NDSGD: A Practical Method to Improve Robustness of Deep Learning Model on Noisy Dataset

Zhi Chen

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2020-55

http://www2.eecs.berkeley.edu/Pubs/TechRpts/2020/EECS-2020-55.html

May 22, 2020

Acknowledgement

This research project was made possible by the constant support from my professors and colleagues at the EECS Department and the Center for Long-Term Cybersecurity, UC Berkeley. I would like to thank Prof. Dawn Song for being my research advisor during my graduate study. Without her strong support and research resources, none of this would be possible. My special thanks go to Dr. Min Du for her guidance leading me into the world of research on deep learning and security. I also want to thank Dr. Ruoxi Jia for her support during the later stages of my M.S. research project, particularly her valuable feedback over this report.

NDSGD: A Practical Method to Improve Robustness of Deep Learning Model on Noisy Dataset

by Zhi Chen

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:

[illegible]

Dawn Song
Research Advisor

05/15/2020

(Date)

* * * * * * *

[illegible]

Jiantao Jiao
Second Reader

05/15/2020

(Date)

Abstract

Noisy labels bring new challenges to deep learning. There are many data sources available for people to download on the web, but they tend to contain inaccurate labels. Training on these datasets causes performance degradation because deep neural network may remember the label noise easily. Existing solutions include constructing noise elimination algorithms to separate noisy labels or proposing noise-robust algorithms to learn directly from noisy labels, but the intrinsic mechanisms and the scalability of deep neural network remain not well designed.

This paper proposes a novel approach called Noisy Dataset Stochastic Gradient Descent (NDSGD), which aims to optimize each step (i.e., noisy data clipping, carry out groups and add robustness factors) of stochastic gradient descent to improve the robustness of deep learning models. The experimental results on MNIST, NEWS and CIFAR-10 demonstrate that NDSGD is superior for noisy dataset and can make the deep learning model robust in noisy environment while maintain high accuracy.

Contents

List of Figures

List of Tables

1 . Introduction

With the development of computing power and the increment of data in recent years, deep learning is widely used in various fields, such as image understanding, natural language, speech, machine translation and robotics research, etc. On the one hand, big data is an indispensable part if we want to train a deep neural network and the characteristics of big data are often defined as 4V's model[49, 47, 11], i.e., large volume, large variety, large velocity and large veracity. On the other hand, supervised learning is used for training deep neural networks on well-annotated datasets. However, with a view to the veracity, low-quality data is prevalent in big data, implied by the fact that there are a large number of incorrect labels, incomplete objects and redundant objects in noisy environment, it's often difficult for us to obtain pure dataset that are carefully annotated because 1) annotation is time-consuming and expensive and 2) some annotation tasks need domain knowledge. As we know, the popular ImageNet[16] dataset was collected from the web[4] and labeled by human labelers for more than one year using Amazon's Mechanical Turk crowd-sourcing tool.

To address the label issues in the real world scenario, some other approaches were proposed by researchers to mitigate the need for expensive annotations. Traditional methods of learning from noisy data can be generally divided into two categories: 1) Propose noise elimination algorithms to get a clean dataset in the data preprocessing phase. 2) Propose some noise-robust algorithms to learn directly from noisy labels in the training phase.

For the noise elimination algorithms to get a clean dataset, the representative methods include majority vote and consensus filters[13], semi-supervised learning[51], self-supervised learning[36, 45], unsupervised learning[27], setting noise layer in the network[40] and minimization in the presence of random classification noise using statistical method[33]. Although these methods allow a clear view of the noises, considering the additional overhead of recognizing the noise and the complexity of the procedures, people have begun using noise-robust algorithms such as deploying semidefinite relaxation for networks (ReLU)[37] or learning directly from noisy labels[20, 22, 21, 10, 23, 25, 30, 32, 42], the newest and representative approaches include Pumpout[20], Mentornet[22], Co-teaching[21]. Pumpout uses gradient decent on good data and uses scaled gradient ascent on bad data. Since the pattern that Pumpout uses is single, this method is hard to be controlled and adjusted in the real-world scenario. Mentornet and Co-teaching need two interactive deep neural networks simultaneously. The results show that they have good performance but the complicated network structure may introduce additional computational overhead. Overall, existing researches have focused mainly on the final accuracy, but the intrinsic mechanisms (It is better to get a simple network structure while perform well on the same noisy dataset) and the scalability (It is better that the network for noisy dataset is controllable and adjustable) of the network structure remain not well designed.

In this paper, we try to address the disadvantages of the above methods and propose a meta approach called NDSGD. NDSGD studies the behavior of standard neural network training procedures in settings with massive label noise and adds several flexible patterns in stochastic gradient descent (SGD). NDSGD learns directly from noisy dataset and contains clipping, group, robustness factors and tuning which make the model robust in noisy environment while the accuracy will not drop. We summarize our contributions as follows:

- We propose an enhanced SGD algorithm called NDSGD which can improve the robustness of deep learning model while the overall accuracy of the model will not drop. Moreover, NDSGD focuses mainly on the intrinsic mechanisms and the scalability of the network structure which are not jointly considered by the existing approaches.

- We firstly use noisy data clipping and group to reduce the influence of noisy data. Then, we add robustness factors to reduce the oscillation of the loss curve. Finally, we tune the hyper-parameters to learn optimal models.

- We evaluate our approaches on several celebrated datasets, i.e., MINST, NEWS and CIFAR-10. The experimental results indicate that our method can achieve a modest cost in the complexity of program, the efficiency of training, and the quality of model. The results also surpass the state-of-the-art methods.

The remainder of this article is organized as follows: We first describe the background in Sec. 2. Then, we introduce our approaches in Sec. 3. We evaluate NDSGD in Sec. 4. The related work and conclusion are presented in Sec. 5 and Sec. 6, respectively.

2 . Background

2.1 Deep Learning

Deep learning, which is known as deep structured learning, is part of a broader family of machine learning methods based on deep neural networks. Unlike the traditional machine learning tasks which need people to perform special feature engineering manually[50, 18], deep learning combine this step in its networks automatically. According to the categories, deep learning can be supervised, semi-supervised or unsupervised[12, 39, 28]. Deep learning architectures such as fully connect feedforward networks, convolutional neural networks, recurrent neural networks, deep residual networks and deep belief networks have been applied to fields including speech recognition, computer vision, audio recognition, natural language processing, machine translation, social network filtering, drug design, material inspection and board game programs, deep learning shows abilities in these areas that are sometimes better than human beings[15, 26].

In deep learning, we want to pick up the best function $f\left(\boldsymbol{\theta}, \boldsymbol{x}_i\right)$ in a large function set $\mathcal{F}$. Specifically, we define a loss function $\mathcal{L}(\theta)$ to represent the deviation between the predict outcome and ground truth for all the training samples, where θ is the set of all parameters, say

$$\theta = \left\{w_1, w_2, \cdots, b_1, b_2, \cdots\right\}.$$

What we want is to optimize the values in θ and minimize $\mathcal{L}(\theta)$, gradient decent has been proposed to solve the optimization problem. Due to the complexity of deep learning, the loss function $\mathcal{L}(\theta)$ is usually non-convex and difficult to minimize. Scientists often use mini-batch stochastic gradient decent (SGD) algorithm instead of the whole sample gradient decent. In SGD, at each step, we first randomly select examples from all samples to form a batch and the batch size is B. Next, we compute the gradient $\mathbf{g}_B$ of the batch, say

$$\mathbf{g}_B = 1/|B| \sum_{x \in B} \nabla_\theta \mathcal{L}(\theta, x).$$

Then θ is updated following the negative gradient. Repeat the above processes, and we will finally get a local minimum.

