fashion, CIFAR10 and CIFAR100 experiments. Initial learning rate of SGD optimizer is set to 0.1 and then, decayed by 0.2 after 50^{th} and

75^{th} epochs for ANIMAL10N. For Clothing1M case, initial learning rate is set to 0.02 and it is decayed by 0.1 after 5^{th} epoch. Momentum and weight decay parameters of SGD optimizer are set to 0.9 and $5e-4$ respectively in all experiments. In the second stage, networks are initialized with outputs of the first stage. Both networks are trained for 100 epochs with batch size of 320 over MNIST, MNIST-Fashion, CIFAR10, CIFAR100, and trained for 30 epochs with batch size of 128 over ANIMAL10N, and trained for 30 epochs with batch size of 448 in Clothing1M V1 and Clothing1M V3 experiments. Similar to first stage, SGD optimizer is utilized in all experiments. Initial learning rate of SGD optimizer is set to 0.001 and then, decayed by 0.1 after 50^{th} epoch for MNIST, MNIST-Fashion, CIFAR10, CIFAR100. For ANIMAL10N case, initial learning rate is set to 0.004 and it is decayed by 0.1 after 5^{th} epoch. For Clothing1M V1 and Clothing1M V3 cases, initial learning rate is set to 0.002 and it is decayed by 0.1 after 5^{th} epoch. Warm-up training is not applied in the second stage of the algorithm. Similar to the first stage, momentum and weight decay parameters of SGD optimizer are set to 0.9 and $5e-4$ respectively in all experiments.

Self-paced MentorNet, Co-teaching, Co-teaching+ and Nested Co-teaching algorithms utilize small loss selection trick during training. Small loss selection trick requires sorting samples according to their loss during each iteration and a desired percentage of samples with smallest loss are chosen for backpropagation calculations. Desired percentage is defined with a parameter k_r named as keep rate and it is related to n_r (noise rate). In theory, k_r should be equal to $1-n_r$ since k_r is used to keep clean samples. In practice, it is hard to know n_r and so k_r but Co-teaching and Co-teaching+ studies claim that it can be estimated by validation over a small clean subset. Co-teaching and Co-teaching+ assumed that n_r is known during its experiments. In our works, k_r is set to $1-n_r$ similar to original implementations and it is also set to a constant value 0.7 to see keep rate effect over performance. During trainings, keep rate starts from 1.0 and it is decreased to desired value k_r linearly until a predefined epoch and then, it remains constant at desired rate. This predefined epoch is set to 10 for SP-MentorNet, Co-teaching and Co-teaching+ while it is set to 0 for Nested Co-teaching over all datasets. The main reason of linearly decreasing keep rate from 1.0 to k_r is that networks learn from clean samples at the beginning of training and then,

start to learn from noisy ones as training progress. Also, networks are not trustful at initial epochs and difference between the clean and noisy samples is not observable. In Nested Co-teaching case, networks are trustful since it is already trained in the first stage. Therefore, mentioned predefined epoch is set to 0 in Nested Co-teaching.

DIVIDEMIX is trained for 300 epochs with batch size of 64 over MNIST, MNIST-Fashion, CIFAR10, CIFAR100, ANIMAL10N, and trained for 80 epochs with batch size of 32 over Clothing1M V1 and Clothing1M V3. Each epoch of Clothing1M experiments contains 1000 random batches instead of iterating over whole data. During all experiments, SGD optimizer is used with a momentum of 0.9 and a weight decay of 0.0005. At the beginning of trainings, a warm-up training is applied for 10 epochs over CIFAR10, 30 epochs over MNIST, MNIST-Fashion, CIFAR100, ANIMAL10N, and 1 epoch over Clothing1M V1, Clothing1M V3. During warm-up training, classical supervised learning is applied instead of semi-supervised learning over labelled sets, and after warm-up training, semi-supervised learning starts. Learning rate of DIVIDEMIX is set to 0.02 for MNIST, MNIST-Fashion, CIFAR10, CIFAR100, and 0.002 for ANIMAL10N, Clothing1M V1, Clothing1M V3. Beta distribution parameter of Mix-Up augmentation is set to 4 for MNIST, MNIST-Fashion, CIFAR10, CIFAR100 and 0.5 for ANIMAL10N, Clothing1M V1, Clothing1M V3. Temperature sharpening is applied for all network predictions and temperature T is set to 0.5.

UNICON is trained for 300 epochs with batch size of 64 over MNIST, MNIST-Fashion, CIFAR10, ANIMAL10N, and 350 epochs with batch size of 64 over CIFAR100, and 40 epochs with batch size of 32 over Clothing1M V1, and 200 epochs with batch size of 32 over Clothing1M V3. Each epoch of Clothing1M V3 experiment contains 1000 random batches instead of iterating over whole data. During all experiments, SGD optimizer is used with a momentum of 0.9 and a weight decay of 0.0005. At the beginning of trainings, a warm-up training is applied for 10 epochs over CIFAR10, MNIST, MNIST-Fashion, and 30 epochs over CIFAR100 while warm-up training is not applied in ANIMAL10N and Clothing1M experiments. During warm-up training, classical supervised learning is applied instead of semi-supervised learning over labelled sets, and after warm-up training, semi-supervised learning starts. Learning rate of UNICON is set to 0.02 for MNIST, MNIST-Fashion, CIFAR10, CIFAR100, and 0.002 for ANIMAL10N, Clothing1M V1, Clothing1M V3. Beta dis-

tribution parameter of Mix-Up augmentation is set to 4 for MNIST, MNIST-Fashion, CIFAR10, CIFAR100 and 0.5 for ANIMAL10N, Clothing1M V1, Clothing1M V3. Temperature sharpening is applied for all network predictions and temperature T is set to 0.5. Weight of the constrastive learning loss is set to 0.025 during weighted sum calculation of the total loss for all experiments.

The proposed method of this thesis contains two training stages. At initial training stage, a classical supervised learning is applied for detection of clean-noisy samples. This first training stage is exactly same with base model training and detail of base model training is already given at the beginning of this section. During clean-noisy classification of samples, voting process is done for first E_l epochs. This E_l is set to 10 for MNIST, MNIST-Fashion, CIFAR10, CIFAR100, ANIMAL10N while it is set set to 3 for Clothing1M V1, Clothing1M V2, Clothing1M V3. If a sample has clean vote larger than V_t, it is assigned as clean. This V_t is set to 6 as majority decision for MNIST, MNIST-Fashion, CIFAR10, CIFAR100, ANIMAL10N while it is set to 3 as consensus decision for Clothing1M V1, Clothing1M V2, Clothing1M V3. In the second stage of training, semi-supervised learning with an additional contrastive learning component is applied to detected clean-noisy sets. During second stage, our network is trained for 200 epochs with batch size of 128 over MNIST, MNIST-Fashion, CIFAR10, CIFAR100, ANIMAL10N and, 50 epochs with batch size of 128 over Clothing1M V1, Clothing1M V3. During second stage of all experiments, SGD optimizer is used with a momentum of 0.9 and a weight decay of 0.0005. At the beginning of trainings, a warm-up training is applied for 30 epochs over MNIST, MNIST-Fashion, and 20 epochs over CIFAR10, CIFAR100, ANIMAL10N and 3 epochs over Clothing1M V1, Clothing1M V3. During warm-up training, classical supervised learning is applied over labelled set instead of semi-supervised learning, and after warm-up training, semi-supervised learning and contrastive learning start. Learning rate of our method is set to 0.01 for MNIST, MNIST-Fashion, and 0.02 for CIFAR10, CIFAR100, ANIMAL10N, and 0.002 for Clothing1M V1, Clothing1M V3. Beta distribution parameter of Mix-Up augmentation is set to 4 for MNIST, MNIST-Fashion, CIFAR10, CIFAR100, ANIMAL10N and 0.5 for Clothing1M V1, Clothing1M V3. Weight of the constrastive learning loss is set to 0.025 during weighted sum calculation of the total loss for all experiments.

During base model, SP-MentorNet, Co-teaching, Co-teaching+, Nested Co-teaching, same augmentation strategies are followed. As augmentation stragies for these methods, random horizontal flipping, random vertical flipping, padding with 4 pixels and random cropping is applied for MNIST, MNIST-Fashon, CIFAR10 and CIFAR100. Random cropping is done with size of $28x28$ for MNIST, MNIST-Fashion and $32x32$ for CIFAR10, CIFAR100. For Clothing1M experiments, images are resized to $256x256$ and then, a region with size of $224x224$ is cropped from center. Then, random horizontal flip is applied to cropped images. During DIVIEMIX same augmentation strategies are followed over all datesets except ignoring random vertical flipping in DIVIDEMIX.

On the other hand, SSL part of our proposed method and UNICON apply two augmentation strategies which are named as weak and strong augmentations. Weak augmentation of these two approaches is same as the one used in the other methods while a new strategy is followed for strong augmentation. Strong augmentation only differs with one step from weak augmentation. Strong augmentation adds one additional step after random flipping of weak augmentation such as CIFAR10Policy[63] for MNIST, MNIST-Fashion, CIFAR10, CIFAR100, ANIMAL10N and ImageNetPolicy[63] for Clothing1M V1, Clothing1M V3.

