Instance-Wise Monotonic Calibration by Constrained Transformation

Yunrui Zhang[*1] **Gustavo Batista**[1] **Salil S. Kanhere**[1]

[1]School of Computer Science and Engineering, University of New South Wales, Australia

Abstract

Deep neural networks often produce miscalibrated probability estimates, leading to overconfident predictions. A common approach for calibration is fitting a post-hoc calibration map on unseen validation data that transforms predicted probabilities. A key desirable property of the calibration map is instance-wise monotonicity (i.e., preserving the ranking of probability outputs). However, most existing post-hoc calibration methods do not guarantee monotonicity. Previous monotonic approaches either use an under-parameterized calibration map with limited expressive ability or rely on black-box neural networks, which lack interpretability and robustness. In this paper, we propose a family of novel monotonic post-hoc calibration methods, which employs a constrained calibration map parameterized linearly with respect to the number of classes. Our proposed approach ensures expressiveness, robustness, and interpretability while preserving the relative ordering of the probability output by formulating the proposed calibration map as a constrained optimization problem. Our proposed methods achieve state-of-the-art performance across datasets with different deep neural network models, outperforming existing calibration methods while being data and computation-efficient. Our code is available at `https://github.com/YunruiZhang/Calibration-by-Constrained-Transformation`

1 INTRODUCTION

Deep learning models have achieved state-of-the-art accuracy across various tasks and applications. These models typically employ a softmax layer at the output, where the resulting outputs are interpreted as a probability distribution. However, despite their high accuracy, deep neural networks often suffer from miscalibration due to overfitting, producing inaccurate probability estimates. As demonstrated in prior work, these softmax-derived probabilities frequently fail to reflect true uncertainty, leading to systematically overconfident predictions [Guo et al., 2017].

Accurate probability estimates are critical for safety-sensitive applications where model confidence serves as a failsafe mechanism or informs human decision-making. For example, in medical diagnosis, miscalibrated predictions could lead to unwarranted trust in automated results, risking misdiagnosis [Van Calster et al., 2019]; in autonomous driving, overconfidence in object detection systems might compromise collision avoidance [Bieshaar et al., 2018]; and in financial decision-making, poorly calibrated uncertainty estimates could propagate costly errors [Tang et al., 2014], and many other application domains where an accurate estimate of the uncertainty is crucial for the task [Zhang et al., 2025b, Tan et al., 2025]. Beyond these domains, downstream tasks such as label shift estimation [Alexandari et al., 2020, Saerens et al., 2002, Zhang et al., 2025a] and out-of-distribution detection [Lee et al., 2018, Hendrycks and Gimpel, 2017] depend heavily on accurate probability estimates.

A common approach to improving the calibration of deep learning models is post-hoc calibration [Guo et al., 2017]. This approach assumes access to a validation set, referred to as the calibration set, consisting of data not seen during the training of a given base model. A parameterized calibration model is then trained on the outputs of the base model using this calibration set. The goal is to learn a calibration map that transforms the model's probabilistic outputs to improve their alignment with true confidence levels.

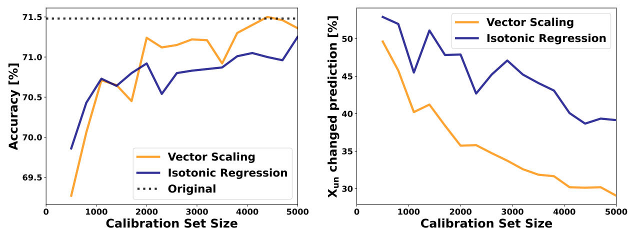
Post-hoc calibration operates by adjusting a model's predicted probabilities through a learned calibration map that modifies confidence scores based on the base model's predictions on the unseen calibration set. Specifically, when

[*]Corresponding author: yunrui.zhang@unsw.edu.au

the model's prediction is incorrect in the calibration set, the calibration map learns to reduce confidence in the misclassified class, whereas, for correct predictions, it increases confidence. This adjustment is achieved by optimizing the calibration map to minimize the loss between the adjusted probabilities and the ground-truth calibration set labels. Empirically, a well-learned calibration map can generalize effectively from the calibration to the test set.

Current post-hoc calibration methods can be broadly categorized into instance-wise monotonic and instance-wise non-monotonic methods, which we refer to as monotonic and non-monotonic. Monotonic methods preserve the ranking of predicted probabilities for each instance, ensuring that for each sample, the transformed probabilities maintain the same order as those produced by the original base model. In contrast, non-monotonic methods do not enforce this constraint.

Non-monotonic calibration maps can be easily parameterized and have demonstrated strong performance in some cases [Rahimi et al., 2022, Kull et al., 2019]. However, without enforcing monotonicity, and given that logits reside in a low-dimensional, abstract feature space derived from the original data, there is a risk of overfitting the calibration set logits. This overfitting can lead to a calibration map that fails to generalize well to testing sets. Moreover, since non-monotonic calibration maps do not preserve the ranking of predicted probabilities from the base model, they can negatively impact classification accuracy. To demonstrate this issue, we perform experiments with non-monotonic methods on CIFAR-100 with a wide range of calibration set sizes. As shown in Figure 1a, we observe a decrease in accuracy after non-monotonic calibration, which becomes more pronounced when the calibration set is smaller.



(a) Accuracy decrease (b) Impact on uncertain samples

Figure 1: (a) Non-monotonic methods (e.g., vector scaling, isotonic regression) reduce accuracy, with the effect worsening as the calibration set shrinks. (b) Non-monotonic methods alter a large proportion of high-uncertainty predictions, with this proportion increasing as the calibration size decreases (ResNet-110 on CIFAR-100).

Additionally, changes in predictions or rankings predominantly occur in samples with high uncertainty— the cases where retaining the base model's ranking precision is crucial. Let X_{un} denote the set of test samples where the top predicted probability from the base model is below 0.7, indicating high uncertainty. As shown in Figure 1b, non-monotonic methods alter the top prediction for a large proportion of samples in X_{un}, with this proportion increasing as the size of the calibration set decreases.

In our opinion, the primary role of post-hoc calibration is to correct systematic over- or under-confidence rather than to refine class rankings. By enforcing monotonicity, the calibration map adjusts probability estimates without altering the relative order of predictions. Thus, monotonicity can be viewed as a regularization constraint that enhances the robustness of the calibration map against a small or out-of-division calibration set.

While monotonicity is an important property for post-hoc calibration methods, how to perform monotonic calibration in a multiclass setting using a parameterized calibration map that is both interpretable and robust remains an open question. Existing monotonic calibration methods either rely on a small number of parameters, such as temperature scaling [Guo et al., 2017] and ensemble temperature scaling [Zhang et al., 2020] or employ black-box neural network-based parameterized models that lack robustness and interpretability for the calibration map [Rahimi et al., 2020, Tomani et al., 2022].