In order to train a deep network and complete SGD, backpropagation is an efficient way to compute gradient in neural network. For the convenience of more scientists, several toolkits have been proposed to support the definition of neural network, such as Tensorflow[6], Caffe[2], theano[7], torch[8], CNTK[3] and so on. Further, Keras[5] has been built beyond the above toolkits. Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result with the least possible delay is key to doing good research. Our work is built on Tensorflow and Keras, which allows the researchers to customize different computation graphs to their purposes.

2.2 The Risk of Deep Learning

Artificial intelligence, as a strategic technology that leads the future, has increasingly become an important engine driving the acceleration of economic and social fields. In recent years, the explosive growth of data volume, the significant improvement of computing power, and the breakthrough application of deep learning algorithms have greatly promoted the development of artificial intelligence. However, the artificial intelligence technology itself has security problems, which bring huge security risks to the intelligently driven digital world, including: (1) network security risks. At present, the research and development of artificial intelligence products and applications are mainly based on open source frameworks and components such as TensorFlow and Caffe. These open source frameworks and components lack strict security testing and certification, and there may be security risks such as vulnerabilities and backdoors. Once maliciously used by attackers, it can jeopardize the integrity and availability of artificial intelligence products and applications, and even cause significant property damage and adverse social impact. (2) Data security risks. The artificial intelligence system can collect countless seemingly irrelevant data fragments based on it. Through data mining analysis, it can get more information related to user privacy, such as personal behavior characteristics or even personality characteristics. Relearning of the data and re-inference causes the current data anonymization and other security protection measures to invalid, while personal privacy becomes more easily mined and exposed. (3) Algorithm security risk. The problems of uncertainty, instability, and unexplainability of the artificial intelligence algorithm itself will bring a variety of security risks. For example, an incorrect algorithm design or implementation can produce results that are inconsistent with expectations or even harmful. The black-box algorithm makes uninterpretable artificial intelligence decisions, limiting its application scenarios. The training data containing noise or deviations can affect the accuracy of the algorithm model and resist sample attacks. The algorithm is induced to identify misjudgments and missed judgments and produce erroneous results. (4) Privacy and security risks. The accuracy of the artificial intelligence technology model is highly dependent on the training and analysis of massive user data. In particular, a large amount of user personal information needs to be obtained in order to provide personalized and customized services, which will lead to the leakage of user personal information. Obtaining convenience in life and protecting personal privacy have become a dilemma. This shows that the security problem of artificial intelligence has become one of the most important problems that restrict the healthy development and application of artificial intelligence technology.

In response to the increasingly serious situation of artificial intelligence security, countries around the world have issued relevant policies to promote the theoretical research and application of artificial intelligence security, which has risen to the highest national strategic issue. For example, the Science and Technology Committee of the House of Commons of the United Kingdom released the report "Robot Technology and Artificial Intelligence" in 2016. It attempts to test and confirm the transparency of decision-making systems, the minimization of prejudice, privacy and right to know, the liability system that strengthens the control of the security of artificial intelligence. In China's "New Generation Artificial Intelligence Development Plan", it is proposed to establish a sound and transparent artificial intelligence supervision system, implement a double-layer supervision structure with equal emphasis on design accountability and application supervision, as well as a comprehensive approach to artificial intelligence algorithm design, product development, and application of results. However, since artificial intelligence is still in the early stages of industrial application, countries are mainly adopting incentive policies focused on promoting development, and a complete artificial intelligence safety supervision system has not yet been formed. Strengthen the supervision of artificial intelligence safety, and the relevant technical means mainly rely on enterprises to carry out construction. The US White House issued ten principles in 2020, the most important of which is the promotion of fair, transparent and safe trusted artificial intelligence. The European Commission 's document on "Building People-Oriented Trust in Artificial Intelligence" stated that artificial intelligence systems should comprehensively design safety mechanisms to ensure that every step of safety can withstand inspections and ensure the physical and mental safety of all relevant personnel.

Therefore, the issue of artificial intelligence security has attracted widespread attention in academia and industry, and has become one of the most popular research fields. The academic community has proposed a series of models (such as robust models, detection-based defense models) and technologies (such as hidden security technologies and decision boundary perturbation technologies) for the basic theories, testing methods, and defense mechanisms of artificial intelligence security. In order to improve the maturity and reliability of artificial intelligence products, the industry also actively improves the security protection capabilities of artificial intelligence. For example, Baidu's Apollo autonomous driving platform provides a complete security framework and system components based on the isolation and trusted security system to achieve the purpose of preventing network intrusion, protecting user privacy and car information security. The 360 Security Research Institute has repeatedly discovered security vulnerabilities in the artificial intelligence technology system architecture and studied the enhancement technology of artificial intelligence security protection capabilities. In addition, artificial intelligence security technology has been widely used in Ali economies. It combines security and artificial intelligence to evolve a flexible, robust, and self-evolving "security AI" technology suitable for security scenarios.

2.3 Robustness of Deep Learning

Deep neural networks have recently shown impressive classification performance on various tasks, such as computer vision, speech recognition, medical image analysis, e-sports and other fields, and their development has aroused widespread concern in academia, industry, and even politics. However, the effectiveness of existing deep learning methods depends on the high quality requirements of the training data set. When the training set presents problems with significant complex noise, abnormal point intrusion, and imbalanced categories, its effectiveness is often not guaranteed. It is called the robustness problem of deep learning [17].

Taking computer vision as an example, its representative tasks mainly include low-level vision and high-level vision. The underlying visual tasks mainly refer to image processing tasks at the pixel level, such as image denoising, image super-resolution, and image blurring. The processing methods are mainly divided into traditional model-driven methods and modern data-driven methods. Most of the model-driven methods need to solve the problem by constructing an optimization model. The objective usually includes two parts, error term and regularization term, to model the a priori information of image noise and structure. Through the algorithm design and optimization solution of the optimization model, the expected restored image can be converged. The data-driven method adopts a simpler and more direct strategy: 1) collecting the image pairs of the input and output of the simulation problem (for example, for the denoising problem, it is necessary to separately collect the data pairs containing the noise image and the corresponding clean image); 2) entering a pre-built Deep network, by calling an algorithm toolkit optimized for network parameters (such as stochastic gradient descent (SGD), etc.); 3) obtaining a deep network prediction function that can directly output clean pictures for noisy pictures.

High-level vision tasks mainly face semantic-level image analysis tasks, such as target recognition and target detection. Similar to the usage of low-level vision, the application of deep learning to such tasks also relies on a pre-labeled data set, including a large number of data pairs that simulate network input and its expected output. Taking the most common target recognition problem as an example, researchers need a large number of labeled pictures which contains a certain type of targets to effectively train a DNN. And then the obtained data set is fed into the network for parameters learning to obtain a DNN that can make precise prediction.

Although the above tasks have made gratifying progress and encouraging experimental results (for most low-level and high-level visual tasks, deep learning methods have become the best performing methods at present), but they all face extremely severe robustness problem. For low-level vision problems, both model-driven and data-driven methods lack sufficient processing accuracy and application promotion for images with non-independent, identically distributed noise of complex shapes. The essential reason is that the model-driven method needs to set the error function form in advance, such as the loss function of the L_2 or L_1 norm. From Bayesian perspective, this setting can be interpreted as the assumption that the image noise distribution follows either independent and identical Gaussian distribution or Laplacian distribution, and the noise in real images is usually neither in this relatively simple distribution form nor non-independently and identically distributed distribution form (such as image noise usually exhibits significant spatial correlation characteristics). Such deviations from actual assumptions often lead to poor performance of corresponding optimization problems for real pictures with complex noise, resulting in robust learning problems. For data-driven methods, a large number of "noisy picture-clean picture pairs" need to be collected in advance for the network to learn parameters. Even if a large number of noisy picture types are designed, the simulated noise is still artificially set and generated, which is far from covering the diversity and complexity of real noise. Therefore, the trained network still tends to overfit and does not perform well on actual pictures that contain large deviations in the training data.

