

Class-Similarity Based Label Smoothing for Confidence Calibration

Chihuang Liu* and Joseph JaJa

University of Maryland, College Park MD 20742, USA
chliu@umd.edu

Abstract. Generating confidence calibrated outputs is of utmost importance for the applications of deep neural networks in safety-critical decision-making systems. The output of a neural network is a probability distribution where the scores are estimated confidences of the input belonging to the corresponding classes, and hence they represent a complete estimate of the output likelihood relative to all classes. In this paper, we propose a novel form of label smoothing to improve confidence calibration. Since different classes are of different intrinsic similarities, more similar classes should result in closer probability values in the final output. This motivates the development of a new smooth label where the label values are based on similarities with the reference class. We adopt different similarity measurements, including those that capture feature-based similarities or semantic similarity. We demonstrate through extensive experiments, on various datasets and network architectures, that our approach consistently outperforms state-of-the-art calibration techniques including uniform label smoothing.

Keywords: Confidence calibration · Uncertainty estimation · Similarity measure

1 Introduction

Machine learning algorithms have progressed rapidly in recent years and are becoming the critical component in a wide variety of technologies [6]. In most of these applications, making wrong decisions could lead to very high costs, including significant business losses or even severe human injuries [2]. As a result, in real-world decision-making systems, machine learning models should not only try to be as accurate as possible, but should also indicate when they are likely to be incorrect, which allows the decision-making to be stopped or passed to human experts when the models are not sufficiently confident to produce a correct prediction. It is therefore strongly desirable that a network provides a calibrated confidence measure in addition to its prediction; that is, the probability associated with the predicted class label should reflect its ground truth likelihood of correctness [19].

However, recent works [7] have shown through extensive empirical studies that even though impressively accurate, modern deep neural networks are poorly

calibrated. It turns out that modern DNNs are overconfident - the prediction accuracy is likely to be lower than what is indicated by the associated confidence. Since the discovery of this challenging problem, several methods [31,18,27,25] have been explored and empirically shown to improve confidence calibration performance on the predictions, which we refer to as *prediction calibration*, for which only the model’s prediction (the winning class) and its associated confidence (the maximum softmax score) are considered.

In the output of a neural network, each score represents the model’s estimated probability of the current input belonging to that corresponding class. Therefore, for a well-calibrated model, each probability should be indicative of the actual likelihood of the input belonging to each class, not just the one with the maximum score [12,10]. This is significant in safety-critical applications. For example, an autonomous driving system predicts an object to be 60% a pedestrian, 30% an obstacle, and 10% a traffic sign. Clearly these probabilities should be calibrated. We refer to this as *output calibration* where the entire output probabilities are considered.

The output scores define a probability distribution over all classes, and we call the perfectly calibrated distribution as the optimal output distribution. Under the optimization scheme with cross-entropy loss that most modern DNNs adopt, a model would achieve perfect confidence calibration if directly trained using the optimal distribution as training label. Unfortunately, as one would expect, it is unclear how to directly compute the optimal distribution in general. However, under certain reasonable assumptions, we can develop good approximations. Different classes are not equally distinct, and a class can be inherently more similar to some classes than others. For example, in the CIFAR-100 dataset, we could generally agree that the class dolphin is much more similar to the class seal than to the class rose, and the probability of a seal should be higher than a rose in the output distribution of a dolphin input. Therefore, we make the following two assumptions:

1. The distribution of other (i.e. non ground-truth) classes is non-uniform in general.
2. The probability value should correlate positively with the similarity between the true class and any other class, i.e. the more similar they are the higher the value.

In this paper, we propose a novel form of the smooth labeling, called *class-similarity based label smoothing*, which uses class similarities to approximate the optimal output distribution. In the proposed smooth label of a reference class, the label for another class is based on its similarity with the reference class, and hence more similar classes result in higher values.