In this paper, we propose a family of novel monotonic post-hoc calibration maps that perform Instance-Wise Monotonic Calibration by Constrained Transformation (MCCT). By applying constrained transformations to sorted logits, the proposed method enables an interpretable and parameterized transformation while ensuring a robust and expressive calibration map that remains strictly monotonic. Implementing the calibration map through constrained optimization allows the model to learn a calibration map in an efficient and data-effective manner. Extensive experiments on CIFAR-10, CIFAR-100, and ImageNet demonstrate that our proposed methods, MCCT and its variant MCCT-I, outperform the existing state-of-the-art calibration methods. The source code is available at `https://github.com/YunruiZhang/Calibration-by-Constrained-Transformation`

2 PRELIMINARY

Consider a multiclass classification problem where we have $\mathbf{x} \in \mathbb{R}^d$ representing the input features and the one-hot encoded label $y = (y_1, \ldots, y_m) \in \{0,1\}^m$, such that $\sum_{i=1}^m y_i = 1$, where m is the number of classes.

Let $\mathcal{F} : \mathbb{R}^d \to \Delta^{m-1}$ be a probabilistic classifier that outputs the predicted class probability vector $\hat{p}(y|x) = (\hat{p}_1, \ldots, \hat{p}_m) \in [0,1]^m$ with $\sum_{i=1}^m \hat{p}_i = 1$ for each sample x. For deep neural networks, the class probability vector is usually generated by the softmax function: $\hat{p}_i =$

$\frac{\exp(z_i)}{\sum_{j=1}^{m}\exp(z_j)}$, where $Z = (z_1, \ldots, z_m) \in \mathbb{R}^m$ are the logits generated by the neural network for each sample before the softmax operation.

2.1 CALIBRATION

We want the predicted class probabilities to be calibrated for a classifier, meaning that the output probabilities should be close to the true probabilities of the corresponding classes. The strongest notion of calibration is complete calibration [Kull et al., 2019], defined as follows:

Definition 1 ***Complete Calibration***. *A classifier $\mathcal{F}$ that outputs $\{\hat{p}(y_i|\mathbf{x})\}_{i=1}^{m}$ is complete calibrated if $\hat{p}(y_i|\mathbf{x}) = p(y_i|\mathbf{x})$ for all classes y_i.*

Definition 1 represents the strictest form of calibration, where the predicted probabilities exactly match the true probabilities. A more relaxed notion is class-wise calibration [Zadrozny and Elkan, 2002], defined as follows:

Definition 2 ***Class-wise Calibration***. *A classifier $\mathcal{F}$ is class-wise calibrated for a class y_i if $\hat{p}(y_i|\mathbf{x}) = p(y_i|\mathbf{x})$.*

Another weaker but commonly used notion of calibration is confidence calibration [Guo et al., 2017], which requires only the top predicted probability (i.e., confidence) to be calibrated:

Definition 3 ***Confidence Calibration***. *A classifier $\mathcal{F}$ is confidence calibrated if $\arg\max_y \hat{p}(y|\mathbf{x}) = \arg\max_y p(y|\mathbf{x})$ and $\max \hat{p}(y|\mathbf{x}) = \max p(y|\mathbf{x})$.*

2.2 CALIBRATION ERROR

While calibration notions are well-defined, measuring calibration error remains an open question. In practical applications, the ground-truth class probabilities are rarely available, making it impossible to directly measure the complete calibration error $\mathbb{E}_X[|\hat{p}(y|\mathbf{x}) - p(y|\mathbf{x})|]$, or even class-wise or top-label calibration error without access to true class probabilities.

A common approach for estimating calibration error is the Expected Calibration Error (ECE) [Naeini et al., 2015, Guo et al., 2017], which approximates the top-label confidence miscalibration. ECE partitions the samples based on their predicted confidence $\max \hat{p}(y|\mathbf{x})$ into K equal-width bins. Each bin B_k contains samples whose confidence falls within the interval $I_k = \left(\frac{k-1}{K}, \frac{k}{K}\right]$. The calibration error is then computed as the weighted average of the absolute difference between the bin's accuracy and average confidence.

$$ECE = \sum_{k=1}^{K} |acc(B_k) - conf(B_k)| \tag{1}$$

$$acc(B_k) = \frac{1}{|B_k|}\sum_{i \in B_k} \mathbf{1}(\hat{y_i} = y_i)$$

$$conf(B_k) = \frac{1}{|B_k|}\sum_{i \in B_k} \max \hat{p}(y|\mathbf{x_i})$$

Other variants of ECE have been proposed in previous work, such as Maximum Calibration Error [Naeini et al., 2015], Class-wise Calibration Error [Kull et al., 2019], and adaptive ECE [Nixon et al., 2020]. However, ECE remains the most commonly used calibration measure due to its data efficiency and convenience for visualization.

2.3 POST-HOC CALIBRATION

For the task of post-hoc calibration, we assume a set of unseen calibration data $\mathbf{X}_c \in \mathbb{R}^d$ with corresponding one-hot encoded labels $\mathbf{Y}_c$, probability outputs $\hat{P}_c$, and logits Z_c provided by a deep neural network classifier $\mathcal{F}$.

The goal of post-hoc calibration is to learn a calibration map $T : \Delta^{m-1} \to \Delta^{m-1}$, trained on $\hat{P}_c$ and $\mathbf{Y}_c$, that transforms the model's probabilistic outputs at test time to improve calibration. In the context of deep neural networks, the calibration map can also be learned in the logit space, where $T : \mathbb{R}^m \to \mathbb{R}^m$ is trained on Z_c and $\mathbf{Y}_c$ to transform the testing time logits before softmax normalization. Logits reside in a less constrained feature space with more degrees of freedom than probability outputs. By operating on the logit space, we avoid the inherent constraints of the probability simplex and leverage the unconstrained logit geometry to improve calibration.

As outlined in the introduction, one desirable property for the calibration map is instance-wise monotonicity, meaning that the calibration transformation preserves the ordering of class probabilities or logits from the original classifier's predictions.

Definition 4 ***Instance-wise monotonic calibration***. *A calibration map $T : \mathbb{R}^m \to \mathbb{R}^m$ is instance-wise monotonic if for every logit vector or probability output $Z \in \mathbb{R}^m$,*

$$\forall i, j \in \{1, \ldots, m\}, \quad Z_i \geq Z_j \implies T(Z)_i \geq T(Z)_j.$$

This property is also called accuracy-preserving [Zhang et al., 2020] and intra-order preserving [Rahimi et al., 2020].