For high-level vision problems, similar robust learning problems have also attracted widespread attention in recent years. The main problem is the deviation of the label in the training data. The most representative is the problem of noise labeling and class imbalance. The noise labeling problem refers to that a large number of erroneous labels appear in the training sample labels, which leads to deviations in network training. This situation is extremely common. For example, in order to obtain supervised labeling data at a relatively low cost, the crowdsourcing method is usually used to outsource the labeling task to a large number of volunteers in a voluntary or cheap way. The certainty and the diversity of areas of expertise lead to labeling noise, especially for labeling tasks that require domain knowledge (such as lesion labeling of medical images). The problem of class imbalance refers to the phenomenon that the number of samples in different categories is extremely different in the training data. This is a common robust machine learning problem, which has received more attention and research in the field of computer vision. The reason for this problem is that on the one hand, different categories of pictures have different difficulties in obtaining them. For example, common categories such as cats and dogs are easier to obtain than prehistoric or endangered animals. In fact, for most of the well-known target recognition data sets, there is a kind of uneven "thick-tailed distribution" phenomenon. On the other hand, when we collect positive and negative samples for a certain category, the collection of negative data has greater freedom and diversity than positive samples determined by domain knowledge, resulting in an uneven phenomenon. Among these negative samples, scholars believe that only the edge samples located near the classification surface have a more significant amount of information. A large number of negative samples do not have a substantial effect on classification, but are easy to cause interference. This phenomenon is often called hard negative mining. This problem will also greatly interfere with deep learning, thus creating the need for robust learning.

2.4 Learn with noisy labels

We consider a classification problem with a training set $\mathcal{D}$,

$$\mathcal{D} = \{(\boldsymbol{x}_1, \boldsymbol{y}_1), \ldots, (\boldsymbol{x}_n, \boldsymbol{y}_n)\},$$

where $\boldsymbol{x}_i$ represents the i^{th} sample and $\boldsymbol{y}_i \in \{0, 1\}^c$ is a one-hot vector representing the corresponding noisy label over c classes. From Sec. 2.1, we know that the goal of deep learning is to compute the best function $f(\boldsymbol{\theta}, \boldsymbol{x}_i)$. If we use softmax as the output layer, the loss function can be considered as

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{x}_i) = -\frac{1}{n} \sum_{i=1}^{n} \boldsymbol{y}_i \cdot \log(f(\boldsymbol{x}_i, \boldsymbol{\theta})),$$

where $\cdot$ represents dot product.

In the real-world environment, $\boldsymbol{y}_i$ is not always clean, if $\boldsymbol{y}_i$ contains noise, the neural network will overfit and perform poorly on the test set. In our approach, we use novel SGD (NDSGD) algorithm to learn from noisy labels, the details will be described in Sec. 3.

3 . Methodology

In this section, we first introduce the overall architecture of the proposed novel mechanism for training deep learning model. We then discuss several important topics in its design, including NDSGD algorithm (clipping, group, robustness factor) and hyper-parameter tuning.

3.1 Overview

The overview of our methodology is illustrated in Figure 3.1. Pure data and noisy data are both fed into the neural network, where pure data have correct labels while noisy data contain incorrect ones. In the training phase, the NDSGD algorithm is applied and we totally have four key components, including clopping, group, robustness factor and tuning. Finally, we will get a robust deep learning model which performs well on various missions in the real-world environment.

3.2 NDSGD Algorithm

In the noisy environment, the training set is not always pure. That means, one may get incorrect $\boldsymbol{y}_i$ from training set $\mathcal{D}$ during the supervised learning. To deal with this problem, one possible way is to check the training set and adjust it manually, but it will take a lot of manpower. Therefore, inspired by previous work[9], we propose a more sophisticated and robust approach which aims to reduce the influence of noisy labels during the training phase, especially in the SGD computation.

Algorithm 1 describes the details of our Noise Data Stochastic Gradient Descent method. We begin the algorithm with random parameters θ_0 and end with outputing the optimal parameters θ_T which can minimize the loss function $\mathcal{L}(\theta)$. Meanwhile, at each step of the SGD, we firstly calculate the gradient $\nabla_{\theta_t}\mathcal{L}\left(\theta_t, x_i\right)$ for a random subset which contains G samples, we then clip ℓ_2 norm after getting the single gradient, add robustness factors and calculate the average. Finally, we take a step in the opposite direction of this average robust gradient. Next we will describe the details of the components used in our algorithm.

Noisy data clipping: To reduce the influence of noisy labels, Algorithm 1 needs to bound the influence of each individual samples. To this end, we clip each gradient in ℓ_2

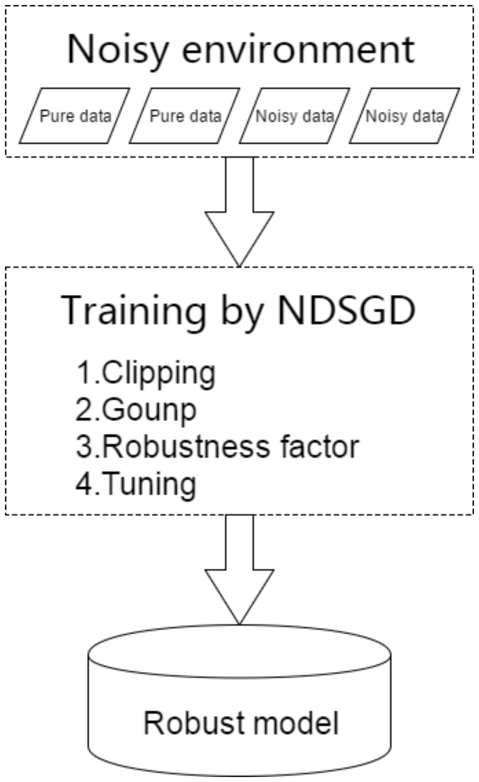


Figure 3.1: System architecture.

Algorithm 1 Noise Data Stochastic Gradient Descent

```
Input:
(1) Traning samples $\{x_1, \dots, x_N\}$;
(2) Loss function $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{x}_i) = -\frac{1}{n}\sum_{i=1}^{n} y_i \cdot \log(f(x_i, \theta))$;
(3) Learning rate $\eta_t$;
(4) Noise scale $\sigma$;
(5) Group size $G$;
(6) Gradient norm bound $C$;
Initialize $\theta_0$ randomly
  for $t \in [T]$ do
    Select random samples into a group $G_t$ with sampling probability $G/N$
    Calculate gradient
    For each $i \in G_t$, calculate $\mathbf{g}_t(x_i) \leftarrow \nabla_{\theta_t}\mathcal{L}(\theta_t, x_i)$
    Norm clipping
    $\overline{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i) / \max\left(1, \frac{\|\mathbf{g}_t(x_i)\|_2}{C}\right)$
    Add Robustness factors (Gaussian noise)
    $\tilde{\mathbf{g}}_t \leftarrow \frac{1}{L}\left(\sum_i \left(\overline{\mathbf{g}}_t(x_i) + \mathcal{N}(0, \sigma_t^2)\right)\right)$
    Gradient Descent
    $\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{\mathbf{g}}_t$
  end for
Output:
The final parameters $\theta_T$.
```

norm, that is, we replace the original gradient vector $\mathbf{g}_t(x_i)$ with $\mathbf{g}_t(x_i) / \max\left(1, \frac{\|\mathbf{g}_t(x_i)\|_2}{C}\right)$, and C is a clipping threshold which is used to control the gradient norm bound. After that, we get a new gradient vector as follow:

$$\overline{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i) / \max\left(1, \frac{\|\mathbf{g}_t(x_i)\|_2}{C}\right).$$

From the above equation, we can easily get that if $\|\mathbf{g}\|_2 \leq C$, $\overline{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i)$. But if $\|\mathrm{g}\|_2 > C$, the values of gradient will be scaled down. As a matter of fact, one can change the hyper-parameter C intelligently to correct the biased gradient direction.

