# NETWORK CALIBRATION BY WEIGHT SCALING

#### Anonymous authors

Paper under double-blind review

#### Abstract

Calibrating neural networks is crucial in applications where the decision making depends on the predicted probabilities. Modern neural networks are not well calibrated and they tend to overestimate probabilities when compared to the expected accuracy. This results in a misleading reliability that corrupts our decision policy. We define a weight scaling calibration method that computes a convex combination of the network output class distribution and the uniform distribution. The weights controls the confidence of the calibrated prediction. Since the goal of calibration is making the confidence prediction more accurate, the most suitable weight is found as a function of the given confidence. We derive an optimization method that is based on a closed form solution for the optimal weight scaling in each bin of a discretized value of the prediction confidence. We report extensive experiments on a variety of image datasets and network architectures. This approach achieves state-of-the-art calibration with a guarantee that the classification accuracy is not altered.

#### **1** INTRODUCTION

Probabilistic machine learning algorithms output confidence scores along with their predictions. Ideally, these scores should match the true correctness probability. However, modern deep learning models still fall short in giving useful estimates of their predictive uncertainty. The lack of connection between the model's predicted probabilities and the confidence of model's predictions constitutes a key obstacle to the application of neural network models to real-world problems, such as decision-making systems. Quantifying uncertainty is especially critical in real-world tasks such as automatic medical diagnosis (Crowson et al., 2016; Jiang et al., 2011; Raghu et al., 2019) and perception tasks in autonomous driving (Amodei et al., 2016). A classifier is said to be calibrated if the probability values it associates with the class labels match the true probabilities of the correct class assignments. Modern neural networks have been shown to be more overconfident in their predictions than their predecessors even though their generalization accuracy is higher, partly due to the fact that they can overfit on the negative log-likelihood loss without overfitting on the classification error (Guo et al., 2017; Lakshminarayanan et al., 2017; Hein et al., 2019).

Various confidence calibration methods have recently been proposed in the field of deep learning to overcome the over-confidence issue. Calibration strategies can be divided into two main types. The first is a model calibration while training the model (e.g. (Kumar et al., 2019; Maddox et al., 2019; Kendall & Gal, 2017; Milios et al., 2018; Mukhoti et al., 2020)). The second approach performs calibration as a post processing step using an already trained model. Post-hoc scaling approaches to calibration (e.g. Platt scaling (Platt et al., 1999), isotonic regression (Zadrozny & Elkan, 2002), and temperature scaling (Guo et al., 2017)) are widely used. They use hold-out validation data to learn a calibration map that transforms the model's predictions to be better calibrated. Temperature scaling is the simplest and most effective calibration method and is the current standard practical calibration method. Guo et al. (2017) investigated several scaling. They reported poor performance for vector/matrix scaling calibration. To avoid overfitting, Kull et al. (2019) suggested regularizing matrix scaling with an  $L_2$  loss on the calibration model weights. Gupta et al. (2021) built a calibration function by approximating the empirical cumulative distribution using a differentiable function via splines. Note that this calibration method can change the accuracy of the model.

Most of these calibration methods extend single parameter temperature scaling by making the selected temperature either a linear or a non-linear function of the logits that are computed for the class-set. For example, in vector scaling (Guo et al., 2017), each class has its own temperature scaling. In this study we take a different approach and propose an alternative to temperature scaling which we dub weight scaling. Weight scaling calibrates the network by computing a suitable convex combination of the original class distribution and the uniform distribution. Since the goal of calibration is to make the confidence prediction more accurate, the suitable weight calibration is found as a function of the confidence (i.e. the probability of the class with the highest logit). We thus choose a suitable scaling weight to a given instance as a function of the confidence of the predicted class. To find the optimal confidence based scaling, we divide the unit interval into bins and compute the optimal weight scaling for the validation set instances whose estimated confidence fall into that bin.

We show that unlike temperature, vector and matrix scaling (Kull et al., 2019) and other recently proposed methods (e.g. Gupta et al. (2021)), we can obtain a closed form solution for the optimal calibration parameters. The proposed calibration procedure is very fast and robust. No hyper parameters need to be tuned. The learned calibration method is easy to implement and yields improved calibration results. The proposed calibration does not change the hard classification decision, which allows it to be applied on any trained network and guarantees to retain the original classification accuracy in all the tested cases. We evaluate our method against leading calibration approaches on various datasets and network architectures and show that it outperforms existing methods (that are more difficult to train and more complicated to implement) on improving the *expected calibration error* (ECE) (Naeini et al., 2015) calibration measure.

