UNIVERSITY OF CALIFORNIA
RIVERSIDE

# The Role of Data Quality and Heterogeneity on the Calibration of Neural Networks

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

Master of Science

in

Computer Science

by

Yuan Zhao

March 2020

Dissertation Committee:

Dr. Samet Oymak, Chairperson
Dr. Christian Shelton
Dr. Evangelos Papalexakis

The Dissertation of Yuan Zhao is approved:

---

---

---

Committee Chairperson

University of California, Riverside

## Acknowledgments

I submit my heartiest gratitude to my respected advisor, Dr. Samet Oymak, who give me the opportunity to explore the fantastic field of machine learning. During my graduate research, he expertly guided and encouraged me. I appreciate all his contributions of time, idea and help to make the project productive. I will never forget the time to discuss the problem and work together at three in the morning. His devoted research attitude will undoubtedly benefit my lifelong career.

I would also like to express my gratitude to the great members of the OptML lab. Special thanks to Mingchen Li and Yahya Sattar, for their generous support both in research and personal lives. I wish them all the best.

I am also thankful to the faculty in the CS department of UCR. I would like to sincerely thank the graduate program coordinator, Vanda Yamaguchi, who patiently helped me with the graduation related affairs.

Finally, I would like to give my thanks to my parents and my girlfriend for their their sacrifice and love.

To my parents for all the support.

To my girlfriend, Xinyi, who encourages me in difficult times.

# ABSTRACT OF THE DISSERTATION

The Role of Data Quality and Heterogeneity on the Calibration of Neural Networks

by

Yuan Zhao

Master of Science, Graduate Program in Computer Science
University of California, Riverside, March 2020
Dr. Samet Oymak, Chairperson

Neural networks have been widely studied and used in recent years due to its high classification accuracy and training efficiency. With the increase of network depth, however, the models become worse calibrated, meaning they cannot reflect the true probabilities. On the other hand, in many applications such as medical diagnosis, facial recognition and self-driving cars, the calibrated output probabilities are of critical importance. Therefore, the understanding of the cause of deep neural network uncalibration is of much concern.

The influence of model structures on the output calibration has been explored. However, the impact of the training dataset quality and heterogeneity, such as dataset size and label noise remains unclear. In this thesis, the impact of data quality and heterogeneity on the output calibration is investigated theoretically and experimentally. Afterwards, the defect of calibration methods using single global parameter are discussed. To overcome the calibration issues resulting from the dataset heterogeneity, we propose an improved calibration technique that can give better performance.

# Contents

# List of Figures

# List of Tables

Nowadays, deep neural networks have been applied to diverse and growing number of domains due to their stellar performance in terms of prediction accuracy. In many safety-critical application, such as medical diagnosis[3, 10, 11], self-driving cars[4, 5, 9] and face recognition[29, 18] etc., however, accuracy is not the only metric we are concerned about. Instead, the models should also give a correct probability of a prediction. For example, if a clinical CT (computer tomography) image diagnosis system gives 0.7 probability of a patient having a tumor, there should indeed be a 70% chance of a tumor being there. Therefore, the correctness of the output confidence is of significance. If the probability of the prediction can reflect the ground truth correctness likelihood, the model is calibrated.

Researches about the impact of model structures (depth, width and batch normalization) on the output calibration have shown that modern deep neural networks exhibit relatively higher uncalibration compared with conventional shallow neural networks[12]. To address this issue, different kinds of postprocessing calibration techniques have been proposed, such as platt scaling[25], vector scaling, histogram method[30], isotonic regression[31], etc. On the other hand, the study of the influence of the dataset quality is insufficient and needs more attention. This is due to that, in practice, data may suffer from error annotation[27, 8] and insufficient sampling[20]. Without having a deep understanding of the relation between dataset quality and output calibration, the postprocessing calibration methods may give a fake satisfying result. In fact, this can happen as is discussed in a later chapter that when the dataset is heterogeneous (partially label-noisy or under-sampled), the calibration schemes treating all classes uniformly may give a good overall calibration but is poorly calibrated for an individual class.

In this thesis, the impact of dataset quality and heterogeneity on the model calibration is explored. Specifically, this thesis first gives an observation and explanation of output confidence of a model trained by data with varying levels of noise or sample sizes. The results show that label noise in training data leads to under-confident models and small-size training data lead to over-confident models. Based on this, an intuitive and general approach to individually calibrate each class is proposed. Different calibration metrics, the expected calibration error (ECE) and the worst-case (maximum) calibration error (max-ECE), are selected to evaluate the performance of the proposed class-wise calibration method (CTS).

In the following chapter 2, the background of neural network calibration is introduced. Including the brief introduction of neural networks (ResNet and Wide ResNet) used in this work, CIFAR-10 and CIFAR-100 datasets, metrics of calibration (ECE and max-ECE) and two classic calibration techniques (temperature scaling and vector scaling). In chapter 3, we will give two important observations and theoretical explanations on how data quality (label corruption and under sampling) effects the model confidence. Based on the results of chapter 3, in chapter 4, a new class-wise algorithm is discussed. Then its calibration performance on classifiers trained by heterogeneous dataset is compared with two classic methods by the metrics mentioned in chapter 2.

# Chapter 1

# Background

This thesis mainly focuses on the model calibration issues rising from data quality and heterogeneity in data classification. In this chapter, the background of neural nework is introduced. Then two neural networks classifiers and two datasets used in the following expriments are described. Afterwards, the concept of model calibration and three calibration metrics are listed. At last, the classic calibration methods (temperature scaling and vector scaling) are discussed.

## 1.1 Neural Netwoks

In supervised learning, the machine learning algorithm gets a labeled training dataset. Each sample is a pair of input data and an output label. The goal of supervised learning is to find a general function or rule that maps the input to the output label. Furthermore, the mapping should be general so that unseen data is also correctly mapped.

One of the commonly used models in supervised learning is the neural network.

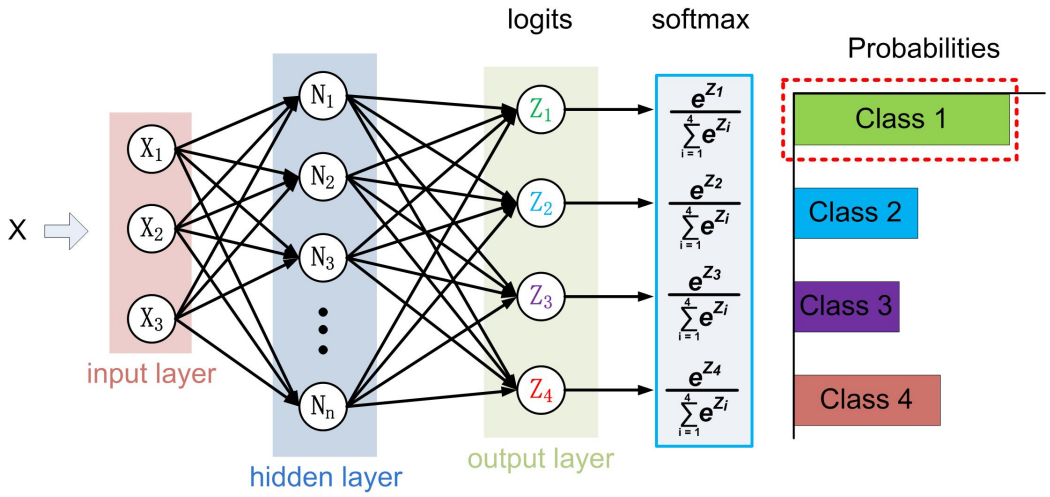



Figure 1.1: Schematic of neural network

Neural networks are a set of algorithms designed to recognize patterns. They are composed of interconnected, simple processing elements called artificial neurons. Neural networks can have one or more layers of neurons. Typically, a neural network consists of an input layer, one or more hidden layers and an output layer, as is shown in Fig. 1.1.

For classification purposes, the output of a neural network is the predicted class. The ability of the model to correctly predict the class of input data can be measured by accuracy. To meet different application requirements with high performance, various neural networks have been proposed and studied [2, 13, 32]. The following subsection introduces Residual Network (ResNet) and Wide Residual Network (WideResNet) that are used in this study.

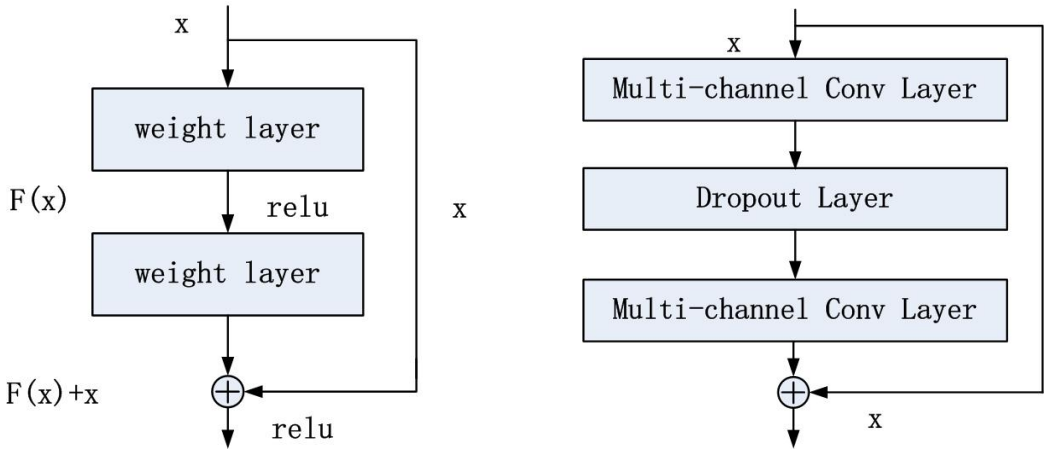
### 1.1.1 ResNet

Before 2015, the depth of the neural network is just dozens of convolutional layers. This is because with the increase of layers in a deep neural network, the accuracy becomes

saturate at a point and eventually get worse than a shallow network which is known as degradation problem. For example, a 20-layer CNN has higher accuracy than a 56-layer CNN both in training and testing phases [13]. This obstacle impedes the application of deeper networks and limited the classifier achieving higher classification accuracy.