Multi-layer parameters: In Algorithm 1, all the parameters in the deep neural network are grouped into a single input θ, where $\theta = \{w_1, w_2, \cdots, b_1, b_2, \cdots\}$, w represents the weight and b represents the bias. Considering multi-layer neural networks, we deal with each layer separately. That is to say, one can customize the gradient norm bound C and noise scale σ in each layer. Also, in a more sophisticated way, the gradient norm bound and noise scale will dynamically change as the number of epochs increases. In our experiment, we use constant setting for these two parameters because the performance and robustness of the deep learning model is well enough by the proposed method.

Groups: Recall that in our noisy data clipping, Algorithm 1 calculates the gradient of the loss function $\mathcal{L}(\theta)$ by computing the gradient of the loss on a group of samples and taking the average. To further eliminate the effects of noisy label, we introduce a new concept called Group. In deep learning, batch is a common way to prevent local minimum, while a Group consists of several batches. To reduce the memory consumption, we set the batch size much smaller than the Group size G, which is a hyper-parameter in our algorithm. We perform the computation in batches, then put several batches into a Group for revising the gradient. As a matter of fact, the construction of batches and Group is done by randomly shuffling the samples for efficiency. For ease of analysis, however, we assume that each Group is formed by independently picking each sample with probability $q = G/N$, where N is the size of the input dataset.

Robustness factors: There is a long tradition of adding random weight noise in classical neural networks, and it has been under-explored in the optimization of modern deep architectures. Inspire by previous work[34], we consider a simple technique of adding time-dependent Gaussian noise to the gradient at each training step t:

$$\tilde{\mathbf{g}}_t \leftarrow \frac{1}{L}\left(\sum_i \left(\overline{\mathbf{g}}_t\left(x_i\right)+\mathcal{N}\left(0,\sigma_t^2\right)\right)\right).$$

Existing work[46] indicates that adding annealed Gaussian noise by decaying the variance works better than using fixed Gaussian noise. We use the following schedule for most of our experiments:

$$\sigma_t^2=\frac{\alpha}{(1+t)^\gamma},$$

where α is selected from $\{0.01, 0.3, 1.0\}$ and $\gamma = 0.55$. If the gradient noise at the beginning of training is high, the gradient can be away from 0 in the early stages.

Limited gradient descent method: According to the characteristic that DNNs tend to prioritize the learning of the LSRS pattern, the main pattern will be learned first if the scale of the main pattern is dominant. The reverse and main patterns are mutually exclusive. We can estimate the generalization performance of the main pattern by observing the training precisions of the leftover and reverse samples. The accuracy of the leftover samples is approximated to that of the main pattern. Meanwhile, the accuracy of the reverse samples is approximated to that of other regular patterns. Training should be stopped when the main pattern is generalized as much as possible and the learning of other regular patterns is suppressed. We design a leftover-Reverse (LoR) metric to estimate the learning performance of the main pattern. When the LoR reaches its maximum value, the main pattern might be best generalized. It is worth mentioning that LGD is different from the methods which only learn reliable samples generally based on confidence or loss. In fact, they are based on a relatively tight learning condition. However, the relatively tight condition could sometimes be difficult to maintain. In other words, selecting reliable samples is sometimes less reliable. Our method relies on the relatively relaxed condition that the main pattern is dominant in the number of samples, rather than on reliable samples. Because DNNs tend to prioritize the learning of the main pattern over the memorization of noise patterns, most traditional noise-robust methods require a clean validation set to determine when to stop training. However, such traditional methods may be sensitive to validation sets. In reality, making a high-quality validation set might be another problem worth exploring. The LGD method was proposed to consider the situation where clean validation sets are not involved. We embed traditional methods into the LGD framework to adapt to such a situation.

3.3 Tuning in Noisy Environment

In order to balance the robustness, accuracy and overall performance of our approach for multiple complex tasks, we tune the hyper-parameters in noisy environment. Especially, in our experiments, we find that the accuracy of deep learning model is more sensitive to the training hyper-parameters such as group size G, learning rate η_t and noise scale σ. The experimental results will be discussed in Sec. 4.

4 . Evaluation

In this section, we will introduce the dataset and the experimental results using NDSGD. The procedure of our evaluation is shown in Figure 4.1.

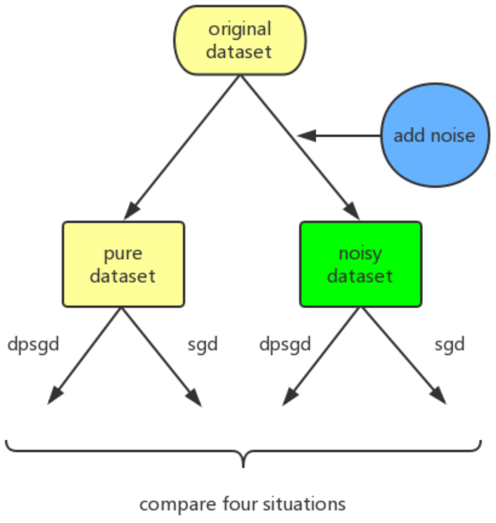


Figure 4.1: The procedure of our evaluation.

4.1 Datasets

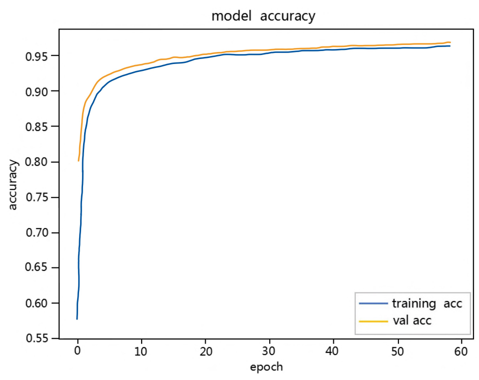
We test the effectiveness of our NDSGD approach on three benchmark datasets, including the hand-written image dataset MNIST, the CIFAR-10 image dataset and NEWS. MNIST consists of $60K + 10K$ (training + testing) 28×28 images of hand-written digits. CIFAR-10 contains $50K + 10K$ (training + testing) 32×32 images of 10 object classes. NEWS consists of $11314 + 7532$ (training + testing) 300-D text data. The reason why we choose these datasets is that they are popularly used for evaluation of noisy labels in the existing works [38, 20, 19, 35, 29].