4.3 Test Accuracy Results

In this section, results of the label noise robust algorithms, which are presented in Section 4.2.2, are given in detail. Results over each dataset are given and analysed in independent sections and additionally, algorithms are compared with each other.

4.3.1 Results over MNIST

All implemented algorithms are trained over MNIST dataset which is corrupted with $\%20$, $\%30$, $\%40$, $\%50$ synthetic uniform noise and Table 4.1 presents test accuracies of these trained models. Small loss selection based algorithms are trained twice for two keep rate cases and so, they are included twice with two tags in Table 4.1. Experiments with **KR=(1-NR)** tag utilize noise rate information and set keep rate to ratio of

the clean samples while experiments with **KR=0.7** assume that noise rate is unknown and set keep rate to a predefined constant number which is 0.7.

Table 4.1 shows that even base model results with good test accuracy larger than 90% for all noise levels. The main reason behind success of the base model is that MNIST is actually a simple dataset and base network can easily capture underlying structure of clean samples in simple datasets even under label noise.

On the other hand, small loss selection based methods(SP-MentorNet, Co-teaching, Co-teaching+, Nested Co-teaching) outperform base model under all four noise levels. When SP-MentorNet, Co-teaching and Co-teaching+ algorithms are compared with each other, similar results are observed for three algorithms and all methods seem to be successful. Nested Co-teaching results with nearly $\%99$ test performance under all four noise levels.

Table 4.1: Test Accuracies of Algorithms over MNIST

Noise Rate	**20**	**30**	**40**	**50**
Base	94.90	93.82	92.48	90.66
SP-MentorNet / KR=(1-NR)	97.49	97.05	96.97	96.70
Co-teaching / KR=(1-NR)	97.53	97.21	97.16	96.82
Co-teaching+ / KR=(1-NR)	97.55	97.27	97.12	97.10
Nested Co-teaching / KR=(1-NR)	99.04	99.11	98.91	98.81
SP-MentorNet / KR=0.7	96.92	97.05	95.06	94.46
Co-teaching / KR=0.7	97.07	97.21	95.18	93.49
Co-teaching+ / KR=0.7	97.06	97.27	96.01	95.19
Nested Co-teaching / KR=0.7	99.04	99.11	98.91	98.81
DIVIDEMIX	98.23	99.12	98.82	98.15
UNICON	93.46	98.98	98.97	98.95
Our Approach with SSL	99.18	99.04	99.16	98.96

When keep rate effect is analysed for small loss selection methods, we see that Nested Co-teaching is not affected by knowledge of noise rate and provides successful results

even under constant keep rate(KR=0.7). On the other hand, a performance degradation is observed in results of SP-MentorNet, Co-teaching and Co-teaching+ under $\%40$ and $\%50$ noise rates when constant keep rate(KR=0.7) is utilized instead of the clean sample rate(KR=(1-NR)). Performance degradation increases with increasing noise rate since constant keep rate of 0.7 is equivalent to removing only $\%30$ of samples as noisy in all cases and increasing noise rate to above $\%30$ increases number of noisy samples observed by neural network during training. Also, reason of observing similar results under $\%20$ noise rate in both keep rate cases is removing $\%30$ of samples as noisy in constant keep rate case. Removing $\%30$ of samples eliminates all noisy samples in $\%20$ noise rate case.

Semi-Supervised Learning based approaches(DIVIDEMIX, UNICON and our proposed method) provide consistently good results under all noise levels except UNICON under $\%20$ noise rate. UNICON presents poor performance under low noise rates($\%20$ in our case) and they also shared this observation in the their paper[58]. According to their explanation, an effective contrastive learning requires an unlabelled dataset with enough number of samples. Unlabelled set is constructed with detected noisy samples during training and so, when noise rate is low algorithm performance degrades because of ineffective contrastive learning.

4.3.2 Results over MNIST-Fashion

All implemented methods are trained over synthetically corrupted MNIST-Fashion with uniform noise. Experiments are repeated for $\%20$, $\%30$, $\%40$ and $\%50$ noise rates and experiment results are provided in Table 4.2. Small loss selection based methods are trained twice for two keep rate cases to analyse effect of prior knowledge about noise rate.

Sharp decreases are observed in base model performance as noise rate increases and this indicates that base model is affected much more by adverse effects of label noise in MNIST-Fashion compared MNIST dataset(see Table 4.1). This is a sign of that MNIST-Fashion is a more complicated dataset compared to MNIST and so, network faces difficulties during discovering an underlying structure in MNIST-Fashion com-

pared to MNIST in existence of label noise. Importance of noise robust algorithms is more clear when performance gained by robust methods compared to base model is analysed in Table 4.2. Larger performance increases are observed with these robust algorithms in MNIST-Fashion compared to MNIST dataset.

On the other hand, when small loss selection based methods(SP-MentorNet, Co-teaching, Co-teaching+, Nested Co-teaching) algorithms are compared with each other according to Table 4.2, similar results are observed and none of them can be claimed to better than others.

Again, similar to MNIST results, Nested Co-teaching preserves its performance for all noise levels when noise rate is assumed to be unknown and a constant keep rate(KR=0.7) is used during its Co-teaching stage. On the other hand, SP-MentorNet, Co-teaching and Co-teaching+ algorithms also preserve its performance under $\%20$ noise rate while a huge performance degradation is observed under $\%40$ and $\%50$ noise rates in constant keep rate case(KR=0.7) since constant keep rate of 0.7 causes updates from noisy samples under $\%40$ and $\%50$ noise rates. Importance of prior knowledge of noise rate for SP-MentorNet, Co-teaching and Co-teaching+ can be understood from this observation.

Different from MNIST experiments, Semi-Supervised Learning(SSL) based approaches DIVIDEMIX and UNICON do not provide successful results and outperform Nested Co-teaching under all noise levels except $\%50$ noise rate case. We suspect from tuned parameters by these algorithms since originally provided parameters are not tuned over MNIST-Fashion and we test these algorithms with their given parameters in our common setup.

Similar to MNIST results, UNICON presents poor performance under low noise rate($\%20$ noise rate case) because of the ineffective contrastive learning caused by less number of unlabelled data. DIVIDEMIX also results with a similar pattern in experiments such as network accuracy increases with increasing noise rate but it can not be related to contrastive learning since DIVIDEMIX does not utilize contrastive learning. We can not find exact reason of low performance of the DIVIDEMIX but we

Table 4.2: Test Accuracies for MNIST-Fashion

Noise Rate	**20**	**30**	**40**	**50**
Base	88.83	85.59	80.69	77.99
SP-MentorNet / KR=(1-NR)	93.78	93.29	92.88	91.82
Co-teaching / KR=(1-NR)	93.96	93.46	93.14	91.99
Co-teaching+ / KR=(1-NR)	94.22	93.66	93.21	92.13
Nested Co-teaching / KR=(1-NR)	94.15	93.63	93.02	91.92
SP-MentorNet / KR=0.7	92.86	93.29	88.69	85.09
Co-teaching / KR=0.7	93.66	93.46	89.14	83.74
Co-teaching+ / KR=0.7	93.38	93.66	90.27	86.81
Nested Co-teaching / KR=0.7	94.26	93.63	93.02	91.69
DIVIDEMIX	88.31	89.97	91.70	92.14
UNICON	89.75	92.66	92.99	93.46
Our Approach with SSL	94.58	94.48	93.98	93.20

conclude that DIVIDEMIX works poorly at low noise rates compared to high noise rates according to MNIST-Fashion results.

4.3.3 Results over CIFAR10

All implemented algorithms are trained over synthetically corrupted CIFAR10 with uniform noise under $\%20$, $\%30$, $\%40$, $\%50$ noise rates.

FIGURE 4.3 presents train and test losses of the base model over CIFAR10 under all noise levels. For all noise levels, train loss decreases continuously throughout training as it can be observed in FIGURE 4.3. On the other hand, test loss decreases for a while and then, starts to increase. Train and test sets are not from the same distribution since there are noisy instances in train set while not in test set. Therefore, when network starts to memorize its train set and so noisy samples, a performance degradation and loss increase are expected over test set which coincides with our observation related to train-test losses. This is the main problem of the noisy labelled datasets: performance degrades once network overfit to noisy samples. One another

important point is that test loss decreases for a while at the beginning of train. This means that network actually learns from clean samples first and so, test loss decreases at the beginning. Then, network starts to memorize noisy instances and so, test loss increases.