From the label smoothing perspective, our proposed smooth label is more intuitive than the traditional uniform smooth label. One-hot labels have all probability mass in one class, which are zero-entropy signals that admit no uncertainty about the input. When a network is trained using such labels, it inevitably becomes overconfident. With uniform smooth labels, the output of a network is trained to be a mixture of a Dirac distribution and a uniform distribution. This

implies that the predicted probabilities for other classes are encouraged to be equal. As we discussed above, the label value should be based on the similarity between classes, which is not accounted for in either the one-hot or the uniform smooth labels. Relational information can be crucial and provides high-order properties that can improve performance of a model in various tasks [20].

In our proposed method, we first measure the similarity between classes which is then mixed with the one-hot labels to serve as the final smooth label. Since measuring similarity is in general an open problem [26,32], we adopt different metrics, including a new notion of semantic similarity, and evaluate their effectiveness in confidence calibration on various data benchmarks and architectures.

2 Related Work

Since the pioneering work [7], confidence calibration has raised a great interest in the machine learning community resulting in a great deal of recent work trying to address this issue. Existing calibration techniques can be broadly categorized based on whether or not they are post-hoc methods.

Post-hoc calibration methods use parametric or non-parametric models to transform the network’s predictions based on a held-out validation set to improve calibration. Traditional techniques include Platt scaling [22], Isotonic Regression [30], Bayesian binning [19,30], Histogram binning [30], matrix and vector scaling [7], Beta calibration [11], and temperature scaling [7]. Among these methods, temperature scaling consistently outperforms the other methods [7]. More recent advances include Dirichlet calibration [10], ensemble temperature scaling [31], mutual information based binning [21], and spline recalibration [8]. Non-post-hoc methods are mostly based on adapting the training procedure, including modifying the training loss [17,27], label smoothing [24,18], and data augmentation [25,29].

Another line of related work is Bayesian Neural Networks [5,14,23]. Bayesian methods provide a natural probabilistic representation of uncertainty in deep learning and are well-suited for providing calibrated uncertainty estimation. However, these methods are resource-demanding and hard to scale to modern datasets and architectures [14].

3 Confidence Calibration

In this section, we formally introduce the definition of confidence calibration. We denote the input as $X \in \mathcal{X}$ and label as $Y \in \mathcal{Y} = \{1, 2, \dots, K\}$. Let h be a neural network and $h(k|x)$ is the confidence estimate of the sample x belonging to class k . The prediction is the winning class $\hat{Y} = \arg \max h(Y|X)$ and its associated confidence is the maximum $\hat{P} = \max h(Y|X)$. The number of samples is n .

3.1 Prediction and Output Calibration

In prediction calibration, only the winning class and its associated confidence are considered. With perfect prediction calibration, the confidence estimate \hat{P}

represents a true probability that indicates the likelihood of correctness

$$\mathbb{P}(Y = \hat{Y} \mid \hat{P} = p) = p, \quad \forall p \in [0, 1] \quad (1)$$

The output of a neural network $h(Y|X)$ is the confidence estimate for all classes. For a calibrated model, we would like not only the maximum but all confidences to be calibrated, which means that $h(k|X)$ represents the actual likelihood of $Y = k$ for all classes k . Perfect output calibration is defined as

$$\mathbb{P}(Y = k \mid h(k|X) = p) = p, \quad \forall p \in [0, 1] \text{ and } \forall k \in \{1, 2, \dots, K\} \quad (2)$$

Note that output calibration infers prediction calibration but not vice versa.

3.2 Histogram-Based ECE

Since the probabilities in (1) and (2) cannot be computed using finitely many samples, empirical approximations are developed. The most widely adopted estimator is histogram-based expected calibration error (ECE) [19,7].

