Network Calibration by Class-based Temperature Scaling

Lior Frenkel Jacob Goldberger
Faculty of Engineering, Bar-Ilan University, Israel
lior.frenkel@biu.ac.il, jacob.goldberger@biu.ac.il

***Abstract*—It is well known that modern neural networks are poorly calibrated. They tend to overestimate or underestimate probabilities when compared to the expected accuracy. This results in misleading reliability and corrupting our decision policy. We show that the amount of calibration error differs across the classes. As a result, we propose to calibrate each class separately. We apply this class-level calibration paradigm to the concept of temperature scaling and describe an optimization method that finds the suitable temperature scaling for each class. We report extensive experiments on a variety of image datasets, and a wide variety of network architectures, and show that our approach achieves state-of-the-art calibration without compromising on accuracy in almost all cases.**

***Index Terms*—neural networks, network calibration, temperature scaling, Expected Calibration Error (ECE)**

I. 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 real 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 [1], [2], [3] and perception tasks in autonomous driving [4]. A classifier is said to be calibrated if the probability values it associates with the class labels match the true probabilities of 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 [5], [6], [7].

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. [8] [9] [10] [11] [12]). The second approach performs calibration as a post processing step using an already trained model. Post-hoc scaling approaches for calibration (e.g. Platt scaling [13], isotonic regression [14], and temperature scaling [5]) are widely used. Their goal is to 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 [5] and is the current standard practical calibration method. Guo et al. [5] investigated several scaling models, ranging from single-parameter based temperature scaling to more complicated vector/matrix scaling. They reported poor performance for vector/matrix scaling calibration. To avoid overfitting, Kull et al. [15] suggested regularizing matrix scaling with a L_2 loss on the calibration model weights.

In this study we propose an extension of temperature scaling that assigns a separate scaling parameter to each class. We use a greedy grid search optimization procedure to directly optimize the Expected Calibration Error (ECE) measure [16], instead of optimizing the negative-log-likelihood score as was done in [5] and [15]. Using this optimization we can guarantee that the calibration does not lead to any performance degradation.

We show that, unlike matrix scaling [15], we can easily find 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 compared to temperature scaling. The proposed approach to calibration does not change the model parameters, which allows it to be applied on any trained network and guarantees to retain the original classification accuracy.

We evaluated our method against existing calibration approaches on various image datasets. Our recalibration approach outperforms existing methods on improving the ECE calibration measures.

II. 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 $\hat{y} = \arg\max_i p(y = i|x)$ 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 *expected calibration error* (ECE) [16], 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 ith 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 ith bin. The accuracy A_i of this bin is computed as $A_i = \frac{1}{|B_i|} \sum_{t \in B_i} \mathbb{1}\left(\hat{y}_t = y_t\right)$, where $\mathbb{1}$ is the indicator function, and $\hat{y}_t$ and y_t are the predicted and ground-truth labels for the tth sample. Similarly, the confidence C_i of the ith 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 a weighted average of the absolute difference between the accuracy and confidence of each bin:

$$\text{ECE} = \sum_{i=1}^{M} \frac{|B_i|}{n} \left|A_i - C_i\right| \tag{1}$$

where n is the number of samples in a validation set. Note that $A_i > C_i$ means the network is under-confident at the ith bin and $C_i > A_i$ implies that the network is over-confident. We note in passing that even though the drawbacks of ECE have been pointed out and some improvements have been proposed [8], [17], [18], the ECE histogram approximation is still used as the standard calibration measure.

The ECE method can also be used to determine the calibration of the prediction for each class separately [15], [19], [8]. 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 ith 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:

$$\mathrm{ECE}_j = \sum_{i=1}^{M} \frac{|B_{ij}|}{n_j} \left|A_{ij} - C_{ij}\right|. \tag{2}$$

Fig. 1a shows the class-level ECE score vs. accuracy of the 100 classes in the CIFAR-100 dataset trained with ResNet110[20]. It can be seen that the ECE of a class depends on its accuracy - the higher the accuracy, the lower the ECE. To better understand the behavior of class level confidence we can break the class-level ECE score (2) into two parts. We can first sum over the bins where the network is under-confident:

$$\mathrm{ECE}_j^{under} = \sum_{i=1}^{M} \frac{|B_{ij}|}{n_j} \max(0, A_{ij} - C_{ij}) \tag{3}$$

and then sum over the bins where the network in over-confident:

$$\mathrm{ECE}_j^{over} = \sum_{i=1}^{M} \frac{|B_{ij}|}{n_j} \max(0, C_{ij} - A_{ij}). \tag{4}$$