4.2 Results

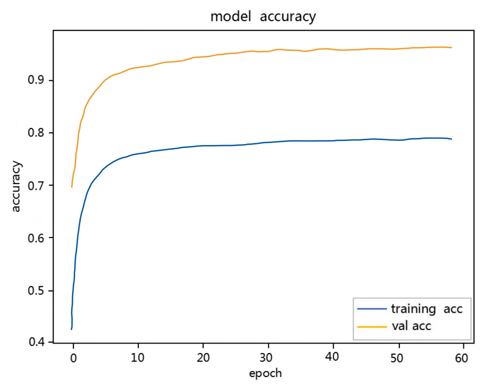
In the experiment, the network structure is a 9-layer CNN from google[1]. The non-linear activation we used was ReLU and we used dropout with parameter 0.5. We trained the network using the Adam optimizer[24] with default parameters. Regarding the configuration of NDSGD, we set the learning rate η_t as 0.001, noise scale σ as 0.01, group size G as 128 and gradient norm bound C as 100. These settings were kept fixed for all the experiments described below. We generated noisy data from clean data by randomly changing some of the labels followed by the existing work[9, 31]. We converted each label with probability p to a different label according to a predefined permutation. The labels of the test data will not change any more in order to validate and compare our method to the regular approach.

In our evaluations, we use two different methods including baseline method and NDSGD. The baseline method is the 9-layer CNN network structure described above[1]. For the NDSGD method, the accuracy of NDSGD at different noise ratios on MNIST is shown in Figure 4.2. We can see that as noise data increase, the training accuracy decreases while the validation accuracy remains the same, which demonstrates the robustness of our method. To reflect the advantages of our work, we perform the same tasks on the baseline method[1]. The accuracy of the baseline at different noise ratios on MNIST is shown in Figure 4.3. Each subgraph shows the test accuracy vs. number of epochs on MNIST dataset, in the 20% noise, 30% noise and 50% noise case, both baseline method and NDSGD perform well. However, as the time goes, NDSGD significantly outperforms baseline method due to the usage of clipping and group. Moreover, we list the overall convergent performance of NDSGD compared against baseline method on MNIST in Table 4.1. All in all, in the 20% noise, 30% noise and 50% noise case, our NDSGD works better and better along with the training epochs though it fluctuates and finally achieves the higher accuracy over baseline method.

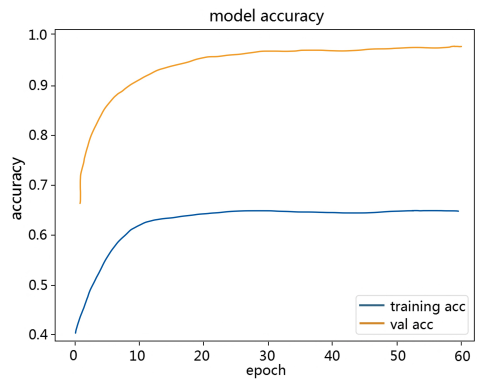
Follow the same way, we evaluated our method on another dataset, NEWS. Figure 4.4 and Figure 4.5 show the accuracy of NDSGD at different noise ratios on NEWS using NDSGD and baseline method, respectively. Table 4.2 lists the final accuracy of baseline methed and NDSGD under the training set with different noise ratios. We can find that the NDSGD method can improve the performance on pure training set. In the 20% noise scenario, the validation accuracy of NDSGD is 63.70%, while the accuracy of baseline is only 37.21%. Though the validation accuracy of NDSGD



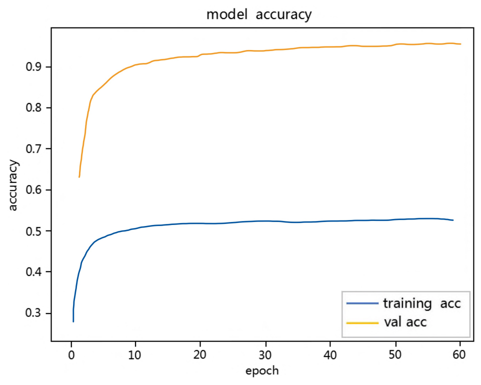
(a) Pure data



(b) 20% noise data

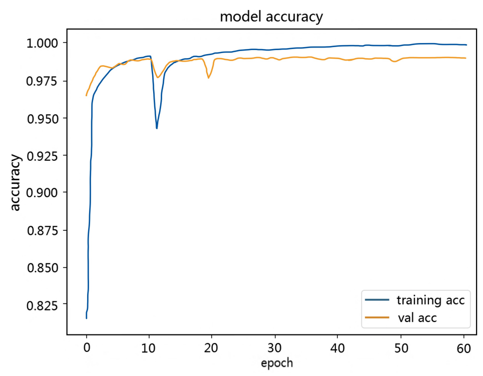


(c) 30% noise data

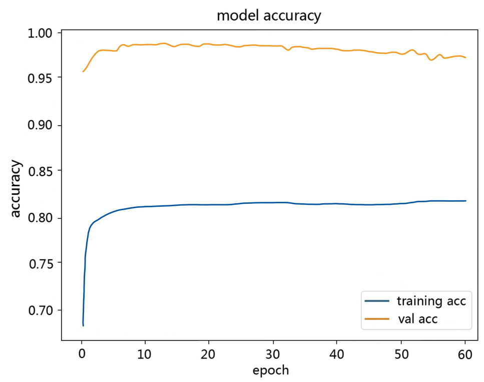


(d) 50% noise data

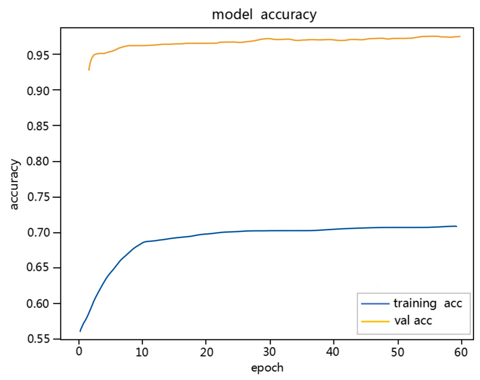
Figure 4.2: The accuracy at different noise ratios on MNIST using NDSGD. The results are shown for several noisy data sizes (0, 20%, 30%, 50%) of the training set for the same CNN network architecture.



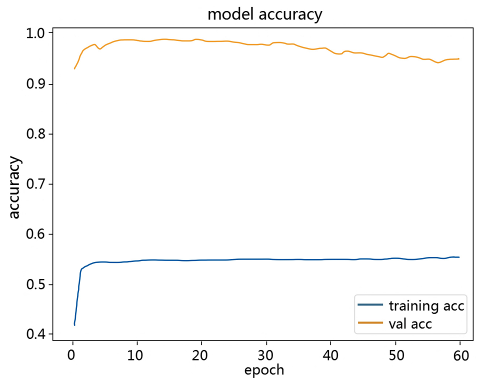
(a) Pure data



(b) 20% noise data



(c) 30% noise data



(d) 50% noise data

Figure 4.3: The accuracy at different noise ratios on MNIST using baseline method. The results are shown for several noisy data sizes (0, 20%, 30%, 50%) of the training set for the same CNN network architecture.

Table 4.1: The validation accuracy of NDSGD compared with other related work on MNIST.

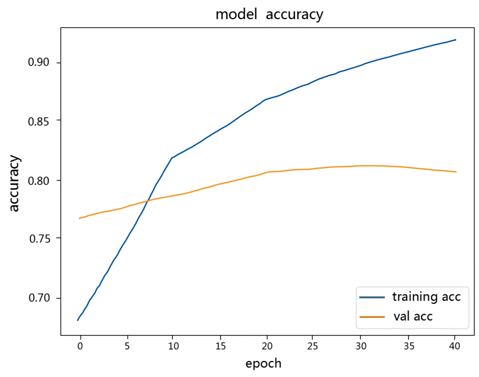
MNIST				
Studies	Pure data	20% noise	30% noise	50% noise
Baseline[1]	99.15%	96.68%	95.42%	93.97%
NDSGD	98.29%	98.32%	98.30%	98.14%

Table 4.2: The validation accuracy of NDSGD compared with other related work on NEWS.