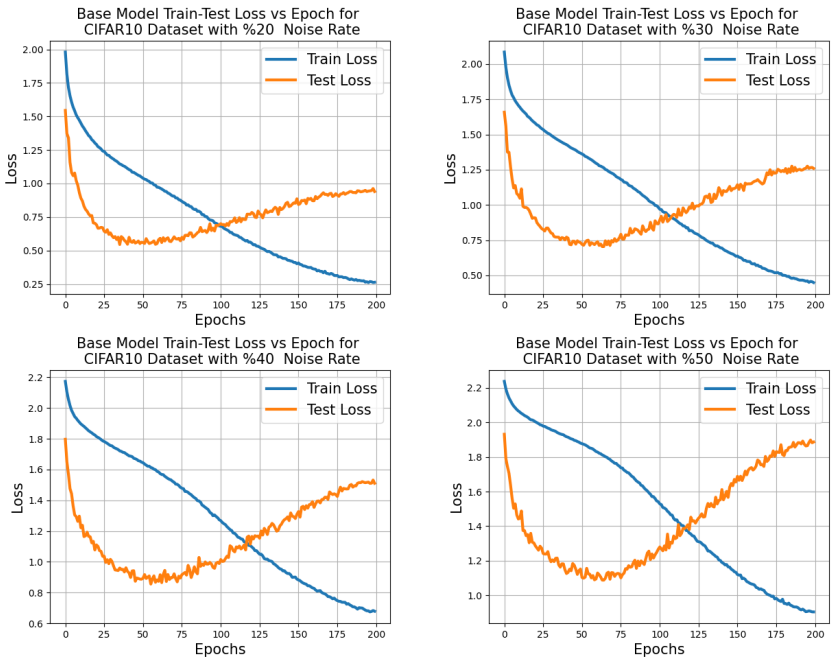

Figure 4.3: Train-Test Losses vs Epochs for CIFAR10 in Base Model

Table 4.3 presents test accuracies of all algorithms over CIFAR10 under all noise levels. Small loss selection based methods are trained twice for two keep rate cases to analyse effect of prior knowledge about noise rate and so, there are two experimental results for these methods in Table 4.3.

When base model results given in Table 4.3 are analysed, more sharp decreases are observed in test accuracies as noise rate increases in CIFAR10 compared to MNIST and MNIST-Fashion results given in Table 4.1 and Table 4.2. These sharper decreases can be related to difficulty of CIFAR10 compared to MNIST and MNIST-Fashion.

All small loss selection based algorithms(SP-MentorNet, Co-teaching, Co-teaching+, Nested Co-teaching) outperform base model with a large margin under all noise levels as it can be seen in Table 4.3. Under $\%50$ noise rate and $KR = (1 - NR)$ case, small

loss selection algorithms provide a performance improvement up to $\%30$ compared to base model which clearly presents effectiveness of small loss selection algorithms and adverse effect of label noise.

When SP-MentorNet, Co-teaching, Co-teaching+ and Nested Co-teaching algorithms are compared with each other for $KR = (1 - NR)$ case according to results of Table 4.3, Nested Co-teaching outperforms the other three ones under all noise levels. Co-teaching+ results are affected more adverse compared to other ones with increasing amount of noise level. The main reason of this observation about Co-teaching+ is related to number of sampled instances after filtering process. Co-teaching+ only keeps small loss instances which fall into disagreement region of two networks and so, as noise rate increases number of filtered possible clean samples decreases sharply. Therefore, Co-teaching+ faces difficulty to find enough samples to update network parameters and results with worse performance compared to other three small loss selection based algorithms at high noise rates. This claim about Co-teaching+ can be proved by comparing SP-MentorNet and Co-teaching+ results for $KR = (1 - NR)$ case. SP-MentorNet is worse than Co-teaching+ at low noise rates such as $\%20$, $\%30$ while it is better than Co-teaching+ at high noise rates such as $\%40$, $\%50$.

Under constant keep rate of 0.7, approximately, $\%1$ performance decrease is observed in Nested Co-teaching algorithm under all noise levels compared to $KR = (1 - NR)$ case besides of $\%30$ noise. Performance does not change under $\%30$ noise rate since 0.7 keep rate corresponds to $\%30$ noise rate. On the other hand, SP-MentorNet, Co-teaching and Co-teaching+ algorithms provide similar performance under $\%20$ noise rate while a huge performance degradation is observed under $\%40$ and $\%50$ noise rates in constant keep rate case($KR = 0.7$) compared to $KR = (1 - NR)$ case because of the reasons explained in Sections 4.3.1 and 4.3.2.

We provide results of our approach with and without Co-training and we observe a very limited performance gain with Co-training under all noise levels. UNICON, DIVIDEMIX and our proposed approach have a clear superiority over all other algorithms. The main reason behind this success is utilizing noisy samples without their labels via Semi-Supervised Learning(SSL) instead of ignoring them like small

Table 4.3: Test Accuracies for CIFAR10

Noise Rate	**20**	**30**	**40**	**50**
Base	79.24	69.18	60.58	50.20
SP-MentorNet / KR=(1-NR)	88.89	87.29	84.77	80.58
SP-MentorNet / KR=0.7	88.69	87.29	79.36	67.66
Co-teaching / KR=(1-NR)	89.52	88.01	85.86	81.32
Co-teaching / KR=0.7	90.32	88.01	79.94	68.12
Co-teaching+ / KR=(1-NR)	89.12	87.04	84.31	78.74
Co-teaching+ / KR=0.7	89.28	87.04	79.88	70.11
Nested Co-teaching / KR=(1-NR)	90.39	88.93	86.91	84.53
Nested Co-teaching / KR=0.7	89.77	88.93	85.91	83.27
DIVIDEMIX	89.71	92.47	93.34	93.50
UNICON	92.11	93.81	94.48	94.53
Our Approach with SSL	93.21	93.07	92.06	90.71
Our Approach with SSL + Co-training	93.34	93.08	92.46	91.17

loss selection methods SP-MentorNet, Co-teaching, Co-teaching+ and Nested Co-teaching. In both UNICON and DIVIDEMIX, a strange result is observed such as performance increases as noise rate increases as in case of MNIST-Fashion experiments. This is probably an outcome of tuning model parameters to high noise level cases in UNICON and DIVIDEMIX. On the other hand, our proposed model results are as expected since our performance decreases with increasing level of noise.

4.3.4 Results over Cifar100

Different from the previous experiments, CIFAR100 experiments are done only for two noise levels since CIFAR100 includes 100 classes and it becomes hard to corrupt dataset at desired noise rate according to exact definition of uniform noise. All implemented algorithms are trained over CIFAR100 under $\%20$, $\%40$ synthetic uniform noise rates.

FIGURE 4.4 presents train and test losses of the base model over CIFAR100 under

two noise levels. The train-test loss trend explained in CIFAR10 experiment is observed in CIFAR100 case too as it can be seen in FIGURE 4.4. Again, train loss decreases throughout training while test loss decreases for a while and then, starts to increase as a result of memorization of noisy samples.

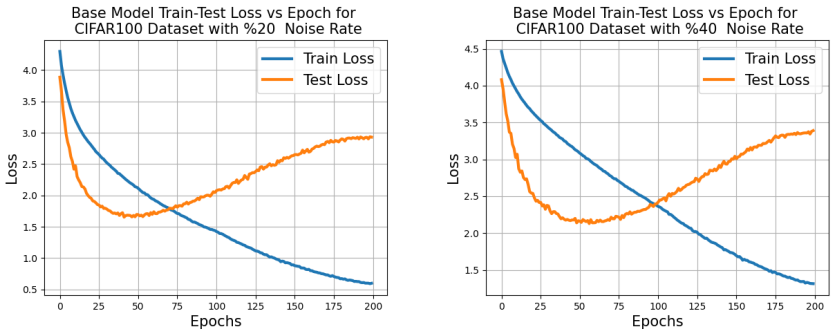

Figure 4.4: Train-Test Losses vs Epochs for Base Model over CIFAR100

Table 4.4 present test accuracy of all algorithms over CIFAR100. According to results given in Table 4.4, small loss selection based algorithms(SP-MentorNet, Co-teaching, Co-teaching+, Nested Co-teaching) outperform base model with a large amount similar to CIFAR10 experiments. On the other hand, when small loss selection based algorithms are compared with each other, Nested Co-teaching seems to be the best algorithm under both noise rate. Additionally, Co-teaching+ algorithm again provides compatible results with SP-MentorNet at low noise rate while SP-MentorNet outperforms Co-teaching+ at high noise rate as a result of filtering approach of Co-teaching+ explained in Section 4.3.3.

Semi-Supervised Learning based approaches DIVIDEMIX and UNICON outperform all other algorithms with a large margin especially under $\%40$ noise rate. Success of these two algorithms is a result of utilization of noisy samples as unlabelled samples during semi-supervised learning. On the other hand, our semi-supervised learning based proposed approach fails under both noise rate. Reason behind failure of our proposed approach is our clean-noisy sample detection process. During noisy sample detection, we utilize artificially corrupted samples to simulate real noisy samples but all these artificially corrupted samples are member of a single class and so, these artificially corrupted samples from a single class face difficulty about simulating real noisy samples of CIFAR100 with large number of classes. Since our artificially cor-

Table 4.4: Test Accuracies for CIFAR100

Noise Rate	**20**	**40**
Base	54.32	41.33
SP-MentorNet / KR=(1-NR)	62.70	57.54
SP-MentorNet / KR=0.7	64.34	53.04
Co-teaching / KR=(1-NR)	63.79	57.55
Co-teaching / KR=0.7	64.92	53.16
Co-teaching+ / KR=(1-NR)	62.17	55.90
Co-teaching+ / KR=0.7	63.37	52.62
Nested Co-teaching / KR=(1-NR)	67.42	61.32
Nested Co-teaching / KR=0.7	68.08	60.34
DIVIDEMIX	72.71	70.28
UNICON	71.53	71.70
Our Approach with SSL	53.71	30.49
Our Approach with SSL and Co-training	54.77	31.36

rupted samples fail to simulate real noisy samples, we have a poor clean-noisy sample detection process and poor detection process results with low test accuracy over CIFAR100.

Despite of failure of our proposed method, we still observe nearly $\%1$ improvement over test accuracy by utilizing Co-training since Co-training combines different learning abilities of two networks during training.