It can be easily verified that for each class j:

$$\mathrm{ECE}_j = \mathrm{ECE}_j^{under} + \mathrm{ECE}_j^{over}. \tag{5}$$

Fig. 1b and 1c show the ECE vs. accuracy for under-confidence bins (3) and for over confidence bins (4) respectively. Fig. 1d shows under and over confidence at each bin summarized over all the classes. We can see that under-confidence situations occur mainly at the lowest bin where the probability of the most likely class is very low. This is why under-confidence is more frequent in classes with low accuracy results. Hence, although the main problem is over-confidence, in classes that are poorly classified there are also problems of under-confidence in smaller bins. Overall we can see from Fig. 1a that the calibration problem is different across the different classes. Therefore, it make sense to calibrate each class separately.

III. Class based temperature scaling

Temperature Scaling (TS), is a simple yet very effective technique for calibrating prediction probabilities [5]. 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. Since the same T is used for all classes, the softmax output with scaling has a monotonic relationship with unscaled output. In overconfident models where $T > 1$, the recalibrated probabilities have a lower value than the original probabilities, and they 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. The temperature T

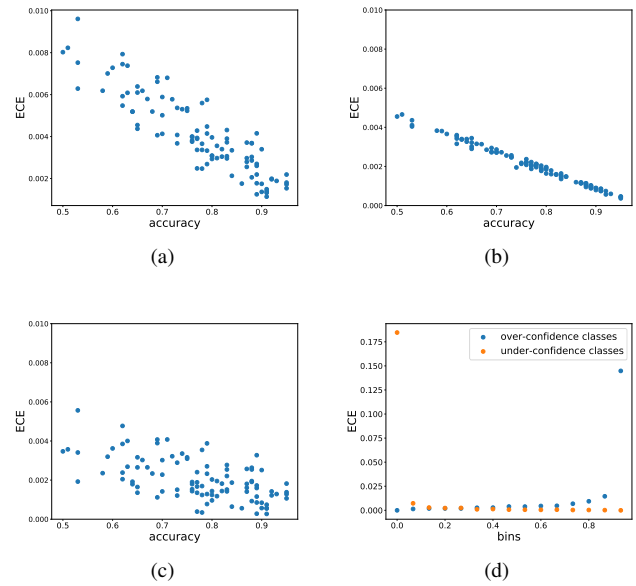

Fig. 1: (a) Classwise ECE vs. accuracy for the 100 classes of CIFAR-100 trained with ResNet110, (b) ECE computed on under-confidence bins, (c) ECE computed on over-confidence bins, (d) under- and over-confidence ECE summarized over the classes at each bin.

usually scales between 1.5 and 3, which indicates that the network learning algorithm produced an overconfident model.

As we demonstrated above, there may be different calibration errors in different classes. Because it only has a single tuneable parameter, TS cannot learn to act differently on different classes. Thus, we propose to assign a separate temperature to each class. Denote the scaling of class i by T_i. The temperature scaling is performed as follows:

$$p(y = i|x) = \frac{\exp(z_i/T_i)}{\sum_{j=1}^{k} \exp(z_j/T_j)}, \quad i = 1, \ldots, k \tag{6}$$

s.t. $z_1, ..., z_k$ are the logit values (the input to the softmax function) obtained by applying the network to input vector x. In the case where we use the same temperature for all classes the model is reduced to standard TS [5]. To find the optimal set of temperature scales we can apply gradient based methods to minimize the negative log likelihood for a held-out validation dataset. In the case of a single temperature parameter, direct minimization of the ECE measure (1) on the validation set was shown to yield better calibration results [12]. This is not surprising since we directly optimize the same calibration measure on the validation set that is finally evaluated on the test set. ECE is not differentiable so that the optimal temperature can be found by a grid search over values between 0 and 10, with a step of 0.1, and finding the one that minimizes the validation set ECE. In our case of assigning a different parameter to each class, the grid size is an exponential function of the number of classes; hence a grid search is no longer feasible. Instead, we suggest a greedy grid search strategy to find a local optimum of the ECE score. Let $\mathrm{ECE}(T_1, T_2, ..., T_k)$ be the validation set ECE score (1) of the system that is calibrated according to Eq. (6). For each class i, while freezing all other temperature values, the algorithm uses a grid search to find the temperature value T_i for that class that minimizes the ECE

measure.

$$T_i = \arg\min_S \text{ECE}(T_1, ..., T_{i-1}, S, T_{i+1}, ..., T_k). \quad (7)$$

The algorithm goes over all the classes in a circular manner. The ECE score is monotonically improved until it converges to a local minimum point. We dub our algorithm "Class based Temperature Scaling (CTS)".