The residual network (ResNet) [13] was proposed to solve this problem. Instead of simply stacking convolutional layers, ResNet added a shortcut connection between the input and the output of a residual block. A typical residual block is shown in 1.2. This structure actually fits a residual function $F(x) = H(x) - x$, where $x$ is input to the layer and $H(x)$ is the output of residual block. The bypass $x$, also called as identity mapping, is later added to $F(x)$. Due to the bypass connection, when $F(x) = 0$, the block is a simple identity mapping of the input, which enables the ResNet to reserve more input information than the stacking CNN. This solves the aforementioned degradation problem. By using a deep ResNet, better accuracy can be achieved which makes ResNet a popular solution to data classifications.

In this work, the residual architecture is used for CIFAR-10 classification. It is constructed of 20 layers, first starting with the 3x3 input convolutional layer. Next, there is a stack of 2n layers (or n residual blocks) for each feature map size 32x16x8 with the filters 16x32x64. After each convolutional layer, batch normalization is also used. At the end of the model, there is used global average pooling and dense layer with softmax which gives confidence values.




(a) A ResNet block (b) A Wide-ResNet block

Figure 1.2: Structures of a ResNet block and a Wide-ResNet block

### 1.1.2 WideResNet

Due to the success of ResNet, different ResNet variants are developed. Among them, the wide residual network (WideResNet) [32] gains much attention because of its faster convergence and higher accuracy. Instead of using deeper residual blocks, the author demonstrated that by increasing the width (number of channels of the weighted layer) and decreases the depth (fewer residual blocks) of a ResNet, the model can achieve higher accuracy in a shorter convergence time. A typical WideResNet block is shown in Fig. 1.2. A WideResNet is often named as WideResNet-k-N, where k representing the widened factor by enlarging the number of channels in each block, and parameter N indicates how many blocks are in one group. In the following study, a WideResNet-10-28 is used to classify the CIFAR-100 dataset.

## 1.2 Dataset

In the following experiments, CIFAR-10 and CIFAR-100 are adopted as standard evaluation datasets.

### 1.2.1 CIFAR-10

CIFAR10 dataset consists of 60 000 32x32 color images in 10 classes. Typically, it is split into 50 000 training and 10 000 test sets. The classes are airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck.

### 1.2.2 CIFAR-100

The CIFAR-100 dataset is similar to the CIFAR-10. There are 100 classes in CIFAR-100 with 600 images in each class. Typically, each class is split into 500 training and 100 testing samples.

## 1.3 Calibration

In most tasks, the accuracy of a trained neural network is the most concerned. However, in many safety-critical applications, such as self-driving cars and medical diagnosis, not only the accurate prediction is needed, but also the exact likelihood of the prediction is desired[21, 22, 14, 25]. For instance, a true probability of a patient having a tumor produced by a CT image classification model can help doctors make a suitable follow-up treatment plan.

In modern neural networks, the output probability is realized by adopting a soft-

max function as is shown in 1.1. The softmax function is a normalization function, which turns the output logits into the $[0, 1]$ interval and guarantees their sum to be one. Therefore, the values after softmax can be interpreted as probabilities. The predicted result is the class with the highest softmax value. However, this value may not reflect the true model confidence, which means when the output probability is 0.8, there may not an 80% correct prediction. Guo [12] showed that in modern neural networks with deep and wide layers, the model becomes much more uncalibrated, specifically over-confident (the model confidence is higher than prediction accuracy), which means the model tends to give a high confident prediction but cannot reach the same prediction correctness. Hence, calibration techniques are proposed to address this mismatch [12, 17, 19, 7, 28, 15, 23, 16].

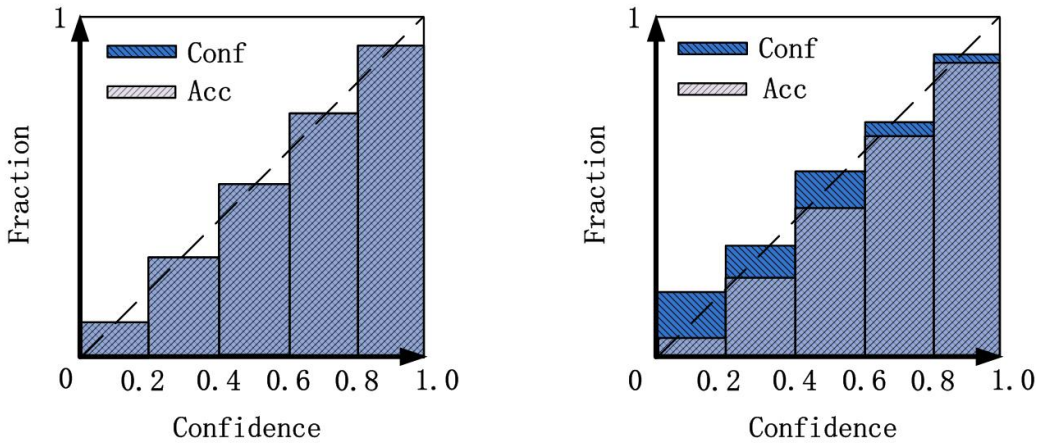
## 1.4 Calibration Metrics

The following subsections introduce three evaluation metrics for calibration which are adopted in chapter 3 and chapter 4.

### 1.4.1 Reliability Diagram

The reliability diagram is an intuitive way to visualize the calibration error. In a reliability diagram figure, the horizontal axis is the output sample confidence from 0 to 1 which is chunking into $M$ bins, with an interval size of $1/M$. The vertical axis is the fraction of average accuracy and average confidence for each bin. A schematic reliability diagram is shown in Fig. 1.3

For a perfect calibration, the average accuracy and average confidence of each




(a) Reliability diagram of an perfect-calibrated model

(b) Reliability diagram of an over-confident model

Figure 1.3: Schematic of a reliability diagrams of a perfect and an over-confident model

bin completely overlap, meaning the accuracy and the confidence align exactly along the diagonal. However, in practical cases, it is almost impossible to achieve perfect calibration. If the average accuracy is lower than average confidence, the classifier is over-confident. On the other hand, If the average accuracy is higher than average confidence, the classifier is under-confident.

### 1.4.2 Expected Calibration Error(ECE)

Although the reliability diagram is an intuitive metric, quantified metrics are needed to make a more precise comparison between different calibration techniques. One commonly used numerical value metric is the expected calibration error (ECE).

First, we consider the description of a supervised classification problem with multiple classes. Denote the joint distribution $\mathcal{D}$ of input/output pairs $(X, Y)$ via

$$P(Y, X) = P(Y|X)P(X).$$

Input $X \in \mathcal{X}$ and output $Y \in \{1, 2, \ldots, K\}$ are random variables where $Y$ is the true class assignment and $\mathcal{X}$ is the input space. $\hat{Y}$ is the predicted class.

For an input $X$, $f$ outputs a class decision $\hat{Y} = \arg\max_{1 \le k \le K} f(X)_k$ with confidence $\hat{P} = f(X)_{\hat{Y}}$, where $f(X)_k$ denotes the $k^{\text{th}}$ entry of the output vector. $\hat{Y}, \hat{P}$ are functions of $f$ and $X$. $\hat{P}_f(X), \hat{Y}_f(X)$ will explicitly highlight this dependence.

ECE measures and combines the distance between model accuracy and confidence at fixed confidence levels on the predicted label. Its continuous version with respect to $\ell_1$ metric is given by

$$\text{ECE}(f) = \text{ECE}(f, \mathcal{D}) = \mathbb{E}_{\hat{P}}[|P(Y = \hat{Y}|\hat{P} = p) - p|].$$

This continuous version operates in infinitesimal confidence intervals. Discrete version of ECE circumvents this by using binned confidences as defined below.

Split the interval $[0, 1]$ into $M$ disjoint intervals $(B_i)_{i=1}^M$. Discrete ECE is given by

$$\text{ECE}(f) = \sum_{i=1}^{M} \mathbb{E}[|P(Y = \hat{Y}|\hat{P} \in B_i) - p|]\mathbb{P}(\hat{P} \in B_i).$$

In the following experiments, ECE bins are chosen to be equally spaced which is the common approach in the related literature. Given a dataset $\mathcal{S} = (\boldsymbol{x}_i, \boldsymbol{y}_i)_{i=1}^n$, we denote the finite-sample versions of ECE by $\text{ECE}(f, \mathcal{S})$ obtained by averaging over the dataset.

### 1.4.3 Maximum ECE

ECE is an effective evaluation indicator, however, it ignores the difference between different classes. This is of great importance if the input data is heterogeneous. To evaluate

the worst-case calibration within different classes, we defined a new metric Maximum ECE (max-ECE).It is quantified via the maximum error over the the class-conditional distributions $\mathcal{D}_k = P(X, Y \big| \hat{Y}_f(X) = k)$ defined as

$$\text{Max-ECE}(f) = \max_{1 \le k \le K} \text{ECE}(f, \mathcal{D}_k) \tag{1.1}$$

## 1.5 Classic Calibration Methods

### 1.5.1 Temperature Scaling

Temperature scaling (TS) is a common and successful calibration technique which is a special case of Platt scaling. Assume that classifier $f$ can be decomposed as a softmax function applied to logits $f_{\text{lgt}}$ i.e. $f(X) = \text{sftmx}(f_{\text{lgt}}(X))$. This is a natural assumption for modern classifiers such as deep networks. TS searches for the calibrated function within the function space parameterized by a scalar $\alpha$ given by

$$\mathcal{F} = \{f_\alpha \quad \text{where} \quad \alpha \in [\alpha_-, \alpha_+]\}$$

where $f_\alpha(X) = \text{sftmx}(\alpha f_{\text{lgt}}(X))$. Given a validation set $\mathcal{S} = (Y_i, X_i)_{i=1}^n$ and calibration loss $\ell_{\text{calib}}$, we obtain optimal $\alpha$ via

$$\alpha_\star = \arg \min_{\alpha \in [\alpha_-, \alpha_+]} \text{calib\_loss}(f_\alpha, \mathcal{S}) \quad \text{where} \tag{1.2}$$

$$\text{calib\_loss}(f_\alpha, \mathcal{S}) = \frac{1}{n} \sum_{i=1}^{n} \ell_{\text{calib}}(Y_i, f_\alpha(X_i)). \tag{1.3}$$

### 1.5.2 Vector Scaling

Vector scaling (VS) is a generalization of temperature scaling and allows for more flexible fit by using $2K$ parameters for calibration (compared to the single parameter of

TS). However, this may lead to overfitting in the calibration process [12]. Specifically, vector scaling calibrates over a larger class of functions given by

$$f_{\boldsymbol{a},\boldsymbol{b}} = \text{sftmx}(\boldsymbol{a} \bigodot f_{\text{lgt}}(X) + \boldsymbol{b}). \tag{1.4}$$

Here $\bigodot$ is the entrywise product and $f_{\boldsymbol{a},\boldsymbol{b}}$ is parameterized by the $K$ dimensional scaling vector $\boldsymbol{a}$ and bias vector $\boldsymbol{b}$.