4.3.5 Results over Clothing1M

4.3.5.1 Clothing1M V1

All algorithms are trained over Clothing1M V1 which is synthetically corrupted version of clean part of Clothing1M dataset. Corruption is applied for $\%10$, $\%20$, $\%30$, $\%40$, $\%50$ noise rates with uniform noise.

FIGURE 4.5 presents train and test losses of the base model over Clothing1M V1 under all noise levels. In all cases, test loss decreases for a while and then, a very large increase in test loss is observed while train loss decreases continuously throughout the training. As it is explained in previous experiments, this is an expected result since network actually learns from clean sample first at the beginning of training and it memorizes noise samples as training progresses.

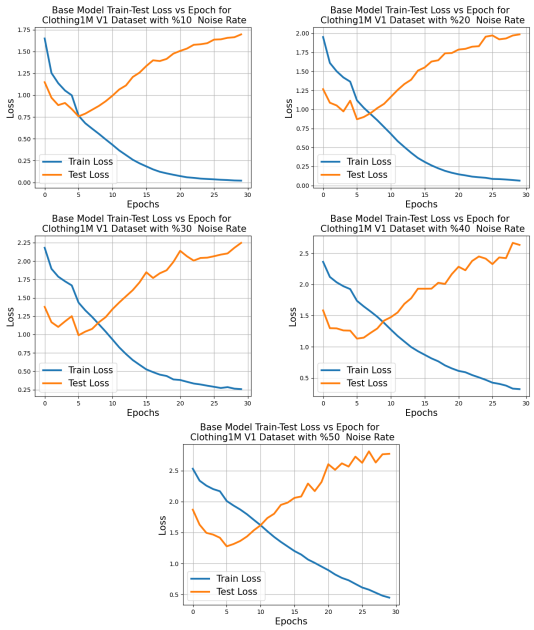

Figure 4.5: Train-Test Losses vs Epochs for Base Model over Clothing1M V1

Table 4.5 presents test accuracy of all algorithms over Clothing1M V1 for all noise levels. Base model provides a baseline to see robustness of all other algorithms. All small loss selection based algorithms(SP-MentorNet, Co-teaching, Co-teaching+, Nested Co-teaching) outperform base model with large margins under all noise levels for both KR=0.7 and KR=(1-NR) cases. Clothing1M V1 results also prove importance of prior knowledge of noise rate for small loss selection algorithms. Similar to results of the toy datasets(CIFAR10, CIFAR100, etc.), Nested Co-teaching preserves its performance in KR=0.7 case compared to KR=(1-NR) case but all other small loss selection algorithms(SP-MentorNet, Co-teaching, Co-teaching+) degrade in KR=0.7 case at high noise rates $\%40$ and $\%50$. On the other hand, under $\%10$ and $\%20$ noise levels, all small loss selection methods provide better results in KR=0.7 case compared to KR=(1-NR) case. This is because of the imperfect filtering performance of

the small loss selection methods. Under $\%10$ and $\%20$ noise levels, theoretically keep rate should be equal to 0.9 and 0.8 but since filtering process of these algorithms is not perfect, keep rate should be less than 0.9, 0.8 to avoid any update from noise samples. KR=0.7 case satisfies this practical requirement and so, algorithms achieve better performance under $\%10$ and $\%20$ noise levels in constant keep rate experiments.

Semi-Supervised Learning(SSL) based approaches(DIVIDEMIX, UNICON and our proposed approach) outperform all other algorithms under all noise levels but power of SSL based approaches is more clear under high noise rates such as $\%40$ and $\%50$ since directly filtering nearly half of the samples is not a good approach to learning in noise label problem.

We provide results of our proposed method over Clothing1M V1 with and without Co-training approach. Power of combining different learning ability of two networks came with Co-training approach is more observable over Clothing1M V1 compared to toy dataset results since it provides an improvement of nearly $\%1$ under all noise levels in Clothing1M V1 dataset.

Table 4.5: Test Accuracies for Clothing1M V1

Noise Rate	**10**	**20**	**30**	**40**	**50**
Base	69.15	63.30	55.33	47.48	39.68
SP-MentorNet / KR=(1-NR)	72.59	71.95	70.59	69.38	66.46
SP-MentorNet / KR=0.7	75.74	75.03	70.59	62.82	52.37
Co-teaching / KR=(1-NR)	73.43	72.60	71.98	71.37	68.64
Co-teaching / KR=0.7	76.82	76.00	71.98	64.29	54.93
Co-teaching+ / KR=(1-NR)	74.97	72.05	65.88	66.77	48.24
Co-teaching+ / KR=0.7	75.34	73.60	65.88	59.65	43.56
Nested Co-teaching / KR=(1-NR)	77.12	75.81	75.04	73.03	67.98
Nested Co-teaching / KR=0.7	78.46	76.89	75.04	72.15	67.90
DIVIDEMIX	79.67	79.35	79.52	78.45	77.40
UNICON	78.12	78.28	78.08	77.79	76.24
Our Approach with SSL	77.45	77.33	75.90	74.60	72.54
Our Approach with SSL-Co-training	78.46	78.20	76.99	75.87	74.23

4.3.5.2 Clothing1M V3

All algorithms are trained over Clothing1M V3 which is class balanced version of noisy part of Clothing1M dataset with unknown noise rate.

FIGURE 4.6 presents train and test losses of the base model over Clothing1M V3. Different from synthetic uniform noise experiments, test loss directly starts to increase after a few epochs. Also, train loss does not decrease directly and oscillates for a while. These both observations can be related to difficulty of the real world noise compared to synthetic uniform noise. Since real world label noise is a hard case compared to synthetic one, learning over train set takes time and loss oscillates. Also, again since real world label noise hard, network faces more difficulties to generalize over a clean test set.

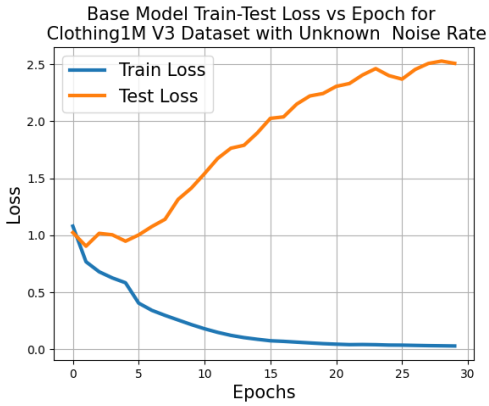

Figure 4.6: Train-Test Losses vs Epochs for Base Model over Clothing1M V3

Table 4.6 presents test accuracies of all algorithms over Clothing1M V3 for all noise levels. Since noise level of the Clothing1M V3 is not known exactly, small loss selection methods are trained only for $KR = 0.7$ case. All loss selection based methods outperform base model and Co-teaching seems to best one among them. Different from synthetic noise experiments, Nested Co-teaching can not outperform other small loss selection methods in real world label noise case. Actually, they report a performance larger than $\%74$ in Clothing1M in [22] but we noticed that they actually utilize clean part of Clothing1M during training. We did not utilize any clean set in our experiments given in this thesis and so, a performance decrease is observed in Nested Co-teaching compared to their reported performance.

Semi-Supervised Learning(SSL) based approaches(DIVIDEMIX, UNICON and our proposed approach) outperform all other methods similar to synthetic uniform noise results with help of utilizing noisy labelled samples instead of removing them.

We provide results of our proposed method over Clothing1M V3 with and without Co-training. We do not observe any improvement with Co-training approach over Clothing1M V3 in our proposed approach.

Table 4.6: Test Accuracies for Clothing1M V3

Noise Rate	**Unknown**
Base	65.59
SP-MentorNet / KR=0.7	71.37
Co-teaching / KR=0.7	72.30
Co-teaching+ / KR=0.7	70.77
Nested Co-teaching / KR=0.7	71.83
DIVIDEMIX	74.50
UNICON	74.49
Our Approach with SSL	73.44
Our Approach with SSL-Cotraining	73.59

We also investigated consistency of the best three algorithms by repeating experiments five different times and report results in Table 4.7. We did not observe large standard deviations in the results since initial newtorks are already pre-trained and the only randomness come from augmentations and sample shuffling.

Table 4.7: Test Accuracies for Clothing1M V3 from 5 Random Runs

	Mean Accuracy	**Standard Deviation**
DIVIDEMIX	74.44	0.244
UNICON	74.30	0.248
Our Approach with SSL-Cotraining	73.62	0.096

4.3.6 Results over Animal10N

All algorithms are trained over Animal10N which is estimated to have $\%8$ noise rate.

FIGURE 4.7 presents train and test losses of the base model over Animal10N. A loss pattern similar to synthetic uniform noise experiments is observed in Animal10N experiment. Network first learns from clean samples and so, test loss decreases at the beginning and then, it starts to increase as network memorizes noisy samples. There is a smooth decrease in Animal10N compared to results of the Clothing1M V3, although Animal10N is a real world noisy labelled dataset. This may be related with estimated low noise rate of Animal10N.