TS has the desirable property that it does not affect its hard-decision accuracy. Since the parameter T does not change the identity of the class that maximizes the softmax function, the class prediction remains unchanged. The proposed CTS algorithm can, in principle, change the model's accuracy. In the next section we empirically show that there is indeed a trade-off between calibration and accuracy. Improving the calibration can cause degradation of the classification accuracy. Since for each class i we perform an exhaustive grid search over the best temperature T_i, we can look for T_i that yields the best ECE result while not decreasing the performance compared to the current value of T_i. Let $\text{Acc}(T_1, T_2, ..., T_k)$ be the classification accuracy computed on the validation set. To ensure that there is no performance degradation in the calibration process, the minimization in (7) is performed only on grid values S such that:

$$\text{Acc}(T_1, ..., S, ..., T_k) \geq \text{Acc}(T_1, ..., T_i, ..., T_k)$$

where T_i is the ith parameter's current value.

Algorithm 1 Class based Temperature Scaling (CTS)

Goal: Find $T_1, ..., T_k \in (0, \infty)$ that minimize the ECE calibration score $\text{ECE}(T_1, ..., T_k)$ on a validation set.
Initialization: Compute temperature scaling T and set
$T_i = T \qquad i = 1, ..., k$
while still not converging **do**
  for $i = 1, ..., k$ **do**

$$\hat{T}_i \leftarrow \arg\min_S \text{ECE}(T_1, ..., T_{i-1}, S, ..., T_k)$$

  s.t. the minimization is done over all grid values S that increase the validation set accuracy.
  end for
end while

In order to find the optimal temperature for each class, we first computed the single temperature that minimized the ECE score. Next, we use it to initialize all the class based temperatures. We found this initializing method to work well for the class-based temperature scaling process. The algorithm usually converges fast and five iterations were enough for all the experiments. The CTS algorithm is summarized in Algorithm box 1.

Matrix and vector scaling are variants of TS. Matrix scaling applies a linear transformation $Wz_i + b$ to the logits before the softmax operation. The number of parameters for matrix scaling grows quadratically with the number of classes k. Vector scaling is a variant where W is restricted to be a diagonal matrix. If W is further restricted to be a scalar matrix, we obtain the TS method. Gau at al. [5] reported poor performance for matrix scaling optimized by minimizing cross entropy on validation set, leading the authors to conclude that a calibration model with many parameters would overfit to a small validation set. Note that applying matrix scaling to a task with 100 classes requires 10100 scaling parameters. They also reported that vector scaling recovers essentially the same solution as temperature scaling. This is because the learned vector has nearly constant entries, and therefore is no different from a scalar transformation. Note that, unlike our approach, they optimized the cross entropy score. Nixon et al. [17] also reported inferior calibration results for vector scaling compared to TS.

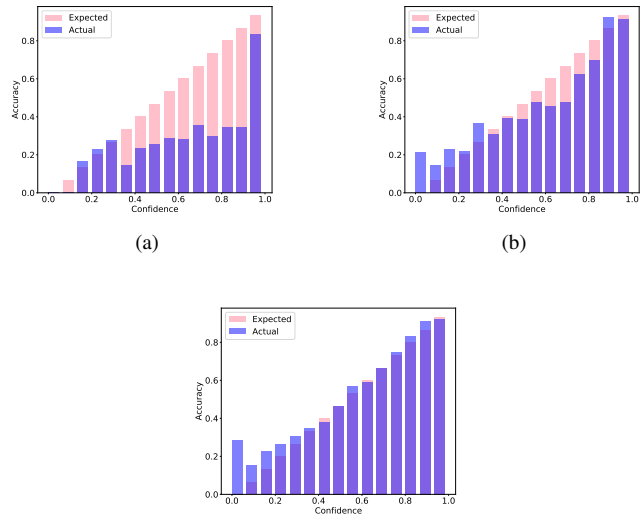

Fig. 2: Reliability plots, before calibration (a) after TS calibration (b) and after CTS calibration (c).