3 METHOD

In this section, we propose two monotonic calibration maps, MCCT and MCCT-I, based on the logit space and the corresponding implementation of the calibration map using constrained optimization.

First, we define a sorting function $S : \mathbb{R}^m \to \mathbb{R}^m$ that sorts a given logit vector Z into its sorted version $S(Z) = (Z_{(1)}, Z_{(2)}, \dots, Z_{(m)})$, where $Z_{(1)} < Z_{(2)} < \dots < Z_{(m)}$. To prevent ties in the sorting process, our proposed calibration map also makes the following assumption on each sample's output logit:

Assumption 1 *For every logit vector $Z = (Z_1, \dots, Z_m) \in \mathbb{R}^m$, its components are pairwise distinct:*

$$\forall i, j \in \{1, \dots, m\},\ (i \neq j) \implies Z_i \neq Z_j.$$

Assumption 1 ensures strict total ordering of logits, avoiding ties in class rankings by assuming no two logits have the same value. This assumption is reasonable for logits produced by a deep neural network, as the logit space is an unconstrained and continuous feature space $\mathbb{R}^m$ with the number of classes m usually relatively small. Empirically, in our experiments across various neural networks and datasets, we have not encountered tied logits in any sample, even for the more extreme cases such as ImageNet-1K, which has 1000 classes.

3.1 MCCT

Now we describe the proposed monotonic calibration map based on the logit space:

Theorem 1 *For transformation $f : \mathbb{R}^m \to \mathbb{R}^m$, defined as*

$$f(Z) = S^{-1}\big(S(Z) \odot w + b\big),$$

is monotonic if:
- $w \in \mathbb{R}^m$ vector satisfy $w_1 \leq w_2 \leq \dots \leq w_m$ and $w_i > 0$
- $b \in \mathbb{R}^m$ vector satisfy $b_1 \leq b_2 \leq \dots \leq b_m$
- S^{-1} is the inverse sorting operator that maps the sorted-and-transformed logit vector back to the original logit Z indices.

The proof for Theorem 1 is intuitive, since $(S(Z)$ and also w and b are non-descending, the transformed sorted vector $\hat{Z} = S(Z) \odot w + b$ preserves the ordering of $S(Z)$. So $\forall i, j \in \{1 \dots m\}$ if $Z_i < Z_j$ then $\hat{Z}_i < \hat{Z}_j$. Since the invert sorting operator S^{-1} maps $\hat{Z}$ back to the original logit indices by reversing the sorting permutation of S, it preserves this inequality in the original indices. Thus proving monotonicity.

The proposed calibration map offers an intuitive way to parameterize the calibration transformation in the logit space while preserving the original order of predictions. Moreover, it provides interpretability, as the effects of the transformation can be directly understood through the applied scaling and bias in the sorted logit domain. By sorting the logits we could connect the parameters to ranking instead of the classes which improves the sample efficient in the cases of many classes or small calibration set.

We employ a constrained optimization approach to implement the proposed calibration map in Theorem 1 and learn the parameters w and b. Specifically, we enforce monotonicity by requiring the consecutive differences of w and b to be positive. Formally, we define the difference vectors $d_w \in \mathbb{R}^{m-1}$ and $d_b \in \mathbb{R}^{m-1}$ for the weight vector $w \in \mathbb{R}^m$ and the bias vector $b \in \mathbb{R}^m$ as :

$$\forall i \in \{1 \dots m-1\} \quad d_{w,i} = w_{i+1} - w_i, \quad d_{b,i} = b_{i+1} - b_i$$

Enforcing the d_w and d_b to be strictly positive in the optimization process guarantees that both w and b are non-decreasing, thereby preserving the order of the logits after transformation. The parameters are learned by minimizing the negative log-likelihood (NLL) loss between the softmax of the transformed logits $\hat{Z}$ and the true labels. Formally, we minimize the following loss to learn the w and b parameters:

$$\min_{w,b} \quad \mathcal{L}_{\text{NLL}}\big(\text{softmax}(S^{-1}(S(Z) \odot w + b)), y\big)$$

subject to:

$$\forall i \in \{1 \dots m-1\}, \quad d_{w,i} \geq 0, \quad d_{b,i} \geq 0$$

Where the gradient of parameter w and b for each sample are

$$\frac{\partial \mathcal{L}}{\partial w} = S(Z) \odot (\hat{p}_c - y), \quad \frac{\partial \mathcal{L}}{\partial b} = \hat{p}_c - y$$

where $\hat{p}_c = \text{softmax}(S^{-1}(S(Z) \odot w + b))$, the probability output after the softmax operator.

We choose not to apply any additional normalization to the logits during post-hoc calibration, as it is standard practice in modern deep neural network training and inference to normalize inputs using the mean and standard deviation computed from the training set. This preprocessing step regularizes the scale and distribution of the logits. Empirically, we observe that for each logit rank, the resulting distribution is approximately normal.

3.2 MCCT-I

We also propose an alternative calibration map as a variant of Theorem 1, where the primary difference is that w is an inverse scaling factor in the transformation. This calibration map is defined as:

Theorem 2 *For transformation $f : \mathbb{R}^m \to \mathbb{R}^m$, defined as*

$$f(Z) = S^{-1}\left(\frac{S(Z)}{w} + b\right),$$

is monotonic if:
- $w \in \mathbb{R}^m$ vector satisfy $w_1 \geq w_2 \geq \ldots \geq w_m$ and $w_i > 0$
- $b \in \mathbb{R}^m$ vector satisfy $b_1 \leq b_2 \leq \ldots \leq b_m$
- S^{-1} is the inverse sorting operator that maps the sorted-and-transformed logit vector back to the original logit Z indices.

Theorem 2 is a variant of Theorem 1, with the only difference being that w acts as an inverse scaling factor. It is straightforward to see that the transformation map described in Theorem 2 is mathematically equivalent to that of Theorem 1, differing only in how the constraint on w is formulated. The key distinction between these two calibration maps lies in the optimization process, where the implementation remains similar, but the parameters have different gradients. The implementation of Theorem 2 is defined as:

$$\min_{w,b} \quad \mathcal{L}_{\text{NLL}} \left(\text{softmax} \left(S^{-1} \left(\frac{S(Z)}{w} + b \right) \right), y \right)$$

subject to:

$$\forall i \in \{1 \ldots m-1\}, \quad g_{w,i} \geq 0, \quad g_{b,i} \geq 0$$
$$\forall j \in \{1 \ldots m\}, w_j > 0$$

Where the constraint terms vector g_w and g_b are defined as:

$$g_{w,i} = w_i - w_{i+1}, \quad g_{b,i} = b_{i+1} - b_i, \quad \forall i \in \{1 \ldots m-1\}$$

The gradient for parameter w and b for each sample are

$$\frac{\partial \mathcal{L}}{\partial w} = -\frac{S(Z)}{w^2} \odot (\hat{p}_c - y), \quad \frac{\partial \mathcal{L}}{\partial b} = \hat{p}_c - y$$

The gradient for the calibration map in Theorem 2 is scaled by $\frac{1}{w^2}$ in comparison with Theorem 1, which acts as an implicit regularization discouraging large values of w. Empirically, this implicit regularization has benefited datasets with many classes, such as ImageNet-1K, where overconfidence is less pronounced compared to datasets with fewer classes. In such cases, enforcing smaller w through implicit regularization improves calibration performance.