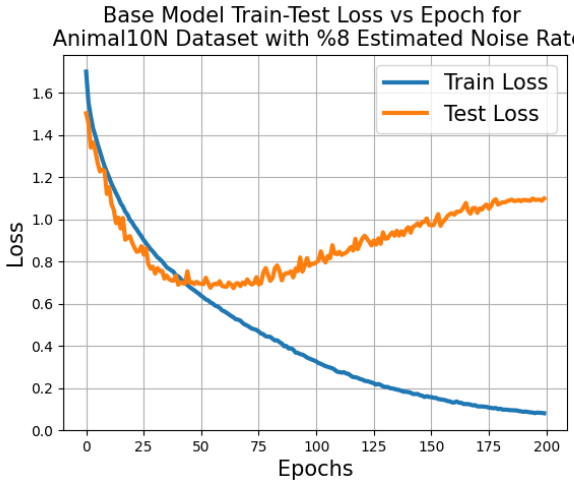

Figure 4.7: Train-Test Losses vs Epochs for Base Model over Animal10N

Table 4.8 presents test accuracy of all algorithms over Animal10N dataset. Since exact noise rate of ANIMAL10N dataset is unknown, small loss selection based algorithms are trained only for a predefined noise rate such as 0.7(KR=0.7) for SP-MentorNet, Co-teaching, Co-teaching+ and 0.8(KR=0.8) for Nested Co-teaching. Nested Co-teaching is trained for predefined keep rate of 0.8 different from the other small loss selection methods since they proposed this keep rate for ANIMAL10N in their work [22]. Results given in the table shows that SP-MentorNet, Co-teaching and Co-teaching+ fail over ANIMAL10N dataset since they are inferior to base model. The main reason behind failure of SP-MentorNet, Co-teaching and Co-teaching+ is their dependency to keep rate and so, a prior knowledge of noise rate. SP-MentorNet,

Co-teaching, Co-teaching+ utilize a keep rate of 0.7 during training which means that they filter $\%30$ of samples as noisy at each iteration but estimated noise rate of ANIMAL10N is nearly $\%8$ and so, these small loss selection algorithms miss some of their clean samples, especially hard ones. Missing these hard clean samples causes their failure compared to base model. On the other hand, Nested Co-teaching is superior to base model since it has a low dependency to keep rate.

Semi-Supervised Learning based approaches(DIVIDEMIX, UNICON and our proposed approach) outperform all other methods over ANIMAL10N. As discussed in previous experiments(CIFAR10, CIFAR100, Clothing1M), superiority of these approaches comes from utilization of noisy samples as unlabelled samples during SSL.

We provide results of our proposed method over ANIMAL10N with and without Co-training. We observed a $\%1$ improvement in test accuracy by utilizing Co-training. Co-training exploits two networks which have different learning abilities and support each other. As a result of utilizing different learning abilities of these networks, we observed an improvement via Co-training. Our proposed approach with SSL and Co-training provides compatible results with state of the art methods DIVIDEMIX and UNICON.

Table 4.8: Test Accuracies for Animal10N

Noise Rate	**Unknown**
Base	79.79
SP-MentorNet / KR=0.7	76.47
Co-teaching / KR=0.7	78.59
Co-teaching+ / KR=0.7	76.84
Nested Co-teaching / KR=0.8	84.08
DIVIDEMIX	86.98
UNICON	86.64
Our Approach with SSL	85.21
Our Approach with SSL and Co-training	86.29

We also investigated consistency of the best three algorithms by repeating experiments five different times and report results in Table 4.9. We did not observe large standard deviations in the results.

Table 4.9: Test Accuracies for Animal10N from 5 Random Runs

	Mean Accuracy	**Standard Deviation**
DIVIDEMIX	87.00	0.103
UNICON	86.61	0.228
Our Approach with SSL-Cotraining	85.91	0.263

4.4 Loss Analysis of Clean-Artificial-Noisy Samples for Proposed Method

In this section, an analysis of train loss dynamics for artificially corrupted datasets is provided. Analysed losses are obtained during clean-noisy classification step of our proposed algorithm(see Section 3.1.2.2). Analysis of the losses is done over Clothing1M V1 and Clothing1M V2 datasets(see Section 4.2.1) since Clothing1M is a more challenging dataset compared to other toy datasets and make possible to compare synthetic and real world noise .

4.4.1 Clothing1M V1

Here, train loss histograms of the clean, noisy and artificial samples of Clothing1M V1 dataset are given for first five epochs. Histograms are provided for two cases such as a low and high noise rate which are $\%20$ and $\%50$ respectively. Noisy samples of this dataset are not real world noisy samples and they are output of uniform noise corruption process.

FIGURE 4.8 and FIGURE 4.9 present histograms for $\%20$ and $\%50$ noise rate cases respectively. Both figures contain 5 columns which correspond to epochs from 0 to 4 in order. Both figures show us neural networks fit to clean samples faster than noisy and artificial samples since peak value in the loss histograms of clean samples moves left(to small values) in a fast manner compared to others. On the other hand, networks

do not fit to noisy and artificial samples easily according to histograms. Additionally, peak values of histograms of artificial samples occurred at a loss value less than noisy ones and higher than clean ones and so, artificial samples can provide a threshold for separation of clean and noisy samples.

High and low noise rate cases have an important difference such as networks fit to clean samples more easily in low noise rate which should coincides with our expectations since learning at high noise rate is hard compared to learning at low noise rate. However, artificial samples still seem to be helpful for separation of clean and noisy samples at high noise rates.

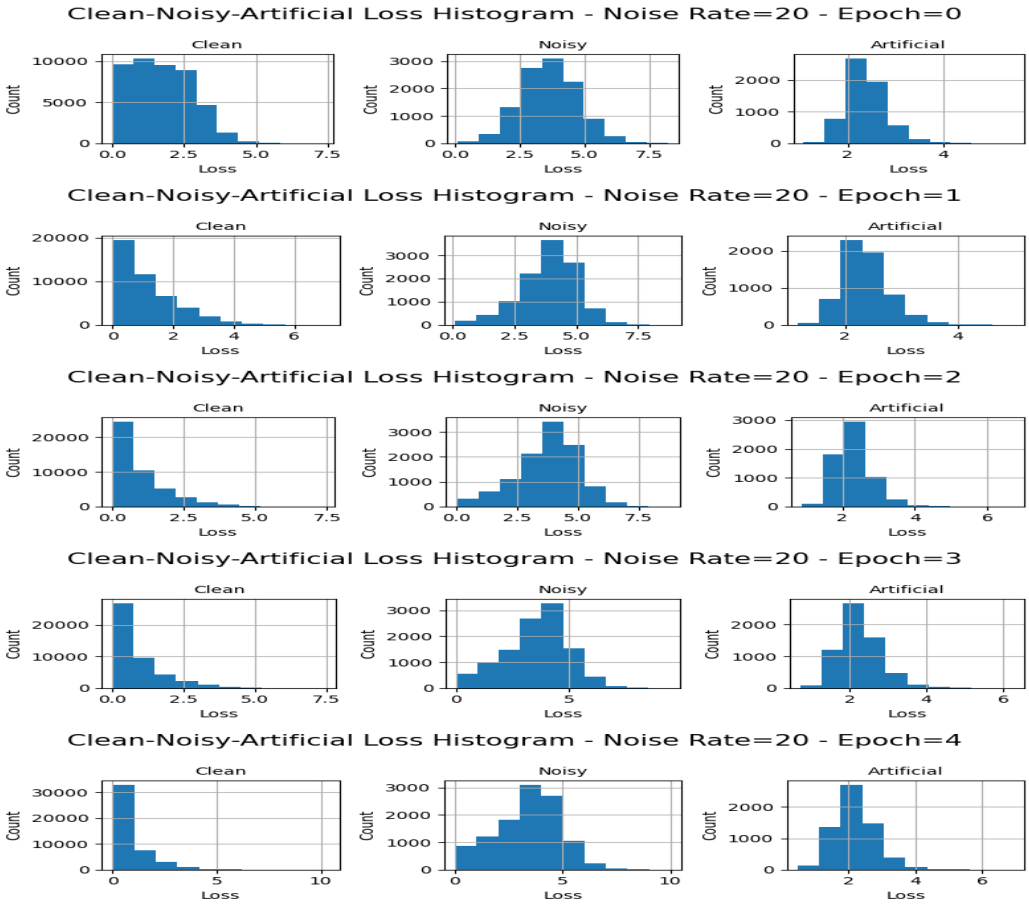

Figure 4.8: Noise Histograms of Clean-Noisy-Artificial Samples of Clothing1M V1 for %20 Noise Rate