## 2 CALIBRATION PROBLEM FORMULATION

Let x be an input vector to a classification network with k classes. The output of the network is a vector of k values  $z_1, ..., z_k$ . Each of these values, which are also called *logits*, represents the score for one of the k possible classes. The logits' vector is transformed into a probabilities vector by a *softmax* layer:  $p(y = i|x) = \frac{\exp(z_i)}{\sum_j \exp(z_j)}$ . Although these values uphold the mathematical terms of probabilities, they do not represent any actual probabilities of the classes.

The predicted class for a sample x is calculated from the probabilities vector by  $\hat{y} = \arg \max_i p(y = i|x) = \arg \max_i z_i$  and the predicted *confidence* for this sample is defined by  $\hat{p} = p(y = \hat{y}|x)$ . The *accuracy* of the model is defined by the probability that the predicted class  $\hat{p}$  is correct. The network is said to be *calibrated* if for each sample the confidence is equal to the accuracy. For example, if we collect ten samples, each having an identical confidence score of 0.8, we then expect an 80% classification accuracy for the ten samples. Calibration can also be defined for each of the k classes separately. Class i is said to be calibrated in the network if the confidence of a sample from this class is equal to the accuracy of the class.

A popular metric used to measure model calibration is the ECE (Naeini et al., 2015), which is defined as the expected absolute difference between the model's confidence and its accuracy. Since we only have finite samples, the ECE cannot in practice be computed using this definition. Instead, we divide the interval [0, 1] into m equispaced bins, where the  $i^{\text{th}}$  bin is the interval  $\left(\frac{i-1}{m}, \frac{i}{m}\right]$ . Let  $B_i$  denote the set of samples with confidences  $\hat{p}$  belonging to the  $i^{\text{th}}$  bin. The accuracy  $A_i$  of this bin is computed as  $A_i = \frac{1}{|B_i|} \sum_{t \in B_i} \mathbb{1}(\hat{y}_t = y_t)$ , where  $\mathbb{1}$  is the indicator function, and  $\hat{y}_t$  and  $y_t$  are the predicted and ground-truth labels for the  $t^{\text{th}}$  sample.  $A_i$  is the relative number of correct predictions of instances that were assigned to  $B_i$  based on the confidence. Similarly, the confidence  $C_i$  of the  $i^{\text{th}}$  bin is computed as  $C_i = \frac{1}{|B_i|} \sum_{t \in B_i} \hat{p}_t$ , i.e.,  $C_i$  is the average confidence of all samples in the bin. The ECE can be approximated as the weighted average of the absolute difference between the accuracy and confidence of each bin:

$$ECE = \sum_{i=1}^{m} \frac{|B_i|}{n} |A_i - C_i|$$
(1)

where n is the number of samples in the validation set. Note that  $A_i > C_i$  means the network is under-confident at the  $i^{th}$  bin and  $C_i > A_i$  implies that the network is over-confident.

One disadvantage of ECE is its uniform bin width. For a well trained model, most of the samples lie within the highest confidence bins; hence, these bins dominate the value of the ECE. For this reason, we can consider another metric, AdaECE (Adaptive ECE), where bin sizes are calculated so as to

evenly distribute samples between bins (Nguyen & O'Connor, 2015):

$$AdaECE = \frac{1}{m} \sum_{i=1}^{m} |A_i - C_i|$$
(2)

such that each bin contains 1/m of the data points with similar confidence values.

The ECE method can also be used to determine the calibration of the prediction for each class separately (Kull et al., 2019; Vaicenavicius et al., 2019; Kumar et al., 2019). We can apply the same procedure described above to compute the ECE score for class j by considering for each sample x the probability p(y = j|x). Let  $B_{ij}$  denote the set of samples x that p(y = j|x) is in the  $i^{th}$  bin,  $A_{ij}$  the accuracy of this class in this bin  $A_{ij} = \frac{1}{|B_{ij}|} \sum_{t \in B_{ij}} 1_{\{y_t=j\}}$  and  $C_{ij} = \frac{1}{|B_{ij}|} \sum_{t \in B_{ij}} p(y_t = j|x_t)$  is the confidence. The classwise-ECE score for class j can be then calculated as:

$$ECE_{j} = \sum_{i=1}^{m} \frac{|B_{ij}|}{n_{j}} |A_{ij} - C_{ij}|.$$
(3)

We note in passing that even though the drawbacks of ECE have been pointed out and some improvements have been proposed (Kumar et al., 2019; Nixon et al., 2019; Gupta et al., 2021; Zhang et al., 2020), the ECE histogram approximation is still used as the standard calibration evaluation measure.