3.3 SCALABILITY IN LARGE NUMBER OF CLASS

For multiclass calibration, a significant challenge is handling many classes. The proposed calibration methods, MCCT and MCCT-I, scale well to problems with many classes since the number of parameters grows linearly with the number of classes. Nonetheless, it is important to consider extreme cases with an even more significant number of classes.

A reasonable assumption about the probabilistic output of a classifier is that, as the number of classes increases, the output remains constrained to the probability simplex Δ^{m-1}, where all class probabilities sum to one. In such cases, the lower-ranked probabilities, and consequently their corresponding logits, may become negligible and contain little helpful information for calibration. Thus, for extremely large-scale classification tasks, it may be beneficial to discard the lowest-ranked logits to reduce computational complexity while preserving calibration effectiveness.

For MCCT and MCCT-I, discarding the lowest-ranked logits can be implemented by truncating $S(Z)$ and the corresponding label y only to consider the top k logits, effectively reducing the number of parameters to k for w and b. Specifically, we define the reduced sorted logit vector as $S(Z)^k = S(Z)_i \quad \forall i \in \{m-k, \ldots, m\}$, with the corresponding reduced label vector y^k indexed accordingly.

During inference, for the lower ranked $S(Z)_i \quad \forall i \in \{1, \ldots, m-k\}$ logits, the transformation is carried out by using w_1 and b_1, the first elements of the weight and bias vectors, respectively, to scale the non-top-k logits.

4 RELATED WORK

Numerous post-hoc calibration methods have been explored in the literature. As discussed in the previous section, a key distinction among these methods is whether they enforce monotonicity. Non-monotonic parametric methods include vector scaling, matrix scaling [Guo et al., 2017], and Dirichlet scaling [Kull et al., 2019]. However, since they are heavily parametrized in an unconstrained manner, these methods are not data-efficient, and their lack of monotonicity becomes more severe as the calibration set size decreases [Rahimi et al., 2020].

Monotonic post-hoc calibration. Temperature scaling [Guo et al., 2017, Platt et al., 1999, Hinton, 2015] is the simplest post-hoc calibration method that enforces monotonicity. It uses a single parameter to scale the output logits for calibration. Later work, such as Ensemble Temperature Scaling [Zhang et al., 2020], extends temperature scaling by incorporating three additional weight parameters to improve its expressivity. More recently, there have been attempts to perform monotonic calibration using more parameterized approaches by learning a neural network for calibration maps. Intra-Order Preserving Calibration [Rahimi et al., 2020] employs a monotonic neural network to transform individual logits for improved calibration. Additionally, parameterized temperature scaling utilizes a nonlinear neural network to select different temperature values for each sample, assuming that different samples have different levels of over or underconfidence and need different temperature parameters for calibration.

Non-parametric post-hoc calibration. Apart from parametric approaches, many non-parametric methods have also been proposed, with binning-based approaches being the most popular category. These methods divide confidence outputs into mutually exclusive bins and assign a calibrated score to each bin. A notable example is histogram binning [Zadrozny and Elkan, 2001]. Several variants of binning have also been introduced, such as Bayesian Binning into Quantiles (BBQ) [Naeini et al., 2015], Platt Binner Marginal Calibrator [Kumar et al., 2019], and other closely related non-parametric methods like isotonic regression [Zadrozny and Elkan, 2002] and spline calibration [Gupta et al., 2021]. Similar to non-monotonic parametric methods, most non-parametric calibration methods suffer from data inefficiency and increased monotonicity violations as the size of the calibration set decreases.

Training calibrated models. Besides post-hoc calibration methods, several approaches focus on training models that are well-calibrated. Techniques such as label smoothing [Müller et al., 2020], focal loss [Mukhoti et al., 2020], dual focal loss [Tao et al., 2023], logit normalization [Wei et al., 2022] have been shown to improve the calibration of deep neural networks.

Calibration Metric. The expected calibration error (ECE) [Naeini et al., 2015] is the most commonly used calibration metric in the literature. Its popularity stems from its data efficiency and close relationship with reliability diagrams, making it both interpretable and easily visualized. However, one of its key weaknesses is its dependence on the number of bins b, which introduces sensitivity to the choice of this hyperparameter. Various variants of ECE have been proposed to mitigate this issue [Kumar et al., 2019, Nixon et al., 2020, Chidambaram et al., 2024], but still rely on alternative assumptions or additional hyperparameters. In our evaluation, in addition to the standard ECE, we use a variant of the adaptive ECE proposed by Nixon et al. [2020], which employs equal-sized binning rather than equal-interval binning based on the top-label confidence. The error is computed as the sum of the un-normalized differences between accuracy and average confidence within each bin. Beyond confidence calibration, other notions of calibration error have been explored, such as class-wise ECE. However, these approaches tend to be less data-efficient and often exhibit inconsistencies with other calibration metrics [Tomani et al., 2022, Nixon et al., 2020]. ECE-KDE [Zhang et al., 2020] is another variant of ECE that replaces histogram-based binning with a non-parametric density estimator for calibration error estimation, implemented using kernel density estimation (KDE). Where the bandwidth is set as $h = 1.06\hat{\sigma} n_e^{-1/5}$, where $\hat{\sigma}$ is the standard deviation of the confidence scores, following a popular bandwidth selection method [Scott, 2015].

5 EXPERIMENTS

We evaluates the proposed calibration methods, MCCT and MCCT-I, on diverse deep-learning models across different datasets and compared their performance with the existing state-of-the-art post-hoc calibration methods.

5.1 DATASETS AND BASE CLASSIFIER

We perform our experiments on three computer vision datasets: CIFAR-10 [Krizhevsky et al., 2009], CIFAR-100 [Krizhevsky et al., 2009], and ImageNet-1K [Deng et al., 2009] with the number of class ranging from 10 to 1000. For all datasets, we assess calibration performance across different neural network architectures. For CIFAR-10/100, we use a calibration set of 5,000 samples and a test set of 10,000 samples. For ImageNet-1K, we use a validation set of 25,000 samples and a test set of 25,000 samples.