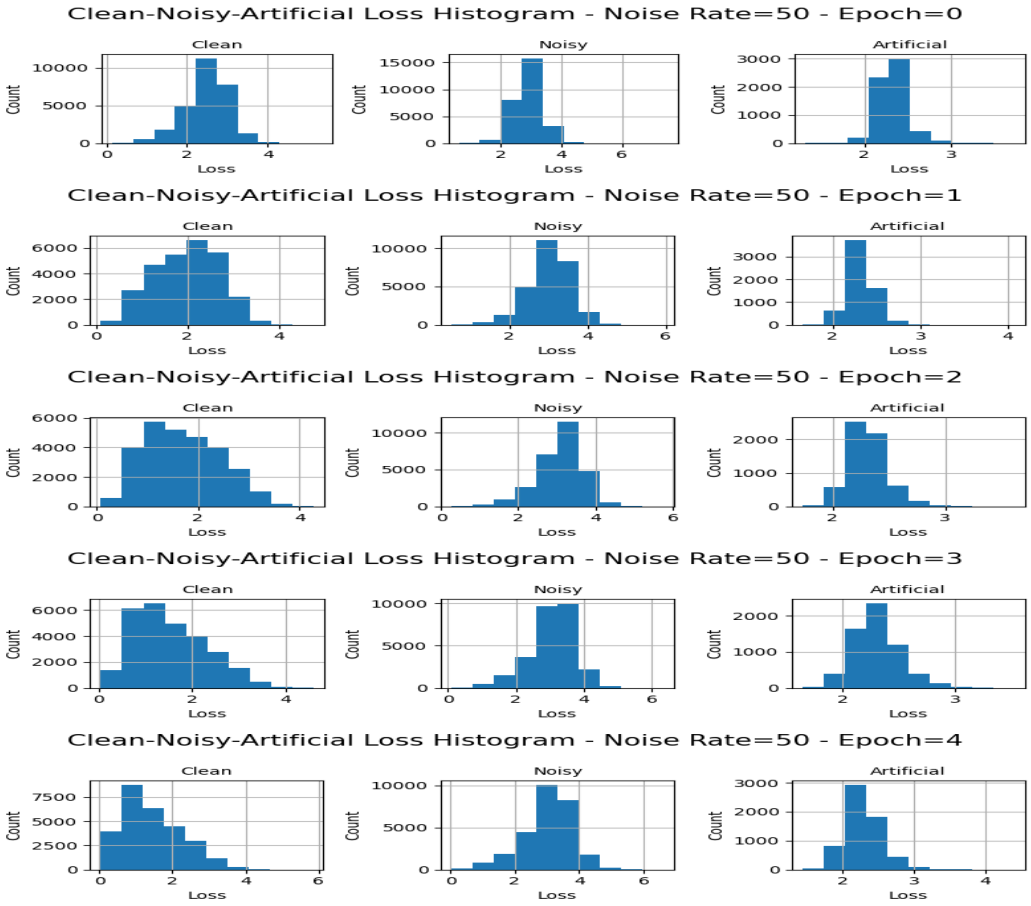

Figure 4.9: Noise Histograms of Clean-Noisy-Artificial Samples of Clothing1M V1 for %50 Noise Rate

All observations related histograms of Clothing1M V1 imply that artificial sample idea should be successful for detection of clean samples under uniform synthetic noise.

4.4.2 Clothing1M V2

Here, train loss histograms of the clean, noisy and artificial samples of Clothing1M V2 dataset are given for first five epochs. Histograms are provided for two cases such as a low and high noise rate which are %5 and %20 respectively. Noisy samples of this dataset are real world noisy samples.

FIGURE 4.10 and FIGURE 4.11 present histograms for %5 and %20 noise rate cases

respectively. Both figures contain 5 columns which correspond to epochs from 0 to 4 in order. Compared synthetic noise loss analysis(see Section 4.4.1), neural networks more easily and fast fit to both clean and noisy samples during training over Clothing1M V2 which contains real world noisy samples. In real world noise case, we observe that peak value in the histogram of artificial samples occurred at a loss value which is higher than both peaks of noisy and clean samples histograms which indicates that networks fit to noisy and clean samples faster than artificial samples. This observation shows us that artificial samples can not mimic noisy samples in real world noise case exactly.

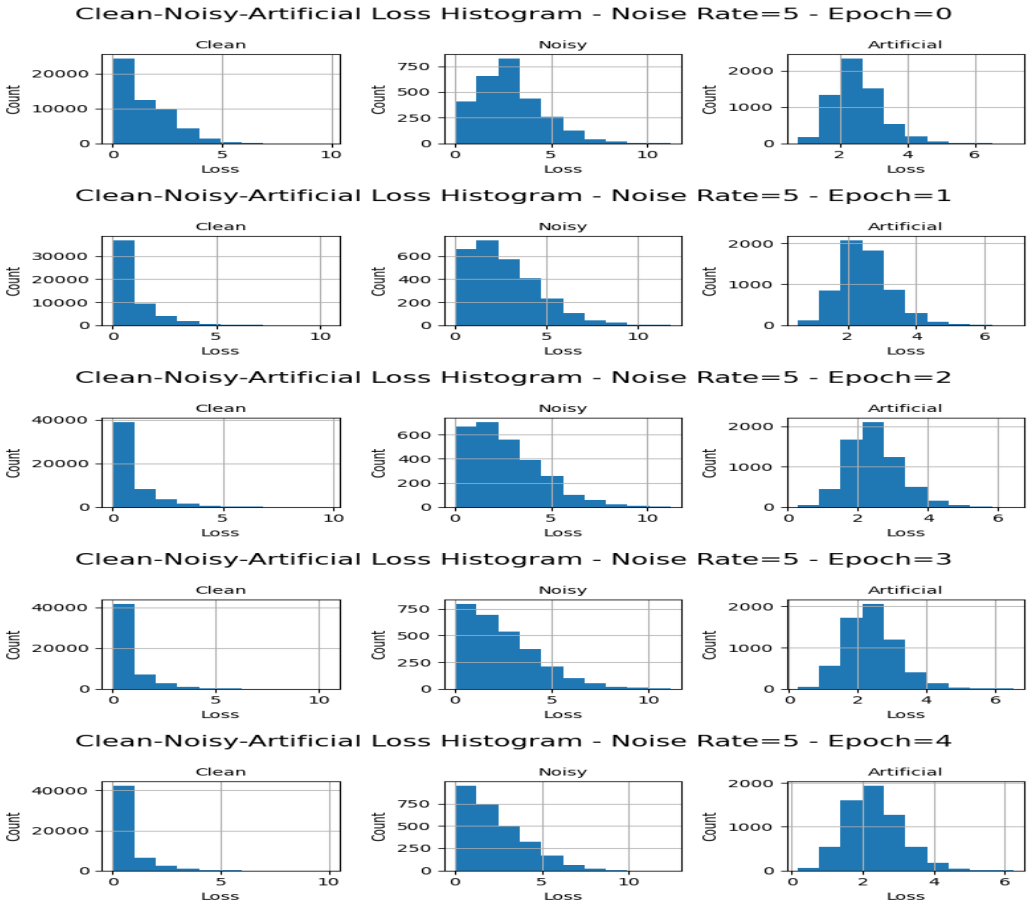

Figure 4.10: Noise Histograms of Clean-Noisy-Artificial Samples of Clothing1M V2 for $\%5$ Noise Rate

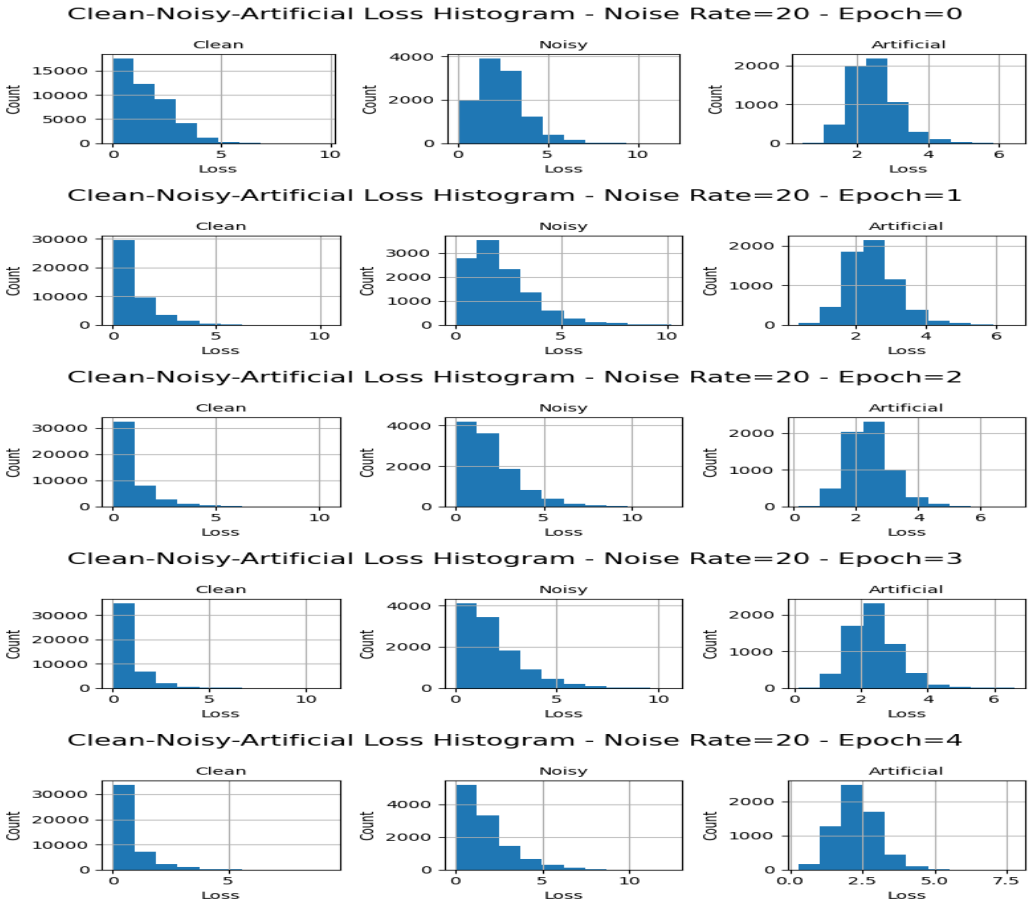

Figure 4.11: Noise Histograms of Clean-Noisy-Artificial Samples of Clothing1M V2 for $\%20$ Noise Rate

Observations from both high and low noise rate cases are consistent and both implies that artificial samples are not successful for setting a threshold to distinguish clean samples from noisy ones since artificial samples can not mimic noisy samples and peak values of the loss histograms of artificial samples do not fall into a region between the peaks of clean and noisy ones.