Kull et al. [15] addressed this over-fitting problem by adding the following off-diagonal L_2 regularization term to the cross entropy score:

$$R(w, b) = \lambda \cdot \frac{1}{k(k-1)} \sum_{i \neq j} w_{ij}^2 + \mu \cdot \frac{1}{k} b_i^2 \quad (8)$$

where λ and μ are hyper-parameters that need to be tuned with the internal cross-validation on the validation data. This approach has several drawbacks. First, having many parameters and adding hyper parameters to the loss makes the calibration procedure difficult to carry out and unstable. Second, and more importantly, due to the large number of parameters and the existence of hyper parameters in the matrix scaling we cannot apply greedy search optimization to minimize the ECE calibration score. In contrast, the proposed CTS method is more general than TS but calibration by minimizing the ECE score is still feasible and there are still guarantees that the performance will not decrease while minimizing the ECE score.

IV. Experimental Results

We conducted image classification experiments to test the performance of the CTS algorithm. We used the following image classification datasets in our experiments:

1) **CIFAR-10** [20]: This dataset has 60,000 color images of size 32×32, divided equally into 10 classes. We use a train/validation/test split of 45,000/5,000/10,000 images.
2) **CIFAR-100** [20]: This dataset has 60,000 color images of size 32×32, divided equally into 100 classes. We again use a train/validation/test split of 45,000/5,000/10,000 images.
3) **Tiny-ImageNet** [21]: 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 CIFAR-10/100 images.

Dataset	Model	Cross-Entropy			Brier Loss			MMCE			LS-0.05		
		Pre T	TS	CTS	Pre T	TS	CTS	Pre T	TS	CTS	Pre T	TS	CTS
CIFAR-100	ResNet-50	17.52	3.42(2.1)	**2.72**	6.52	3.64(1.1)	**1.59**	15.32	**2.38(1.8)**	2.48	7.81	4.01(1.1)	**2.37**
	ResNet-110	19.05	4.43(2.3)	**1.74**	7.88	4.65(1.2)	**2.65**	19.14	3.86(2.3)	**2.33**	11.02	5.89(1.1)	**1.99**
	Wide-ResNet-26-10	15.33	2.88(2.2)	**2.14**	4.31	2.70(1.1)	**1.79**	13.17	**4.37(1.9)**	4.49	4.84	4.84(1.0)	**1.68**
	DenseNet-121	20.98	4.27(2.3)	**2.06**	5.17	2.29(1.1)	**1.98**	19.13	3.06(2.1)	**2.27**	12.89	7.52(1.2)	**1.79**
CIFAR-10	ResNet-50	4.35	1.35(2.5)	**0.99**	1.82	**1.08(1.1)**	1.33	4.56	1.19(2.6)	**0.83**	2.96	1.67(0.9)	**1.04**
	ResNet-110	4.41	1.09(2.8)	**0.90**	2.56	1.25(1.2)	**0.67**	5.08	1.42(2.8)	**0.66**	2.09	2.09(1.0)	**0.68**
	Wide-ResNet-26-10	3.23	0.92(2.2)	**0.80**	1.25	1.25(1.0)	**0.66**	3.29	0.86(2.2)	**0.34**	4.26	1.84(0.8)	**0.72**
	DenseNet-121	4.52	1.31(2.4)	**1.00**	1.53	1.53(1.0)	**1.05**	5.10	1.61(2.5)	**1.32**	1.88	1.82(0.9)	**1.49**
Tiny-ImageNet	ResNet-50	15.32	5.48(1.4)	**4.20**	4.44	4.13(0.9)	**2.96**	13.01	5.55(1.3)	**4.17**	15.23	6.51(0.7)	**4.73**

TABLE I: ECE (in %) computed for different approaches for pre-temperature scaling, single temperature scaling (TS) and class-based temperature scaling. (CTS).

