The Dynamic of Consensus in Deep Networks and the Identification of Noisy Labels

By

DANIEL SHWARTZ

THE HEBREW
UNIVERSITY
OF JERUSALEM

Faculty of Computer Science and Engineering
THE HEBREW UNIVERSITY OF JERUSALEM

A dissertation submitted to the Hebrew University of Jerusalem as a partial fulfillment of the requirements of the degree of MASTER OF SCIENCE in the Faculty of Computer Science and Engineering.

UNDER THE SUPERVISION OF **PROF. DAPHNA WEINSHALL**

DECEMBER 2022

Abstract

Deep neural networks have incredible capacity and expressibility, and can seemingly memorize any training set. This introduces a problem when training in the presence of noisy labels, as the noisy examples cannot be distinguished from clean examples by the end of training. Recent research has dealt with this challenge by utilizing the fact that deep networks seem to memorize clean examples much earlier than noisy examples. Here we report a new empirical result: for each example, when looking at the time it has been memorized by each model in an ensemble of networks, the diversity seen in noisy examples is much larger than the clean examples. We use this observation to develop a new method for noisy labels filtration. The method is based on a statistics of the data, which captures the differences in ensemble learning dynamics between clean and noisy data. We test our method on three tasks: (i) noise amount estimation; (ii) noise filtration; (iii) supervised classification. We show that our method improves over existing baselines in all three tasks using a variety of datasets, noise models, and noise levels. Aside from its improved performance, our method has two other advantages. (i) Simplicity, which implies that no additional hyperparameters are introduced. (ii) Our method is modular: it does not work in an end-to-end fashion, and can therefore be used to clean a dataset for any other future usage.

Table of Contents

List of Tables

Table **Page**

List of Figures

CHAPTER 1

INTRODUCTION

Deep neural networks dominate the state of the art in an ever increasing list of application domains, but for the most part, this incredible success relies on very large datasets of annotated examples available for training. Unfortunately, large amounts of high-quality annotated data are hard and expensive to acquire, whereas cheap alternatives (obtained by way of crowd-sourcing or automatic labeling, for example) often introduce noisy labels into the training set. By now there is much empirical evidence that neural networks can memorize almost every training set, including ones with noisy and even random labels Zhang et al. (2017), which in turn increases the generalization error of the model. As a result, the problems of identifying the existence of label noise and the separation of noisy labels from clean ones, are becoming more urgent and therefore attract increasing attention.

Henceforth, we will call the set of examples in the training data whose labels are correct "clean data", and the set of examples whose labels are incorrect "noisy data". While all labels can be eventually learned by deep models, it has been empirically shown that most noisy datapoints are learned by deep models late, after most of the clean data has already been learned (Arpit et al., 2017). Therefore many methods focus on the learning time of an example in order to classify it as noisy or clean, by looking at its loss (Pleiss et al., 2020; Arazo et al., 2019) or loss per epoch (Li et al., 2020) in a single model. However, these methods struggle to classify correctly clean and noisy datapoints that are learned at the same time, or worse - noisy datapoints that are learned early. Additionally, many of these methods work in an end-to-end manner, and thus neither provide noise level estimation nor do they deliver separate sets of clean and noisy data for novel future usages.

Our first contribution is a new empirical results regarding the learning dynamics of an ensemble of deep networks, showing that the dynamics is different when training with clean data vs. noisy data. The dynamics of clean data has been studied in (Hacohen et al., 2020;

Pliushch et al., 2021), where it is reported that different deep models learn examples in the same order and pace. This means that when training a few models and comparing their predictions, a binary occurrence (approximately) is seen at each epoch e: either all the networks correctly predict the example's label, or none of them does. Namely, this result is coming on par with other results (e.g (Nakkiran et al., 2021; Neal et al., 2018)) on the subject of deep neural networks variance. The variance of neural networks tends to behave in an interesting way when the amount parameter is changed. Like classical models, when the number of parameters is small, the network exhibits low variance and low accuracy. When the number of parameters is higher, in the so-called "interpolation zone" where the number of parameters is roughly equal to the data points size - neural networks exhibit large variance and lower accuracy. But unlike classical models, when the number of parameters is much bigger than the number of data points, a unique behavior emerges in the over-parametrized zone, as the variance gets smaller, and the accuracy gets higher. This provides additional evidence that different deep networks learn data at the same time simultaneously, as the variance decrease during training.

In Section 3 we describe a new empirical result: when training an ensemble of deep models with noisy data, and *in contrast to what happens when using clean data, different models learn different datapoints at different times* (see Fig. 1.1). This empirical finding tells us that in an ensemble of networks, the learning dynamics of clean data and noisy data can be distinguished. When training such an ensemble with a mixture of clean and noisy data, the emerging dynamics reflects this observation, as well as the tendency of clean data to be learned faster as previously observed.

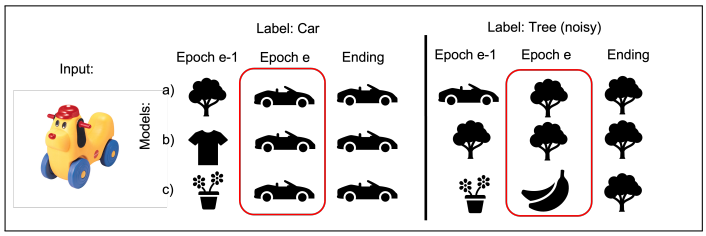


Figure 1.1: With noisy labels models show higher disagreement. The noisy examples are not only learned at a later stage, but each model learns the example at its own different time.

In our second contribution, we use this result to develop a new algorithm for noise level estimation and noise filtration, which we call *DisagreeNet* (see Section 4). Importantly, unlike most alternative methods, our algorithm is simple (it does not introduce any new hyperparameters), parallelizable, easy to integrate with any supervised or semi-supervised learning method and any loss function, and does not rely on prior knowledge of the noise amount. When used for noise filtration, our empirical study (see Section 5) shows the superiority of *DisagreeNet* as compared to the state of the art, using different datasets, different noise models and different noise levels. When used for supervised classification by way of pre-processing the training set prior to training a deep model, it provides a significant boost in performance, more so than alternative methods.

In a broader sense, this work is part of a growing line of works that focus on fundamental experiments with deep neural networks. In this field, the aim of the research is to advance the understanding of deep neural networks by taking a somewhat "behavioral" approach. This is

done by treating our models as black boxes and observing the response when we control the experiment settings. The setting may be the training data, model size, hyperparameters, or anything else, and the response may be the test accuracy, different statistics of the weights, or any traceable behavior of the model. The aim of this field is to provide a qualitative description of the models, if not prove them rigorously.

CHAPTER 2

INTER-NETWORK AGREEMENT

Measuring the similarity between deep models is not a trivial challenge, as modern deep neural networks are complex functions defined by a huge number of parameters, which are invariant to transformations hidden in the model's architecture. Here we measure the similarity between deep models in an ensemble by measuring inter-model prediction agreement at each datapoint. Accordingly, in Section 2.2 we describe scores that are based on the state of the networks at each epoch e, while in Section 2.3 we describe cumulative scores that integrate these states through many epochs. Practically (see Section 4), our proposed method relies on the cumulative scores, which are shown empirically to provide more accurate results in the noise filtration task. These scores promise added robustness, as it is no longer necessary to identify the epoch at which the score is to be evaluated.

2.1 Preliminaries

Notations Let $f^e : \mathbb{R}^d \to [0,1]^{|C|}$ denote a deep model, trained with Stochastic Gradient Descent (SGD) for e epochs on training set $\mathbb{X} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^M$, where $\boldsymbol{x}_i \in \mathbb{R}^d$ denotes a single example and $y_i \in [C]$ its corresponding label. Let $\mathscr{F}^e(\mathbb{X}) = \{f_1^e, ..., f_N^e\}$ denote an ensemble of N such models, where each model $f_{i\in[N]}^e$ is initialized and trained independently on $\mathbb{X}$.

Noise model We analyze the training dynamics of an ensemble of models in the presence of label noise. Label noise is different from data noise (like image distortion or additive Gaussian noise). Here it is assumed that after the training set $\mathbb{X} = \{(\boldsymbol{x}_i, l_i)\}_{i=1}^M$ is sampled, the labels $\{l_i\}$ are corrupted by some noise function $g : [C] \to [C]$, and the training set becomes $\mathbb{X} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^M$, $y_i = g(l_i)$. The two most common models of label noise are termed *symmetric noise* and *asymmetric noise* (Patrini et al., 2017). In both cases it is assumed that some fixed percentage of the labels are corrupted by $g(l)$. With symmetric noise, $g(l)$ assigns any new label from the set $[C] \setminus \{l\}$ with

equal probability. With asymmetric noise, $g(l)$ is the deterministic permutation function. Note that the asymmetric noise model is considered much harder than the symmetric noise model.

2.2 Per-Epoch Agreement Score

Following Hacohen et al. (2020), we define the *True Positive Agreement* (TPA) score of ensemble $\mathscr{F}^e(\mathbb{X})$ at each datapoint $(\boldsymbol{x}, y)$, where

$$TPA(\boldsymbol{x}, y; \mathscr{F}^e(\mathbb{X})) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}_{[f_i^e(\boldsymbol{x})=y]}$$

. The TPA score measures the average accuracy of the models in the ensemble, when seeing $\boldsymbol{x}$, after each model has been trained for exactly e epochs on $\mathbb{X}$. Note that TPA measures the average accuracy of multiple models on one example, as opposed to the generalization error that measures the average error of one model on multiple examples.

2.3 Cumulative Scores

When inspecting the dynamics of the TPA score on clean data, we see that at the beginning the distribution of $\{TPA(\boldsymbol{x}_i, y_i)\}$ is concentrated around 0, and then quickly shifts to 1 as training proceeds (see side panels in Fig. 3.1(a)). This implies that empirically, data is learned in a specific order by all models in the ensemble. To measure this phenomenon we use the *Ensemble Learning Pace* (ELP) score defined below, which essentially integrates the TPA score over a set of epochs $\mathscr{E}$:

$$ELP(\boldsymbol{x}, y) = \frac{1}{|\mathscr{E}|} \sum_{e \in \mathscr{E}} TPA(\boldsymbol{x}, y; \mathscr{F}^e(\mathbb{X})) \tag{2.1}$$

$ELP(\boldsymbol{x}, y)$ captures both the time of learning by a single model, and its consistency across models. For example, if all the models learned the example early, the score would be high. It would be significantly lower if some of them learned it later than others (see pseudo-code in Sec. 4.1).

In our study we evaluated two additional cumulative scores of inter-model agreement:

1. Cumulative loss:
$$CumLoss(\boldsymbol{x}, y) = \frac{1}{N|\mathscr{E}|} \sum_{i, e \in \mathscr{E}} CE(f_i^e(\boldsymbol{x}), y)$$
Above CE denotes the cross entropy function. This score is very similar to ELP, engaging the average of the cross-entropy loss instead of the accuracy indicator $\mathbb{1}_{[f_i^e(\boldsymbol{x})=y]}$.

2. Area under the margin: following (Pleiss et al., 2020), the MeanMargin score is defined as follows
$$MeanMargin(\boldsymbol{x}, y) = \frac{1}{N|\mathscr{E}|} \sum_{i, e \in \mathscr{E}} [f_i^e(\boldsymbol{x})]_{y_i} - \underset{j \neq y_i}{\text{argmax}} [f_i^e(\boldsymbol{x})]_j$$
The MeanMargin score is the mean of the 'margin', the difference between the value of the ground-truth logit (before softmax) and the value of the otherwise maximal logit.