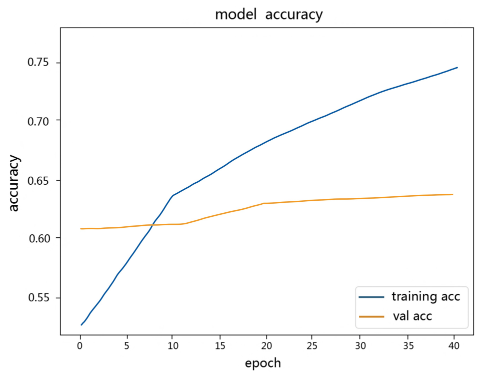
NEWS				
Studies	Pure data	20% noise	30% noise	50% noise
Baseline[1]	53.91%	37.21%	36.33%	36.62%
NDSGD	81.15%	63.70%	46.70%	37.05%

under the 50% noise drops to 37.05%, the main reason is that the noise ratio is outside the acceptable range of the model, but it also performs better than the baseline method. Generally speaking, the experiment on NEWS dataset shows that NDSGD can surpass the baseline method.

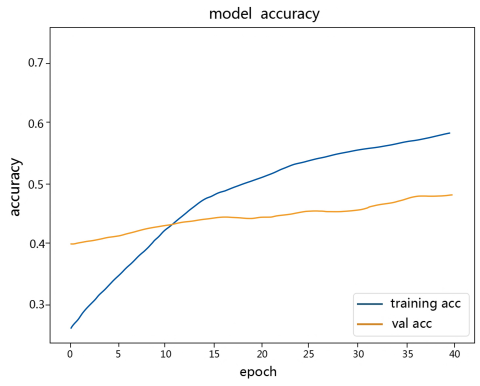
Similarly, we evaluated our method on CIFAR-10. Figure 4.6 and Figure 4.7 show the accuracy of NDSGD at different noise ratios on CIFAR-10 using NDSGD and baseline method, respectively. Table 4.3 lists the final accuracy of baseline methed and NDSGD under the training set with different noise ratios on CIFAR-10.



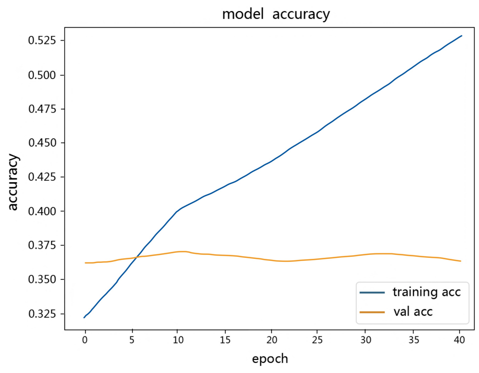
(a) Pure data



(b) 20% noise data

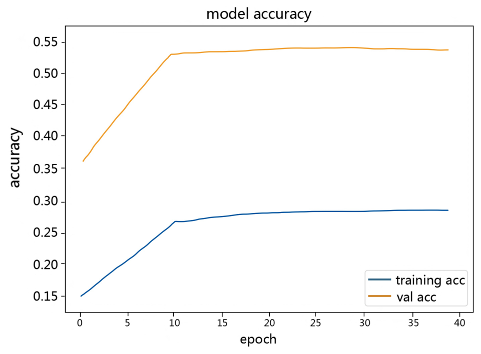


(c) 30% noise data

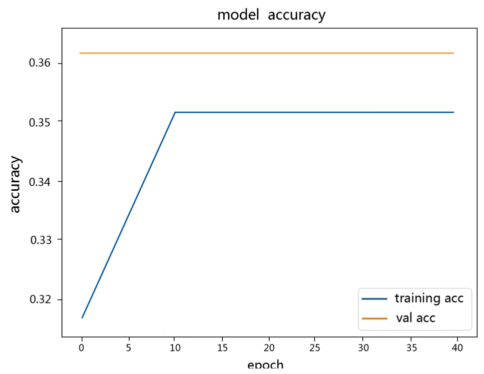


(d) 50% noise data

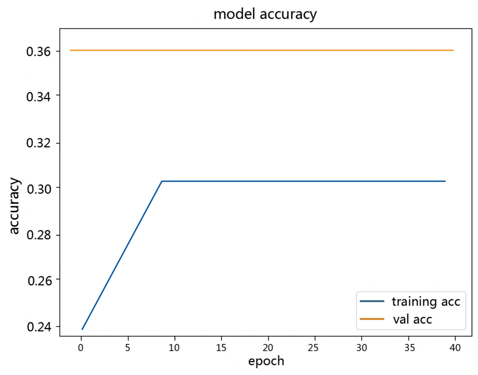
Figure 4.4: The accuracy at different noise ratios on NEWS using NDSGD. The results are shown for several noisy data sizes (0, 20%, 30%, 50%) of the training set for the same CNN network architecture.



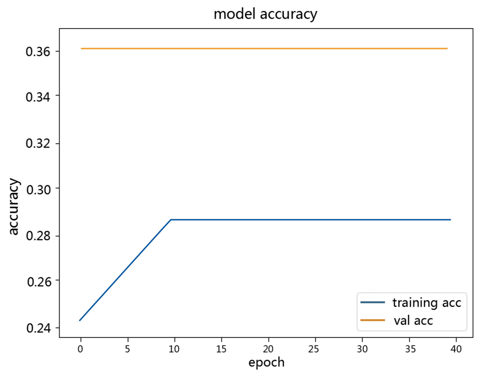
(a) Pure data



(b) 20% noise data

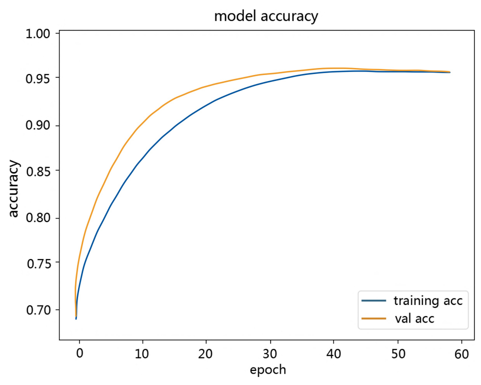


(c) 30% noise data

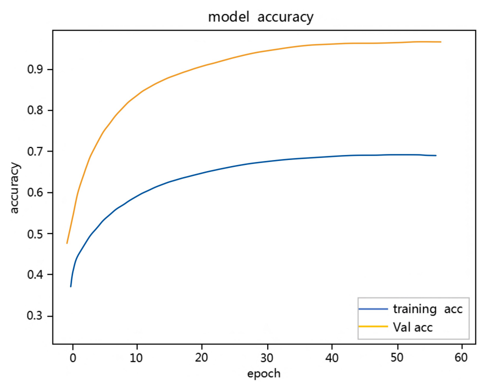


(d) 50% noise data

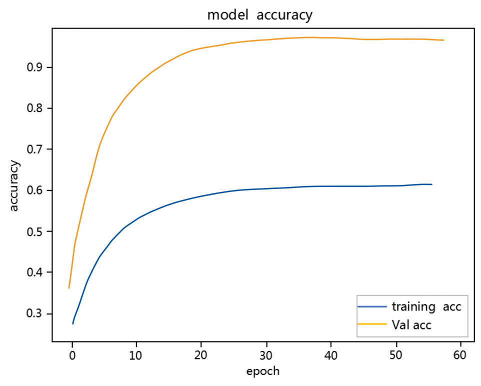
Figure 4.5: The accuracy at different noise ratios on NEWS using baseline method. The results are shown for several noisy data sizes (0, 20%, 30%, 50%) of the training set for the same CNN network architecture.