4.5 Clean Sample Extraction Performance of the Proposed Method

In this section, clean sample extraction performance of our proposed method is presented. Firstly, performance over toy datasets such as MNIST, MNIST-Fashion, CIFAR10, CIFAR100 is provided. These toy datasets are corrupted with synthetic uniform noise. Secondly, performance over two variations of Clothing1M(Clothing1M

V1 and Clothing1M V2) is presented. Clothing1M versions are more realistic datasets compared to toy ones. Also, Clothing1M V2 contains real world noisy samples which makes it different from other datasets.

Tables 4.10, 4.11, 4.12, 4.13 present clean sample extraction performance over MNIST, MNIST-Fashion, CIFAR10, CIFAR100 respectively. False positive percentage value given in the tables corresponds to new noise rate of the filtered dataset and recall of the clean samples shows us how much of the clean samples are detected correctly in the filtered dataset. Besides of CIFAR100, our proposed algorithm is successful in terms of both finding clean samples and eliminating noisy samples over toy datasets. This observation coincides with our expectations mentioned in Section 4.4 about detecting clean samples successfully using artificial samples in synthetic uniform noise case.

The only unexpected results from toy datasets are obtained over CIFAR100 dataset. While noisy samples are detected successfully, we lost most of the clean samples during filtering operation. The most probable reason of bad results obtained over CIFAR100 is large number of classes exist in dataset. CIFAR100 has 100 classes inside and this large number of classes complicates things. We utilize artificial samples to mimic noisy samples during clean-noisy classification of samples. Our artificially corrupted samples are from a single class while original noisy samples(samples corrupted with synthetic uniform noise) of CIFAR100 can be from any of the 100 classes and so, mimicking original noisy samples via our artificial samples is getting difficult in CIFAR100 dataset.

Table 4.10: Clean Sample Extraction Performance over MNIST

Noise Rate	**True Positive**	**False Positive**	**Recall of Clean Samples**
20	99.85	0.14	99.90
30	99.65	0.34	99.88
40	99.36	0.63	99.86
50	98.79	1.20	99.82

Table 4.11: Clean Sample Extraction Performance over MNIST-Fashion

Noise Rate	True Positive	False Positive	Recall of Clean Samples
20	98.99	1.01	98.53
30	98.39	1.60	98.43
40	97.42	2.57	98.25
50	95.42	4.57	97.99

Table 4.12: Clean Sample Extraction Performance over CIFAR10

Noise Rate	True Positive	False Positive	Recall of Clean Samples
20	97.54	2.45	94.14
30	95.17	4.82	92.29
40	92.11	7.88	91.02
50	86.79	13.2	87.90

Table 4.13: Clean Sample Extraction Performance over CIFAR100

Noise Rate	True Positive	False Positive	Recall of Clean Samples
20	99.72	0.27	34.27
30	99.66	0.33	13.65

Tables 4.14 and 4.15 present clean sample extraction performance over Clothing1M V1 and Clothing1M V2 respectively. Similar to toy datasets case, our expectation is obtaining good results over Clothing1M V1 since it contains uniform synthetic noise but not real world noise. Table 4.14 shows that clean samples extraction performance of Clothing1M V1 coincides with our expectations which are obtained in Section 4.4 since our algorithm able to eliminates majority of noisy samples without losing too much clean samples.

On the other hand, we expect a poor clean sample extraction performance over Clothing1M V2 because of our observations from Section 4.4. Table 4.15 shows that we have high recall rate for clean samples under all noise levels but our noisy sample detection performance is poor. Poor performance of the noisy sample detection pro-

cess can be understood from false positive rates which represent new noise rates of the filtered datasets. We observed that false positive rates given in the table are very close to original noise rates and so, we can claim that most of the noisy samples are classified as clean wrongly.

Table 4.14: Clean Sample Extraction Performance over Clothing1M V1

Noise Rate	**True Positive**	**False Positive**	**Recall of Clean Samples**
10	97.69	2.31	95.77
20	94.98	5.02	94.86
30	91.92	8.08	93.81
40	88.63	11.37	90.28
50	85.53	14.47	85.60

Table 4.15: Clean Sample Extraction Performance over Clothing1M V2

Noise Rate	**True Positive**	**False Positive**	**Recall of Clean Samples**
5	96.86	3.14	96.72
10	92.81	7.19	96.99
15	88.03	11.97	97.18
20	83.40	16.60	97.22

CHAPTER 5

CONCLUSION

In this work, the strong and weak sides of the sample selection-based methods are investigated. To achieve a fair comparison of these methods, a common setup is constructed, and extensive experimentation results of the algorithms are provided. In addition to investigated methods, a novel pipeline is proposed and compared with methods from the literature over a common setup. Firstly, non-robust base models are trained over datasets with both synthetic uniform and real-world label noise to prove the tendency of the deep neural networks to learn from clean samples first and then learn from noisy samples. This tendency justifies the small loss selection idea since clean samples should have small loss values compared to noisy ones as a result of learning from clean samples firstly. Also, the experimental results of the small loss selection methods proved this conclusion. Secondly, small loss selection methods are trained twice by utilizing and not utilizing prior knowledge related to the noise rate of the dataset. Experimental results showed that the performance of the small loss selection methods degrades when prior knowledge of noise rate is not utilized during training, and this is the main weakness of these methods.

Thirdly, semi-supervised learning-based methods from the literature are trained over noisy datasets. Experiments proved their superiority over supervised learning-based methods in general. The superiority of semi-supervised learning-based methods revealed that each sample in the dataset is important and should be utilized during training even if it is noisy labeled. Fourthly, a weak side of contrastive learning over noisy labeled datasets is proved. At low noise rates, contrastive learning results with low performance since it utilizes noisy labeled samples as unlabelled samples during training, and a contrastive loss calculated over a small amount of unlabeled data de-

grades network performance.

Fifthly, we proposed our overall pipeline, which is based on classifying samples as clean or noisy utilizing artificially corrupted samples and then training classified samples via semi-supervised learning. Experimental results of the proposed method proved the power of the idea of utilizing artificially corrupted samples during the clean sample detection process. We have investigated loss histograms of the clean, noisy, and artificial samples for both uniform synthetic and real-world noise cases. Histograms of the synthetic noise cases showed that artificial samples can be utilized successfully to separate noisy and clean samples in datasets with synthetic uniform noise. On the other hand, histograms of the real-world noise case showed that artificial samples are not good enough to mimic real-world noisy samples. These observations are also supported by clean sample extraction experiments. Clean sample extraction experiments revealed that our proposed algorithm is successful in terms of detecting both clean and noisy samples in synthetically corrupted datasets with uniform noise while not successful good enough in terms of detecting noisy samples in real-world noisy datasets. Despite being unsuccessful in terms of detecting noisy samples in real-world noisy datasets, our algorithm has a high recall rate for clean samples, and it is still able to decrease the noise rate of real-world noisy datasets with small amounts. Finally, our models trained with semi-supervised learning over samples classified as clean-noisy outperform small loss selection methods in general and provide compatible results with other semi-supervised learning based state of the art methods. Also, our proposed method is trained twice with and without the co-training approach to see its effectiveness. These experiments about co-training showed that utilizing two networks during training makes it possible to combine different learning abilities of these networks, and combining different learning abilities is an effective approach for learning in the existence of label noise since we observe a performance increase in general with the co-training approach.

REFERENCES

[1] 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?," *CoRR*, vol. abs/2012.03061, 2020.

[2] T. Xiao, T. Xia, Y. Yang, C. Huang, and X. Wang, "Learning from massive noisy labeled data for image classification," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2691–2699, 2015.

[3] X. Yu, B. Han, J. Yao, G. Niu, I. W. Tsang, and M. Sugiyama, "How does disagreement help generalization against label corruption?," *CoRR*, vol. abs/1901.04215, 2019.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems* (F. Pereira, C. Burges, L. Bottou, and K. Weinberger, eds.), vol. 25, Curran Associates, Inc., 2012.

[5] Q. Xie, E. H. Hovy, M. Luong, and Q. V. Le, "Self-training with noisy student improves imagenet classification," *CoRR*, vol. abs/1911.04252, 2019.

[6] L. Schmarje, J. Brünger, M. Santarossa, S. Schröder, R. Kiko, and R. Koch, "Beyond cats and dogs: Semi-supervised classification of fuzzy labels with overclustering," *CoRR*, vol. abs/2012.01768, 2020.

[7] L. Schmarje, M. Santarossa, S.-M. Schroder, and R. Koch, "A survey on semi-, self- and unsupervised learning for image classification," *IEEE Access*, vol. 9, pp. 82146–82168, 2021.

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

[9] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context," *CoRR*, vol. abs/1405.0312, 2014.