## 3 WEIGHT SCALING BASED ON THE PREDICTED CONFIDENCE

Temperature Scaling (TS), is a simple yet highly effective technique for calibrating prediction probabilities (Guo et al., 2017). It uses a single scalar parameter T > 0, where T is the temperature, to rescale logit scores before applying the softmax function to compute the class distribution. In overconfident models where T > 1, the recalibrated probabilities of the most likely class have a lower value than the original probabilities, and all the probabilities are more evenly distributed between 0 and 1. To get an optimal temperature T for a trained model, we can minimize the negative log likelihood for a held-out validation dataset. Alternatively, the ECE measure can be used as the objective score when finding the optimal T.

Let  $A_i$  and  $C_i$  be the accuracy and confidence of the validation-set points in the *i*-th set  $B_i$ . Denote the average confidence in bin *i* after temperature scaling of all the instances in  $B_i$  by a temperature *T* by  $C_i(T)$ :

$$C_i(T) = \frac{1}{|B_i|} \sum_{t \in B_i} \max_{j=1}^k \frac{\exp(z_{tj}/T)}{\sum_{l=1}^k \exp(z_{tl}/T)}$$
(4)

s.t.  $z_{t1}, ..., z_{tk}$  are the logit values computed by the network that is fed by  $x_t$ . The optimal temperature T can be found by minimizing the following adaECE score:

$$L_{\rm TS}(T) = \frac{1}{m} \sum_{i=1}^{m} |A_i - C_i(T)|.$$
(5)

The minimization is carried out by a grid search over the possible values of T. Direct minimization of the ECE measure (1) on the validation set was shown to yield better calibration results than maximizing the likelihood on a validation set (Mukhoti et al., 2020). This is not surprising since we optimize the same calibration measure directly on the validation set that is finally evaluated on the test set. It is better to use here the adaECE variant (2) rather than the ECE since in the ECE accuracy at low confidence bins is computed using a small number of validation samples which makes the scaling parameters' estimates less robust.

Ji et al. (2019) extended TS to a bin-wise setting, denoted bin-wise temperature scaling (BTS), by setting separate temperatures for each bin. BTS is trained by maximizing the log-likelihood function. We can also directly minimize the gap between the confidence and the accuracy in each bin by minimizing the following adaECE score:

$$L_{\text{CTS}}(T_1, ..., T_m) = \frac{1}{m} \sum_{i=1}^m |A_i - C_i(T_i)|, \qquad (6)$$

We need to apply a grid search to find  $T_i$  that satisfies  $A_i = C_i(T_i)$ . We denote this calibration method Confidence based Temperature Scaling (CTS). Similar to the case of single temperature scaling it can be shown that CTS consistently yields better calibration results than BTS. We use CTS as one of the baseline methods that are compared with the calibration method we propose next.

Varying the distribution temperature T from 1 to  $\infty$  induces a continuous path from the original class distribution  $p = (p_1, ..., p_k)$  to the uniform distribution u = (1/k, ..., 1/k). The notion of temperature scaling of a distribution originated in statistical physics. There is no intrinsic reason to specifically use a temperature to make the network output distribution smoother. The relevant features of temperature scaling as a smoothing procedure are that the entropy increases monotonically and the confidence decreases monotonically as a function of T, the order of probabilities from smallest to largest is maintained in the smoothing operation and it is a continuous function of T.

In this study we put forward a different way to make a distribution smoother. For each weight  $\alpha \in [0, 1]$  we define a smooth version of the original distribution p as follow:

$$p_{\alpha} = \alpha p + (1 - \alpha)u. \tag{7}$$

Varying the weight  $\alpha$  from 1 to 0 induces a different path from the class distribution p to the uniform distribution u. We denote the calibration approach based on shifting from p to  $p_{\alpha}$  (7) as Weight Scaling (WS). Figure 1 shows the trajectories of temperature scaling and weight scaling from p = [0.6, 0.3, 0.1] to u = [1/3, 1/3, 1/3].



Figure 1: Smoothing trajectories

WS has the following probabilistic interpretation. Denote the un-calibrated network output distribution by  $p(y = i|x; \theta)$  s.t. that  $\theta$  is the network parameter set. Let z be a binary random variable s.t.  $p(z = 1) = \alpha$ . Define the following conditional distribution:

$$p(y|x, z; \theta) = \begin{cases} p(y|x; \theta) & z = 1\\ \\ 1/k & z = 0 \end{cases}$$

The class distribution of a weight scaling calibrated network is:

$$p(y|x;\theta,\alpha) = p(z=1;\alpha)p(y|x,z=1;\theta) + p(z=0;\alpha)\frac{1}{k} = p_{\alpha}(y).$$

In other words, the calibration is carried out by flipping a coin z and if the result is zero the network ignores the input and reports a random class. It can be easily verified that the entropy  $H(p_{\alpha})$  is a concave function of  $\alpha$  and obtains its global maximum at  $\alpha = 0$ . Hence, as  $p_{\alpha}$  moves from p to u, the entropy of  $p_{\alpha}$  monotonically increases. The confidence after weight scaling by  $\alpha$  is simply  $\hat{p}_{\alpha} = \alpha \hat{p} + (1 - \alpha)1/k$  where  $\hat{p}$  is the confidence before calibration. It can be verified that in distributions where all the non-maximum probabilities are equal (e.g. (0.6,0.2,0.2)) weight scaling and temperature scaling are the same.