As for neural network classifier model, for CIFAR-10, we use LeNet-5 [Lecun et al., 1998], DenseNet-40 [Huang et al., 2017], and ResNet-110 [He et al., 2016]. For CIFAR-100, we use Wide-ResNet-32 [Zagoruyko, 2016] and ResNet-110 [He et al., 2016]. For ImageNet-1K, we use ResNet-50/152 [He et al., 2016], ViT-B/16 [Dosovitskiy et al., 2021], Inception-V3 [Szegedy et al., 2016], and Wide-ResNet-50-2 [Zagoruyko, 2016].

5.2 BASELINES

To compare the calibration performance of MCCT and MCCT-I, we select the following seven baseline methods: Temperature Scaling (TS) [Guo et al., 2017], Vector Scaling (VS) [Guo et al., 2017], Histogram Binning (HB) [Zadrozny and Elkan, 2001], Ensemble Temperature Scaling with Negative Log-Likelihood Loss (ETS) [Zhang et al., 2020], Ensemble Temperature Scaling with Mean Squared Error Loss (ETS-MSE) [Zhang et al., 2020], Parameterized Temperature Scaling (PTS) [Tomani et al., 2022], Intra-Order Preserving Calibration (DIAG) [Rahimi et al., 2020].

We also include the original uncalibrated classifier result (Uncalibrated) for comparison. Among all the baselines, the VS and HB are not monotonic and do not preserve the accuracy or the prediction ranking in the calibration process.

5.3 RESULTS

Table 1 summarizes the results of our proposed calibration methods compared to the baselines in terms of ECE and average ranking, calculated across all datasets and base classifiers. Overall, MCCT and MCCT-I achieve the highest rankings among all methods, with MCCT-I slightly outperforming MCCT. Additionally, MCCT and MCCT-I achieve the highest number of best-performing results, ranking first in 7 out of 11 cases. Even in rare instances where MCCT or MCCT-I is not the best-performing calibration method, they remain competitive, typically ranking second and not far behind in terms of ECE. The performance of MCCT and MCCT-I is similar on datasets with fewer classes. How-

Model	Uncalibrated	TS	VS	HB	ETS	ETS MSE	PTS	DIAG	MCCT	MCCT-I
CIFAR10										
Lenet5	5.18	1.67	1.28	4.67	1.61	1.49	11.48	1.42	**1.19**	**1.19**
Densenet 40	5.50	**0.95**	0.98	1.61	1.17	1.09	3.96	1.05	1.06	1.06
W-Resnet 32	4.51	0.78	0.85	0.73	0.68	0.69	3.49	0.73	**0.54**	**0.54**
Resnet 110	4.75	1.13	1.30	1.13	0.82	**0.58**	4.26	0.79	0.76	0.76
CIFAR100										
W-Resnet 32	18.78	1.47	1.58	8.43	1.32	1.25	5.82	2.00	**1.06**	**1.11**
Resnet 110	18.48	2.38	2.54	8.53	1.48	1.68	7.15	1.50	**1.28**	**1.32**
ImageNet										
Resnet 50	3.66	2.25	1.77	7.32	1.39	1.35	1.09	1.81	0.99	**0.95**
Resnet 152	4.78	2.02	1.86	7.73	1.10	**1.07**	1.22	1.69	1.28	1.18
ViT-B/16	5.86	3.72	4.23	7.04	3.36	3.10	2.50	**0.67**	1.79	1.79
Inception v3	18.27	5.41	27.93	7.61	2.94	2.52	1.62	0.98	0.72	**0.61**
W-ResNet 50	5.43	2.95	3.11	7.70	1.77	1.56	1.16	1.14	1.13	**1.08**
Average rank	9.45	6.32	6.64	8.55	4.82	4.00	6.55	4.32	2.32	**2.05**

Table 1: ECE (with the number of bins set to 15, smaller is better) on benchmark datasets and models with different calibration methods. Calibration methods VS and HB are not monotonic and suffer from changes in accuracy and prediction rankings. The reported numbers are averaged over 10 runs and are in the scale of 10^{-2}.

ever, as the number of classes increases, MCCT-I begins to outperform MCCT due to the implicit regularization effect introduced by the different gradients on parameter w, as explained in the previous section. Table 2 presents the results using the ECE-KDE metric [Zhang et al., 2020], which remain mostly consistent with and correlate closely to ECE. Table 3 presents the results for Equal Bin ECE (EQ-BIN ECE), a variant of ECE that, instead of dividing samples by grouping the top-confidence predictions into bins of equal confidence intervals. The results are consistent with those of ECE and ECE-KDE, with MCCT and MCCT-I remaining the top-performing calibration methods by a significant margin.

Similar to TS, MCCT, and MCCT-I are interpretable since the calibration map is directly parameterized by intuitive parameters. By simply examining the w and b parameters, we can assess the degree of overconfidence and underconfidence in the model, as well as how these effects vary across probability rankings. In contrast, neural network-based approaches such as PTS and DIAG lack this level of interpretability.

Another observation from our experiments is that MCCT and MCCT-I are not sensitive to random seeds or initialization. In our experiments, the random seed has no impact on the results for MCCT and MCCT-I. In contrast, neural network-based methods such as PTS and DIAG exhibit high variance in ECE when trained with different random seeds.

For initialization, MCCT sets the weight vector w by assigning it m equally spaced values in the interval $[0, 1]$, and b vector is initialized to 0. For MCCT-I, the w vector is initialized to ones, and b is initialized to zeros. Our experiments found that neither MCCT nor MCCT-I are sensitive to parameter initialization when optimized by sequential least squares programming (SLSQP) optimizer [Kraft, 1988].

5.4 DATA EFFICIENCY

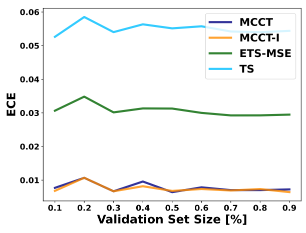
A crucial aspect of a calibration method is its data efficiency, i.e., how well the post-hoc calibration method performs when the calibration set is small. In many real-world applications, setting aside a large unseen validation set is not feasible. As shown in Figure 1a, a smaller calibration set increases monotonicity violations for non-monotonic methods. For monotonic post-hoc calibration methods, it is essential to maintain calibration performance even when the calibration set is limited.

Calibration methods with fewer parameters, such as TS and ETS, have a notable advantage in data efficiency since they are less prone to overfitting on small calibration sets. To assess the data efficiency of MCCT and MCCT-I, we conduct experiments on ImageNet and CIFAR-100 using Inception-NetV3 and ResNet110 as base models. We fit MCCT and MCCT-I on varying calibration set sizes, ranging from 10% to 90% of the original size. We compare their performance with data-efficient methods such as TS and ETS. The results are presented in Figure 2.