2.4 Overfit and inter-model correlation

In this section we formally analyze the relation between two type of scores, which measure either overfit or inter-model agreement. *Overfit* is a condition that can occur during the training of deep neural networks. It is characterized by the co-occurring decrease of train error or loss, which is continuously minimized during the training of a deep model, and the increase of test error or loss, which is the ideal measure one would have liked to minimize and which determines the network's generalization error. An *agreement* score measures how similar the models are in their predictions.

We start by introducing the model and some notations in Section 2.4.1. In Section 2.4.2 we prove the main result (Prop. 2.4.2): the occurrence of overfit at time s in all the models of the ensemble implies that the agreement between the models decreases.

2.4.1 Model and notations

Model. We analyze the agreement between an ensemble of Q models, computed by solving the linear regression problem with Gradient Descent (GD) and random initialization. In this problem, the learner estimates a linear function $f(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}$, where $\boldsymbol{x} \in \mathbb{R}^d$ denotes an input vector and $y \in \mathbb{R}$ the desired output. Given a training set of M pairs $\{\boldsymbol{x}_m, y_m\}_{m=1}^M$, let $X \in \mathbb{R}^{d \times M}$ denote the training input - a matrix whose m^{th} column is $\boldsymbol{x}_m \in \mathbb{R}^d$, and let row vector $\boldsymbol{y} \in \mathbb{R}^M$ denote the output vector whose m^{th} element is y_m. Let N denote the size of the test set. When solving a linear regression problem, we seek a row vector $\hat{\boldsymbol{w}} \in \mathbb{R}^d$ that satisfies

$$\hat{\boldsymbol{w}} = \text{argmin}_{\boldsymbol{w}} L(\boldsymbol{w}), \qquad L(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w}X - \boldsymbol{y}\|_F^2 \tag{2.2}$$

To solve (2.2) with GD, we perform at each iterative step $s \geq 1$ the following computation:

$$\begin{aligned} \boldsymbol{w}^{s+1} &= \boldsymbol{w}^s - \mu \Delta \boldsymbol{w}^s \\ \Delta \boldsymbol{w}^s &= \frac{\partial L(\mathbb{X})}{\partial \boldsymbol{w}}\Big|_{\boldsymbol{w}=\boldsymbol{w}^s} = \boldsymbol{w}^s \Sigma_{XX} - \Sigma_{YX} \qquad \Sigma_{XX} = XX^\top,\ \Sigma_{YX} = \boldsymbol{y}X^\top \end{aligned} \tag{2.3}$$

for some random initialization vector $\boldsymbol{w}_0 \in \mathbb{R}^d$ where usually $\mathbb{E}[\boldsymbol{w}_0] = 0$, and learning rate μ. Henceforth we omit the index s when self evident from context.

As a final remark, when we use the notation $\|A\|$ below for some matrix A, differently from $\|A\|_F$, it denotes the operator norm of the symmetric matrix A, namely, its largest singular value.

Additional notations

- Index $i \in [Q]$ denotes a network instance, and t denotes the test data. For simplicity and with some risk of notation abuse, let Q and Q' also denote sets of indices, either training or test. Specifically, $Q = [1, \ldots, Q]$ and $Q' = [1, \ldots, Q, t]$.

- We use function notation, where $\{X(i), y(i)\}$ is the training set of network i and $\{X(t), y(t)\}$ is the test set. Thus

$$\Sigma_{XX}(j) = \mathbb{X}(j)\mathbb{X}(j)^\top, \quad \Sigma_{YX}(j) = \boldsymbol{y}(j)\mathbb{X}(j)^\top \qquad j \in Q'$$

- Similarly, $\boldsymbol{w}(i) \in \mathbb{R}^d$ is the model learned by network i, and $\Delta\boldsymbol{w}(i)$ is the gradient step of $\boldsymbol{w}(i)$, where

$$\Delta\boldsymbol{w}(i) = \boldsymbol{w}(i)\Sigma_{XX}(i) - \Sigma_{YX}(i) \qquad i \in Q$$

- $\boldsymbol{e}(i,j)$ denotes a function, which maps indices $i \in Q, j \in Q'$ to the cross error of model i on data j - the classification error vector when using model $\boldsymbol{w}(i)$ to estimate $y(j)$. Let $M' = M$ if $j \in Q$ is a training index, and $M' = N$ if $j \in \{t\}$ .Then we can write

$$\boldsymbol{e}(i,j) : Q \times Q' \to \mathbb{R}^{M'} \qquad \boldsymbol{e}(i,j) = \boldsymbol{w}(i)\mathbb{X}(j) - \boldsymbol{y}(j)$$
$$\implies \qquad \Delta\boldsymbol{w}(i) = \boldsymbol{e}(i,i)\mathbb{X}(i)^\top$$

  Note that in this notation, $e(i,t)$ is the classification error vector when using model i, which is trained on data $\mathbb{X}(i)$, to estimate the desired outcome on the test data - $y(t)$. $\|e(i,t)\|_F$ is the test error, estimate of the generalization error, of classifier i.

- Let $\Delta(i,j)$ denote the cross gradient:

$$\Delta(i,j) = \boldsymbol{e}(i,j)\mathbb{X}(j)^\top = \boldsymbol{w}(i)\Sigma_{XX}(j) - \Sigma_{YX}(j) \quad \implies \quad \Delta\boldsymbol{w}(i) = \Delta(i,i) \tag{2.4}$$

After each GD step, the model and the error are updated as follows:

$$\tilde{\boldsymbol{w}}(i) = \boldsymbol{w}(i) - \mu\Delta\boldsymbol{w}(i)$$
$$\tilde{\boldsymbol{e}}(i,j) = \tilde{\boldsymbol{w}}(i)\mathbb{X}(j) - \boldsymbol{y}(j) = \boldsymbol{e}(i,j) - \mu\Delta(i,i)\mathbb{X}(j)$$

We note that at step s and $\forall i, j \in Q$, $\tilde{\boldsymbol{w}}(i)$ is a random vector in $\mathbb{R}^d$, and $\tilde{\boldsymbol{e}}(i,j)$ is a random vector in $\mathbb{R}^M$. If $j \in \{t\}$, then $\tilde{\boldsymbol{e}}(i,j) = \tilde{\boldsymbol{e}}(i,t)$ is a random vector in $\mathbb{R}^N$.

Test error random variable. Using the above notations, $\{\boldsymbol{e}(i,t)\}_{i=1}^{Q}$ is a set of Q test errors vectors in $\mathbb{R}^N$, where the n^{th} component of the i^{th} vector $\boldsymbol{e}(i,t)_n$ captures the test error of model i on test example n. In effect, it is a sample of size Q from the random variable $\boldsymbol{e}(*,t)_n$. This random variable captures the error over test point n of a model computed from a random sample of size M. The empirical variance of this random variable will be used to estimate the agreement between the models.

Overfit. Overfit occurs at step s if

$$\|\tilde{\boldsymbol{e}}(i,t)\|_F^2 > \|\boldsymbol{e}(i,t)\|_F^2 \tag{2.5}$$

Measuring inter-model agreement. In classification problems, bi-modality of the ELP score captures the agreement between a set of classifiers, all trained on the same training matrix

$\mathbb{X}(i) = X$. Since here we are analyzing a regression problem, we need a comparable score to measure agreement between the predictions of Q linear functions. This measure is chosen to be the variance of the test error among models. Accordingly, we will measure *disagreement* by the empirical variance of the test error random variable $\tilde{\boldsymbol{e}}(*,t)_n$, average over all test examples $n \in [N]$.

More specifically, consider an ensemble of linear models $\{w(i)\}_{i=1}^{Q}$ trained on set $\mathbb{X}$ to minimize (2.2) with s gradient steps, where i denotes the index of a network instance and Q the number of network instances. Using the test error vectors of these models $e(i,t)$, we compute the empirical variance of each element $\text{var}[\boldsymbol{e}(*,t)_n]$, and sum over the test examples $n \in [N]$:

$$\sum_{n=1}^{N} \sigma^2[e(*,t)_n] = \sum_{n=1}^{N} \frac{1}{2Q^2} \sum_{i=1}^{Q}\sum_{j=1}^{Q} |e(i,t)_n - e(j,t)_n|^2 = \frac{1}{2Q^2} \sum_{i=1}^{Q}\sum_{j=1}^{Q} \|e(i,t) - e(j,t)\|_F^2$$

Definition 1 (Inter-model DisAgreement.)**.** *The disagreement among a set of Q linear models $\{w(i)\}_{i=1}^{Q}$ at step s is defined as follows*

$$DisAg(s) = \frac{1}{2Q^2} \sum_{i=1}^{Q}\sum_{j=1}^{Q} \|e(i,t) - e(j,t)\|_F^2 \tag{2.6}$$

2.4.2 Overfit and Inter-Network Agreement

We first prove Lemma 1, which has the following intuitive interpretation: overfit occurs in model i iff the gradient step of model i (denoted $\Delta\boldsymbol{w}(i)$), which is computed using the training set, is negatively correlated with the 'correct' gradient step - the one we would have obtained had we known the test set (this unattainable vector is denoted $\Delta(i,t)$).

Lemma 1. *Assume that the learning rate μ is small enough so that we can neglect terms that are $O(\mu^2)$. Then in each gradient descent step s, overfit occurs iff the gradient step $\Delta\boldsymbol{w}(i)$ of network i is negatively correlated with the cross gradient $\Delta(i,t)$.*

Proof. Starting from (2.5)

$$\begin{aligned}
\text{(overfit)} &\iff \|\tilde{\boldsymbol{e}}(i,t)\|_F^2 > \|\boldsymbol{e}(i,t)\|_F^2 \\
&\iff \|\tilde{\boldsymbol{e}}(i,t)\|_F^2 - \|\boldsymbol{e}(i,t)\|_F^2 = \|\boldsymbol{e}(i,t) - \mu\Delta(i,i)\mathbb{X}(t)\|_F^2 - \|\boldsymbol{e}(i,t)\|_F^2 > 0 \\
&\iff -2\mu\Delta(i,i)\mathbb{X}(t)\boldsymbol{e}(i,t)^\top + O(\mu^2) > 0 \qquad (2.7) \\
&\iff \Delta(i,i)\cdot\Delta(i,t) < 0 \\
&\iff \Delta\boldsymbol{w}(i)\cdot\Delta(i,t) < 0
\end{aligned}$$

■

Lemma 2 claims that if the magnitude of the gradient step μ is small enough, then the operator norm of matrix $I - \mu\Sigma_{XX}$ is smaller than 1. The implication is that a geometric sum of this matrix converges, a technical result which will be used later.

Lemma 2. *For any invertible covariance matrix Σ_{XX} there exists $\hat{\mu} > 0$, such that $\mu < \hat{\mu} \implies \|I - \mu\Sigma_{XX}\| < 1$.*

Proof. Since Σ_{XX} is positive-definite, we can write $\Sigma_{XX} = USU^\top$ for orthogonal matrix U and the diagonal matrix of singular values $S = diag\{s_i\}$. It follows that $I - \mu\Sigma_{XX} = Udiag\{1 - \mu s_i\}U^\top$, a matrix whose largest singular value is $1 - \mu s_d$. Since by assumption $s_d > 0$, the lemma follows. ■

Our last Lemma 3 claims that eventually, after sufficiently many gradient steps, the expected value of the solution is exactly the closed-form solution of the vetor that minimizes the loss.