# Chapter 2

# The Impact of Data Quality on Model Confidence

This chapter first studies the influence of dataset quality (label noise and small size) on the model confidence. To simplify the analysis, the discussion in this chapter focuses on binary classification with linear classifiers and minimizes binary negative logistic loss (NLL) for training. Specifically, our classifier $f$ will be parameterized by a vector a and intercept b via

$$f_{\boldsymbol{a},b}(X) = \text{sftmx}(\boldsymbol{a}^T X + b) = \frac{\text{e}^{\boldsymbol{a}^T X + b}}{1 + \text{e}^{\boldsymbol{a}^T X + b}}.$$

## 2.1 The Role of the Sample Size

### 2.1.1 Small Size Training Dataset Leads to Over-confident Model

We discuss model confidence influenced by a small training dataset. This scenario frequently appears in anomaly detection and rare-event classification. Deep networks are often trained until they achieve 100% training accuracy ([33]) and sufficiently large deep networks can provably achieve 100% accuracy if data is not degenerate. Once a network $f = \text{sftmx}(f_{\text{lgt}})$ achieves 100% accuracy, it will still attempt to push NLL to zero. Loss can be pushed to 0 by scaling up the logits i.e. letting $\alpha \to \infty$ in the class of functions $\text{sftmx}(\alpha f_{\text{lgt}})$. This eventually leads to classifiers with 100% confidence in training data as well as in the test data. The reason is as soon as one entry of $f_{\text{lgt}}$ is favorable over the others (which is guaranteed to happen except for degenerate distributions/classifiers), letting $\alpha \to \infty$ will lead to 100% confidence in the predicted class. The following result formalizes this intuition and states that a small sample size can provably lead to further over-confidence.

**Theorem 1** *There exists a distribution $\mathcal{D}$ (with unit $\ell_2$ norm input set $\mathcal{X}$) as follows. Generate datasets $\mathcal{S}_1 = (X_i, Y_i)_{i=1}^{n} \overset{i.i.d.}{\sim} \mathcal{D}$ and $\mathcal{S}_2 = (X_i, Y_i)_{i=1}^{50n} \overset{i.i.d.}{\sim} \mathcal{D}$ and fix $R > 0$. Minimize the empirical NLL loss on these datasets to find linear classifiers $f_1, f_2$ as follows.*

$$f_i = \arg \min_{f \in \{f_{\boldsymbol{a},b} \mid \|\boldsymbol{a}\|_{\ell_2} \le R\}} NLL(f, \mathcal{S}_i).$$

*Given precision $\varepsilon > 0$, choose $R \ge 6 \log(50n + \varepsilon^{-1})$. With probability at least $9/10$ (over the proper set $\mathcal{S}_1$ or $\mathcal{S}_2$), we have the following accuracy and confidence behavior.*

- *For all inputs $X \in \mathcal{X}$ and $i \in \{1, 2\}$: $\hat{P}_{f_i}(X) \ge 1 - \varepsilon$.*

- $\mathbb{P}_{\mathcal{D}}(\hat{Y}_{f_1}(X) = Y) \leq 1 - \frac{1}{20n}$ *and* $\mathbb{P}_{\mathcal{D}}(\hat{Y}_{f_2}(X) = Y) = 1$.

In the setup above, both large dataset ($\mathcal{S}_2$) and small dataset ($\mathcal{S}_1$) problems lead to arbitrarily high confidence classifiers (over all viable inputs in $\mathcal{D}$); however, the model trained on the small dataset is provably less accurate, which indicates that the smaller dataset makes the model over-confident. The proof idea is constructing a distribution where certain features have low probability, thus requiring more data to learn them.

**Proof.** The NLL (cross-entropy) loss on a dataset $\mathcal{S}$ is given by

$$\mathrm{NLL}(f_{\boldsymbol{a},b}, \mathcal{S}) = -\frac{1}{n}\sum_{i=1}^{n} \log\left(\frac{\mathrm{e}^{y_i(\boldsymbol{a}^T X_i + b)}}{1 + \mathrm{e}^{y_i(\boldsymbol{a}^T X_i + b)}}\right) = \frac{1}{n}\sum_{i=1}^{n} \log(1 + \mathrm{e}^{-y_i(\boldsymbol{a}^T X_i + b)})$$

Fix orthogonal unit $\ell_2$ norm vectors $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^p$. Set $\boldsymbol{v'} = (\boldsymbol{u} + \boldsymbol{v})/\sqrt{2}$. Define the binary distribution $\mathcal{D}$ as follows.

$$\mathbb{P}(Y = 1 \big| X = \boldsymbol{v}) = \mathbb{P}(Y = 0 \big| X = \boldsymbol{v'}) = \mathbb{P}(Y = 0 \big| X = -\boldsymbol{v}) = 1 \tag{2.1}$$

$$\mathbb{P}(X = \boldsymbol{v}) = 1/2, \quad \mathbb{P}(X = \boldsymbol{v'}) = 1/N, \quad \mathbb{P}(X = -\boldsymbol{v}) = 1/2 - 1/N. \tag{2.2}$$

Let $E_i$ be the event that $\boldsymbol{v'}$ appears as an input in dataset $\mathcal{S}_i$. Observe that

$$\mathrm{e}^{-n/N} \geq 1 - \mathbb{P}(E_1) = (1 - 1/N)^n \geq 1 - n/N.$$

Thus, setting $N = 20n$, we find $\mathbb{P}(E_1) \leq 0.05$ and $\mathbb{P}(E_2) \geq 1 - \mathrm{e}^{-50n/20n} = 1 - \mathrm{e}^{-2.5} \geq 0.91$. Also let $B$ be the event that at least $1/3$ of the training inputs are equal to $\boldsymbol{v}$ and at least $1/3$ are equal to $-\boldsymbol{v}$. Applying a standard Chernoff bound yields that $\mathbb{P}(B) \geq 1 - 2\mathrm{e}^{-\frac{n}{100}}$.

Before proceeding further, we also note that for all $x \geq 0$, we have

$$\mathrm{e}^{-x}/2 \leq \log(1 + \mathrm{e}^{-x}) \leq \mathrm{e}^{-x}$$

**Analyzing $\mathcal{S}_1$ on the event $E_1 \cap B$:** Suppose $E_1$ holds. Note that the training dataset only contains inputs $\boldsymbol{v}$ and $-\boldsymbol{v}$. Thus, it can be concluded that the optimal classifier has the form $R\boldsymbol{v} + b$ for some scalar $b$ i.e. $\boldsymbol{a} = R'\boldsymbol{v}$ for $|R'| \le R$. Let $0 \le \gamma \le 1$ denote the fraction of $+\boldsymbol{v}$ inputs within the training data. The empirical (training) NLL is given by

$$\mathrm{NLL}(f_{\boldsymbol{a},b}, \mathcal{S}_1) = \gamma \log(1 + \frac{1}{\mathrm{e}^{R'+b}}) + (1-\gamma)\log(1 + \frac{1}{\mathrm{e}^{R'-b}})$$

Minimizing NLL over $R'$ reveals $R' = R$ and the loss is given by

$$\mathrm{NLL}(f_{\boldsymbol{a},b}, \mathcal{S}_1) = \gamma \log(1 + \frac{1}{\mathrm{e}^{R+b}}) + (1-\gamma)\log(1 + \frac{1}{\mathrm{e}^{R-b}})$$

We next bound the optimal $b$ choice. Under event $B$, $\gamma, 1-\gamma \ge 1/3$. Using $\boldsymbol{a} = R\boldsymbol{v}, b = 0$ as an upper bound, we have that

$$\mathrm{e}^{-R} \ge \gamma \log(1 + \frac{1}{\mathrm{e}^{R+b}}) + (1-\gamma)\log(1 + \frac{1}{\mathrm{e}^{R-b}}) \tag{2.3}$$

$$\ge \frac{1}{3}\log(1 + \frac{1}{\mathrm{e}^{R-|b|}}) \ge \min(\frac{1}{6}\frac{1}{\mathrm{e}^{R-|b|}}, \frac{\log(2)}{3}) \tag{2.4}$$

which implies $|b| \le \log 6$. Now observe that optimal classifier (on training), which obeys $\boldsymbol{a} = R\boldsymbol{v}, |b| \le \log 6$, outputs the wrong decision on $\boldsymbol{v}'$ since

$$\hat{Y}_{f_1}(\boldsymbol{v}') = \mathrm{sign}(\boldsymbol{a}^T\boldsymbol{v}' + b) = \mathrm{sign}(\frac{R}{\sqrt{2}} + |b|) = 1$$

as $R \ge \sqrt{2}\log 6$. This implies $\mathbb{P}(\hat{Y}_{f_1}(X) = Y) \le 1 - 1/20n$. However, confidence on $\boldsymbol{v}'$ (as well as on $\pm\boldsymbol{v}$) is lower bounded as follows

$$\hat{P}_{f_1}(X) \ge \frac{1}{1 + \mathrm{e}^{-(R/\sqrt{2} - \log 6)}} \ge 1 - \mathrm{e}^{-(R/\sqrt{2} - \log 6)} \ge 1 - \varepsilon$$

whenever $R \ge \sqrt{2}(\log 6 + \log(1/\varepsilon))$ which is implied by $R \ge 3\log\max(6, 1/\varepsilon)$.