Both temperature scaling and weight scaling maintain the order of the predicted classes and therefore do not change the original hard classification decision. Another desired property of a calibration method is maintaining the order of instances based on their network prediction confidence. It can easily be verified that a network that is more confident at point x than at point y can become less confident at x than y after a temperature scaling calibration using the same temperature in both cases. Since weight scaling is a linear operation, it maintains the order of data points based on the confidence. If the network is more confident at x than y it remains more confident after weight scaling by the same  $\alpha$ .

We next use the adaECE score to learn a calibration procedure based on weight scaling instead of temperature scaling. In the case of weight scaling let

$$C_i(\alpha) = \frac{1}{|B_i|} \sum_{t \in B_i} \max_{j=1}^k (\alpha p_{tj} + (1-\alpha)\frac{1}{k}) = \alpha C_i + (1-\alpha)\frac{1}{k}$$
(8)

be the confidence in bin i after scaling by a weight  $\alpha$  where  $p_{t1}, ..., p_{tk}$  are the soft-max probability values computed by the network that is fed by  $x_t$ . In the case of single parameter weight scaling, we

look for a weight  $\alpha$  that minimizes the following adaECE score:

$$L_{\rm WS}(\alpha) = \frac{1}{m} \sum_{i=1}^{m} |A_i - C_i(\alpha)| = \frac{1}{m} \sum_{i=1}^{m} |A_i - \alpha C_i - (1 - \alpha) \frac{1}{k}|.$$
 (9)

Here there is no closed form solution for the optimal  $\alpha$ . However, if we replace the  $|\cdot|$  operation in Eq. (9) by  $|\cdot|^2$ , the optimal weight scaling  $\alpha$  is:

$$\alpha = \frac{\sum_{i} (C_{i} - \frac{1}{k}) (A_{i} - \frac{1}{k})}{\sum_{i} (C_{i} - \frac{1}{k})^{2}}.$$
(10)

We found that the  $L_2$  score yields worse calibration results than  $L_1$  (9). This is due to the fact that there is a different optimal weight for each bin and  $L_1$  is more robust to this diversity. This motivated us to allow a different weight in each bin. To find the weight set that minimizes the following adaECE score:

$$L_{\text{CWS}}(\alpha_1, ..., \alpha_m) = \frac{1}{m} \sum_{i=1}^m |A_i - C_i(\alpha_i)|, \qquad (11)$$

we can perform the minimization in each bin separately. In the case of weight scaling (unlike temperature scaling) there is a closed form solution to the equation  $A_i = C_i(\alpha_i)$  which is

$$\alpha_i = \frac{A_i - \frac{1}{k}}{C_i - \frac{1}{k}}.$$
(12)

The definition of confidence as the probability of the most likely class implies that always  $1/k \le C_i$ . If  $1/k \le A \le C_i$  then  $\alpha_i \in [0, 1]$ . In the (rare) case of accuracy less than random, i.e.  $A_i < 1/k$ , we set  $\alpha_i = 0$  and in the (rare) case of under-confidence, i.e.  $C_i < A_i$ , we set  $\alpha_i = 1$ .

This proposed calibration method is denoted Confidence based Weight Scaling (CWS). The train and inference phases of the CWS algorithm are summarized in Algorithm boxes 1 and 2, respectively. CWS has the desirable property that it does not affect the hard-decision accuracy since the same weight scaling is applied to all the logits. This guarantees that the calibration does not impact the accuracy. Note that both vector and matrix scaling do affect model accuracy and may decrease it.

## Algorithm 1 Confidence based Weight Scaling (CWS) - Train

**input:** A validation dataset  $x_1, ..., x_n$ . Each  $x_t$  is fed into a k-class classifier network to produce class distribution  $p_{t1}, ..., p_{tk}$ .