Lemma 3. *Assume that $\|I - \mu\Sigma_{XX}\| < 1$ and Σ_{XX} is invertible. If the number of gradient steps s is large enough so that $\|I - \mu\Sigma_{XX}\|^s$ can be neglected, then*

$$\mathbb{E}[\boldsymbol{w}^s] \approx \Sigma_{YX}\Sigma_{XX}^{-1} \tag{2.8}$$

Proof. Starting from (2.3), we can show that

$$\boldsymbol{w}^s = \boldsymbol{w}^0(I - \mu\Sigma_{XX})^{s-1} + \mu\Sigma_{YX}\sum_{k=1}^{s-1}(I - \mu\Sigma_{XX})^{k-1}$$

Since $\mathbb{E}(\boldsymbol{w}^0) = 0$

$$\mathbb{E}(\boldsymbol{w}^s) = \mathbb{E}(\boldsymbol{w}^0)(I - \mu\Sigma_{XX})^{s-1} + \mu\Sigma_{YX}\sum_{k=1}^{s-1}(I - \mu\Sigma_{XX})^{k-1} = \mu\Sigma_{YX}\sum_{k=1}^{s-1}(I - \mu\Sigma_{XX})^{k-1}$$

Given the lemma's assumptions, this expression can be evaluated and simplified:

$$\begin{aligned}\mathbb{E}(\boldsymbol{w}^s) &= \mu\Sigma_{YX}[I - (I - \mu\Sigma_{XX})]^{-1}[I - (I - \mu\Sigma_{XX})^{s-1}] \\ &= \Sigma_{YX}\Sigma_{XX}^{-1} - \Sigma_{YX}\Sigma_{XX}^{-1}(I - \mu\Sigma_{XX})^{s-1} \\ &\approx \Sigma_{YX}\Sigma_{XX}^{-1}\end{aligned} \tag{2.9}$$

■

From (2.6) it follows that a decrease in inter-model agreement at step s, which is implied by increased test variance among models, is indicated by the following inequality:

$$\begin{aligned}\mathbb{C} &= DisAg(s) - DisAg(s-1) \\ &= \frac{1}{2Q^2}\sum_{i,j=1}^{Q}\|\tilde{\boldsymbol{e}}(i,t) - \tilde{\boldsymbol{e}}(j,t)\|_F^2 - \frac{1}{2Q^2}\sum_{i,j=1}^{Q}\|\boldsymbol{e}(i,t) - \boldsymbol{e}(j,t)\|_F^2 \;>\; 0\end{aligned} \tag{2.10}$$

Theorem. *Assume that all models see the same training set, denoted as $\mathbb{X}(i) = X \;\forall i \in [Q]$, and that the training data covariance matrix Σ_{XX} is full rank.*

We make the following asymptotic assumptions, which are loosely phrased but can be rigorously defined with additional notations:

1. *The learning rate μ is small enough so that $\|I-\mu\Sigma_{XX}\|<1$ (from Lemma 2), and additionally we can neglect terms that are $O(\mu^2)$.*

2. *The number of gradient steps s is large enough so that $\|I-\mu\Sigma_{XX}\|^s$ can be neglected.*

3. *The number of models Q is large enough so that using the law of large numbers, we get $\frac{1}{Q}\sum_{i=1}^{Q}\boldsymbol{w}(i)\approx\mathbb{E}[\boldsymbol{w}]$.*

Finally, we assume that overfit occurs at time s in all the models of the ensemble. In other words, at time s the generalization error does not decrease in all the models.

When these assumptions hold, the agreement between the models decreases.

Proof. (2.10) can be rearranged as follows

$$\begin{aligned}\mathbb{C}&=\frac{1}{2Q^2}\sum_{i,j=1}^{Q}\|[\boldsymbol{e}(i,t)-\mu\Delta(i,i)\mathbb{X}(t)]-[\boldsymbol{e}(j,t)-\mu\Delta(j,j)\mathbb{X}(t)]\|_F^2-\frac{1}{2Q^2}\sum_{i,j=1}^{Q}\|\boldsymbol{e}(i,t)-\boldsymbol{e}(j,t)\|_F^2\\&=\frac{1}{Q^2}\sum_{i,j=1}^{Q}-\mu[\boldsymbol{e}(i,t)-\boldsymbol{e}(j,t)]\cdot[\Delta(i,i)\mathbb{X}(t)]-\Delta(j,j)\mathbb{X}(t)]+O(\mu^2)\\&=\frac{\mu}{Q^2}\sum_{i,j=1}^{Q}[\Delta(i,i)\cdot\Delta(j,t)+\Delta(j,j)\cdot\Delta(i,t)]-[\Delta(i,i)\cdot\Delta(i,t)+\Delta(j,j)\cdot\Delta(j,t)]+O(\mu^2)\end{aligned}$$

where the last transition follows from $\boldsymbol{e}(i,t)\mathbb{X}(t)^\top=\Delta(i,t)$. Using assumption 2

$$\mathbb{C}=\mu(\mathbb{C}'-\mathbb{C}'')+O(\mu^2)\approx\mu(\mathbb{C}'-\mathbb{C}'') \tag{2.11}$$

where

$$\mathbb{C}''=\frac{1}{Q^2}\sum_{i,j=1}^{Q}[\Delta(i,i)\cdot\Delta(i,t)+\Delta(j,j)\cdot\Delta(j,t)]=\frac{2}{Q}\sum_{i=1}^{Q}\Delta(i,i)\cdot\Delta(i,t) \tag{2.12}$$

and

$$\begin{aligned}\mathbb{C}'&=\frac{1}{Q^2}\sum_{i,j=1}^{Q}[\Delta(i,i)\cdot\Delta(j,t)+\Delta(j,j)\cdot\Delta(i,t)]\\&=\frac{1}{Q}\sum_{i=1}^{Q}\Delta(i,i)\cdot\frac{1}{Q}\sum_{j=1}^{Q}\Delta(j,t)+\frac{1}{Q}\sum_{j=1}^{Q}\Delta(j,j)\cdot\frac{1}{Q}\sum_{i=1}^{Q}\Delta(i,t)\\&=\frac{1}{Q}\sum_{i=1}^{Q}\Delta(i,i)\cdot\frac{2}{Q}\sum_{j=1}^{Q}\Delta(j,t)\end{aligned} \tag{2.13}$$

Next, we prove that $\mathbb{C}'$ is approximately 0. We first deduce from assumptions 1 and 4 that

$$\frac{1}{Q}\sum_{i=1}^{Q}\Delta(i,i)=\frac{1}{Q}\sum_{i=1}^{Q}\boldsymbol{w}(i)\Sigma_{XX}(i)-\Sigma_{YX}(i)=\left(\frac{1}{Q}\sum_{i=1}^{Q}\boldsymbol{w}(i)\right)\Sigma_{XX}-\Sigma_{YX}\approx\mathbb{E}[\boldsymbol{w}]\Sigma_{XX}-\Sigma_{YX}$$

From assumption 3 and Lemma 3, we have that $\mathbb{E}[\boldsymbol{w}] \approx \Sigma_{YX}\Sigma_{XX}^{-1}$. Thus

$$\frac{1}{Q}\sum_{i=1}^{Q}\Delta(i,i) \approx \mathbb{E}[\boldsymbol{w}]\Sigma_{XX} - \Sigma_{YX} \approx \Sigma_{YX}\Sigma_{XX}^{-1}\Sigma_{XX} - \Sigma_{YX} = 0$$

From this derivation and (2.13) we may conclude that $\mathbb{C}' \approx 0$. Thus

$$\mathbb{C} \approx -\mu\mathbb{C}'' = -\mu\frac{2}{Q}\sum_{i=1}^{Q}\Delta(i,i)\cdot\Delta(i,t) \tag{2.14}$$

If overfit occurs at time s in all the models of the ensemble, then $\mathbb{C} > 0$ from Lemma 1 and (2.14). From (2.10) we may conclude that the inter-model agreement decreases, which concludes the proof.

■

CHAPTER 3

DEALING WITH NOISY LABELS

In this section we analyze, both theoretically and empirically, how measures of inter-network agreement may indicate the detrimental phenomenon of 'overfit'. *Overfit* is a condition that can occur during the training of deep neural networks. It is characterized by the co-occurring decrease of *train error or loss* and the increase of *test error or loss*. Recall that train loss is the quantity that is being continuously minimized during the training of deep models, while the test error is the quantity linked to generalization error. When these quantities change in opposite directions, training harms the final performance and thus early stopping is recommended.

We begin by showing in Section 3.1 that in an ensemble of linear regression models, overfit and the agreement between models are negatively correlated. When this is the case, an epoch in which the agreement between networks reaches its maximal value is likely to indicate the beginning of overfit.

Our next goal is to examine the relevance of this result to deep learning in practice. Yet inexplicably, at least as far as image datasets are concerned, overfit rarely occurs in practice when deep learning is used for image recognition. However, when label noise is introduced, significant overfit occurs. Capitalizing on this observation, we report in Section 3.3 that when overfit occurs in the independent training of an ensemble of deep networks, the agreement between the networks starts to decrease.

The approach we describe in Section 4 is motivated by these results: Since it has been observed that noisy data are memorized later than clean data, we hypothesize that overfit occurs when the memorization of noisy labels becomes dominant. This suggests that measuring the dynamics of agreement between networks, which is correlated with overfit as shown below, can be effectively used for the identification of label noise.

3.1 Overfit and Agreement: Theoretical Result

Since deep learning models are not amenable to a rigorous theoretical analysis, and in order to gain computational insight into such general phenomena as overfit, simpler models are sometimes analyzed (e.g. Weinshall and Amir, 2020). Accordingly, in Sec. 2.4 we formally analyze the relation between overfit and inter-model agreement in an ensemble of linear regression models. In this framework, it can be shown that the two phenomena are negatively correlated, namely, increase in overfit implies decrease in inter-model agreement. Thus, we prove (under some assumptions) the following result:

Theorem 1. *Assume an ensemble of models obtained by solving linear regression with gradient descent and random initialization. If overfit* increases *at time t in all the models in the ensemble, then the agreement between the models in the ensemble at time t* decreases.

3.2 Measuring the Agreement between Models

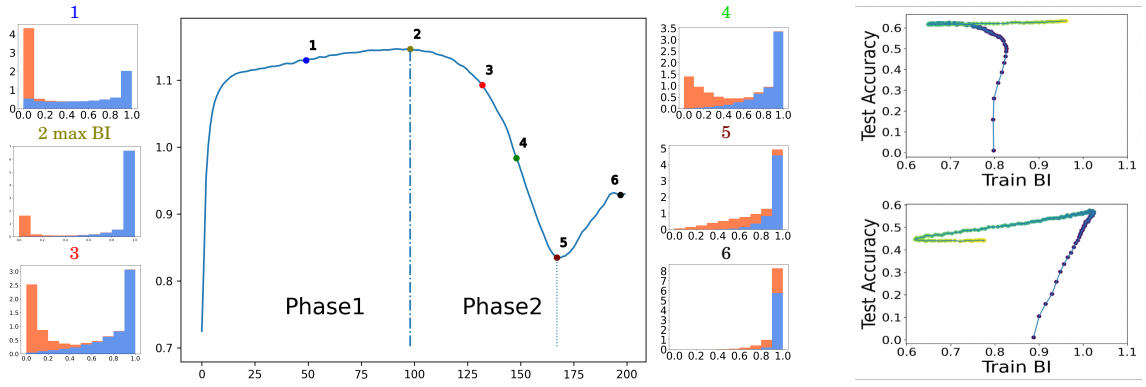
In order to obtain a score that captures the level of disagreement between networks, we inspect more closely the distribution of $TPA(\boldsymbol{x}, y; \mathscr{F}^e(\mathbb{X}))$, defined in Section 2.2, over a sample of datapoints, and analyze its dynamics as training proceeds. First, note that if all of the models in ensemble $\mathscr{F}^e(\mathbb{X})$ give identical predictions at each point, the TPA score would be either 0 (when all the networks predict a false label) or 1 (when all the networks predict the correct label). In this case, the TPA distribution is perfectly bimodal, with only two peaks at 0 and 1. If the predictions of the models at each point are independent with mean accuracy p, then it can be readily shown that TPA is approximately the binomial random variable with a unimodal distribution around p.