**Analyzing $\mathcal{S}_2$ on the event $E_2$:** We claim that the classifier achieves small loss on all examples $\boldsymbol{v}, -\boldsymbol{v}, \boldsymbol{v}'$ which will help show the result. First we pick a baseline classifier $\boldsymbol{a} = R\frac{\boldsymbol{v}-2\sqrt{2}\boldsymbol{u}}{3}$ and $b = 0$. This guarantees that for all $(Y, X) \sim \mathcal{D}$

$$YX^T\boldsymbol{a} \geq R/3.$$

Thus empirical NLL over $\mathcal{S}_2$ is at most $-\log(\frac{e^{R/3}}{1+e^{R/3}}) = \log(1 + \frac{1}{e^{R/3}}) \leq e^{-R/3}$. The overall loss will bound the individual losses i.e. at the optimal classifier $(\boldsymbol{a}, b)$ (on training data) for any training example $(X, Y) \in \mathcal{S}_2$ we have

$$e^{-R/3} \geq \text{NLL}(f_{\boldsymbol{a},b}, \mathcal{S}_2) \geq \frac{-1}{50n}(Y \log f(X) + (1-Y)\log(1-f(X)))$$

Since $R \geq 6\log(50n)$, we find $50ne^{-R/3} \leq e^{-R/6}$. Without losing generality, let us assume $Y = 1$. This implies

$$e^{-R/6} \geq -\log f(X) \implies f(X) \geq e^{-e^{-R/6}} \implies f(X) \geq 1 - e^{-R/6}.$$

To achieve $1-\varepsilon$ probability, we need $e^{-R/6} \leq \varepsilon$ which holds whenever $R \geq 6\log(1/\varepsilon)$. For $\varepsilon < 1/2$, this also implies the classification is correct i.e. $Y = \hat{Y}$ since $f(X) > 1/2$. The identical argument holds when $Y = 0$. ■

## 2.2 The Role of Label Noise

### 2.2.1 Label-noisy Training Dataset Leads to Under-confident Model

For label noise, we work with a noisy dataset model with a discrete distribution over $\mathcal{X} = \{\boldsymbol{v}, -\boldsymbol{v}\}$.

**Definition 2** ($\mathcal{D}_{\text{noisy}}(p_+, p_-)$) *Fix a vector $\boldsymbol{v} \in \mathbb{R}^d$ with unit $\ell_2$ norm and let $\mathcal{X} = \{\boldsymbol{v}, -\boldsymbol{v}\}$. Fix the noise levels $0 \le p_-, p_+ \le 1/2$. Suppose that $\mathbb{P}(X = \boldsymbol{v}) = 1/2$ and the conditional class distributions obey*

$$\mathbb{P}(Y = 1 \big| X = \boldsymbol{v}) = 1 - p_+ \text{ and } \mathbb{P}(Y = 0 \big| X = -\boldsymbol{v}) = 1 - p_-.$$

The next lemma is a straightforward result that captures the properties of the linear classifier on this noisy data model.

**Lemma 3** *Fix $1/2 > p_+, p_-, p_{test} \ge 0$. Suppose the data is distributed with $\mathcal{D} = \mathcal{D}_{noisy}(p_{test}, p_{test})$, but the training set is corrupted by label noise in an unbalanced way with distribution $\mathcal{D}_{noisy}(p_+, p_-)$. A linear classifier $f$ minimizing population (infinite sample) training NLL loss obeys*

$$\begin{aligned}
&\textit{Test confidence over } +: && \hat{P}(\boldsymbol{v}) = f(\boldsymbol{v}) = 1 - p_+ \\
&\textit{Test confidence over } -: && \hat{P}(-\boldsymbol{v}) = 1 - f(-\boldsymbol{v}) = 1 - p_- \\
&\textit{Test accuracy over either:} && \mathbb{P}(\hat{Y} = Y \big| X) = 1 - p_{test}.
\end{aligned}$$

This lemma highlights that if the training data is noisier than the test (e.g. $p_+, p_- > p_{\text{test}}$), the classifier will be under-confident at test time, explaining the behavior in Fig. 2.1e. It also shows that individual classes or inputs can have different confidence levels as a function of noise.

**Proof.** Classifier outputs the probability $f(X) = \frac{e^{\boldsymbol{a}^T X + b}}{1 + e^{\boldsymbol{a}^T X + b}}$. Note that $X = x\boldsymbol{v}$ for $x \in \{-1, 1\}$ hence without losing generality, we can assume $\boldsymbol{a} = a\boldsymbol{v}$ since any direction orthogonal to $\boldsymbol{v}$ has zero inner product with input. Then, classifier simplifies to a single dimension as

follows

$$f(X) = \frac{e^{ax+b}}{1+e^{ax+b}}$$

We need to find $a_*, b_*$ that maximizes the negative NLL loss

$$-\mathcal{L}(\boldsymbol{a}, b) = \mathbb{E}[Y \log(f(X)) + (1-Y)\log(1-f(X))]$$

This expectation leads to the scalar optimization

$$-2\mathcal{L}(\boldsymbol{a}, b) = (1-p_+)\log(\frac{e^{a+b}}{1+e^{a+b}}) + p_+ \log(\frac{1}{1+e^{a+b}}) + (1-p_-)\log(\frac{e^{a-b}}{1+e^{a-b}}) + p_- \log(\frac{1}{1+e^{a-b}}).$$

Note that we can re-parameterize the loss by considering it as a function of $\alpha = a+b$ and $\beta = a-b$. Together it gives

$$-2\mathcal{L}(\alpha, \beta) = (1-p_+)\log(\frac{e^{\alpha}}{1+e^{\alpha}}) + p_+ \log(\frac{1}{1+e^{\alpha}}) + (1-p_-)\log(\frac{e^{\beta}}{1+e^{\beta}}) + p_- \log(\frac{1}{1+e^{\beta}}).$$

Right hand side is maximized when partial derivatives with respect to $\alpha$ and $\beta$ are zero i.e.

$$\begin{aligned}
\frac{-2\partial\mathcal{L}(\alpha,\beta)}{\partial\alpha} &= [(1-p_+)\frac{1}{1+e^{\alpha}} - p_+\frac{1}{1+e^{-\alpha}})] \\
\frac{-2\partial\mathcal{L}(\alpha,\beta)}{\partial\beta} &= [(1-p_-)\frac{1}{1+e^{\beta}} - p_-\frac{1}{1+e^{-\beta}}].
\end{aligned}$$

Note that partial derivative w.r.t. $\alpha$ depends only on $p_+$ and partial derivative w.r.t. $\beta$ depends only on $p_-$ which greatly simplifies our life. Proceeding, we find that $\alpha_* = a_* + b_*$ satisfies the likelihood ratio

$$\frac{1-p_+}{1+e^{\alpha_*}} - \frac{p_+}{1+e^{-\alpha_*}} = 0 \implies \frac{1+e^{\alpha_*}}{1+e^{-\alpha_*}} = \frac{1-p_+}{p_+} \tag{2.5}$$

Note that this implies that the classifier output is

$$f(\boldsymbol{v}) = \frac{e^{a_*+b_*}}{1+e^{a_*+b_*}} = \frac{e^{\alpha_*}}{1+e^{\alpha_*}} = 1-p_+.$$

Similarly, following $\beta_* = a_* - b_*$, we find $\frac{1+e^{\beta_*}}{1+e^{-\beta_*}} = \frac{1-p_-}{p_-}$ and $f(-\boldsymbol{v}) = p_-$. On the other hand, this classifier always predicts 1 for $X = \boldsymbol{v}$ and 0 for $X = -\boldsymbol{v}$ (as $1 - p_+ > 1/2$ and $p_- < 1/2$). As a result, since the test data is distributed with $\mathcal{D}_{\text{noisy}}(p_{\text{test}}, p_{\text{test}})$, the test accuracy will be $1 - p_{\text{test}}$ for both classes. ■

### 2.2.2 Model Confidence of a Fully Trained Neural Network

Recall that once a network achieves 100% training accuracy further training will eventually lead to 100% confidence. Thus large capacity and sufficiently trained networks should not be under-confident. On the other hand, for noisy datasets (e.g. Fig. 2.1e), the training stops before the model achieves 100% training accuracy which is the key source of under-confidence.

## 2.3 Verification Experiments

To verify the previous analysis of the impact of data quality, two sets of experiments are designed with different neural networks and datasets. The first set of experiments uses a ResNet-20 model to classify the CIFAR-10 dataset, and the second experiment is based on the WideResNet-28-10 model and CIFAR-100 dataset.

### 2.3.1 Experiments Setup

#### ResNet-20 model and CIFAR-10 preprocessing

For CIFAR-10, 60 000 samples are split into a training set with 50 000 samples and a 10 000 samples test set. The samples are preprocessed by subtracting the per-channel

mean of the training images and dividing by the standard deviation. Further, they are augmented by flipped horizontally in 50 percent of cases. Additionally, the training images are also padded by 4 pixels from every side and then randomly cropped, so the final image size is 32x32. The padded pixels are either reflections of the image or constant value 0.

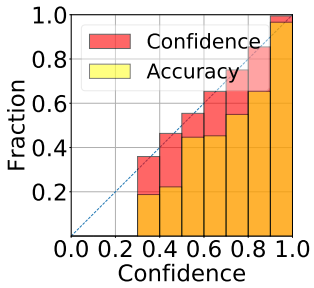
To train the ResNet-20, 200 training epochs of the ResNet-20 with Adam optimizer are used to fit the data, with cross-entropy as the loss function using Keras and TensorFlow [1, 6]. The initial learning rate is initially set to $10^{-3}$ and decreases to $10^{-4}$ after 80 epochs. The epoch with maximum testing accuracy is recorded.