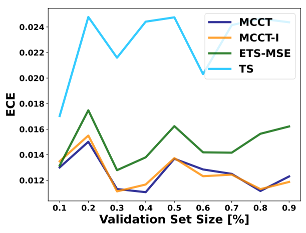
As shown in Figure 2, MCCT and MCCT-I exhibit strong data efficiency, with no significant increase in ECE for ImageNet even when the calibration set size is reduced to as low as 10% of its original size. For CIFAR-100, the ECE fluctuates by less than 30% across different calibration set sizes, following a similar trend observed in TS and ETS. Moreover, MCCT and MCCT-I consistently outperform TS and ETS-MSE across all calibration set sizes. A potential

Model	Uncalibrated	TS	VS	HB	ETS	ETS MSE	PTS	DIAG	MCCT	MCCT-I
CIFAR10										
Lenet5	4.66	1.26	1.19	3.68	1.22	1.22	11.01	1.10	**0.88**	**0.88**
Densenet 40	5.01	**1.14**	1.25	2.16	1.31	1.35	3.58	1.31	1.30	1.30
W-Resnet 32	4.06	1.16	1.28	1.44	1.16	1.17	3.26	1.35	**1.04**	**1.04**
Resnet 110	4.34	1.24	1.41	1.66	1.25	**1.08**	3.94	1.23	**1.11**	**1.11**
CIFAR100										
W-Resnet 32	18.16	**1.09**	1.17	7.75	1.25	1.33	5.63	2.11	1.13	**1.12**
Resnet 110	17.82	1.71	1.91	8.94	1.45	1.47	6.90	1.69	**1.17**	**1.18**
ImageNet										
Resnet 50	3.02	1.85	1.29	7.99	1.44	1.57	**1.00**	2.05	1.07	**1.00**
Resnet 152	4.24	1.61	1.40	7.40	1.32	1.38	**1.13**	1.95	1.23	1.21
ViT-B/16	5.78	3.79	4.09	6.68	3.25	2.97	2.40	**0.92**	1.74	1.75
Inception v3	17.71	5.05	27.60	8.30	2.88	2.33	1.39	1.17	**0.67**	**0.68**
W-ResNet 50	4.83	2.42	2.57	7.46	1.78	1.79	**0.94**	1.34	1.07	1.02
Average rank	9.45	5.32	6.00	8.90	4.95	5.05	5.86	5.05	2.27	**2.14**

Table 2: ECE-KDE on benchmark datasets and models with different calibration methods. Calibration methods VS and HB are not monotonic and suffer from changes in accuracy and prediction rankings. The reported numbers are averaged over 10 runs and are in the scale of 10^{-2}.



(a) ImageNet



(b) CIFAR-100

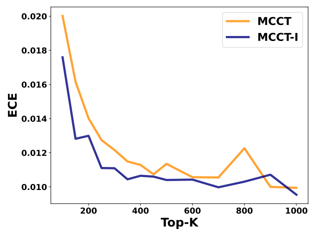
Figure 2: The impact of calibration set size on ECE for MCCT and MCCT-I compared to data-efficient methods TS and ETS. (a) ImageNet with InceptionNetV3. (b) CIFAR-100 with ResNet110.

reason for MCCT and MCCT-I performing well in low-data regimes is that the constraints on the transformation parameters act as a form of regularization, restricting the parameter search space and improving generalization.

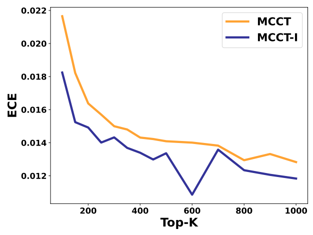
5.5 IMAGENET TOP-K LOGITS

As discussed in Section 3.3, scalability can become a concern when handling an extremely large number of classes. In our experiments, both MCCT and MCCT-I complete training within a reasonable time on a single-threaded CPU. For CIFAR-10 and CIFAR-100, training time is negligible requiring less than 2 seconds and 6 seconds, respectively. For larger-scale datasets such as ImageNet, both methods finish training in under 55 minutes. While this training time is acceptable and does not render the method impractical, it may become a limitation as the number of classes increases.

To evaluate the top-k logits selection method introduced in Section 3.3, which aims to improve efficiency on datasets with a large number of classes, we conduct experiments on ImageNet-1K using ResNet-50 and ResNet-152 as base models. We explore a wide range of top-k parameter values, varying k from 100 to 1000. Our goal is to investigate the impact of top-k selection on ECE, determining how many logits can be trimmed before significantly affecting calibration performance. Additionally, we analyze the computational efficiency gains achieved by reducing the number of selected logits.



(a) ResNet50



(b) ResNet152

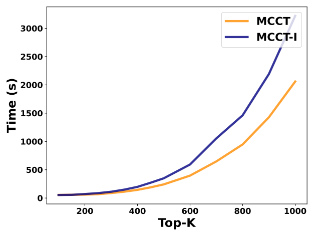
Figure 3: The impact of top-k logit selection on ECE. The x-axis represents the top-k value, and the y-axis represents ECE. (a) ImageNet with ResNet-50, (b) ImageNet with ResNet-152.

Figure 3 presents the plot between ECE and the top-k logit. For ImageNet-1K, both MCCT and MCCT-I maintain stable performance in terms of ECE until the top-k parameter is reduced below 300. When the top-k falls below 200, performance begins to deteriorate rapidly. Consistent with our previous results, MCCT-I generally outperforms MCCT in datasets with many classes.

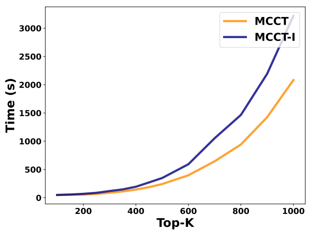
Model	Uncalibrated	TS	VS	HB	ETS	ETS MSE	PTS	DIAG	MCCT	MCCT-I
					CIFAR10					
Lenet5	5.04	1.48	1.03	4.19	1.31	1.30	11.46	1.09	**0.82**	**0.82**
Densenet40	5.49	**0.92**	0.93	2.67	1.19	1.32	3.95	1.10	1.09	1.09
W-Resnet 32	4.48	0.70	0.56	2.84	0.69	0.71	3.40	0.52	**0.42**	**0.42**
Resnet 110	4.75	0.93	1.13	2.29	0.87	**0.42**	4.26	0.92	0.79	0.79
					CIFAR100					
W-Resnet 32	18.78	1.17	1.33	8.21	1.35	1.45	5.79	1.86	**0.96**	**0.97**
Resnet 110	18.48	2.09	2.25	9.65	1.36	1.47	7.14	1.42	**1.07**	**1.06**
					ImageNet					
Resnet 50	8.65	5.60	4.52	22.41	3.95	4.17	2.88	4.68	**2.56**	**2.43**
Resnet 152	11.84	5.20	4.54	21.69	3.95	4.04	**3.10**	4.43	3.37	3.20
ViT-B/16	14.70	10.51	10.57	18.43	8.24	7.54	6.32	**2.25**	4.71	4.72
Inception v3	45.68	13.44	69.82	24.41	8.24	6.71	4.07	2.79	**2.23**	2.40
W-ResNet 50	13.45	7.23	7.77	22.70	5.02	4.82	2.97	3.03	3.05	**2.92**
Average rank	9.45	6.00	6.00	8.91	5.00	5.09	6.09	4.36	2.18	**1.91**