Empirically, (Hacohen et al., 2020) showed that in ensembles of deep models trained on "real" datasets as we use here, the TPA distribution is highly bimodal. Since commonly used measures of bimodality, such as the Pearson bimodality score, are ill-fitted for the discrete TPA distribution, we measure bimodality with the following *Bimodal Index* score:

$$BI(e) = \sqrt{\frac{1}{M}\sum_{i=1}^{M} \mathbb{1}_{[TPA(\boldsymbol{x}_i, y_i; \mathscr{F}^e(\mathbb{X}))=N]}} + \sqrt{\frac{1}{M}\sum_{i=1}^{M} \mathbb{1}_{[TPA(\boldsymbol{x}_i, y_i; \mathscr{F}^e(\mathbb{X}))=0]}} \tag{3.1}$$

$BI(e)$ measures how many examples are either correctly or incorrectly classified by *all* the models in the ensemble, rewarding distributions where points are (roughly) equally divided between 0 and 1. Here we use this score to measure the agreement between networks at epoch e.

If we were to draw the *Bimodality Index (BI)* of the TPA score as a function of the epochs (Fig. 3.1(a)), we often see two distinct phases. Initially (phase 1), BI is monotonically increasing, namely, both test accuracy and agreement are on the rise. We call it the "learning" phase. Empirically, in this phase most of the clean examples are being learned (or memorized), as can also be seen in the left side panels of Fig. 3.1(a) (cf. Li et al., 2015). At some point BI may



((a)) BI (Y-axis) vs. epochs (X-axis)

((b))

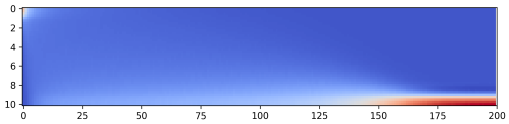
Figure 3.1: (a) Main panel: bimodality in an ensemble of 10 DenseNet networks, trained to classify Cifar10 with 20% symmetric noise. Side panels: TPA distribution in 6 epochs (blue - clean examples, orange - noisy ones). (b) Scatter plots of test accuracy vs train bimodality, measured by $BI(e)$ as defined in (3.1), where changes in color from blue to yellow correspond with advancing epochs.

start to decrease, followed by another possible ascent. This is phase 2, in which empirically the memorization of noisy examples dominates the learning (see the right side panels of Fig. 3.1(a)). This fall and rise is explained by another set of empirical observations, that noisy labels are **not** being learned in the same order by an ensemble of networks, which therefore predicts a decline in BI when noisy labels are being learned. To see this, we measure the distance between the TPA distribution, computed separately for clean examples and for noisy examples, and the binomial distribution, which is the expected distribution of iid classifiers with the same overall accuracy. Specifically, we compute the Wasserstein distance between the agreement distribution at each epoch and the binomials BIN (k,p_{clean}) and BIN(k,p_{noisy}), where p_{clean} is the average accuracy on the clean examples, and p_{noisy} is the average accuracy on the noisy examples, see Fig. 3.2. We see that while the distribution of model agreement on clean examples is very far from the binomial distribution, the distribution of model agreement on noisy examples is much closer.

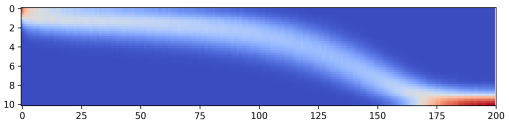
3.3 Overfit and Agreement: Empirical Evidence

Earlier work, investigating the dynamics of learning in deep networks, suggests that examples with noisy labels are learned later (Krueger et al., 2017; Zhang et al., 2017; Arpit et al., 2017; Arora et al., 2019). Since the learning of noisy labels is unlikely to improve the model's test accuracy, we hypothesize that this may be correlated with the occurrence (or increase) of *overfit*. The theoretical result in Section 3.1 suggests that this may be correlated with a decrease in the agreement between networks. Our goal now is to test this prediction empirically.

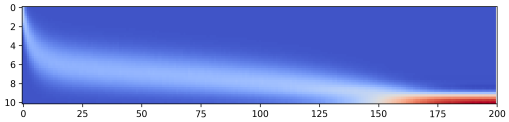
We next outline empirical evidence that this is indeed the case in actual deep models. In order to boost the strength of overfit, we adopt the scenario of recognition with label noise, where the occurrence of overfit is abundant. When overfit indeed occurs, our experiments show that if the



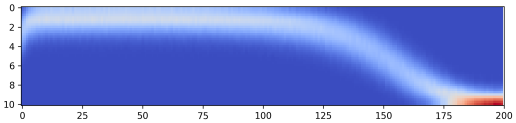
((a)) Empirical distribution - Clean examples



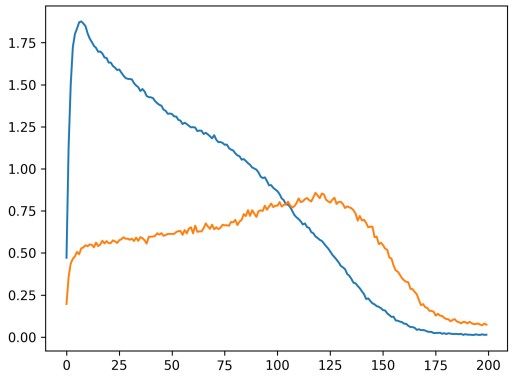
((b)) Empirical distribution - Noisy examples



((c)) Binomial distribution - Clean examples



((d)) Binomial distribution - Noisy examples



((e))

Figure 3.2: (a)-(d): The empirical distribution of the agreement values over epochs (X-axis: epochs, Y-axis: agreement, color code: blue for low and red for high). Clearly, the distribution of noisy examples resembles the binomial distribution with matched expected value, while the clean examples distribution is far from binomial. (e) Wasserstein distance between the binomial distribution and the empirical agreement distribution over epochs.

test accuracy drops, then the disagreement score BI also decreases (see example in Fig. 3.1(b)-bottom). This observation is confirmed with various noise models and different datasets. When overfit does not occur, the prediction is no longer observed (see example in Fig. 3.1(b)-top).

These results suggest that a consistent drop in the BI index of some training set $\mathbb{X}$ can be used to estimate the occurrence of overfit, and possibly even the beginning of noisy label memorization.

CHAPTER 4

DEALING WITH NOISY LABELS: PROPOSED APPROACH

When dealing with noisy labels, there are essentially three intertwined problems that may require separate treatment:

1. **Noise level estimation**: estimate the number of noisy examples.
2. **Noise filtration**: flag points whose label is to be removed.
3. **Classifier construction**: train a model without the examples that are flagged as noisy.

4.1 DisagreeNet

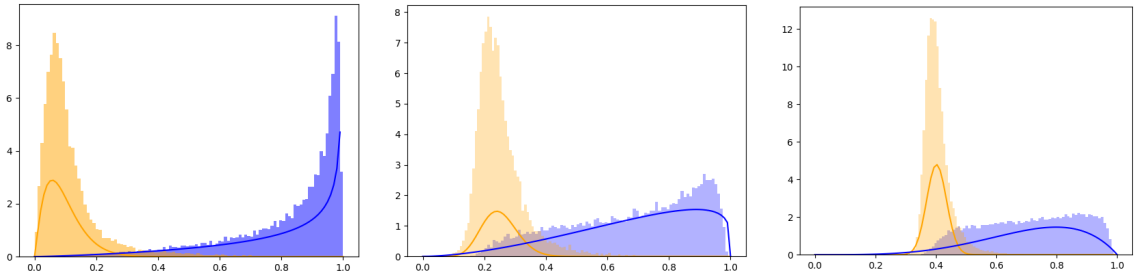
Guided by Section 3, we propose a method to estimate the noise level in a training set denoted *DisagreeNet*, which is further used to filter out the noisy examples (see pseudo-code below in Alg. 1 and Alg. 2):

1. Compute the ELP score from (2.1) at each training example (Alg. 2)
2. Fit a two component BMM to the ELP distribution (see Fig. 4.1).
3. Use the intersection between the 2 components of the BMM fit to divide the data to two groups.
4. Call the group with lower ELP 'noisy data'.
5. Estimate noise level by counting the number of datapoints in the noisy group.

As part of our ablation study, we evaluated the two alternative scores defined in Section 2.3: CumLoss and MeanMargin, where Step 2 of *DisagreeNet* is executed using one of them instead of the ELP score. Results are shown in Section 5.4, revealing the superiority of the ELP score in noise filtration.

Algorithm 1: *DisagreeNet*

Input: ELP_arr, specifying the ELP score of each point in training set $\mathbb{X}$
Output: Noise level estimate, and the list of indices of noisy points
$\{G_{low-ELP}, G_{high-ELP}\} \leftarrow$ divide the data to two groups using **fit_BMM**(ELP_arr);
noise_indices $\leftarrow$ indices of ELP_arr assigned to $G_{low-ELP}$;
noise_estim $\leftarrow \frac{|G_{low-ELP}|}{|\text{ELP\_arr}|}$;
return noise_estim, noise_indices



Cifar10, 40% Symm. noise | Cifar100, 20% Asymm. noise | TinyImagenet, 40% Symm. noise

Figure 4.1: ELP distribution, shown separately for the clean data in blue and the noisy data in orange. Superimposed, in blue and orange lines, is the bi-modal BMM fit to the ELP total (not separated) distribution

4.2 Classifier construction

Aiming to achieve modular handling of noisy labels, we propose the following two-step approach:

1. Run *DisagreeNet*.
2. Run SOTA supervised learning method using the filtered data.

In step 2 it is possible to invoke semi-supervised SOTA methods, using the noisy group as unsupervised data. However, given that semi-supervised learning typically involves additional assumptions (or prior knowledge) as well as high computational complexity (that restricts its applicability to smaller datasets), as discussed in Section 1, we do not consider this scenario here.

Algorithm 2: Computing the ELP score

Input: Training dataset $\mathbb{X}$ with N examples, potentially noisy, network architecture $\mathscr{A}$, batch size b, learning rate η, number of networks K, number of epochs E

Output: Array, containing the ELP score for each data point

compute agreement during training;

initialization $\theta_1^0 ... \theta_K^0$ different initialization of $\mathscr{A}$;

Initialize $agreement\_arr$[E,N]$\leftarrow 0$;

for $e = 0; E$ **do**

  for $k = 0; K$ **do**

    sample $indices_b$ with size b from [1...N] uniformly;

    $x_b, y_b \leftarrow \mathscr{X}[indices_b], \mathscr{Y}[indices_b]$; /* Gets a mini batch */

    compute p_b on x_b using θ_k^e;

    compute loss l_b with respect to p_b and y_b;

    $\theta_k^{e+1} \leftarrow SGD(\theta_k^e; l_b)$;

    $agreement\_arr[e, indice_b]$ += (**argmax**$(p_b) == y_b$); /* Store whether the network k predicted correctly on the examples at epoch e */

  end

end

$agreement\_arr \leftarrow agreement\_arr/(E \cdot K)$; /* normaliztion */

$ELP\_arr \leftarrow mean\_over\_epochs(agreement\_arr)$; /* mean over X-axis */

return ELP_arr

CHAPTER 5

EMPIRICAL EVALUATION

We evaluate our method in the following scenarios and tasks:

1. Noise identification (Section 5.2), with two complementary sub-tasks: (i) estimate the noise level in the given dataset; (ii) identify the noisy examples.
2. Supervised classification (Section 5.3), after the removal of the noisy examples.