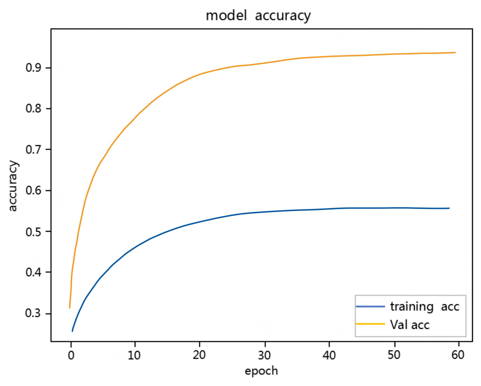
(a) Pure data



(b) 20% noise data

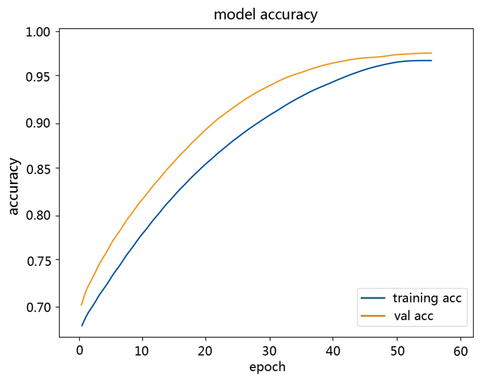


(c) 30% noise data

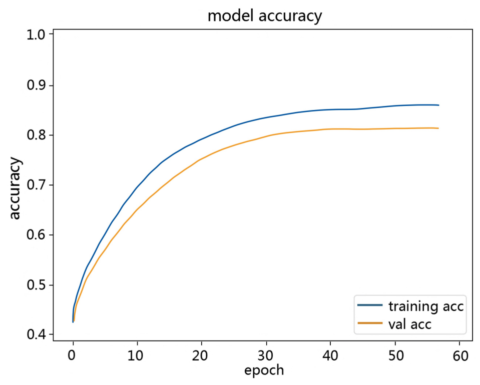


(d) 50% noise data

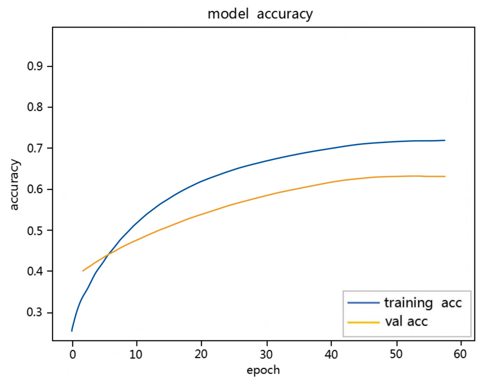
Figure 4.6: The accuracy at different noise ratios on CIFAR-10 using NDSGD. The results are shown for several noisy data sizes (0, 20%, 30%, 50%) of the training set for the same CNN network architecture.



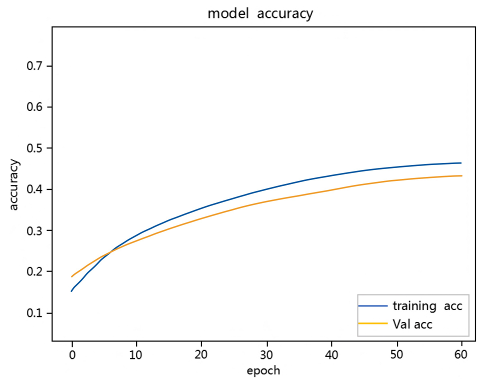
(a) Pure data



(b) 20% noise data



(c) 30% noise data



(d) 50% noise data

Figure 4.7: The accuracy at different noise ratios on CIFAR-10 using baseline method. The results are shown for several noisy data sizes (0, 20%, 30%, 50%) of the training set for the same CNN network architecture.

Table 4.3: The validation accuracy of NDSGD compared with other related work on CIFAR-10.

CIFAR-10				
Studies	Pure data	20% noise	30% noise	50% noise
Baseline[1]	95.91%	86.02%	71.18%	42.03%
NDSGD	95.32%	94.10%	93.28%	92.04%

5 . Related Work

In this section, we describe related work from two aspects: learning from noisy labels and analyzing the robustness of neural networks.

Learn from noisy labels. Several studies have investigated the impact of noisy labels on deep learning classifiers. Approaches to learn from noisy labels can generally be categorized into two groups: In the first group, existing approaches aim to propose noise elimination algorithms to get a clean dataset in the data preprocessing phase. Methods in this group frequently face the challenge of disambiguating between mislabeled and hard training samples. In order to overcome this difficulty, people often use semi-supervised approaches by combining noisy data with a small set of clean labels[51]. Some approaches use unsupervised learning[27] and self-supervised learning[36, 45] to filter out the noisy data. Some methods model the label noise as conditionally independent from the input image[33, 40] and some propose image-conditional noise models[43, 48]. In the second group, existing methods propose some noise-robust algorithms to learn directly from noisy labels in the training phase[20, 22, 21, 10, 23, 25, 30, 32, 42]. Therein, the newest and representative approaches include Pumpout[20], Mentornet[22], Co-teaching[21]. Pumpout aims to squeeze out the negative effects of noisy labels actively from the model being trained, instead of passively forgetting these effects. The realization of Pumpout is to train deep neural networks by stochastic gradient descent "fitting" labels; while train deep neural networks by scaled stochastic gradient ascent on "not-fitting" labels. Since the pattern which Pumpout uses is single, this method is hard to be controlled and adjusted in the real-world scenario. MentorNet employs the small-loss trick. Specifically, MentorNet pre-trains an extra network, and then uses it to select small loss samples as clean samples to guide the training. However, the idea of MentorNet is similar to the approach[14], thus MentorNet inherits the same drawback of accumulated error. Co-teaching aims to maintain two networks simultaneously, and collaboratively-trains on samples screened by the "small loss" criteria. They have good performance but the complicated network structure may introduce additional computational overhead.

Our work differs from these approaches in that we not only consider the final accuracy, but also the intrinsic mechanisms and the scalability of the network structure. We study the behavior of standard neural network training procedures in settings with different proportions of label noise. We demonstrate that utilizing our proposed approach, neural networks can learn from data that has been diluted by a large amount of label noise.

Analyzing the robustness of neural networks. Some studies aim to improve the understanding of deep neural networks, such as CNN. One area of research is to investigate neural networks by analyzing their robustness. Veit et al.[44] show that the network architecture with residual connections has high redundancy in terms of parameters and is robust to deleting multiple complete layers during the test time. Another area of research is to investigate the robustness of neural networks against adversarial examples. Szegedy et al.[41] show that even for fully trained networks, small changes in the input can lead to large changes in the output results. Different from the above researches, we do not consider complex network structure which may introduce a large amount of computational overhead. Also, we are focusing on non-adversarial noise at the training phase. The study which is the closest to our work is Pumpout[20]. As mentioned before, Pumpout's pattern is single, thus this method is hard to be controlled and adjusted in the real-world scenario. On the contrary, NDSGD adds several flexible patterns in stochastic gradient descent and learns directly from noisy dataset, and our approach makes the deep learning model robust in noisy environment and also maintains high accuracy.