Compute the confidence values:  $\hat{p}_t = \arg \max_j p_{tj}, \qquad t = 1, ..., n.$ 

Order the points based on the confidence values and divide them into m equal size sets  $B_1, ..., B_m$ . for i = 1, ..., m do

Compute the average accuracy  $A_i$  and confidence  $C_i$  based on the points in  $B_i$ . Compute the calibration weight:

$$\alpha_i = \max(0, \min(1, \frac{A_i - \frac{1}{k}}{C_i - \frac{1}{k}}))$$

end for

output: The weight set and the bins' interval borders.

The CWS algorithm finds weight values  $\alpha_1, ..., \alpha_m$  such that the adaECE loss function (11) of the validation is exactly zero. This does not imply, however, that the adaECE (2) score of the calibrated validation set is zero. Since there is a different weight in each bin, the calibration can change the order of the validation points when sorted according to their confidence. This alters the partition of the validation set into bins and causes that the adaECE score (2) of the calibrated validation set is not necessarily zero. We can thus apply the optimization of the adaECE loss function (11) on the calibrated validation set in an iterative manner. There was no significant performance change when iterating the weight scaling procedure.

#### Algorithm 2 Confidence based Weight Scaling (CWS) - Inference

**input:** A data point x with network outputs class distribution  $p_1, ..., p_k$ .

calibration parameters: weights  $\alpha_1, ..., \alpha_m$  and a division of the unit interval into m bins.

Compute the predicted confidence:  $\hat{p} = \max_{i} p_{i}$ .

Find the index  $i \in \{1, ..., m\}$  s.t.  $\hat{p}$  is within the borders of *i*-th bin.

output: The calibrated prediction is:

$$p(y = j|x) = \alpha_i p_j + (1 - \alpha_i) \frac{1}{k}, \qquad j = 1, ..., k$$

Table 1: ECE (%) computed for different approaches for pre-scaling, post-single temperature scaling (TS) and post-single weight scaling (WS) (with the optimal weight (%) in brackets).  $W \approx 100$  indicates an innately calibrated model.

Dataset	Model	Cross-Entropy			Brier Loss			MMCE			LS-0.05		
		Pre T	TS	WS	Pre T	TS	WS	Pre T	TS	WS	Pre T	TS	WS
CIFAR-100	ResNet-50	17.52	3.42	4.28 (85.8)	6.52	3.64	3.97 (95.2)	15.32	2.38	6.39 (90.2)	7.81	4.01	<b>4.00</b> (95.4)
	ResNet-110	19.05	4.43	4.12 (84.3)	7.88	4.65	4.16 (94.9)	19.14	3.86	4.31 (84.4)	11.02	5.89	3.97 (91.2)
	Wide-ResNet-26-10	15.33	2.88	4.05 (86.5)	4.31	2.70	3.68 (98.1)	13.17	4.37	4.78 (90.8)	4.84	4.84	3.83 (98.6)
	DenseNet-121	20.98	4.27	4.37 (82.6)	5.17	2.29	3.82 (95.2)	19.13	3.06	4.97 (83.7)	12.89	7.52	3.67 (89.1)
CIFAR-10	ResNet-50	4.35	1.35	1.12 (96.4)	1.82	1.08	1.17 (99.2)	4.56	1.19	0.98 (96.0)	2.96	1.67	2.96 (100)
	ResNet-110	4.41	1.09	0.65 (95.8)	2.56	1.25	1.71 (99.0)	5.08	1.42	<b>0.70</b> (95.1)	2.09	2.09	2.09 (100)
	Wide-ResNet-26-10	3.23	0.92	0.84 (97.1)	1.25	1.25	1.25 (100)	3.29	0.86	0.79 (97.2)	4.26	1.84	4.26 (100)
	DenseNet-121	4.52	1.31	<b>1.03</b> (96.1)	1.53	1.53	1.53 (100)	5.10	1.61	1.25 (95.7)	1.88	1.82	1.88 (100)
Tiny-ImageNet	ResNet-50	15.32	5.48	6.89 (86.5)	4.44	4.13	4.32 (99.0)	13.01	5.55	5.25 (75.7)	15.23	6.51	15.23 (100)

## 4 EXPERIMENTAL RESULTS

We first illustrate the CWS algorithm on the CIFAR-10 and CIFAR-100 datasets (Krizhevsky, 2009) with network architecture ResNet110 trained with a cross-entropy loss. Fig. 2a and 2b present the CWS weight and the reciprocal of the CTS temperature in each bin that minimizes the adaECE score for CIFAR-10 and CIFAR-100, respectively. The horizontal axis contains the bins' indices from 0 to 12 (13 bins) and not an actual confidence value for each bin for purpose of better visualization (the confidences in high bins are very dense).