5.1 Dataset and Baselines

Datasets We evaluated our method on a few standard image classification datasets, including Cifar10 and Cifar100 (Krizhevsky et al., 2009) and Tiny imagenet (Le and Yang, 2015). Cifar10/100 consist of 60k 32×32 color images of 10 and 100 classes respectivaly. Tiny ImageNet consists of 100,000 images from 200 classes of ImageNet (Deng et al., 2009), downsampled to size 64×64. Animal10N dataset contains 5 pairs of confusing animals with a total of 55,000 64x64 images. Clothing1M (Xiao et al., 2015) contains 1M clothing images in 14 classes. These datasets were used in earlier work to evaluate the success of noise estimation (Pleiss et al., 2020; Arazo et al., 2019; Li et al., 2020; Liu et al., 2020).

Baseline methods for comparison We evaluate our method in the context of two approaches designed to deal with label noise: methods that focus on improving the supervised learning by identifying noisy labels and removing/reducing their influence on the training, and methods that use iterative methods and utilize semi-supervised algorithms in order to learn with noisy labels.

First approach: ◇ ***DY-BMM*** and ***DY-GMM*** (Arazo et al., 2019) estimate mixture models on the loss to separate noisy and clean examples. ◇ ***INCV*** (Chen et al., 2019) iteratively filter out noisy examples by using cross-validation. ◇ ***AUM*** (Pleiss et al., 2020) inserts corrupted examples

to determine a filtration threshold, using the mean margin as a score. ⋄ ***Bootstrap*** (Reed et al., 2014) interpolates between the net predictions and the given label. ⋄ ***D2L*** (Ma et al., 2018) follows *Bootstrap*, and uses the examples dimensional attributes for the interpolation. ⋄ ***Co-teaching*** (Han et al., 2018) use two networks to filter clean data for the other net training. ⋄ ***O2U*** (Huang et al., 2019b) varies the learning rate to identify the noisy samples, based on a loss-based metric. ⋄ ***MentorNet*** (Jiang et al., 2018) trains a mentor network, whose outputs are used as a curriculum to the student network. ⋄ ***LEC*** (Lee and Chung, 2019) trains multiple networks, and uses the intersection of their small loss examples (using a given noise rate as a threshold) to construct a filtered dataset for the next epoch.

Second approach: ⋄ ***SELF*** (Nguyen et al., 2019) iteratively uses an exponential moving average of a net prediction over the epochs, compared to the ground truth labels, to filter noisy labels and retrain . ⋄ ***Meta learning*** (Li et al., 2019) uses a gradient based technique to update the networks weights with noise tolerance. ⋄ ***DivideMix*** (Li et al., 2020) uses 2 networks to flag examples as noisy and clean with two component mixture, after which the SSL technique MixMatch (Berthelot et al., 2019) is used. ⋄ ***ELR*** (Liu et al., 2020) identifies early learned example, and uses them to regulate the learning process. ⋄ ***C2D*** (Zheltonozhskii et al., 2022) uses the same algorithm as ELR and Dividemix, and uses a pretrain net with unsupervised loss.

We also report the results of two absolute baselines: (i) ***Oracle***, which trains the model on the clean dataset; (ii) ***Random***, which trains the model after the removal of a random fraction of the whole data, equivalent to the noise level.

Other methods The following methods use additional prior information, such as a clean validation set or known level of noise: ***Co-teaching*** (Han et al., 2018), ***O2U*** (Huang et al., 2019b), ***LEC*** (Lee and Chung, 2019) and ***SELFIE*** (Song et al., 2019). Direct comparison does injustice to the previous group of methods, and so the comparison is done separately, in section 5.5. Another group of methods is excluded from the comparison because they invoke semi-supervised or contrastive learning (e.g., Ortego et al., 2020; Li et al., 2020; Karim et al., 2022; Li et al., 2022; Wei et al., 2020; Yao et al., 2021), which is a different learning paradigm (see discussion of prior art in Section 1).

Implementation details We used DenseNet (Iandola et al., 2014), ResNet-18 and ResNet50 (He et al., 2016) when training on CIFAR-10/100 and Tiny imagenet , and ResNet50 for Clothing1M and Animal10N.

Technical Details Unless stated otherwise, we used an SGD optimizer with 0.9 momentum and a learning rate of 0.01, weight decay of 5e-4, and batch size of 32. We used a Cosine-annealing scheduler in all of our experiments and used standard augmentation (horizontal flips, random crops) during training. We inspected the effect of different hyperparameters in the ablation study.

All of our experiments were conducted on the internal cluster of the Hebrew University, on GPU type AmpereA10.

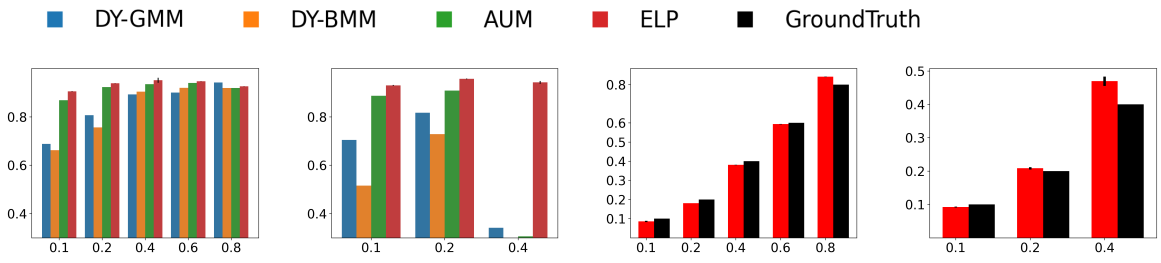
5.2 Results: Noise Identification

The performance of *DisagreeNet* is evaluated in two tasks: (i) The detection of noisy examples, shown in Fig. 5.1(a)-5.1(b) (see also Figs. A.1 and A.2 in App. A), where *DisagreeNet* is seen to outperform the three baselines - *AUM*, *DY-GMM* and *DY-BMM*. (ii) Noise level estimation, shown in Fig. 5.1(c)-5.1(d), showing good noise level estimation especially in the case of symmetric noise. We also compare *DisagreeNet* to MeanMargin and CumLoss, see Fig. 5.2.

5.3 Result: Supervised Classifications

DisagreeNet is used to remove noisy examples, after which we train a deep model from scratch using the remaining examples only. We report our main results using the Densenet architecture, and report results with other architectures in the ablation study. Table 5.1 summarizes the results for simulated symmetric and asymmetric noise on 5 datasets, and 3 repetitions. It also shows results on 2 real datasets, which are assumed (in previous work) to contain significant levels of 'real' label noise. Additional results are reported in Sec. 5.5, including methods that require additional prior knowledge.

Not surprisingly, dealing with datasets that are presumed to include inherent label noise proved more difficult, and quite different, than dealing with synthetic noise. As claimed in (Ortego et al., 2020), non-malicious label noise does less damage to networks' generalization than random label noise: on Clothing1M, for example, hardly any overfit is seen during training, even though the data is believed to contain more than 35% noise. Still, here too, *DisagreeNet* achieves improved accuracy without access to a clean validation set or known noise level (see Table 5.1). In Sec. 5.5, Table 5.5 we compare *Disagreenet* to methods that *do* use such prior



((a)) Cifar100 sym ((b)) Cifar100 asym ((c)) Cifar100 sym ((d)) Cifar100 asym

Figure 5.1: (a)-(b) **Noise identification**: F1 score for noisy label identification task, using different noise levels (X-axis), with asymmetric (a) and asymmetric (b) noise models. Results reflect 3 repetitions involving an ensemble of 10 Densenets each. (c)-(d) **Noise level estimation**: different noise levels are evaluated (X-axis), with asymmetric (c) and asymmetric (d) noise models (the 3 comparison baselines did not report this estimate).

Table 5.1: Test accuracy (%), average and standard error, in the best epoch of retraining after filtration. Results of benchmark methods (see Section 5.1) are taken from (Pleiss et al., 2020). The top and middle tables show CIFAR-10, CIFAR-100 and Tiny Imagenet, with simulated noise. The bottom table shows three 'real noise' datasets, and includes in addition results of noise level estimation (when applicable). The presumed noise level for these datasets is indicated in the top line following (Huang et al., 2019a; Song et al., 2019).

Method/**Dataset**	**CIFAR-10 sym**			**CIFAR-100 sym**		
Noise level	20%	40%	60%	20%	40%	60%
random	87.18 ± 0.6	81.59 ± 0.4	64.35 ± 0.4	65.49 ± 0.4	49.1 ± 0.2	28.7 ± 0.5
Bootstrap	77.6 ± 0.2	62.6 ± 0.4	48.0 ± 0.2	51.4 ± 0.2	41.1 ± 0.2	29.7 ± 0.2
MentorNet	86.7 ± 0.1	81.9 ± 0.2	–	64.2 ± 0.3	57.5 ± 0.2	–
D2L	87.7 ± 0.2	84.4 ± 0.3	72.7 ± 0.6	54.0 ± 1.0	29.7 ± 1.8	–
INCV	89.5 ± 0.1	86.8 ± 0.1	81.1 ± 0.3	58.6 ± 0.5	55.4 ± 0.2	43.7 ± 0.3
AUM	90.2 ± 0.0	87.5 ± 0.1	82.1 ± 0.0	65.5 ± 0.2	61.3 ± 0.1	53.0 ± 0.5
DisagreeNet+SL	$\mathbf{93.1 \pm 0.2}$	$\mathbf{91.1 \pm 0.1}$	$\mathbf{83.9 \pm 0.08}$	$\mathbf{77.3 \pm 0.2}$	$\mathbf{71.8 \pm 0.3}$	$\mathbf{64.7 \pm 0.3}$
oracle	95.1 ± 0.2	94.1 ± 0.2	92.4 ± 0.1	78.2 ± 0.3	75.4 ± 0.1	70.3 ± 0.2

Method/**Dataset**	**CIFAR-10 constant asym**		**CIFAR-100 constant asym**		**Tiny Imagenet sym**	
Noise level	20%	40%	20%	40%	20%	40%
random	89.5 ± 0.2	79.3 ± 0.4	65.2 ± 0.1	44.64 ± 0.2	49.8 ± 0.4	29.9 ± 0.3
Bootstrap	76.2 ± 0.2	55.0 ± 0.6	53.4 ± 0.3	38.7 ± 0.3	-	-
D2L	88.6 ± 0.2	76.4 ± 1.5	43.6 ± 0.7	16.9 ± 1.2	-	-
DY-BMM	77.9 ± 0.1	59.4 ± 0.6	53.2 ± 0.0	37.9 ± 0.0	41.8 ± 0.1	36.3 ± 0.2
INCV	88.3 ± 0.1	79.8 ± 0.4	56.8 ± 0.1	44.4 ± 0.7	45.2 ± 0.1	42.6 ± 0.1
AUM	89.7 ± 0.1	58.7 ± 0.2	59.7 ± 0.2	40.2 ± 0.1	48.9 ± 0.2	44.7 ± 0.1
DisagreeNet+SL	$\mathbf{94.4 \pm 0.1}$	$\mathbf{91.9 \pm 0.0}$	$\mathbf{73.9 \pm 0.5}$	$\mathbf{61.3 \pm 0.2}$	$\mathbf{64.5 \pm 0.1}$	$\mathbf{58.5 \pm 0.2}$
oracle	95.2 ± 0.0	94.3 ± 0.0	78.1 ± 0.1	75 ± 0.1	65.4 ± 0.0	60.8 ± 0.2

Method/**Dataset**	**animal10N, 8% noise**		**Clothing1M, 38% noise**	
Noise level	noise est	test accuracy	noise est	test accuracy
Cross-Entropy	-	84.1 ± 0.3	-	69
AUM	-	-	10.7	70.4
DisagreeNet+SL	7.8	85.1 ± 0.1	17	70.8

knowledge. Surprisingly, we see that *DisagreeNet* still achieves better results even without using any additional prior knowledge.