Table 3: EQ-BIN ECE on benchmark datasets and models with different calibration methods. Calibration methods VS and HB are not monotonic and suffer from changes in accuracy and prediction rankings. The reported numbers are averaged over 10 runs and are in the scale of 10^{-1}.

This result confirms our previous assumption that lower-ranked logits contain little helpful information for calibration. For datasets with a large number of classes, the probability distribution follows a probability simplex with a degree of freedom of $m - 1$, and due to the NLL loss used in neural network models, classifiers tend to produce a long-tailed top-1 probability distribution. As a result, lower-ranking probability outputs usually have tiny values, which makes the corresponding logits contribute no information for calibration. Thus, our approach, which involves sorting logits and discarding the lower-ranking logits to a certain extent, does not significantly degrade calibration performance.



(a) ResNet50



(b) ResNet152

Figure 4: The impact of top-k logit selection on training time. The x-axis represents the top-k value, and the y-axis represents training time in seconds. (a) ImageNet with ResNet-50, (b) ImageNet with ResNet-152.

Another interesting phenomenon we observed during the top-k experiment relates to the interpretability aspect of MCCT and MCCT-I. When examining the fitted parameters w and b of MCCT and MCCT-I across the full 1000 class ImageNet, we find that there exists a rank beyond which the parameters stabilize and cease to change. Notably, this stabilization point aligns closely with the elbow point observed in Figure 3. This observation provides insight into when the sorted logits begin to resemble noise and no longer contribute meaningful information, thereby shedding light on model behavior in the many-class setting.

By reducing the number of top-k logits, we observe a significant decrease in training time for both MCCT and MCCT-I. As shown in Figure 4, the training time is reduced to under 10 minutes when top-k is set to 600 and further drops to 3 minutes and 14 seconds when top-k is reduced below 400. In practical applications, we recommend using all logits for training MCCT and MCCT-I on datasets with fewer than 500 classes. However, for datasets with more than 500 classes, a top-k scheme that retains the top 400 logits can be used for both training and inference to improve efficiency without significantly affecting performance.

6 CONCLUSION

In this paper, we introduce a family of novel calibration methods for monotonic post-hoc calibration. By proposing a monotonic calibration map that is parameterized linearly with respect to the number of classes and has an intuitive interpretation of the parameters, the introduced calibration method is both expressive and interpretable. Furthermore, by formulating the implementation as a constrained optimization problem, the proposed method achieves state-of-the-art performance across various datasets and neural network models, outperforming previous post-hoc calibration methods such as ETS, PTS and DIAG while being data-efficient and robust in the sense of prediction and ranking preservation.

References

Amr Alexandari, Anshul Kundaje, and Avanti Shrikumar. Maximum likelihood with bias-corrected calibration is hard-to-beat at label shift adaptation. In *International Conference on Machine Learning*, pages 222–232. PMLR, 2020.

Maarten Bieshaar, Stefan Zernetsch, Andreas Hubert, Bernhard Sick, and Konrad Doll. Cooperative starting movement detection of cyclists using convolutional neural networks and a boosted stacking ensemble. *IEEE Transactions on Intelligent Vehicles*, 3(4):534–544, 2018.

Muthu Chidambaram, Holden Lee, Colin McSwiggen, and Semon Rezchikov. How flawed is ece? an analysis via logit smoothing. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

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, 2009. doi: 10.1109/CVPR.2009.5206848.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR*, 2021.

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*, pages 1321–1330. PMLR, July 2017. URL `https://proceedings.mlr.press/v70/guo17a.html`. ISSN: 2640-3498.

Kartik Gupta, Amir Rahimi, Thalaiyasingam Ajanthan, Thomas Mensink, Cristian Sminchisescu, and Richard Hartley. Calibration of Neural Networks using Splines, December 2021. URL `http://arxiv.org/abs/2006.12800`. arXiv:2006.12800 [cs, stat].

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.

Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=Hkg4TI9xl`.

Geoffrey Hinton. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. Densely Connected Convolutional Networks . In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, Los Alamitos, CA, USA, July 2017. IEEE Computer Society. doi: 10.1109/CVPR.2017.243. URL `https://doi.ieeecomputersociety.org/10.1109/CVPR.2017.243`.

Dieter Kraft. A software package for sequential quadratic programming. *Forschungsbericht- Deutsche Forschungs- und Versuchsanstalt fur Luft- und Raumfahrt*, 1988.

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

Meelis Kull, Miquel Perello Nieto, Markus Kängsepp, Telmo Silva Filho, Hao Song, and Peter Flach. Beyond temperature scaling: Obtaining well-calibrated multiclass probabilities with Dirichlet calibration. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.

Ananya Kumar, Percy S Liang, and Tengyu Ma. Verified Uncertainty Calibration. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL `https://papers.nips.cc/paper_files/paper/2019/hash/f8c0c968632845cd133308b1a494967f-Abstract.html`.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=ryiAv2xAZ`.

Jishnu Mukhoti, Viveka Kulharia, Amartya Sanyal, Stuart Golodetz, Philip Torr, and Puneet Dokania. Calibrating deep neural networks using focal loss. *Advances in Neural Information Processing Systems*, 33:15288–15299, 2020.

Rafael Müller, Simon Kornblith, and Geoffrey Hinton. When Does Label Smoothing Help?, June 2020. URL `http://arxiv.org/abs/1906.02629`. arXiv:1906.02629 [cs, stat].

Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29, 2015.

Jeremy Nixon, Mike Dusenberry, Ghassen Jerfel, Timothy Nguyen, Jeremiah Liu, Linchuan Zhang, and Dustin Tran. Measuring Calibration in Deep Learning, August 2020. URL `http://arxiv.org/abs/1904.01685`. arXiv:1904.01685 [cs, stat].

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

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.