**WideResNet-28-10 model and CIFAR-100 preprocessing**

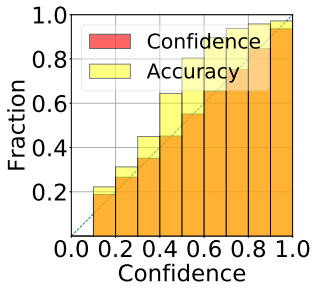
Similar to CIFAR-10, for CIFAR-100, 60 000 samples are split into 50 000 training samples and 10 000 test samples. 200 training epochs of the WideResNet-28-10 with SGD optimizer were used to fit the data, with cross-entropy as the loss function using PyTorch[24]. The initial learning rate is initially set to 0.1 and decreases to 0.02, 0.004, 0.0008 after 60, 120, 160 epochs respectively. The epoch with maximum testing accuracy is recorded.

### 2.3.2 Experiments Results

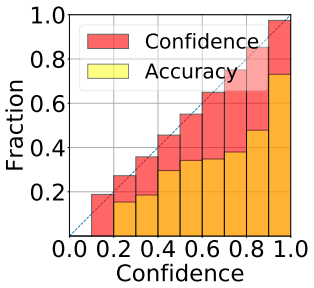
In the following experiments, clean and standard-split test datasets are always used (10 000 samples for both CIFAR-10 and CIFAR-100). No postprocessing calibration algorithm is applied. The reliability diagrams are obtained on the test set with uncalibrated models. The number of confidence bins is 10. Average confidence is represented by red bars, average accuracy is represented by yellow bars, and their overlap is represented by orange bars. Note that the confidence always (approximately) follows the diagonal line by



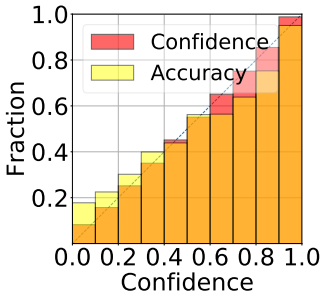

(a) Standard CIFAR-10 model trained with standard data



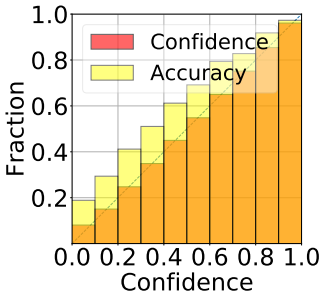

(b) CIFAR-10 model trained with noisy data (30% noisy rate)



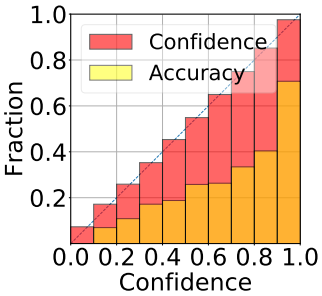

(c) CIFAR-10 model trained with small classes (2% sampling rate)




(d) Standard CIFAR-100 model trained with standard data




(e) CIFAR-100 model trained with noisy data (30% noisy rate)




(f) CIFAR-100 model trained with small class sizes (10% sampling rate)

Figure 2.1: Reliability diagram of ResNet-20 models and WideResNet-28-10 Models

construction.

Fig. 2.1 shows empirically that data quality can greatly affect the model confidence. Specifically, in Fig. 2.1b, a CIFAR-10 model with noisy data (*i.e.,* 30% chance of label corruption) is trained. Compared to the standard CIFAR-10 model with perfect labels (Fig. 2.1a), the model with noisy data suffers from under-confidence on the test set. On the other hand, a CIFAR-10 model trained with small sizes (only 100 labels per class rather than the standard 5000 labels) results in over-confident models (Fig. 2.1c), especially compared to the default CIFAR-10 model (Fig. 2.1a).

The same conclusion can be drawn by the WideResNet-28-10 model with CIFAR-100 dataset. As is shown in Fig. 2.1 (d-f), WideResNet-28-10 exhibits under-confident when the training dataset is label-noisy (Fig. 2.1e) and over-confident when training dataset is much smaller than standard one.

In summary, the results of the experiment are in agreement with the aforementioned theoretical analysis that label corruption leads to a under-confident model and small size leads to a more over-confident model.

# Chapter 3

# The Model Calibration on Heterogeneous Dataset

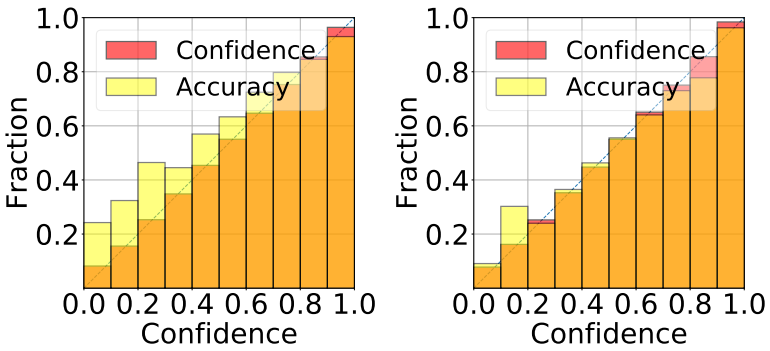
In the practical application of a neural network, the training dataset may be heterogeneous, meaning some of the classes in the training dataset may contain noisy labels while other classes remain clean, or some of the classes contain a smaller number of samples than others. For example, when gathering the monitoring data of mixed industrial organic gases emitting from a factory, one of the specific gas-sensitive sensors gathered insufficient data due to out of power, and the overall dataset of collected gases forms a heterogeneous dataset.

In this chapter, the impact of heterogeneous datasets on model calibration is first investigated. Then a class-wise algorithm is proposed. After that, a class-wise temperature scaling algorithm (CTS) is discussed followed by the comparisons of CTS, TS and VS.

## 3.1 The Impact of Heterogeneous Datasets on Model Calibration

### 3.1.1 The Role of Noise-Imbalanced Dataset

To investigate the influence of the noise-imbalanced dataset, we add 30% label noise on the training data (50 000 samples) of the CIFAR-100 classes 0-49 and keep the classes 50-99 clean. Then the constructed heterogeneous training set is used to train a WideResNet-28-10 model with the setup same as that in chapter 3. Fig 3.1 provides separate reliability diagrams for the noisy and clean subsets of the overall CIFAR-100 dataset at the end of training and before any calibration. The contrasting subsets are determined by the actual test labels. In consistency with theoretical intuition, this figure demonstrates that noisy classes tend to be underconfident and clean classes tend to be over-confident. The average accuracy over noisy classes 0-49 is 0.689 and average confidence is 0.627. In contrast, average accuracy over clean classes 50-99 is 0.768 and average confidence is 0.781.

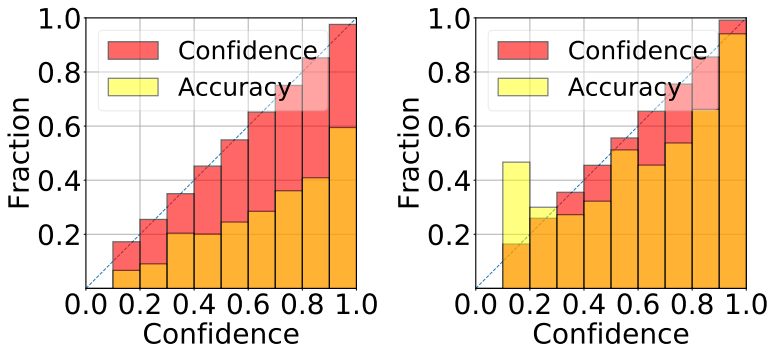



(a) Noisy classes (0-49) (b) Clean classes (50-99)

Figure 3.1: Reliability diagrams for the noisy and clean subsets of the dataset

### 3.1.2 The Role of Size-Imbalanced Dataset

Our next experiment explores the heterogeneity of the sample sizes within the classes. We use the same model and training setup in section 3.1.1. For the training set, we under-sample classes 0-49 at 10% (i.e. 50 per class rather than 500) and classes 50-99 remain untouched. Fig 3.2 provides reliability diagrams for undersampled vs fully-sampled classes. The contrasting subsets are determined by the actual test labels. This figure demonstrates that under-sampled classes tend to be more over-confident than fully-sampled classes. The average accuracy over under-sampled classes 0-49 is 0.396 and average confidence is 0.728. In contrast, average accuracy over fully sampled classes 50-99 is 0.841 and average confidence is 0.909.




(a) Undersampled classes (b) Fully sampled classes

Figure 3.2: Reliability diagrams for the clean subsets and undersampled subsets

## 3.2 Class-wise Calibration Algorithm

From section 3.1 we can refer that, when dealing with the calibration of a model trained on a heterogeneous dataset, without discriminating the difference between classes may result in a fake well-calibrated model. For example, at one extreme, when the model is trained on a partially label-noisy dataset, the under-confident output of noisy classes and over-confident output of clean classes cancel out, making the model perfect calibrated. Furthermore, popular calibration schemes such as temperature scaling [12] typically try to find optimal global parameters that are used to calibrate all samples uniformly (e.g. using a single calibration parameter). Thus this kind of one size fits all approach may be ineffective. Global calibration may fail to treat such heterogeneities, leading to worse calibration performance.