5.4 Ablation Study

How many networks are needed? We report in Table. 5.2 the F1 score for noisy label identification, using *DisagreeNet* with varying numbers of networks. The main boost in performance provided by the use of additional networks is seen when using *DisagreeNet* on hard noise scenarios, such as the asymmetric noise, or with small amounts of noise.

Additional ablation results Results in Table. 5.3, Table 5.4 indicate robustness to architecture, scheduler, and usage of augmentation, although the standard training procedures achieve the best results. Additionally, we see robustness to changing the backbone architecture

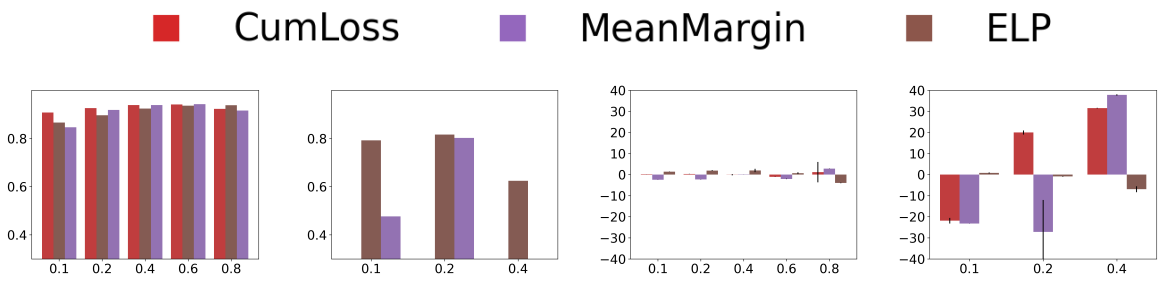
Table 5.2: F1 score of DisagreeNet, using different numbers of models.

Dataset	**Noise**	**size of ensemble (number of networks)**					
Method		1	2	3	4	7	10
Cifar10 sym	10%	0.605 ± 0.01	0.77 ± 0.0	0.862 ± 0.0	0.906 ± 0.0	0.941 ± 0.0	0.936 ± 0.0
	20%	0.861 ± 0.0	0.939 ± 0.0	0.95 ± 0.0	0.949 ± 0.0	0.943 ± 0.0	0.941 ± 0.0
	40%	0.954 ± 0.0	0.953 ± 0.0	0.952 ± 0.0	0.952 ± 0.0	0.951 ± 0.0	0.951 ± 0.0
Cifar100 sym	10%	0.225 ± 0.05	0.855 ± 0.0	0.855 ± 0.01	0.854 ± 0.0	0.860 ± 0.01	0.864 ± 0.01
	20%	0.89 ± 0.0	0.895 ± 0.0	0.896 ± 0.0	0.897 ± 0.0	0.901 ± 0.0	0.899 ± 0.0
	40%	0.89 ± 0.0	0.917 ± 0.0	0.921 ± 0.0	0.924 ± 0.0	0.924 ± 0.0	0.927 ± 0.0
Cifar10 asym	10%	0.355 ± 0.0	0.469 ± 0.01	0.568 ± 0.01	0.631 ± 0.01	0.748 ± 0.0	0.814 ± 0.0
	20%	0.553 ± 0.0	0.642 ± 0.01	0.703 ± 0.01	0.734 ± 0.0	0.799 ± 0.01	0.829 ± 0.01
	40%	0.739 ± 0.0	0.795 ± 0.0	0.816 ± 0.0	0.826 ± 0.0	0.824 ± 0.0	0.812 ± 0.0
Cifar100 asym	10%	0.703 ± 0.0	0.708 ± 0.0	0.712 ± 0.0	0.716 ± 0.0	0.718 ± 0.0	0.717 ± 0.0
	20%	0.727 ± 0.0	0.732 ± 0.0	0.732 ± 0.0	0.735 ± 0.0	0.736 ± 0.0	0.737 ± 0.0
	40%	0.594 ± 0.0	0.606 ± 0.0	0.614 ± 0.0	0.614 ± 0.0	0.618 ± 0.0	0.62 ± 0.0

of *DisagreeNet*, using ResNet18 and ResNet50, see Table 5.3. Finally, in Fig. 5.2 we compare *DisagreeNet* using ELP to disagreeNet using the MeanMargin and CumLoss scores, as defined in Section 2.3. In symmetric noise scenarios all scores perform well, while in asymmetric noise scenarios the ELP score performs much better, as can be seen in Figs. 5.2(b),5.2(d). Additional analysis of the 3 scores are reported in Sec.5.6.

Table 5.3: Final accuracy results when changing the backbone architecture.

Method/**Dataset**	**CIFAR-10 sym**			**CIFAR-100 sym**		
Noise level	20%	40%	60%	20%	40%	60%
DisagreeNet+SL R18	93.3 ± 0.1	91.1 ± 0.6	87.6 ± 0.1	75.1 ± 0.1	71.5 ± 0.1	63.1 ± 0.4
DisagreeNet+SL R50	93.4 ± 0.1	91.0 ± 0.2	87.0 ± 0.1	75.7 ± 0.3	70.2 ± 1.2	61.0 ± 0.4



((a)) Cifar100 sym ((b)) Cifar100 asym ((c)) Cifar100 sym ((d)) Cifar100 asym

Figure 5.2: (a)-(b): F1 score (Y-axis) for the noisy label identification task, using different noise levels (X-axis), with asymmetric (a) and asymmetric (b) noise models. Results with 3 variants of *DisagreeNet* are shown, based on 3 scores: MeanMargin, ELP and CumLoss. (c)-(d): Error in noise level estimation (Y-axis) using different noise levels (X-axis), with asymmetric (c) and asymmetric (d) noise models. As can be seen, ELP very significantly outperforms the other 2 scores when handling asymmetric noise.

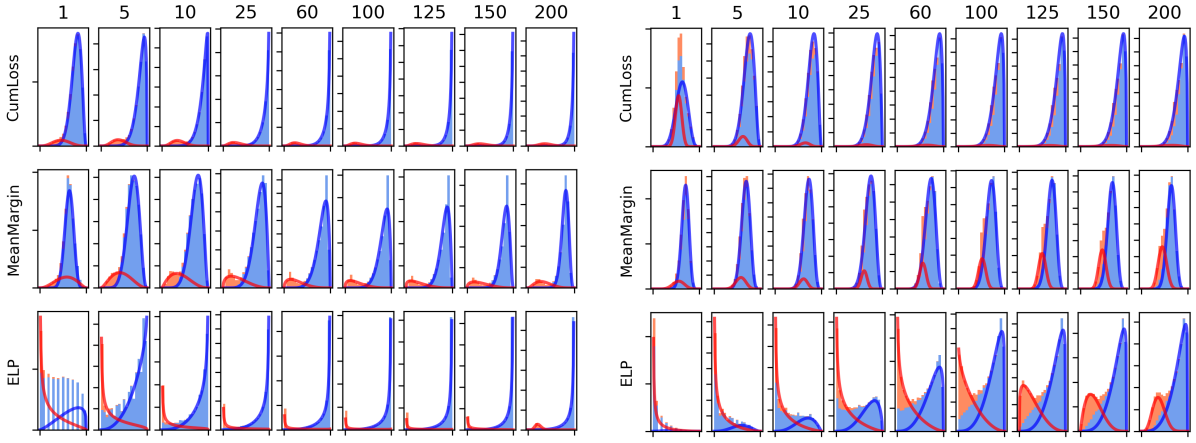
Table 5.4 summarize experiments relating to architecture, scheduler, and augmentation usage.

Alternative scores We evaluate the two alternative scores defined in Section 2.3: CumLoss and MeanMargin, in which case Step 2 of *DisagreeNet* is executed using one of them instead of

Table 5.4: F1 score for Cifar100 with 2 levels of symmetric noise. Different ablation conditions are marked in columns: *ResNet34* indicates a change of architecture, *no Aug* indicates that image augmentations are not used, and *lr 0.01* indicates that no scheduler or learning rate drop are used during training.

Noise level	No change	ResNet34	No Aug	Constant lr 0.01	Lr 0.01 + no Aug
20%	0.903 ± 0.01	0.839 ± 0.0	0.832 ± 0.0	0.859 ± 0.01	0.869 ± 0.01
40%	0.918 ± 0.01	0.887 ± 0.0	0.887 ± 0.01	0.877 ± 0.0	0.888 ± 0.01

the ELP score. Fig. 5.3 shows the Probability Distribution Function (PDF) of the three scores, revealing that ELP is more consistency bimodal (especially in the difficult asymmetric case), with modes (peaks) that appear more separable. This benefit translates to superior performance in the noise filtration task (Figs. 5.2(b),5.2(d)).



((a)) Cifar10 with 40% Symmetric noise

((b)) Cifar100 with 20% Asymmetric noise

Figure 5.3: Distribution of the CumLoss, MeanMargin and ELP scores during training. ELP remains bimodal even for hard noise models, where the other scores become unimodal.

We believe that this empirical observation, of increased mode separation, is due to significant difference in the pace of change in agreement values during training between clean and noisy data, in contrast with the pace of change in smoother measures of confidence like *Margin* and *Loss* (see Sec. 5.6). Note that with the easier symmetric noise, we do not see this difference, and indeed the other scores exhibit two nicely separated modes, sometimes achieving even better results in noise filtration than ELP (Fig. A.1 in App. A.1). However, when comparing the test accuracy after retraining (see App. A), we observe that ELP still achieves superior results.