6 . Conclusion

This paper presents a meta approach called NDSGD, which can improve robustness of deep learning model in noisy environment. In our heuristic algorithm, we firstly use noisy data clipping and group to reduce the influence of noisy data. Then, we use robustness factors to reduce the oscillation of the loss curve. Finally, we tune the hyper-parameters to learn optimal models. Unlike previous related studies, NDSGD focuses mainly on the intrinsic mechanisms and the scalability of the network structure which are not jointly considered before. The experimental results indicate that our method can achieve a modest cost in the quality of model. By deploying NDSGD in the noisy environment on a large scale, our results also surpass the well-known method.

7 . Acknowledgements

This research project was made possible by the constant support from my professors and colleagues at the EECS Department and the Center for Long-Term Cybersecurity, UC Berkeley.

I would like to thank Prof. Dawn Song for being my research advisor during my graduate study. Without her strong support and research resources, none of this would be possible.

My special thanks go to Dr. Min Du for her guidance leading me into the world of research on deep learning and security. I also want to thank Dr. Ruoxi Jia for her support during the later stages of my M.S. research project, particularly her valuable feedback over this report.

I want to extend my appreciation and respectfulness to all the authors of relevant literatures referred to in this study whose research excellence inspires me to grow and be better every day as a researcher in the field.

References

[1] 9-layer CNN from Google. https://www.tensorflow.org/js/models.

[2] Caffe. https://caffe.berkeleyvision.org/.

[3] CNTK. https://docs.microsoft.com/en-us/cognitive-toolkit/.

[4] ImageNet Overview. http://www.image-net.org/about-overview.

[5] Keras: The Python Deep Learning library. https://keras.io/.

[6] TensorFlow. https://www.tensorflow.org/.

[7] theano. http://deeplearning.net/software/theano/.

[8] torch. http://torch.ch/.

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

[10] Eyal Beigman and Beata Beigman Klebanov. Learning with annotation noise. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 280–287. Association for Computational Linguistics, 2009.

[11] Gema Bello-Orgaz, Jason J Jung, and David Camacho. Social big data: Recent achievements and new challenges. *Information Fusion*, 28:45–59, 2016.

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

[13] Carla E Brodley and Mark A Friedl. Identifying mislabeled training data. *Journal of Artificial Intelligence Research*, 11:131–167, 1999.

[14] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009.

[15] Dan Cireşan, Ueli Meier, and Jürgen Schmidhuber. Multi-column deep neural networks for image classification. *arXiv preprint arXiv:1202.2745*, 2012.

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

[17] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. The robustness of deep networks: A geometrical perspective. *IEEE Signal Processing Magazine*, 34(6):50–62, 2017.

[18] Lei Fu, Tiantian Zhu, Kai Zhu, and Yiling Yang. Condition monitoring for the roller bearings of wind turbines under variable working conditions based on the fisher score and permutation entropy. *Energies*, 12(16):3085, 2019.

[19] Jacob Goldberger and Ehud Ben-Reuven. Training deep neural-networks using a noise adaptation layer. 2016.

[20] Bo Han, Gang Niu, Jiangchao Yao, Xingrui Yu, Miao Xu, Ivor Tsang, and Masashi Sugiyama. Pumpout: A meta approach for robustly training deep neural networks with noisy labels. *arXiv preprint arXiv:1809.11008*, 2018.

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

[22] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. *arXiv preprint arXiv:1712.05055*, 2017.

[23] Armand Joulin, Laurens van der Maaten, Allan Jabri, and Nicolas Vasilache. Learning visual features from large weakly supervised data. In *European Conference on Computer Vision*, pages 67–84. Springer, 2016.

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[25] Jonathan Krause, Benjamin Sapp, Andrew Howard, Howard Zhou, Alexander Toshev, Tom Duerig, James Philbin, and Li Fei-Fei. The unreasonable effectiveness of noisy data for fine-grained recognition. In *European Conference on Computer Vision*, pages 301–320. Springer, 2016.

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[27] Quoc V Le, Marc'Aurelio Ranzato, Rajat Monga, Matthieu Devin, Kai Chen, Greg S Corrado, Jeff Dean, and Andrew Y Ng. Building high-level features using large scale unsupervised learning. *arXiv preprint arXiv:1112.6209*, 2011.

[28] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436, 2015.

[29] Kimin Lee, Sukmin Yun, Kibok Lee, Honglak Lee, Bo Li, and Jinwoo Shin. Robust determinantal generative classifier for noisy labels and adversarial attacks. 2018.

[30] Naresh Manwani and PS Sastry. Noise tolerance under risk minimization. *IEEE Transactions on Cybernetics*, 43(3):1146–1151, 2013.

[31] H Brendan McMahan and Galen Andrew. A general approach to adding differential privacy to iterative training procedures. *arXiv preprint arXiv:1812.06210*, 2018.

[32] Ishan Misra, C Lawrence Zitnick, Margaret Mitchell, and Ross Girshick. Seeing through the human reporting bias: Visual classifiers from noisy human-centric labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2930–2939, 2016.

[33] Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. In *Advances in Neural Information Processing Systems*, pages 1196–1204, 2013.

[34] Arvind Neelakantan, Luke Vilnis, Quoc V Le, Ilya Sutskever, Lukasz Kaiser, Karol Kurach, and James Martens. Adding gradient noise improves learning for very deep networks. *arXiv preprint arXiv:1511.06807*, 2015.

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

[36] Lerrel Pinto, Dhiraj Gandhi, Yuanfeng Han, Yong-Lae Park, and Abhinav Gupta. The curious robot: Learning visual representations via physical interactions. In *European Conference on Computer Vision*, pages 3–18. Springer, 2016.

[37] Aditi Raghunathan, Jacob Steinhardt, and Percy S Liang. Semidefinite relaxations for certifying robustness to adversarial examples. In *Advances in Neural Information Processing Systems*, pages 10877–10887, 2018.

[38] Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596*, 2014.

[39] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

[40] Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. Training convolutional networks with noisy labels. *arXiv preprint arXiv:1406.2080*, 2014.

[41] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[42] Grant Van Horn, Steve Branson, Ryan Farrell, Scott Haber, Jessie Barry, Panos Ipeirotis, Pietro Perona, and Serge Belongie. Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 595–604, 2015.

[43] Andreas Veit, Neil Alldrin, Gal Chechik, Ivan Krasin, Abhinav Gupta, and Serge Belongie. Learning from noisy large-scale datasets with minimal supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 839–847, 2017.

[44] Andreas Veit, Michael J Wilber, and Serge Belongie. Residual networks behave like ensembles of relatively shallow networks. In *Advances in Neural Information Processing Systems*, pages 550–558, 2016.

[45] Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2794–2802, 2015.

[46] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 681–688, 2011.

[47] Xindong Wu, Xingquan Zhu, Gong-Qing Wu, and Wei Ding. Data mining with big data. *IEEE Transactions on Knowledge and Data Engineering*, 26(1):97–107, 2013.

[48] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2691–2699, 2015.

[49] Qingchen Zhang, Laurence T Yang, Zhikui Chen, and Peng Li. A survey on deep learning for big data. *Information Fusion*, 42:146–157, 2018.

[50] Tiantian Zhu, Zhengyang Qu, Haitao Xu, Jingsi Zhang, Zhengyue Shao, Yan Chen, Sandeep Prabhakar, and Jianfeng Yang. Riskcog: Unobtrusive real-time user authentication on mobile devices in the wild. *IEEE Transactions on Mobile Computing*, 2019.

[51] Xiaojin Jerry Zhu. Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2005.