Dataset	Model	Cross-Entropy			Brier Loss			MMCE			LS-0.05		
		CTS	U-CTS	Acc-diff	CTS	U-CTS	Acc-diff	CTS	U-CTS	Acc-diff	CTS	U-CTS	Acc-diff
CIFAR-100	ResNet-50	2.72	**2.50**	4.6	**1.59**	1.99	2.4	**2.48**	2.92	6.2	2.37	**1.86**	0.1
	ResNet-110	1.74	**1.33**	0.7	2.65	**2.19**	4.9	2.33	**1.80**	0.1	**1.99**	2.58	0.1
	Wide-ResNet-26-10	2.14	**1.57**	5.1	1.79	**1.32**	4.9	4.49	**3.37**	9.8	**1.68**	2.14	0.0
	DenseNet-121	**2.06**	2.43	1.8	1.98	**1.67**	6.2	2.27	**2.12**	1.0	**1.79**	2.42	0.0
CIFAR-10	ResNet-50	0.99	**0.95**	0.1	1.33	**1.06**	0.3	**0.83**	0.88	0.0	1.04	1.04	0.0
	ResNet-110	0.90	**0.73**	0.0	**0.67**	0.94	0.2	0.66	**0.63**	0.0	0.68	**0.61**	0.1
	Wide-ResNet-26-10	0.80	**0.76**	0.0	**0.66**	0.81	0.3	**0.34**	0.47	0.0	0.72	0.72	0.0
	DenseNet-121	**1.00**	1.09	0.0	**1.05**	1.07	0.4	1.32	**1.07**	0.0	1.49	**0.77**	0.0
Tiny-ImageNet	ResNet-50	4.20	**1.47**	6.1	2.96	**1.74**	3.2	4.17	**1.74**	5.2	4.73	**1.47**	10.3

TABLE II: ECE (in %) computed for different approaches for CTS with an accuracy constraint (CTS), and without an accuracy constraint (U-CTS) and the drop in the accuracy (in %) when using U-CTS for each approach.

Dataset	Model	Uncalibrated	TS	Vector Scaling	MS-ODIR	Dir-ODIR	CTS
ImageNet	DenseNet-161	5.720	**2.059**	2.637	4.337	3.989	<u>2.158</u>
	ResNet-152	6.545	<u>2.166</u>	2.641	5.377	4.556	**2.149**
SVHN	ResNet-152-SD	0.877	0.675	<u>0.630</u>	0.646	0.651	**0.458**

TABLE III: ECE (in %) using 25 bins (with the lowest in bold and the second lowest underlined).

We tested the results on several pre-trained deep neural networks. We followed the experiment setup in [12] and used their trained networks which are available online [1].

Table I describes a comparison on ECE% (computed using 15 bins) obtained by evaluation on the test set. The results are before calibration, with scaling by a single temperature (TS) and with our class based temperature scaling (CTS). The optimal TS was achieved by a greedy algorithm to minimize the ECE calibration score [12]. Along with the cross-entropy loss, we tested our results on three other models which were trained on different loss functions:

1) **Brier loss** [22]: The squared error between the predicted softmax vector and the one-hot ground truth encoding.
2) **MMCE** (Maximum Mean Calibration Error) [23]: A continuous and differentiable proxy for calibration error that is normally used as a regulariser alongside cross-entropy.
3) **Label smoothing** (LS) [24]: Given a one-hot ground-truth distribution $\mathbf{q}$ and a smoothing factor α (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 ith 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 btained from LS-0.05 with $\alpha = 0.05$, which was found to achieve the best performance [12].

[1]https://github.com/torrvision/focal_calibration

The comparative calibration results are presented in Table I. As can be seen, the ECE score after CTS calibration was lower than the ECE after TS in almost all cases. Fig. 3 presents the optimal temperature of each class after class-based scaling vs. class accuracy for ResNet-110 with a cross-entropy loss trained on CIFAR-100. The red line marks the optimal single temperature that minimized the ECE score.

The CTS algorithm ensures no performance degradation by applying a constraint on the model's accuracy at each iteration. We compared the ECE results of CTS to the Unconstrained CTS (U-CTS) that allows the accuracy to drop when optimizing the ECE score. We also checked the difference in accuracy before and after the Unconstrained CTS to see if and how much the performance actually decreases. The results are shown in Table II.

Another way of visualizing calibration is to use a *reliability plot* [25], which plots the accuracies of the confidence bins as a bar chart. For a perfectly calibrated model, the accuracy for each bin matches the confidence, hence all of 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, then it is over-confident. Fig. 2 compares reliability plots over 15 bins on models trained on ResNet-110 with cross-entropy loss and on CIFAR-100 for three cases. It shows that the CTS method yielded the best calibrated system of the three, especially for high probability bins. This is yet another

indication that the network is better calibrated after CTS than after TS. Note that some of the bins lie above the diagonal, which indicates the under-confidence of the model.