5.5 Comparing to methods with different assumptions

Here we compare DisagreeNet to methods that assume known noise level - O2U (Huang et al., 2019b) and LEC (Lee and Chung, 2019), using a 9-layered CNN for the training (with standard hyper parameters as detailed in Sec. 5.1). Since the noise level is assumed known, we replace the estimation provided by DisagreeNet with the actual noise level. The results are summarized

in Table. 5.5. We also compare DisagreeNet to other methods that use prior knowledge, where DisagreeNet does not use prior knowledge. The results are summarized in Table. 5.6

Dataset	**Noise level**	**Method**		
Dataset - noise type	Noise level	O2U	LEC	DisagreeNet
Cifar10 sym	10%	87.64%	-	**92.57%**
	20%	85.24%	88.31%	**91.44%**
	40%	79.64%	-	**88.48%**
	60%	-	80.52%.	**81.81%**
Cifar100 sym	10%	62.32%	-	**69.86%**
	20%	60.53%	59.98%	**67.99%**
	40%	52.47%	-	**62.89%**
	60%	-	46.63%	**53.76%**
Cifar10 asym	10%	88.22%	-	**91.96%**
	20%	-	89.41%.	**90.94%**
	40%	-	86.50%	**86.93%**
Cifar100 asym	10%	64.50%	-	**69.83%**
	20%	-	58.86%.	**67.99%**
	40%	-	47.82%.	**62.89%**

Table 5.5: Test accuracy (%) comparison with methods that utilize prior knowledge with 9-layered CNN

Method/**Dataset**	**CIFAR-10 sym**			**CIFAR-100 sym**		
Noise level	20%	40%	60%	20%	40%	60%
Co-teaching	88.8 ± 0.1	86.5 ± 0.1	80.7 ± 0.1	64.1 ± 0.1	60.2 ± 0.2	48.0 ± 0.3
LEC	88.3	-	80.5	60	-	46.63
O2U	92.5	90.3	-	74.1	69.2	-
DisagreeNet+SL (no prior knowledge)	$\mathbf{93.1 \pm 0.2}$	$\mathbf{91.1 \pm 0.1}$	$\mathbf{83.9 \pm 0.08}$	$\mathbf{77.3 \pm 0.2}$	$\mathbf{71.8 \pm 0.3}$	$\mathbf{64.7 \pm 0.3}$

Method/**Dataset**	**CIFAR-10 asym**		**CIFAR-100 asym**		**Tiny Imagenet sym**	
Noise level	20%	40%	20%	40%	20%	40%
LEC	89.4	86.5	58.9	47.8	-	-
DisagreeNet+SL (no prior knowledge)	$\mathbf{94.4 \pm 0.1}$	$\mathbf{91.9 \pm 0.0}$	$\mathbf{73.9 \pm 0.5}$	$\mathbf{61.3 \pm 0.2}$	$\mathbf{62.5 \pm 0.2}$	$\mathbf{55.7 \pm 0.4}$

Method/**Dataset**	**animal10N, 8% noise**	
Noise level	noise est	test accuracy
Co-teaching	-	82.5 ± 0.1
SELFIE	-	83 ± 0.1
DisagreeNet+SL (no prior knowledge)	7.8	$\mathbf{85.1 \pm 0.1}$

Table 5.6: Test accuracy (%) comparison with methods that utilize prior knowledge of the real noise level.

5.6 Comparing agreement to confidence in noise filtration

While the learning time of an example has been shown to be effective for noise filtration, it fails to separate noisy and clean data that are learned more or less at the same time. To tackle this problem, one needs additional information, beyond the learning time of a single network. When

using an ensemble, we can use the TPA score, or else the average probability assigned to the ground truth label (denoted the "correct" logit) by the networks. The latter score conveys the model's confidence in the ground truth label, and is used by our two alternative scores - CumLoss and MeanMargin.

Going beyond learning time, we propose to look at "how quickly" the agreement value rises from 0 to 1, denoted as the "slope" of the agreement. Since our empirical results indicate that the learning time of noisy data is much more varied, we expect a slower rise in agreement over noisy data as compared to clean data. In our experiments, ELP achieved superior results in noise filtration. We hypothesize that the difference in slope between clean and noisy data may underlie the superiority of ELP in noise filtration.

To check this hypothesis, we compare between two scores computed at each data example: ELP and Logits Mean (denoted LM for simplicity). LM is defined as follows:

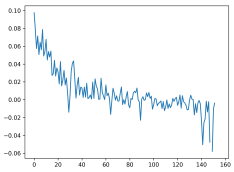
$$LM(x) = \frac{\sum_{i=1}^{k}\sum_{j=1}^{T}[p_{i,j}(x)]_y}{kT}$$

where k is the number of networks, T is the number of epochs during training, (x, y) is a data example and its assigned label, and $[p_{i,j}(x)]_y$ is the probability assigned by network i in epoch j to y (the ground truth label).

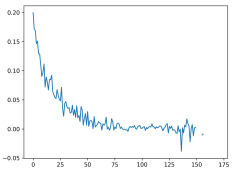
In order to compare between the pace of increase (slope) of ELP and LM, we conduct the following analysis: We select the two groups of clean and noisy data that are learned (roughly) at the same time by some net in the ensemble, and then compute the average agreement and "correct" logit functions as a function of epoch, separately for clean and noisy data. We then compute the difference per epoch between the noisy and clean average agreement, which we denote as $\Delta Agreement$ and $\Delta logit$. Note that $\Delta Agreement$ and $\Delta logit$ encode the difference in the slope between noisy and clean data, since they begin to rise at (roughly) the same time. Finally, we plot in Fig. 5.4 the difference between $\Delta Agreement$ and $\Delta logit$, recalling that larger Δ indicates stronger separation between the clean and noisy data.

Indeed, our analysis shows that with asymmetric noise, the difference between the agreement slope on clean and noisy data of the ELP score is consistently larger than the agreement slope difference between the average logits on clean and noisy data. This, we believe, is the key reason as to why ELP outperforms LM in noise filtration. Note that this effect is much less pronounced when using the easier symmetric noise, and indeed, our empirical results show that ELP does not outperform LM significantly in this case.

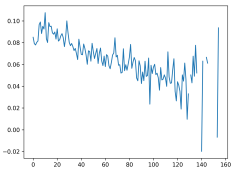
To conclude, we believe that the signal encoded by the agreement values is stronger than the signal encoded in measures of confidence in the networks' prediction when true labels are concerned, which explains its capability to classify correctly even some hard-clean examples and easy-noisy examples as clean and noise (respectively). This, we believe, is a result of the polarization effect caused by the binary indicators inside TPA, which disregard misleading positive probabilities assigned to noisy labels even before they are learned by the networks.



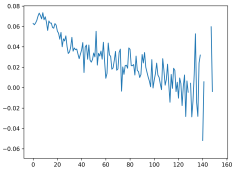
((a)) 20% symmetric noise



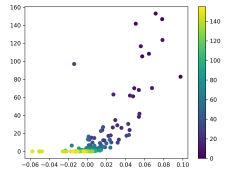
((b)) 40% symmetric noise



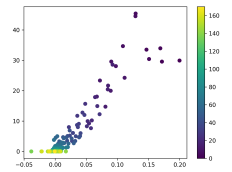
((c)) 20% Asymmetric noise



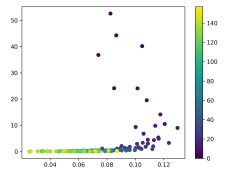
((d)) 40% Asymmetric noise



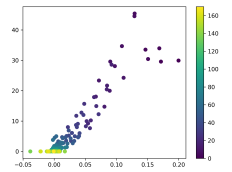
((e)) 20% symmetric noise



((f)) 40% symmetric noise



((g)) 20% Asymmetric noise



((h)) 40% Asymmetric noise

Figure 5.4: Top: X-axis is the learning time of the chosen clean and noisy data; Y-axis is the difference between $\Delta Agreement$ and $\Delta logit$. We see that for most of the training, the difference is positive, implying that ELP provides stronger separation between these groups. Bottom: X-axis is the difference between $\Delta Agreement$ and $\Delta logit$; Y-axis is the ratio between the amount of clean and noisy data. The color represents the learning time of the groups. These graphs show that while at the end of the training the difference between $\Delta Agreement$ and $\Delta logit$ is negative, implying that LM would be better at separating these groups, these are in fact very small sets of data, as most of the data is learned by some network at an earlier stage of the training

CHAPTER 6

DISCUSSION

We presented a new empirical observation, that the variability in the predictions of an ensemble of deep networks is much larger when labels are noisy, than it is when labels are clean. This observation is used as a basis for a new method for classification with noisy labels, addressing along the way the tasks of noise level estimation, noisy labels identification, and classifier construction. Our method is easy to implement, and can be readily incorporated into existing methods for deep learning with label noise, including semi-supervised methods, to improve the outcome of the methods.

Importantly, our method achieves this improvement without making additional assumptions, which are commonly made by alternative methods: (i) Noise level is expected to be unknown. (ii) There is no need for a clean validation set, which many other methods require, but which is very difficult to acquire when the training set is corrupted. (iii) Almost no additional hyperparameters are introduced.

6.1 Future work

6.1.1 ELP score

Working with the ELP score rises multiple questions on the process of learning in deep neural networks. Having established that the neural networks models have a global order of learning, we can ask ourselves what this order means, and what is its relation to other metrics of the hardness of the data. For example, we may wonder about the relation between sample selection and the ELP score, in the practical context of active learning or datasets distillation. Furthermore, we can relate the ELP to curriculum learning, as we try to understand what the optimal order is to present the data to the model, and whether the natural order is the best order. Another point of

interest is the connection between the classification task with given labels and other tasks, in the context of the learning order. Does this order is preserved with different training schemes, such as transfer learning? or by adding auxiliary tasks to the training? Can we define a meaningful metric on the unsupervised representation learning process, so the order of learning will be similar to the learning order of the supervised task? Is there an order of learning in unsupervised tasks at all? Those questions stem from the same origin of understanding the roots of the ELP score, and the fundamental connections between the data points and their labels. Other questions that are worth exploring are less practical but still interesting. What would happen if we removed the examples with the lowest score from the training process? Would the test set have the same ELP score or something fundamental will change, as the neural network learns in a local fashion, in each epoch? Does the big dataset behave as a 'regulator', or we can remove many data points that have 'repetitive' knowledge, and are not essential to the deep neural networks learning process? Invariants or preserved statistics in complex models are an interesting subject, and this course of exploration may shed new perspectives on the behavior of SGD and deep neural networks.