To address this issue, we proposed a class-wise calibration algorithm. Our approach is summarized in Alg. 1 and applies post-processing on a given classifier $f$. It can use an arbitrary calibration function $\mathcal{C}$ (chosen from a set $\mathcal{F}_{\text{cal}}$) which takes a classifier $f$ and outputs a calibrated classifier $\mathcal{C}(f)$ (e.g. $\mathcal{C}$ applies Platt scaling on $f$) . The core idea is splitting a heterogeneous dataset $\mathcal{S}$ into homogenous subsets so that $\mathcal{C}$ can calibrate each subset individually. The appropriate splitting is a function of the dataset (i.e. its size and type of heterogeneity), and prior information can guide the subset selection. A good example is related to fair machine learning where a dataset may be heterogeneous with respect to a sensitive input feature (e.g. race, sex) [26]. We can create the sub-datasets, (e.g. corresponding to different demographic groups) based on the distinct values of the sensitive feature. While our approach can apply to any general splitting policy, in this work, we

**Algorithm 1** Class-wise Calibration

**Inputs:** Classifier $f$, validation dataset $\mathcal{S}$, regularization $\Gamma$

Calibration loss function calib_loss$(\cdot)$ (e.g. NLL, ECE)

Set of calibrators $\mathcal{F}_{\text{cal}}$ (e.g. Platt scalings)

**Outputs:** Calibrated classifier $f_{\text{cal}}$

$\mathcal{S}_k = \{(X, Y) \in \mathcal{S} \mid k = \hat{Y}\},\ \forall\ 1 \le k \le K.$

Solve the calibration optimization

$$\mathcal{C}_k^\star = \min_{\mathcal{C}_k} \sum_{i=1}^{K} \text{calib\_loss}(\mathcal{C}_k(f), \mathcal{S}_k) \quad \text{s.t.} \tag{CC}$$

$$\|\mathcal{C}_k - \mathcal{C}_0\| \le \Gamma,\ \mathcal{C}_k \in \mathcal{F}_{\text{cal}} \quad \forall\ 0 \le k \le K.$$

For any fresh input sample $X$, $f_{\text{cal}}$ returns

$f_{\text{cal}}(X) = \mathcal{C}_{\hat{Y}}^\star(f(X))$

restrict our attention to the heterogeneity across different classes and focus on class-wise splitting to address unbalanced class distributions.

Specifically, $\mathcal{S}$ is split into $K$ subsets $(\mathcal{S}_k)_{k=1}^K$ where $\mathcal{S}_k$ is the set of samples whose predicted labels $\hat{Y}$ are class $k$. Note that, we use *predicted labels* for calibration rather than the actual labels, because at the time of inference, we won't have access to the labels and have to infer them.

Our algorithm takes a calibration loss (e.g. NLL, ECE) and solves the Class-wise Calibration problem (CC). The key idea is individually calibrating each class to obtain $\mathcal{C}_k^\star(f)$ from the base function $f$. (CC) admits a regularization parameter $\Gamma$ which quantifies the level of multi-task learning. $\Gamma = 0$ reduces to standard (non-class-wise) calibration whereas $\Gamma = \infty$ means each class is calibrated by themselves which may be more prone to

over-fitting. Finally, for inference in test time, the final calibrated classifier $f_{\text{cal}}$ calls the sub-classifier $\mathcal{C}_k^\star(f)$ whenever the predicted tag is class $k$.

## 3.3 Class-Wise Temperature Scaling Method (CTS)

When Algorithm 1 is specialized to TS, we get the Class-wise Temperature Scaling (CTS) algorithm. When $\Gamma = \infty$, CTS picks $K$ distinct scalars $(\alpha_k)_{k=1}^K$ by training on the sets $\mathcal{S}_k$ with predicted label $\hat{Y} = k$. For a fresh sample $X$, CTS outputs class probabilities $f_{\text{cal}}(X) = \text{sftmx}(\alpha_{\hat{Y}} f_{\text{lgt}}(X))$.

In the following sections, to simplify the CTS, we keep $\Gamma = \infty$ and demonstrate the effectiveness of CTS and its advantages over classic TS and VS methods.

## 3.4 Experiments Setup

**Datasets:** CIFAR-10 and CIFAR-100 datasets are used to demonstrate the proposed class-wise temperature scaling algorithm. In the experiments, whenever CIFAR-10 and CIFAR-100 validation is needed, the original training set is split into 45k training samples and 5k validation samples. We only modify the training data. The validation set is always clean (*i.e.,* not noisy). All experiments use the standard data augmentation by shifting the width and height of the image as well as flipping the image horizontally. Experiments are repeated five times with different random seeds. To evaluate the impact of heterogeneous data, two variants of CIFAR-10 and CIFAR-100 are constructed:

- *Noise-imbalanced dataset construction (§3.5):* In the training dataset, we add label noise to classes 0 to 4 for CIFAR-10 and 0 to 49 for CIFAR-100 with noise rate $\rho$

varying from 0 to 1. The remaining Classes are unchanged. This results in a noise-imbalance training set.

- *Size-imbalanced dataset construction (§3.6):*

  We under-sample the classes 0 to 4 in the CIFAR-10 training set and 0 to 49 in the CIFAR-100 training set, with the sampling rate $\rho \in [0.01, 1]$ and $[0.05, 1]$ respectively. Instead of the usual $n$ training samples, under-sampled classes have only $n\rho$ training samples. For instance, with $\rho = 0.01$, the smaller classes of CIFAR-10 contain only 45 samples resulting in a highly unbalanced dataset. The overall training set is obtained by combining the downsampled classes and the other classes.

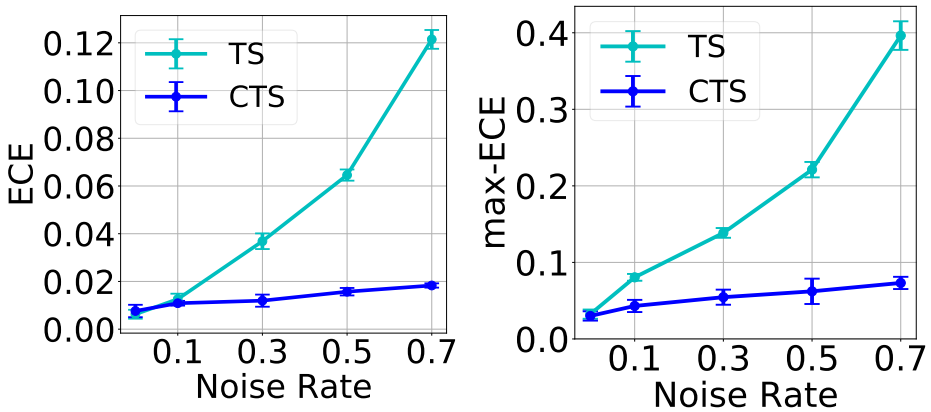
**Comparison algorithms:** We compare the performance of two class-wise approaches (class-wise temperature scaling and vector-scaling) versus two standard approaches that globally apply to all samples (temperature scaling and no calibration). The reported ECE and max-ECE metrics of each algorithm are generated from the test dataset.

**Metrics:** We evaluate the performance of the above algorithms through the ECE and max-ECE: We optimize the NLL loss for calibration optimization (*e.g.,* fitting TS, CTS, VS) as a proxy for ECE and max-ECE in all experiments.

**Neural network model:** To perform image classification, we utilize the ResNet-20 for CIFAR-10 and WideResNet-28-10 for CIFAR-100 network models. The training processes are the same with those in Chapter 2.

## 3.5 Noise-Imbalanced Training Data

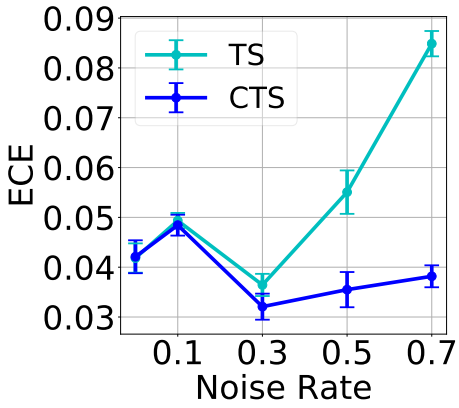
### 3.5.1 Comparison between TS and CTS



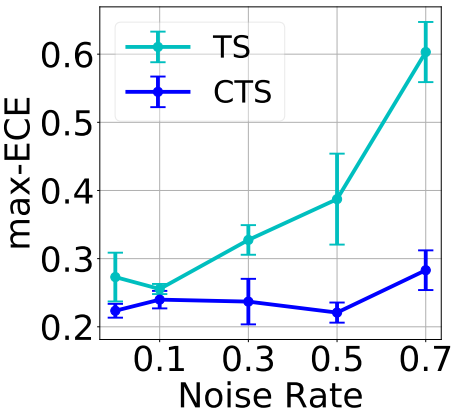

(a) ECE (CIFAR-10)




(b) max-ECE (CIFAR-10)




(c) ECE (CIFAR-100)




(d) max-ECE (CIFAR-100)

Figure 3.3: Impact of training data noise on the TS and CTS algorithms

In this experiment, we evaluate the impact of noise-imbalanced training data on the calibration error.

In Fig. 3.3, we plot the ECE and max-ECE as we sweep across different noise

rates. In both experiments of CIFAR-10 and CIFAR-100, the CTS method shows significant improvement over simple TS, especially when there is more noise in the dataset. These results suggest that not only can CTS achieve better calibration on individual classes (as shown by the max-ECE plot), but can also result in a better calibrated model from global perspective (as shown from the global ECE plot).

### 3.5.2 Comparison with VS

VS is another class-wise calibration method that may give better performance than non-class-wise methods, due to better fitting capability (as long as overfitting does not occur). In this set of simulations, we compare the class-wise CTS and VS methods with non-class-wise TS and uncalibrated methods.