The model's output probabilities $p_i^k = h(k|x_i)$ are grouped into M interval bins, where p_i^k is the model's output confidence for sample i belonging to class k . Let B_m be the set of indices of which the confidence falls into bin m , i.e. $p_i^k \in (\frac{m-1}{M}, \frac{m}{M}]$. The likelihood and confidence of B_m are defined as

$$\text{lik}(B_m) = \frac{1}{|B_m|} \sum_{(i,k) \in B_m} \mathbb{1}(y_i = k) \quad \text{and} \quad \text{conf}(B_m) = \frac{1}{|B_m|} \sum_{(i,k) \in B_m} p_i^k \quad (3)$$

where y_i is true class label for sample i . The histogram-based ECE is defined as

$$\text{Histogram ECE} = \sum_{m=1}^M \frac{|B_m|}{nK} |\text{lik}(B_m) - \text{conf}(B_m)| \quad (4)$$

3.3 KDE-Based ECE

While a histogram-based estimator is easy to implement, it inevitably inherits drawbacks from histograms, for example being sensitive to the binning schemes and data-inefficient [12]. KDE-based estimator is proposed in [31] by replacing histograms with density estimation using a continuous kernel.

Let $\phi : \mathbb{R} \rightarrow \mathbb{R}_{\geq 0}$ denote a kernel function and h denote the bandwidth. The density function and canonical calibration function are given by

$$f(p) = \frac{h^{-K}}{n} \sum_{i=1}^n \prod_{k=1}^K \phi\left(\frac{p - p_i^k}{h}\right) \quad \text{and} \quad c(p) = \frac{\sum_{i=1}^n y_i \prod_{k=1}^K \phi\left(\frac{p - p_i^k}{h}\right)}{\sum_{i=1}^n \prod_{k=1}^K \phi\left(\frac{p - p_i^k}{h}\right)} \quad (5)$$

and the KDE-based ECE is computed as

$$\text{KDE ECE} = \int \|p - c(p)\| f(p) dp \quad (6)$$

While KDE alleviates the dependence on histogram binning, it heavily depends on the kernel choice and may induce error from the integral approximation procedure. Therefore, we use both ECE metrics in our evaluation.

4 Learning with Different Labels

In this section, we discuss the learning objective with respect to different labels and present the intuition behind our method. The output of a neural network is a probability distribution over all classes $h(k|x) = \exp(z_k) / \sum_{i=1}^K \exp(z_i)$, where z_k is the activation in the last layer. Let $\pi(k|x)$ be the label corresponding to input x . The model is trained by minimizing the cross-entropy loss

$$l(x) = -\sum_{i=1}^K \pi(k|x) \log(h(k|x)) \quad (7)$$

With one-hot label, the cross entropy loss can be simplified to

$$l(x) = -\log(h(y|x)) \quad (8)$$

Under this loss, the model is not only trained to make a correct prediction but also with the highest confidence z_y possible to reduce the loss, which causes the model to become overconfident. Uniform smooth labels are defined by

$$\pi(k|x) = (1 - \alpha)e_y + \alpha u(k)/K \quad (9)$$

where e_y is the coordinate vector and u is a uniform distribution. The cross-entropy loss with uniform smooth label can be written as

$$l(x) = -(1 - \alpha) \log(h(y|x)) + \alpha H(u, h) \quad (10)$$

The second part of this loss encourages the model to match its output with a uniform distribution, which implies that the example is regarded to be equally probable in any other class. This is clearly not the case in general as some classes are inherently more similar than others. The intuition behind our method is that we assume there is an unknown optimal distribution for perfect calibration $q^*(k|x)$ that satisfies the two assumptions made in Section 1 and can be approximated by estimating the class similarities. Let $\hat{q} \approx q^*$ be the approximated optimal distribution, then our proposed smooth label is defined as

$$\pi(k|x) = (1 - \alpha)e_y + \alpha \hat{q}(k|x) \quad (11)$$

With this label, the cross entropy loss can be written as

$$l(x) = -(1 - \alpha) \log(h(y|x)) + \alpha H(\hat{q}, h) \quad (12)$$

and the model is trained to make a correct classification while matching it output with the approximated optimal probability distribution. Hence, *minimizing $H(\hat{q}, h)$ is a direct optimization for both prediction and output calibration.*

5 Approach

In this section, we describe our approach to capture class similarities in order to compute the approximation distribution \hat{q} and generate our proposed smooth labels. Capturing semantic similarity has been a longstanding and still a wide-open problem [26,32]. Different metrics quantify the similarity from different perspectives, therefore we propose to use several distance metrics, including a novel one based on word2vec mapping of label words, and evaluate the performance under varying notions of the captured similarities.