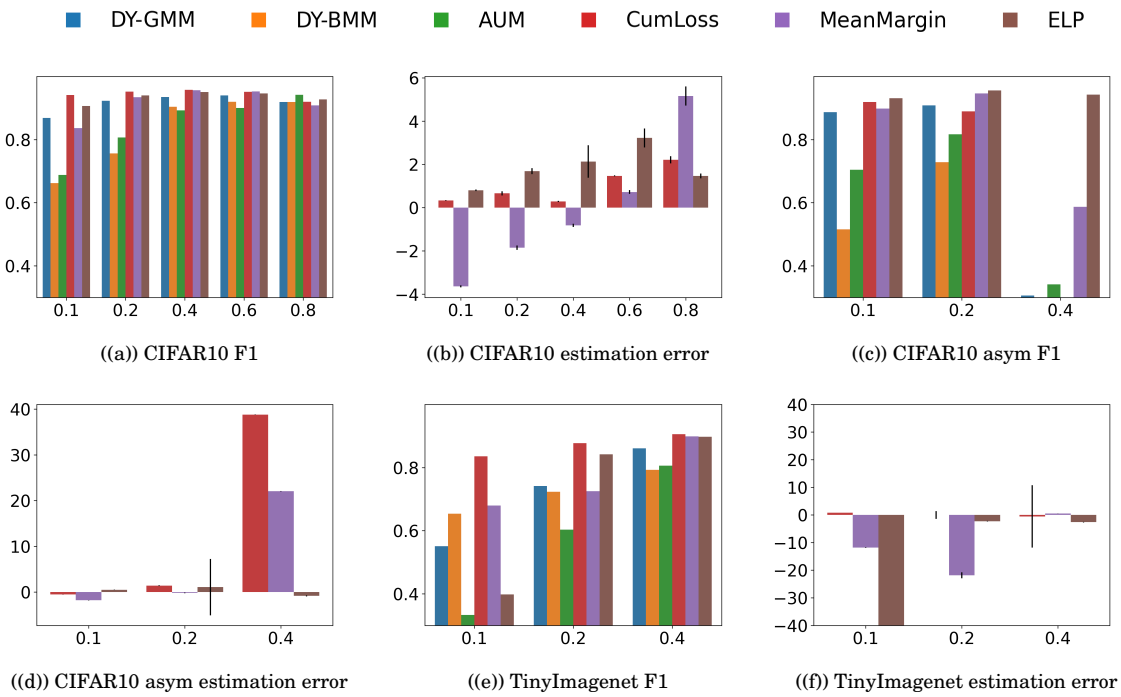
6.1.2 Broader view

In a broader sense, we can relate this work to one of the most interesting phenomenon in deep learning research today: low variance (Neal et al., 2018). As we mentioned in the introduction, when the depth and number of parameters of the neural networks grow, we traditionally expect the bias to descend and variance to ascent. There are some interpretations of this phenomenon, and it may be interesting to look at those interpretations in light of this work. To summarize the relevant part of the work, deep neural networks learn examples in the same order and pace, and noisy examples in different paces and order. This behavior on clean examples is an extension of the low variance (and even strengthens the phenomenon), but the low variance on the noisy examples still requires an explanation. Many works discussed the subject of low variance in deep neural networks, using the tools of statistical physics (Gerace et al., 2021), or statistics (Bartlett et al., 2021). Some of the explanation deal with terms like benign overfit (Bartlett et al., 2021), inductive bias, and implicit regularization (Neyshabur, 2017). These works attempt to analyze simpler models (shallow neural networks, linear regressors), so the extrapolation to more complex models can be problematic. We can ask which approach our results support, and future (and careful) examination of this question is still open and interesting.

APPENDIX A

ADDITIONAL RESULTS

A.1 Noise level estimation on additional datasets



((a)) CIFAR10 F1

((b)) CIFAR10 estimation error

((c)) CIFAR10 asym F1

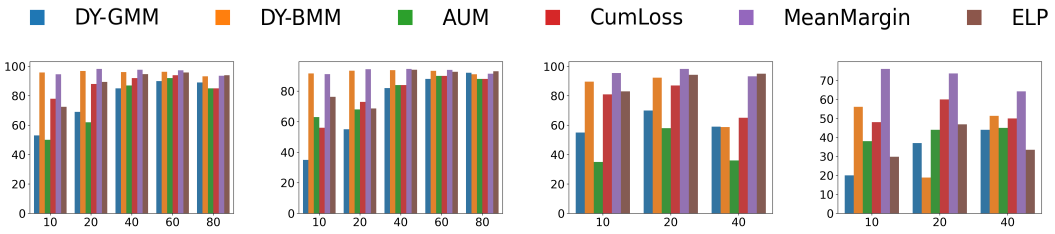
((d)) CIFAR10 asym estimation error

((e)) TinyImagenet F1

((f)) TinyImagenet estimation error

Figure A.1: Additional results on CIFAR10 and Tiny Imagenet.

A.2 Precision and Recall results



((a)) CIFAR10 ((b)) CIFAR100 ((c)) CIFAR10 asym ((d)) CIFAR100 asym

((e)) CIFAR10 ((f)) CIFAR100 ((g)) CIFAR10 asym ((h)) CIFAR100 asym

Figure A.2: Noisy label identification. Top: precision; bottom: recall.

BIBLIOGRAPHY

Eric Arazo, Diego Ortego, Paul Albert, Noel O,ÄôConnor, and Kevin McGuinness.
Unsupervised label noise modeling and loss correction.
In *International conference on machine learning*, pages 312–321. PMLR, 2019.

Sanjeev Arora, Simon Du, Wei Hu, Zhiyuan Li, and Ruosong Wang.
Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks.
In *International Conference on Machine Learning*, pages 322–332. PMLR, 2019.

Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al.
A closer look at memorization in deep networks.
In *International conference on machine learning*, pages 233–242. PMLR, 2017.

Peter L. Bartlett, Andrea Montanari, and Alexander Rakhlin.
Deep learning: a statistical viewpoint, 2021.
URL https://arxiv.org/abs/2103.09177.

David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel.
Mixmatch: A holistic approach to semi-supervised learning.
Advances in Neural Information Processing Systems, 32, 2019.

Pengfei Chen, Ben Ben Liao, Guangyong Chen, and Shengyu Zhang.
Understanding and utilizing deep neural networks trained with noisy labels.
In *International Conference on Machine Learning*, pages 1062–1070. PMLR, 2019.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei.
Imagenet: A large-scale hierarchical image database.
In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

Federica Gerace, Bruno Loureiro, Florent Krzakala, Marc Mé zard, and Lenka Zdeborová.
Generalisation error in learning with random features and the hidden manifold model.

Journal of Statistical Mechanics: Theory and Experiment, 2021(12):124013, dec 2021.
doi: 10.1088/1742-5468/ac3ae6.
URL https://doi.org/10.1088%2F1742-5468%2Fac3ae6.

Guy Hacohen, Leshem Choshen, and Daphna Weinshall.
Let‚Äôs agree to agree: Neural networks share classification order on real datasets.
In *Int. Conf. Machine Learning ICML*, pages 3950–3960, 2020.

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.
Advances in neural information processing systems, 31, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun.
Deep residual learning for image recognition.
In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Jinchi Huang, Lie Qu, Rongfei Jia, and Binqiang Zhao.
O2u-net: A simple noisy label detection approach for deep neural networks.
In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3326–3334, 2019a.

Jinchi Huang, Lie Qu, Rongfei Jia, and Binqiang Zhao.
O2u-net: A simple noisy label detection approach for deep neural networks.
In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3325–3333, 2019b.
doi: 10.1109/ICCV.2019.00342.

Forrest Iandola, Matt Moskewicz, Sergey Karayev, Ross Girshick, Trevor Darrell, and Kurt Keutzer.
Densenet: Implementing efficient convnet descriptor pyramids.
arXiv preprint arXiv:1404.1869, 2014.

Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei.
Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels.
In *International Conference on Machine Learning*, pages 2304–2313. PMLR, 2018.

Nazmul Karim, Mamshad Nayeem Rizve, Nazanin Rahnavard, Ajmal Mian, and Mubarak Shah.
Unicon: Combating label noise through uniform selection and contrastive learning, 2022.
URL https://arxiv.org/abs/2203.14542.

Alex Krizhevsky, Geoffrey Hinton, et al.
Learning multiple layers of features from tiny images.
Online, 2009.

David Krueger, Nicolas Ballas, Stanislaw Devansh Arpit, Maxinder S. Kanwal, Tegan Maharaj, Emmanuel Bengio, Asja Fischer, and Aaron C. Courville.
Deep nets don't learn via memorization.
In *Int. Conf. Learning Representations ICLR*, 2017.

Ya Le and Xuan Yang.
Tiny imagenet visual recognition challenge.
CS 231N, 7(7):3, 2015.

Jisoo Lee and Sae-Young Chung.
Robust training with ensemble consensus.
arXiv preprint arXiv:1910.09792, 2019.

Junnan Li, Yongkang Wong, Qi Zhao, and Mohan S Kankanhalli.
Learning to learn from noisy labeled data.
In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5051–5059, 2019.

Junnan Li, Richard Socher, and Steven CH Hoi.
Dividemix: Learning with noisy labels as semi-supervised learning.
arXiv preprint arXiv:2002.07394, 2020.

Shikun Li, Xiaobo Xia, Shiming Ge, and Tongliang Liu.
Selective-supervised contrastive learning with noisy labels, 2022.
URL https://arxiv.org/abs/2203.04181.

Yixuan Li, Jason Yosinski, Jeff Clune, Hod Lipson, and John E Hopcroft.
Convergent learning: Do different neural networks learn the same representations?
In *Adv. Neural Inform. Process. Syst. NeurIPS*, pages 196–212, 2015.

Sheng Liu, Jonathan Niles-Weed, Narges Razavian, and Carlos Fernandez-Granda.
Early-learning regularization prevents memorization of noisy labels.
Advances in neural information processing systems, 33:20331–20342, 2020.

Xingjun Ma, Yisen Wang, Michael E Houle, Shuo Zhou, Sarah Erfani, Shutao Xia, Sudanthi Wijewickrema, and James Bailey.
Dimensionality-driven learning with noisy labels.
In *International Conference on Machine Learning*, pages 3355–3364. PMLR, 2018.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever.
Deep double descent: Where bigger models and more data hurt.
Journal of Statistical Mechanics: Theory and Experiment, 2021(12):124003, 2021.

Brady Neal, Sarthak Mittal, Aristide Baratin, Vinayak Tantia, Matthew Scicluna, Simon Lacoste-Julien, and Ioannis Mitliagkas.
A modern take on the bias-variance tradeoff in neural networks.
arXiv preprint arXiv:1810.08591, 2018.

Behnam Neyshabur.
Implicit regularization in deep learning, 2017.
URL https://arxiv.org/abs/1709.01953.

Duc Tam Nguyen, Chaithanya Kumar Mummadi, Thi Phuong Nhung Ngo, Thi Hoai Phuong Nguyen, Laura Beggel, and Thomas Brox.
Self: Learning to filter noisy labels with self-ensembling.
arXiv preprint arXiv:1910.01842, 2019.

Diego Ortego, Eric Arazo, Paul Albert, Noel E. O'Connor, and Kevin McGuinness.
Multi-objective interpolation training for robustness to label noise.
CoRR, abs/2012.04462, 2020.
URL https://arxiv.org/abs/2012.04462.

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.

Geoff Pleiss, Tianyi Zhang, Ethan Elenberg, and Kilian Q Weinberger.
Identifying mislabeled data using the area under the margin ranking.
Advances in Neural Information Processing Systems, 33:17044–17056, 2020.

Iuliia Pliushch, Martin Mundt, Nicolas Lupp, and Visvanathan Ramesh.
When deep classifiers agree: Analyzing correlations between learning order and image statistics.
arXiv preprint arXiv:2105.08997, 2021.

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.

Hwanjun Song, Minseok Kim, and Jae-Gil Lee.
Selfie: Refurbishing unclean samples for robust deep learning.
In *International Conference on Machine Learning*, pages 5907–5915. PMLR, 2019.

Hongxin Wei, Lei Feng, Xiangyu Chen, and Bo An.
Combating noisy labels by agreement: A joint training method with co-regularization, 2020.
URL https://arxiv.org/abs/2003.02752.

Daphna Weinshall and Dan Amir.
Theory of curriculum learning, with convex loss functions.
Journal of Machine Learning Research, 21(222):1–19, 2020.

Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang.
Learning from massive noisy labeled data for image classification.
In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2691–2699, 2015.
doi: 10.1109/CVPR.2015.7298885.

Yazhou Yao, Zeren Sun, Chuanyi Zhang, Fumin Shen, Qi Wu, Jian Zhang, and Zhenmin Tang.
Jo-src: A contrastive approach for combating noisy labels, 2021.
URL https://arxiv.org/abs/2103.13029.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals.
Understanding deep learning requires rethinking generalization.
In *Int. Conf. Learning Representations ICLR*, 2017.

Evgenii Zheltonozhskii, Chaim Baskin, Avi Mendelson, Alex M Bronstein, and Or Litany.
Contrast to divide: Self-supervised pre-training for learning with noisy labels.
In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1657–1667, 2022.