We construct a training dataset with a 30% label corruption rate for half of the classes. We compare the calibration error of VS, TS and CTS is according to accuracy, ECE, and max-ECE.

| **Alg.** | **Acc. (%)** | **ECE (%)** | **max-ECE (%)** |
|---|---|---|---|
| Uncal. | $71.53 \pm 0.13$ | $3.78 \pm 0.30$ | $36.83 \pm 4.30$ |
| VS | $72.82 \pm 0.22$ | $3.26 \pm 0.21$ | $18.30 \pm 1.31$ |
| TS | $71.53 \pm 0.13$ | $3.64 \pm 0.22$ | $32.74 \pm 2.17$ |
| CTS | $71.53 \pm 0.13$ | $3.21 \pm 0.26$ | $23.70 \pm 3.34$ |

Table 3.1: Comparison of class-wise (VS, CTS) and non-class-wise (uncalibration, TS) calibration methods (CIFAR-100)

Table 3.1 and 3.2 show the results. In terms of max-ECE, VS is the most preferable, while CTS also has good performance. In terms of ECE, CTS outperforms other methods

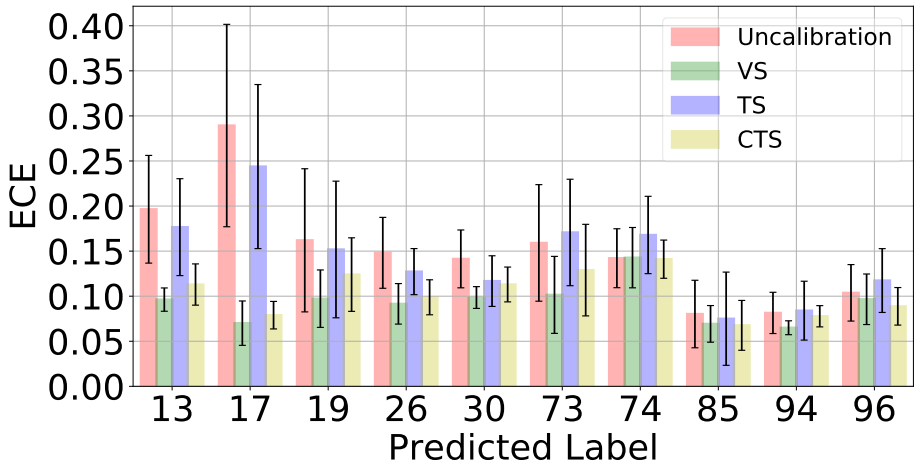
| **Alg.** | **Acc. (%)** | **ECE (%)** | **max-ECE (%)** |
|---|---|---|---|
| Uncal. | $86.98 \pm 0.45$ | $6.50 \pm 0.46$ | $21.09 \pm 0.49$ |
| VS | $87.27 \pm 0.39$ | $1.31 \pm 0.11$ | $4.37 \pm 0.56$ |
| TS | $86.98 \pm 0.45$ | $3.69 \pm 0.33$ | $13.85 \pm 0.64$ |
| CTS | $86.98 \pm 0.45$ | $1.19 \pm 0.25$ | $5.46 \pm 0.98$ |

Table 3.2: Comparison of class-wise (VS, CTS) and non-class-wise (uncalibration, TS) calibration methods (CIFAR-10).

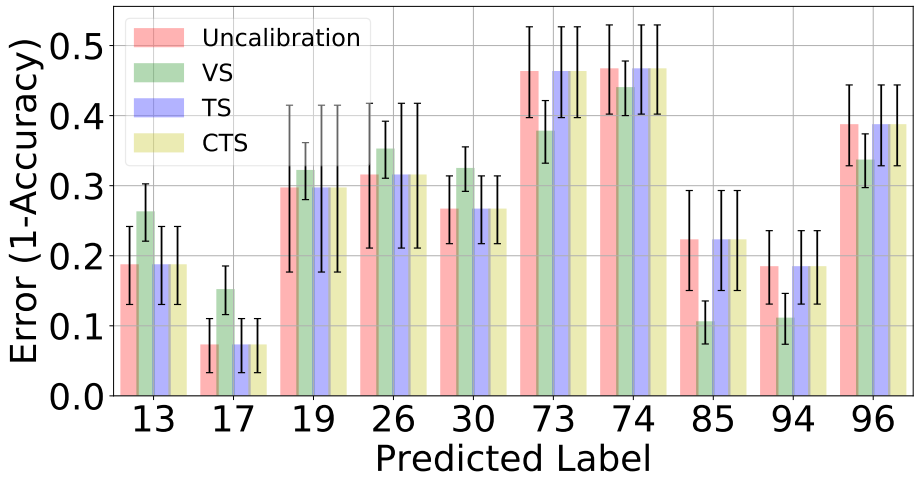
due to the benefits from the class-wise calculation procedure. Both class-wise methods, VS and CTS, have improvement over the non-class-wise TS method.

Aside from the similar calibration performance of VS and CTS, VS *slightly improves the prediction accuracy* , which is a surprising observation.

For instance, as shown in Fig. 3.4b and 3.5b, VS uniformly degrades the prediction accuracy over noisy classes (classes 0-49) and uniformly improves the average accuracy over clean classes (classes 50-99). Note that noisy classes are already suffering from lower accuracy due to the noise, and VS ends up amplifying this while improving the overall accuracy. In contrast, by construction the CTS prediction is guaranteed to be consistent with the original classifier as discussed. Fig. 3.4a and 3.5a breaks down the results from Table 3.1 and 3.2 respectively, and shows that ECE is lower for VS and CTS in every class when compared to TS and no calibration.

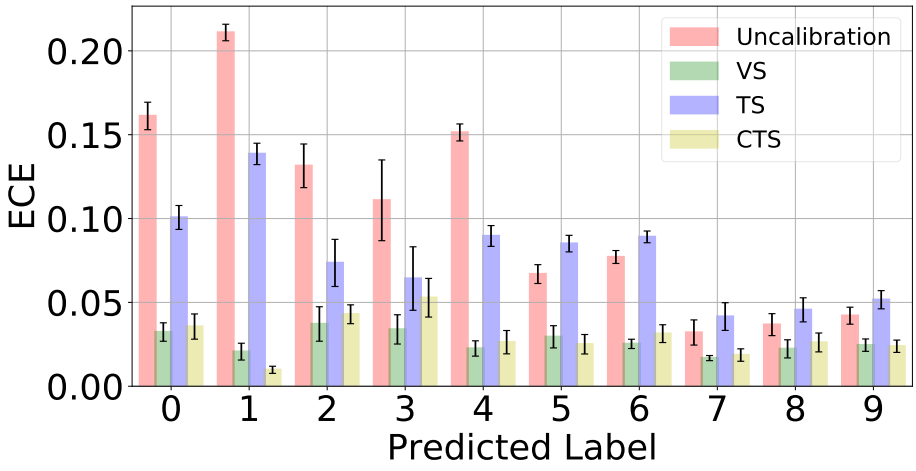



(a) ECE)

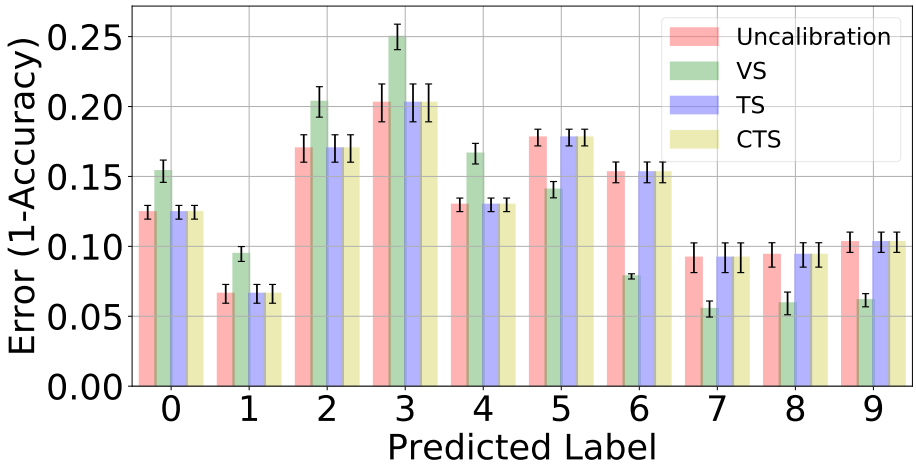



(b) Classification error

Figure 3.4: ECE and accuracy for five random classes (from each of 0-49 and 50-99) are visualized (CIFAR-100)




(a) ECE)

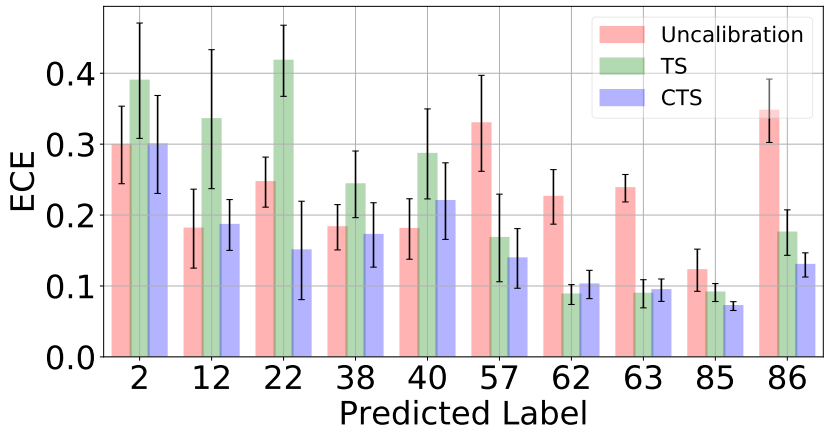



(b) Classification error

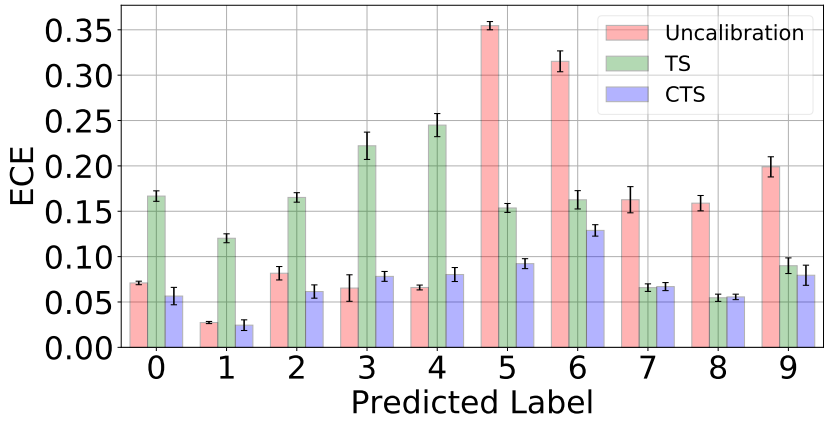
Figure 3.5: Per-class error in terms of ECE and classification accuracy (0-4 are noisy and 5-9 are clean) (CIFAR-10).

## 3.6 Size-Imbalanced Training Data

We next investigate the effectiveness of CTS on size-imbalanced training set. We construct the unbalanced training dataset as described in §3.4.