As we go up the bin range, we can see an increase in the optimal weight per bin. This is because the difference between the average confidence and accuracy in high bins is small compared to low bins, so the movement of probabilities towards the average accuracy should also be small. This is the reason why a single weight for all samples is not accurate enough. Fig. 2c and 2d show the difference between confidence and accuracy in each bin before calibration. Fig. 2a and 2b also indicate that the value of the (reciprocal of the) CTS temperature is different in each bin, but less consistently than the CWS weights.

We implemented the CWS method on various image classification tasks to test the algorithm's performance. The experimental setup followed the setup in (Mukhoti et al., 2020) and included several pre-trained deep neural networks which are available online<sup>1</sup>, trained on the following image classification datasets:

- 1. **CIFAR-10** (Krizhevsky, 2009): This dataset has 60,000 color images of size  $32 \times 32$ , divided equally into 10 classes. We used a train/validation/test split of 45,000/5,000/10,000 images.
- 2. **CIFAR-100** (Krizhevsky, 2009): This dataset has 60,000 color images of size  $32 \times 32$ , divided equally into 100 classes. We again used a train/validation/test split of 45,000/5,000/10,000 images.
- 3. **Tiny-ImageNet** (Deng et al., 2009): Tiny-ImageNet is a subset of ImageNet with 64 x 64 dimensional images, 200 classes and 500 images per class in the training set and 50 images per class in the validation set. The image dimensions of Tiny-ImageNet are twice those of the CIFAR-10/100 images.

<sup>&</sup>lt;sup>1</sup>https://github.com/torrvision/focal\_calibration



Figure 2: Comparison of the optimal weights achieved by CWS and the (reciprocal of the) optimal temperatures achieved by CTS in each bin for (a) CIFAR-10 trained on ResNet110 and (b) CIFAR-100 trained on ResNet110. The corresponding difference between confidence and accuracy before calibration in each bin are shown in (c) and (d).



Figure 3: Reliability diagrams, (a) before calibration, (b) after CTS calibration and (c) after CWS calibration with 13 bins for CIFAR-100 trained on ResNet110.

Table 1 compares the ECE% (computed using 15 bins) obtained by evaluating the test set. The results are divided into ECE before calibration, after scaling by a single temperature (TS) and after our single Weight Scaling (WS). The optimal TS was achieved by a greedy algorithm to minimize the ECE calibration score over a validation set (Mukhoti et al., 2020). The optimal WS was calculated in practice by a greed search and not by the closed formula (10). This is because the  $L_2$  score is not

Dataset	Model	Uncalibrated	TS	Vector Scaling	MS-ODIR	Dir-ODIR	Spline	BTS	CTS	CWS
CIFAR-100	ResNet-110	18.480	2.428	2.722	3.011	2.806	1.868	1.907	1.828	1.745
	ResNet-110-SD	15.861	1.335	2.067	2.277	2.046	1.766	1.373	1.421	1.370
	Wide-ResNet-32	18.784	1.667	1.785	2.870	2.128	1.672	1.796	1.182	1.599
	DenseNet-40	21.159	1.255	1.598	2.855	1.410	2.114	1.336	1.252	1.223
	Lenet-5	12.117	1.535	1.350	1.696	2.159	1.029	1.659	1.402	0.926
CIFAR-10	ResNet-110	4.750	1.224	1.092	1.276	1.240	1.011	1.224	0.829	0.812
	ResNet-110-SD	4.135	0.777	0.752	0.684	0.859	0.992	1.020	0.776	0.864
	Wide-ResNet-32	4.512	0.905	0.852	0.941	0.965	1.003	1.064	0.881	0.759
	DenseNet-40	5.507	1.006	1.207	1.250	1.268	1.389	0.957	0.855	1.314
	Lenet-5	5.188	1.999	1.462	1.504	1.300	1.333	1.865	1.420	1.406
ImageNet	DenseNet-161	5.720	2.059	2.637	4.337	3.989	0.798	1.224	1.097	0.830
	ResNet-152	6.545	2.166	2.641	5.377	4.556	0.913	1.165	0.918	0.912
SVHN	ResNet-152-SD	0.877	0.675	0.630	0.646	0.651	0.832	0.535	0.508	0.727

Table 2: ECE for top-1 predictions (in %) using 25 bins (with the lowest in bold and the second lowest underlined) on various image classification datasets and models with different calibration methods.

Table 3: ECE for top-1 predictions (in %) using various numbers of bins on CIFAR-100 dataset and models with equal size bins (ECWS) and an equal number of samples in each bin (CWS). Lowest ECE in bold.