[10] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona, "Caltech-UCSD Birds 200," Tech. Rep. CNS-TR-2010-001, California Institute of Technology, 2010.

[11] D. K. Wallace, G. E. Quinn, S. F. Freedman, and M. F. Chiang, "Agreement among pediatric ophthalmologists in diagnosing plus and pre-plus disease in retinopathy of prematurity," *J AAPOS*, vol. 12, pp. 352–356, Mar. 2008.

[12] D. Mahajan, R. B. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. van der Maaten, "Exploring the limits of weakly supervised pretraining," *CoRR*, vol. abs/1805.00932, 2018.

[13] W. Li, L. Wang, W. Li, E. Agustsson, and L. V. Gool, "Webvision database: Visual learning and understanding from web data," *CoRR*, vol. abs/1708.02862, 2017.

[14] D. Rolnick, A. Veit, S. J. Belongie, and N. Shavit, "Deep learning is robust to massive label noise," *CoRR*, vol. abs/1705.10694, 2017.

[15] J. Krause, B. Sapp, A. Howard, H. Zhou, A. Toshev, T. Duerig, J. Philbin, and L. Fei-Fei, "The unreasonable effectiveness of noisy data for fine-grained recognition," *CoRR*, vol. abs/1511.06789, 2015.

[16] D. Arpit, S. Jastrzundefinedbski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio, and S. Lacoste-Julien, "A closer look at memorization in deep networks," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, p. 233–242, JMLR.org, 2017.

[17] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," *CoRR*, vol. abs/1611.03530, 2016.

[18] E. Malach and S. Shalev-Shwartz, "Decoupling "when to update" from "how to update"," in *Advances in Neural Information Processing Systems 30* (I. Guyon,

U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), pp. 960–970, Curran Associates, Inc., 2017.

[19] 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," in *Proceedings of the 35th International Conference on Machine Learning* (J. Dy and A. Krause, eds.), vol. 80 of *Proceedings of Machine Learning Research*, (Stockholmsmässan, Stockholm Sweden), pp. 2304–2313, PMLR, 10–15 Jul 2018.

[20] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. W.-H. Tsang, and M. Sugiyama, "Co-teaching: Robust training of deep neural networks with extremely noisy labels," in *NeurIPS*, 2018.

[21] H. Wei, L. Feng, X. Chen, and B. An, "Combating noisy labels by agreement: A joint training method with co-regularization," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13723–13732, 2020.

[22] Y. Chen, X. Shen, S. X. Hu, and J. A. K. Suykens, "Boosting co-teaching with compression regularization for label noise," *CoRR*, vol. abs/2104.13766, 2021.

[23] D. T. Nguyen, C. K. Mummadi, T. Ngo, T. H. P. Nguyen, L. Beggel, and T. Brox, "SELF: learning to filter noisy labels with self-ensembling," *CoRR*, vol. abs/1910.01842, 2019.

[24] G. Pleiss, T. Zhang, E. R. Elenberg, and K. Q. Weinberger, "Identifying mislabeled data using the area under the margin ranking," *CoRR*, vol. abs/2001.10528, 2020.

[25] O. Litany and D. Freedman, "SOSELETO: A unified approach to transfer learning and training with noisy labels," *CoRR*, vol. abs/1805.09622, 2018.

[26] D. Hendrycks, M. Mazeika, D. Wilson, and K. Gimpel, "Using trusted data to train deep networks on labels corrupted by severe noise," *CoRR*, vol. abs/1802.05300, 2018.

[27] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

[28] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. W. Tsang, and M. Sugiyama, "Co-sampling: Training robust networks for extremely noisy supervision," *CoRR*, vol. abs/1804.06872, 2018.

[29] G. Algan and I. Ulusoy, "Label noise types and their effects on deep learning," *CoRR*, vol. abs/2003.10471, 2020.

[30] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015. cite arxiv:1503.02531Comment: NIPS 2014 Deep Learning Workshop.

[31] D. Angluin and P. Laird, "Learning from noisy examples," *Machine Learning*, vol. 2, pp. 343–370, Apr 1988.

[32] G. Algan and I. Ulusoy, "Image classification with deep learning in the presence of noisy labels: A survey," *CoRR*, vol. abs/1912.05170, 2019.

[33] H. Song, M. Kim, D. Park, and J. Lee, "Learning from noisy labels with deep neural networks: A survey," *CoRR*, vol. abs/2007.08199, 2020.

[34] D. Karimi, H. Dou, S. K. Warfield, and A. Gholipour, "Deep learning with noisy labels: exploring techniques and remedies in medical image analysis," *CoRR*, vol. abs/1912.02911, 2019.

[35] X. Xia, T. Liu, N. Wang, B. Han, C. Gong, G. Niu, and M. Sugiyama, "Are anchor points really indispensable in label-noise learning?," 2019.

[36] T. Liu and D. Tao, "Classification with noisy labels by importance reweighting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, pp. 447–461, mar 2016.

[37] C. Scott, "A Rate of Convergence for Mixture Proportion Estimation, with Application to Learning from Noisy Labels," in *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics* (G. Lebanon and S. V. N. Vishwanathan, eds.), vol. 38 of *Proceedings of Machine Learning Research*, (San Diego, California, USA), pp. 838–846, PMLR, 09–12 May 2015.

[38] N. Natarajan, I. S. Dhillon, P. K. Ravikumar, and A. Tewari, "Learning with noisy labels," in *Advances in Neural Information Processing Systems*

(C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, eds.), vol. 26, Curran Associates, Inc., 2013.

[39] S. Sukhbaatar, J. Bruna, M. Paluri, L. Bourdev, and R. Fergus, "Training convolutional networks with noisy labels," Jan. 2015. 3rd International Conference on Learning Representations, ICLR 2015 ; Conference date: 07-05-2015 Through 09-05-2015.

[40] G. Patrini, A. Rozza, A. Menon, R. Nock, and L. Qu, "Making deep neural networks robust to label noise: a loss correction approach," 2016.

[41] Y. Wang, W. Liu, X. Ma, J. Bailey, H. Zha, L. Song, and S.-T. Xia, "Iterative learning with open-set noisy labels," 2018.

[42] M. Ren, W. Zeng, B. Yang, and R. Urtasun, "Learning to reweight examples for robust deep learning," 2018.

[43] 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," 2019.

[44] K.-H. Lee, X. He, L. Zhang, and L. Yang, "Cleannet: Transfer learning for scalable image classifier training with label noise," 2017.

[45] X. Wang, Y. Hua, E. Kodirov, and N. M. Robertson, "Emphasis regularisation by gradient rescaling for training deep neural networks with noisy labels," *CoRR*, vol. abs/1905.11233, 2019.

[46] A. Ghosh, H. Kumar, and P. S. Sastry, "Robust loss functions under label noise for deep neural networks," 2017.

[47] X. Wang, E. Kodirov, Y. Hua, and N. M. Robertson, "Improving MAE against CCE under label noise," *CoRR*, vol. abs/1903.12141, 2019.

[48] Y. Wang, X. Ma, Z. Chen, Y. Luo, J. Yi, and J. Bailey, "Symmetric cross entropy for robust learning with noisy labels," *CoRR*, vol. abs/1908.06112, 2019.

[49] M. Andrychowicz, M. Denil, S. G. Colmenarejo, M. W. Hoffman, D. Pfau, T. Schaul, and N. de Freitas, "Learning to learn by gradient descent by gradient descent," *CoRR*, vol. abs/1606.04474, 2016.

[50] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," *CoRR*, vol. abs/1703.03400, 2017.

[51] Y. Li, J. Yang, Y. Song, L. Cao, J. Luo, and J. Li, "Learning from noisy labels with distillation," *CoRR*, vol. abs/1703.02391, 2017.

[52] J. Li, Y. Wong, Q. Zhao, and M. S. Kankanhalli, "Learning to learn from noisy labeled data," *CoRR*, vol. abs/1812.05214, 2018.

[53] O. Rippel, M. A. Gelbart, and R. P. Adams, "Learning ordered representations with nested dropout," in *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, p. II–1746–II–1754, JMLR.org, 2014.

[54] J. Li, R. Socher, and S. C. H. Hoi, "Dividemix: Learning with noisy labels as semi-supervised learning," *CoRR*, vol. abs/2002.07394, 2020.

[55] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, COLT' 98, (New York, NY, USA), p. 92–100, Association for Computing Machinery, 1998.

[56] E. A. Sanchez, D. Ortego, P. Albert, N. E. O'Connor, and K. McGuinness, "Unsupervised label noise modeling and loss correction," *CoRR*, vol. abs/1904.11238, 2019.

[57] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *CoRR*, vol. abs/1710.09412, 2017.

[58] N. Karim, M. N. Rizve, N. Rahnavard, A. Mian, and M. Shah, "Unicon: Combating label noise through uniform selection and contrastive learning," 2022.

[59] L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[60] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *CoRR*, vol. abs/1708.07747, 2017.

[61] H. Song, M. Kim, and J.-G. Lee, "SELFIE: Refurbishing unclean samples for robust deep learning," in *ICML*, 2019.

[62] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *CoRR*, vol. abs/1512.03385, 2015.

[63] E. D. Cubuk, B. Zoph, D. Mané, V. Vasudevan, and Q. V. Le, “Autoaugment: Learning augmentation policies from data,” *CoRR*, vol. abs/1805.09501, 2018.