(a) The first fifty classes are downsampled at 5% (CIFAR-100)

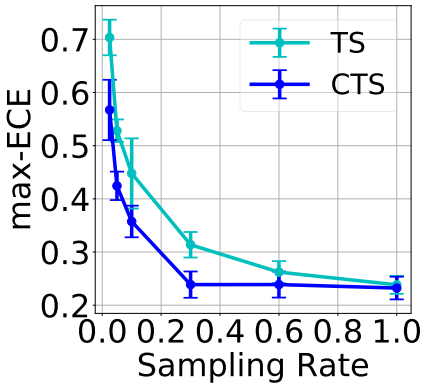



(b) The first five classes are downsampled at 6% (CIFAR-10)

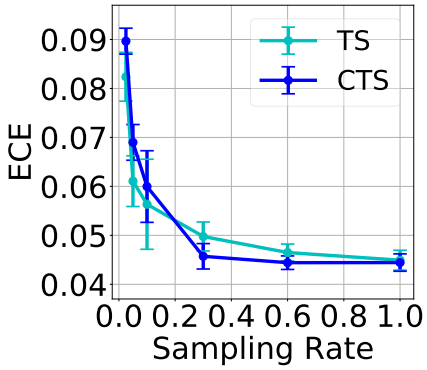
Figure 3.6: ECE when the first half classes are downsampled. CTS generally has lower ECE for both the small and large classes.

Fig. 3.6 shows the ECE errors associated with individual classes as labeled by the

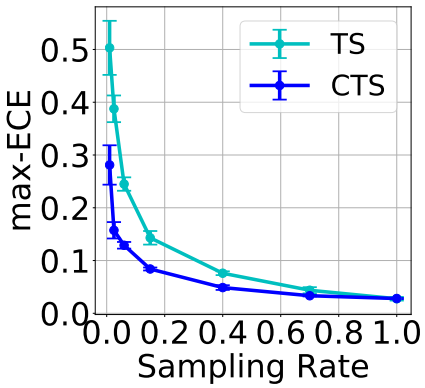
classifier, *i.e.*, $\text{ECE}_k = \text{ECE}(f, \mathcal{D}_k)$ where $\mathcal{D}_k$ is the conditional distribution $P(Y, X | \hat{Y} = k)$. Here smaller classes are 5% (CIFAR-100) and 6% (CIFAR-10) as large as the non-down-sampled classes. The results show that CTS provides uniform improvement over original uncalibrated classifier for all classes. In contrast, TS actually inflates the calibration errors of the under-represented smaller classes, while improving the performance over larger classes. This suggests that class-wise calibration provides a more fair treatment of the classes.



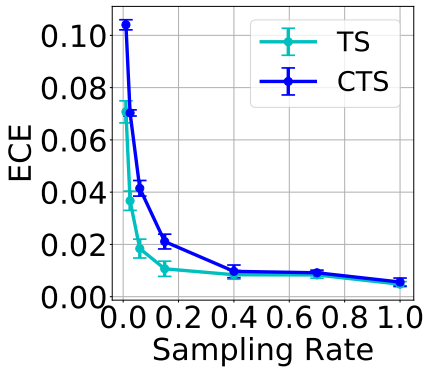

(a) max-ECE error (CIFAR-100)




(b) ECE error (CIFAR-100)




(c) max-ECE error (CIFAR-10)




(d) ECE error (CIFAR-10)

Figure 3.7: Calibration error as a function of the training set sampling rate

To further understand the impact of sample size on calibration error, we plot the

ECE as a function of sampling rate in Fig. 3.7.

Fig. 3.7a and 3.7c shows that CTS uniformly outperforms TS in terms of max-ECE metric for all sampling rates, highlighting the fairness benefit of CTS. However, perhaps surprisingly, we find that in terms of overall ECE (where all samples are aggregated), TS outperforms CTS (Fig. 3.7b and 3.7d). Upon digging deeper into this, we found that this is due to the way that individual class confidences output by TS combine in a favorable fashion when they are merged in a given confidence bin, as is done in the overall ECE metric. For example, suppose there are only two classes with equal sizes, and fix a confidence bucket, *e.g.,* $[0.4, 0.6]$.

- Suppose Class 1 has average accuracy of 0.52, TS confidence of 0.6, and CTS confidence of 0.54.
- Suppose Class 2 has average accuracy of 0.48, TS confidence of 0.4, and CTS confidence of 0.5.

In this case, TS will achieve $\text{ECE}_1 = \text{ECE}_2 = 0.08$ whereas CTS will achieve $\text{ECE}_1 = \text{ECE}_2 = 0.02$, so CTS is better. However, CTS is overconfident in both classes whereas TS is perfectly calibrated when both classes are combined, resulting in $\text{ECE}_{TS} = 0$ and $\text{ECE}_{CTS} = 0.02$.

# Chapter 4

# Summary

In this thesis, we investigated the influence of the training data quality on the model calibration. Specifically, we make the following contributions.

We find that label noise in the training data leads to under-confident classifiers, and we provide a theoretical justification explaining this observation. This is surprisingly in contrast to over-confidence of deep networks trained with noiseless data.

Training sample size similarly has a major effect on classifier confidence. Specifically, in CIFAR-10 and CIFAR-100 experiments, smaller sample size leads to more over-confident classifiers due to lower accuracy.

Both of these observations are surprisingly transferable to classifiers trained on heterogeneous data. For instance, if label noise levels of classes are unbalanced (*e.g.,* some classes have more noise than the others), we find that classifier tends to be under-confident over noisy classes and over-confident over noiseless.

These observations motivate us to investigate class-wise calibration algorithms.

We propose an intuitive and general approach that allows for individually calibrating each class. Specifically, we slice the validation set by predicted class assignments and calibrate each slice separately. Our approach, coupled with temperature scaling method (TS), leads to class-wise temperature scaling (CTS) as a special case. We demonstrate the benefit of this approach when the classes exhibit noise and sample size imbalances. We also demonstrate the benefit of vector scaling as an alternative approach and contrast with CTS.

# Bibliography


[1] Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pages 265–283, 2016.

[2] Mikhail Belkin, Alexander Rakhlin, and Alexandre B. Tsybakov. Does data interpolation contradict statistical optimality? 06 2018.

[3] Arpit Bhardwaj and Aruna Tiwari. Breast cancer diagnosis using genetically optimized neural network model. *Expert Systems with Applications*, 42(9):4611, 2015.

[4] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

[5] Mariusz Bojarski, Philip Yeres, Anna Choromanska, Krzysztof Choromanski, Bernhard Firner, Lawrence Jackel, and Urs Muller. Explaining how a deep neural network trained with end-to-end learning steers a car. *arXiv preprint arXiv:1704.07911*, 2017.

[6] François Chollet et al. Keras, 2015.

[7] Jeffrey De Fauw, Joseph R Ledsam, Bernardino Romera-Paredes, Stanislav Nikolov, Nenad Tomasev, Sam Blackwell, Harry Askham, Xavier Glorot, Brendan O'Donoghue, Daniel Visentin, et al. Clinically applicable deep learning for diagnosis and referral in retinal disease. *Nature medicine*, 24(9):1342, 2018.

[8] Santosh K. Divvala, Ali Farhadi, and Carlos Guestrin. Learning everything about anything: Webly-supervised visual concept learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[9] George Hotz Eder Santana. Learning a driving simulator. *arXiv preprint arXiv:1608.01230*, 2016.

[10] Orhan Er, Nejat Yumusak, and Feyzullah Temurtas. Chest diseases diagnosis using artificial neural networks. *Expert Systems with Applications*, 37(7):7648, 2010.

[11] Orhan Er, Nejat Yumusak, and Feyzullah Temurtas. Tuberculosis disease diagnosis using artificial neural networks. *Journal of Medical Systems*, 34(3):299, 2010.

[12] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1321–1330. JMLR. org, 2017.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[14] Xiaoqian Jiang, Melanie Osl, Jihoon Kim, and Lucila Ohno-Machado. Calibrating predictive model estimates to support personalized medicine. *Journal of the American Medical Informatics Association*, 19(2):263–274, 2011.

[15] Ananya Kumar, Percy Liang, and Tengyu Ma. Verified uncertainty calibration. *arXiv preprint arXiv:1909.10155*, 2019.

[16] Aviral Kumar, Sunita Sarawagi, and Ujjwal Jain. Trainable calibration measures for neural networks from kernel mean embeddings. In *International Conference on Machine Learning*, pages 2810–2819, 2018.

[17] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pages 6402–6413, 2017.

[18] Ya Li, Guangrun Wang, Lin Niew, Qing Wang, and Wenwei Tan. Distance metric optimization driven convolutional neural network for age invariant face recognition. *attern Recognition*, 75(11):51, 2018.

[19] Shiyu Liang, Yixuan Li, and R Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv preprint arXiv:1706.02690*, 2017.

[20] Maciej A. Mazurowski, Piotr A. Habas, Jacek M. Zurada, Joseph Y. Lo, Jay A.Baker, and Georgia D. Tourassi. *Neural Networks*.

[21] Allan H Murphy. A new vector partition of the probability score. *Journal of applied Meteorology*, 12(4):595–600, 1973.

[22] Allan H Murphy and Robert L Winkler. Reliability of subjective probability forecasts of precipitation and temperature. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 26(1):41–47, 1977.

[23] Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[24] Adam Paszke, Sam Gross, and Francisco Massa etc. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, 2019.

[25] John Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

[26] Geoff Pleiss, Manish Raghavan, Felix Wu, Jon Kleinberg, and Kilian Q Weinberger. On fairness and calibration. In *Advances in Neural Information Processing Systems*, pages 5680–5689, 2017.

[27] F. Schroff, A. Criminisi, and A. Zisserman. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

[28] Sunil Thulasidasan, Gopinath Chennupati, Jeff Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. *arXiv preprint arXiv:1905.11001*, 2019.

[29] Xiaogang Wang Xiaoou Tang Yi Sun, Ding Liang. Deepid3: Face recognition with very deep neural networks. *arXiv preprint arXiv:1502.00873*, 2015.

[30] Bianca Zadrozny and Charles Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *Proceedings of the 18th International Conference on Machine Learning*, pages 609–616. JMLR. org, 2001.

[31] Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the 8th ACM SIGKDD*, pages 694–699, 2002.

[32] Sergey Zagoruyko and Komodakis Nikos. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[33] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *International Conference on Learning Representations*, 2016.