Model	Uncalibrated	# of bins	ECWS	CWS	
		5	9.946	2.741	
DecNet 110	10 100	13	2.512	1.745	
Resinet-110	18.480	25	3.198	1.957	
		35	3.027	1.866	
		5	6.185	2.379	
DecNet 110 CD	15 061	13	2.774	1.370	
Resinet-110-5D	13.801	25	2.659	1.965	
		35	2.075	2.068	
		5	11.73	1.372	
W. J. D. N. 4 22	10 704	13	2.282	1.599	
wide-KesiNet-52	18.784	25	5.172	1.586	
		35	1.746	1.527	
		5	13.78	1.461	
DansaNat 40	21.159	13	11.07	1.223	
Delisemet-40		25	3.971	1.694	
		35	1.768	1.988	
		5	1.868	2.069	
Lonot 5	12 117	13	1.658	0.926	
Lenet-J	12.117	25	1.402	1.538	
		35	2.183	1.284	

robust to outliers, so the closed formula yields a non-accurate weight. Along with the cross-entropy loss, we tested our results on three other models which were trained on different loss functions:

- 1. **Brier loss** (Brier, 1950): The squared error between the predicted softmax vector and the one-hot ground truth encoding.
- 2. **MMCE** (Maximum Mean Calibration Error) (Kumar et al., 2018): A continuous and differentiable proxy for the calibration error that is normally used as a regulariser alongside cross-entropy.
- 3. Label smoothing (LS) (Müller et al., 2019): Given a one-hot ground-truth distribution  $\mathbf{q}$  and a smoothing factor  $\alpha$  (hyper-parameter), the smoothed vector  $\mathbf{s}$  is obtained as  $\mathbf{s}_i = (1 \alpha)\mathbf{q}_i + \alpha(1 \mathbf{q}_i)/(k 1)$ , where  $\mathbf{s}_i$  and  $\mathbf{q}_i$  denote the  $i^{th}$  elements of  $\mathbf{s}$  and  $\mathbf{q}$  respectively, and k is the number of classes. Instead of  $\mathbf{q}$ ,  $\mathbf{s}$  is treated as the ground truth. The reported results were obtained from LS-0.05 with  $\alpha = 0.05$ , which was found to achieve the best performance (Mukhoti et al., 2020).

The comparative calibration results are presented in Table 1. As can be seen, the ECE score after WS calibration was lower than the ECE after TS in half of all cases. However, the advantage of WS over TS is its simplicity.

In another set of experiments, we followed the setup in (Gupta et al., 2021). In addition to CIFAR-10 and CIFAR-100, we evaluated our CWS method on the SVHN dataset (Netzer et al., 2011) and ImageNet (Deng et al., 2009). Pre-trained network logits are available online <sup>2</sup>. The CWS was compared to TS, vector scaling, two variants of matrix scaling (Kull et al., 2019), BTS (Ji et al., 2019) and Spline fitting (Gupta et al., 2021). It was also compared to our CTS algorithm, which calculates the optimal TS in each bin. As shown in Table 2, CWS achieved the best or second best results in most cases.

The weights that yielded the ECE scores for CWS in Table 2 were set to  $\alpha_i \in [0, 1]$ , as specified in Algorithm box 1. There were three cases in Table 2 where the original weight calculated by (10) for a specific bin was higher than 1. This occurred for ResNet-110-SD on CIFAR100, Lenet-5 on CIFAR-10 and ResNet-152-SD on the SVHN dataset. In all three cases, the corresponding bin contained under-confidence samples, i.e., the average accuracy in that bin was higher than the average confidence. As mentioned above, this case is rare.

Another way of visualizing calibration is to use a *reliability plot* (Niculescu-Mizil & Caruana, 2005), which shows the accuracy values of the confidence bins as a bar chart. For a perfectly calibrated model, the accuracy for each bin matches the confidence; hence, all the bars lie on the diagonal. By contrast, if most of the bars lie above the diagonal, the model is more accurate than it expects, and is under-confident, and if most of the bars lie below the diagonal, it is over-confident. Fig. 3 compares reliability plots over 13 bins on models trained on ResNet-110 with cross-entropy loss and on CIFAR-100 before calibration, after CTS calibration and after CWS calibration. It shows that both CTS and CWS methods yielded a calibrated system, although CWS was simpler to implement using a closed form. Note that some of the bins were located above the diagonal, which indicates the under-confidence of the model.

As an ablation study we examined learning the CWS by maximizing ECE (1) instead of adaECE (2) for a varied number of bins. Table 3 shows that the original split for same number of samples in each bin (CWS) yielded a lower ECE than the case of equal bin size (ECWS) in most cases. No number of bins in a reasonable range in the training step of the CWS algorithm had a significant impact on the ECE.