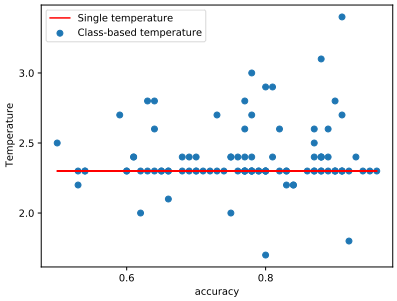

Fig. 3: Optimal scaling values vs. accuracy for the classes in CIFAR-100 (trained with ResNet110).

In another set of experiments we followed the setup in [18]. We evaluated our CTS method on the SVHN dataset [26] and ImageNet [27]. Pre-trained network logits are available online [2]. The proposed CTS is compared to TS, vector scaling and to two variants of matrix scaling (see details at [18]). As can be seen in Table III, CTS achieved the best results in two cases and was on a par with TS in the third case.

V. Conclusion

Temperature scaling is the simplest, fastest, and most straightforward calibration method. In spite of this, it is often the most effective and widely used method. In this paper we proposed CTS which is the simplest considerable extension of the TS method which enables a different temperature scaling for each class. CTS has all the advantages of TS, i.e., it is easy to train and implement. In addition, it consistently produces better calibration results on a large variety of tasks and network architectures without any performance degradation.

Acknowledgment

The research was partially supported by the Israeli Ministry of Science & Technology.

References

[1] Cynthia S Crowson, Elizabeth J Atkinson, and Terry M Therneau, "Assessing calibration of prognostic risk scores," *Statistical methods in medical research*, vol. 25, no. 4, pp. 1692–1706, 2016.

[2] 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*, vol. 192, no. 2, pp. 263–274, 2011.

[3] 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.

[4] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mane, "Concrete problems in AI safety," *arXiv preprint arXiv:1606.06565*, 2016.

[5] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger, "On calibration of modern neural networks," in *International Conference on Machine Learning*, 2017.

[6] 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.

[7] Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf, "Why ReLU networks yield high-confidence predictions far away from the training data and how to mitigate the problem," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[8] Ananya Kumar, Percy S Liang, and Tengyu Ma, "Verified uncertainty calibration," in *Advances in Neural Information Processing Systems (NeurIPs)*, 2019.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] John Platt et al., "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," *Advances in Large Margin Classifiers*, vol. 10, no. 3, pp. 61–74, 1999.

[14] Bianca Zadrozny and Charles Elkan, "Transforming classifier scores into accurate multiclass probability estimates," in *International Conference on Knowledge Discovery and Data Mining (KDD)*, 2002.

[15] 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.

[16] Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht, "Obtaining well calibrated probabilities using bayesian binning," in *AAAI Conference on Artificial Intelligence*, 2015.

[17] Jeremy Nixon, Michael W Dusenberry, Linchuan Zhang, Ghassen Jerfel, and Dustin Tran, "Measuring calibration in deep learning.," in *CVPR Workshops*, 2019.

[18] 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.

[19] Juozas Vaicenavicius, David Widmann, Carl Andersson, Fredrik Lindsten, Jacob Roll, and Thomas Schön, "Evaluating model calibration in classification," in *Artificial Intelligence and Statistics*, 2019.

[20] Alex Krizhevsky, "Learning multiple layers of features from tiny images," Tech. Rep., Department of Computer Science, University of Toronto, 2009.

[21] 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.

[22] Glenn W Brier, "Verification of forecasts expressed in terms of probability," *Monthly weather review*, vol. 78, no. 1, pp. 1–3, 1950.

[23] Aviral Kumar, Sunita Sarawagi, and Ujjwal Jain, "Trainable calibration measures for neural networks from kernel mean embeddings," in *International Conference on Machine Learning*, 2018.

[24] Rafael Müller, Simon Kornblith, and Geoffrey Hinton, "When does label smoothing help?," *arXiv preprint arXiv:1906.02629*, 2019.

[25] Alexandru Niculescu-Mizil and Rich Caruana, "Predicting good probabilities with supervised learning," in *International Conference on Machine Learning*, 2005.

[26] 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.

[27] 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.

[2] https://github.com/markus93/NN_calibration