Amir Rahimi, Amirreza Shaban, Ching-An Cheng, Richard Hartley, and Byron Boots. Intra order-preserving functions for calibration of multi-class neural networks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Amir Rahimi, Thomas Mensink, Kartik Gupta, Thalaiyasingam Ajanthan, Cristian Sminchisescu, and Richard Hartley. Post-hoc Calibration of Neural Networks by g-Layers, February 2022. URL `http://arxiv.org/abs/2006.12807`. arXiv:2006.12807 [cs, stat].

Marco Saerens, Patrice Latinne, and Christine Decaestecker. Adjusting the Outputs of a Classifier to New a Priori Probabilities: A Simple Procedure. *Neural Computation*, 14(1):21–41, January 2002. ISSN 0899-7667. doi: 10.1162/089976602753284446. URL `https://doi.org/10.1162/089976602753284446`.

David W Scott. *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons, 2015.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, 2016. doi: 10.1109/CVPR.2016.308.

Xingyu Tan, Xiaoyang Wang, Qing Liu, Xiwei Xu, Xin Yuan, and Wenjie Zhang. Paths-over-graph: Knowledge graph empowered large language model reasoning. In *Proceedings of the ACM on Web Conference 2025*, pages 3505–3522, 2025.

Fengchun Tang, Traci J Hess, Joseph S Valacich, and John T Sweeney. The effects of visualization and interactivity on calibration in financial decision-making. *Behavioral Research in Accounting*, 26(1):25–58, 2014.

Linwei Tao, Minjing Dong, and Chang Xu. Dual focal loss for calibration. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

Christian Tomani, Daniel Cremers, and Florian Buettner. Parameterized Temperature Scaling for Boosting the Expressive Power in Post-Hoc Uncertainty Calibration. In *European Conference on Computer Vision – ECCV 2022*, pages 555–569, Cham, 2022.

Ben Van Calster, David J McLernon, Maarten Van Smeden, Laure Wynants, Ewout W Steyerberg, Topic Group 'Evaluating diagnostic tests, and prediction models' of the STRATOS initiative Bossuyt Patrick Collins Gary S. Macaskill Petra McLernon David J. Moons Karel GM Steyerberg Ewout W. Van Calster Ben van Smeden Maarten Vickers Andrew J. Calibration: the achilles heel of predictive analytics. *BMC medicine*, 17(1):230, 2019.

Hongxin Wei, Renchunzi Xie, Hao Cheng, Lei Feng, Bo An, and Yixuan Li. Mitigating neural network overconfidence with logit normalization. In *International conference on machine learning*, pages 23631–23644. PMLR, 2022.

Bianca Zadrozny and Charles Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, page 609–616, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1558607781.

Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 694–699, 2002.

Sergey Zagoruyko. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Jize Zhang, Bhavya Kailkhura, and T. Yong-Jin Han. Mix-n-Match : Ensemble and Compositional Methods for Uncertainty Calibration in Deep Learning. In *Proceedings of the 37th International Conference on Machine Learning*, pages 11117–11128. PMLR, November 2020. URL `https://proceedings.mlr.press/v119/zhang20k.html`. ISSN: 2640-3498.

Yunrui Zhang, Gustavo Batista, and Salii S Kanhere. Label shift estimation with incremental prior update. In *Proceedings of the 2025 SIAM International Conference on Data Mining (SDM)*, pages 134–142. SIAM, 2025a.

Yunrui Zhang, Gustavo Batista, and Salil S Kanhere. Revisit time series classification benchmark: The impact of temporal information for classification. *arXiv preprint arXiv:2503.20264*, 2025b.

A SUPPLEMENTARY MATERIAL

A.1 GRADIENT OF MCCT AND MCCT-I PARAMETERS

A.1.1 MCCT

First, let's revisit the calibration map defined in Theorem 1 we have:

$$f(Z) = S(Z) \odot W + b \tag{1}$$

For the purpose of calculating the gradient, we could ignore the constraints. To simplify the notation, instead of reverse sorting the transformed logits, here we remove the S^{-1} the inverse sorting operation, and instead of calculating the loss between the S(Z) and the permuted one-hot encoded label Y that each vector Y is permuted in accordance to the sorting index of the corresponding logit vector Z.

$$\hat{p}_j = \frac{e^{f(Z)_j}}{\sum_{i=1}^{n} e^{f(Z)_i}} \tag{2}$$

The transformed logits are then go through softmax operation to produce the probability estimation P.

$$L = -\sum_{j=1}^{n} Y_j \log \hat{p}_j \tag{3}$$

And the loss L is calculated by taking the Negative log-likelihood loss between the transform probability P and the sorted one-hot encoded logit Y.

$$\frac{\partial L}{\partial f(Z)} = \hat{p} - Y \tag{4}$$

Equation 4 shows the gradient vector of L w.r.t. $f(Z)$, this is the well known derivative of the Softmax Function combine with the Categorical Cross-Entropy Loss.

$$\frac{\partial f(Z)}{\partial W} = S(Z) \tag{5}$$

Equation 5 describes the gradient of $f(Z)$ w.r.t. W.

$$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial f(Z)_j} \cdot \frac{\partial f(Z)}{\partial W} = S(Z) \odot (\hat{p} - Y) \tag{6}$$

Then by using the chain rule we have the gradient vector of L w.r.t. W described in Equation 6

$$\frac{\partial L}{\partial B} = \frac{\partial L}{\partial f(Z)} \cdot \frac{\partial f(Z)}{\partial B} = (\hat{p} - Y) \tag{7}$$

From equation 1 we know that the $\frac{\partial f(Z)}{\partial B}$ is simply 1. Then by using the chain rule we have the gradient vector of L w.r.t. B described as in Equation 7

A.1.2 MCCT-I

For the calibration map described in the Theorem 2 we have

$$f(Z) = \frac{S(Z)}{W} + b \tag{8}$$

We can see that the $\frac{\partial L}{\partial B}$ is the same as the first calibration map.

$$\frac{\partial f(Z)}{\partial W} = -\frac{S(Z)}{W^2} \tag{9}$$

Equation 9 describes the gradient of $f(Z)$ w.r.t. W.

$$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial f(Z)} \cdot \frac{\partial f(Z)}{\partial W} = -\frac{S(Z)}{W^2} \odot (\hat{p} - Y), \tag{10}$$

Then by using the chain rule we have the gradient vector of L w.r.t. W described in Equation 10

A.2 ADDITIONAL INFORMATION FOR BASE CLASSIFIER

For CIFAR-10 and CIFAR-100, we use the logits provided by previous work [Kull et al., 2019], available at `https://github.com/markus93/NN_calibration`. For ImageNet-1K, we use the PyTorch pre-trained models [Paszke et al., 2019] from `https://pytorch.org/vision/main/models.html` and perform inference on the original ImageNet-1K test set, which is randomly divided 50-50 into a calibration set and a testing set.