# 5 CONCLUSION

Calibrated confidence estimates of predictions are critical to increasing our trust in the performance of neural networks. As interest grows in deploying neural networks in real work decision making systems, the predictable behavior of the model will be a necessity especially for critical tasks such as automatic navigation and medical diagnosis. In this work, we introduced a simple and effective calibration method based on weight scaling of the prediction confidence. Most calibration methods are trained by optimizing the cross entropy score. CWS function learning can be done by explicitly optimizing the ECE measure. We compared our CWS method to various state-of-the-art methods and showed that it yielded the lowest calibration error in the majority of our experiments. CWS is very easy to train and there is no need to tune any hyper-parameter. We believe that it can be used in place of the standard temperature scaling method.

# REFERENCES

- Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mane. Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*, 2016.
- Glenn W Brier. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3, 1950.
- Cynthia S Crowson, Elizabeth J Atkinson, and Terry M Therneau. Assessing calibration of prognostic risk scores. *Statistical methods in medical research*, 25(4):1692–1706, 2016.

<sup>&</sup>lt;sup>2</sup>https://github.com/markus93/NN\_calibration

- Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE conference on computer vision and pattern recognition*, 2009.
- Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, 2017.
- Kartik Gupta, Amir Rahimi, Thalaiyasingam Ajanthan, Thomas Mensink, Cristian Sminchisescu, and Richard Hartley. Calibration of neural networks using splines. In *International Conference on Learning Representations (ICLR)*, 2021.
- Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf. Why ReLU networks yield highconfidence predictions far away from the training data and how to mitigate the problem. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- Byeongmoon Ji, Hyemin Jung, Jihyeun Yoon, Kyungyul Kim, and Younghak Shin. Bin-wise temperature scaling (BTS): Improvement in confidence calibration performance through simple scaling techniques. In *IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, 2019.
- 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*, 192(2):263–274, 2011.
- Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? Advances in Neural Information Processing Systems (NeurIPs), 2017.
- Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, Department of Computer Science, University of Toronto, 2009.
- Meelis Kull, Miquel Perello Nieto, Markus Kängsepp, Telmo Silva Filho, Hao Song, and Peter Flach. Beyond temperature scaling: Obtaining well-calibrated multi-class probabilities with Dirichlet calibration. In *Advances in Neural Information Processing Systems (NeurIPs)*, 2019.
- Ananya Kumar, Percy S Liang, and Tengyu Ma. Verified uncertainty calibration. In Advances in Neural Information Processing Systems (NeurIPs), 2019.
- Aviral Kumar, Sunita Sarawagi, and Ujjwal Jain. Trainable calibration measures for neural networks from kernel mean embeddings. In *International Conference on Machine Learning*, 2018.
- Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems (NeurIPs)*, 2017.
- Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. In *Advances in Neural Information Processing Systems (NeurIPs)*, 2019.
- Dimitrios Milios, Raffaello Camoriano, Pietro Michiardi, Lorenzo Rosasco, and Maurizio Filippone. Dirichlet-based Gaussian processes for large-scale calibrated classification. In Advances in Neural Information Processing Systems (NeurIPs), 2018.
- Jishnu Mukhoti, Viveka Kulharia, Amartya Sanyal, Stuart Golodetz, Philip HS Torr, and Puneet K Dokania. Calibrating deep neural networks using focal loss. In *Advances in Neural Information Processing Systems (NeurIPs)*, 2020.
- Rafael Müller, Simon Kornblith, and Geoffrey Hinton. When does label smoothing help? *arXiv* preprint arXiv:1906.02629, 2019.
- Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In AAAI Conference on Artificial Intelligence, 2015.
- Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop*, 2011.

- Khanh Nguyen and Brendan O'Connor. Posterior calibration and exploratory analysis for natural language processing models. *arXiv preprint arXiv:1508.05154*, 2015.
- Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities with supervised learning. In International Conference on Machine Learning, 2005.
- Jeremy Nixon, Michael W Dusenberry, Linchuan Zhang, Ghassen Jerfel, and Dustin Tran. Measuring calibration in deep learning. In CVPR Workshops, 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.
- Maithra Raghu, Katy Blumer, Rory Sayres, Ziad Obermeyer, Bobby Kleinberg, Sendhil Mullainathan, and Jon Kleinberg. Direct uncertainty prediction for medical second opinions. In *International Conference on Machine Learning*, 2019.
- Juozas Vaicenavicius, David Widmann, Carl Andersson, Fredrik Lindsten, Jacob Roll, and Thomas Schön. Evaluating model calibration in classification. In *Artificial Intelligence and Statistics*, 2019.
- Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, 2002.
- Jize Zhang, Bhavya Kailkhura, and T Yong-Jin Han. Mix-n-match: Ensemble and compositional methods for uncertainty calibration in deep learning. In *International Conference on Machine Learning*, 2020.