5.1 Image Space

For directly computing the distance in the image space, we use L_p norms and choose $p = 1$ and $p = 2$. For two inputs x in class k and x' in class k' , the pairwise distances are given by

$$\mathbf{L}_p \text{ Distance} \quad d(x, x') = \|x - x'\|_p \quad (13)$$

5.2 Representation Space

Studies have shown that features learned by neural networks are often surprisingly useful as a representational space for a much wider variety of tasks and match with human perception [4,32]. The latent representations of autoencoders have been shown to be useful for various downstream tasks [1]. Therefore, we also propose to use an autoencoder to map the data to the representation space and compute distances between their encodings.

Autoencoder Distance

$$d(x, x') = \|r(x) - r(x')\|_2 \quad (14)$$

where $r(x)$ is the autoencoder latent encoding of input x .

The inter-class distance between classes k and k' is determined by averaging the distances between all pairs of inputs that belong to them

$$d_k(k') = \frac{1}{|C(k)||C(k')|} \sum_{x \in C(k), x' \in C(k')} d(x, x') \quad (15)$$

where $C(k)$ is the set of data points in class k , and d_k is a vector that contains the distances between class k and all other classes.

5.3 Semantic Space

The previous distance metrics use either the original features of the objects or those generated by a neural network. In general, samples in the same class are expected to share a common semantic meaning, which may not be captured in either feature space. It was observed for example that image visual similarity is not necessarily the same as semantic similarity [3]. The problem of defining semantic similarity has been studied extensively in the NLP literature as well as in various disciplines for which knowledge can be captured through an ontology or a hierarchy of classes [13]. For most datasets, the words used to label each class capture significant semantics of the class [16]. We make use of the advances in NLP based on the labels to define the semantic similarity. Vector representation of words has been shown to successfully capture semantic similarities such that words with similar meaning are mapped to similar points in the vector space [15]. Therefore, we propose to use a word2vec model to map the label words into Euclidean space, and then compute the distances between the vectors.

Word Embedding Distance

$$d_k(k') = \|\mathcal{V}(w(k)) - \mathcal{V}(w(k'))\|_2 \quad (16)$$

where \mathcal{V} is a word2vec model and $w(k)$ be the natural language word associated with class k . This notion can be generalized to the case in which each class is defined by a set of words or a sentence.

5.4 Class-Similarity Based Smooth Label

Class distances are converted into class similarities using the softmax function

$$s_k(k') = \frac{\exp(-\beta d_k(k'))}{\sum_{i=1}^K \exp(-\beta d_k(i))} \quad (17)$$

where $\beta \geq 0$ is a hyperparameter that controls how “uniform” the similarity distribution is. In order to ensure the consistency of β for different metrics, we normalize the distances d_k to zero mean and standard deviation of one before applying the softmax function. Note that this normalization is equivalent to scaling β and does not affect the relative relationship between classes.

Finally, the class-similarity based smooth label is defined as follows. Let e_k be the one-hot label vector, the smooth label for class k is

$$y_k = (1 - \alpha)e_k + \alpha s_k \quad (18)$$

where α is the label smoothing factor. Note that in practice we set $s_k(k) = 0$ and scale $\sum_{k' \neq k} s_k(k') = 1$ to make sure that α consistently represents the total mass in the label over the other (non ground-truth) classes. Otherwise the total mass will be different because of different values of $s_k(k)$.

6 Experiments

6.1 Setup

We compare the four variants of our proposed method with L_1 distance, L_2 distance, Autoencoder distance (AE) and Word Embedding distance (WE) to the vanilla training using one-hot labels, as well as various techniques that improve confidence calibration: temperature scaling (TS) [7], uniform label smoothing [24,18], mixup training [25], Dirichlet calibration with off-diagonal regularization (Dir-ODIR) [10], and ensemble temperature scaling (ETS) [31].

We perform experiments on CIFAR-100 and Tiny-ImageNet using DenseNet (DN), ResNet (RN), and WideResNet(WRN). For all experiments, the network architectures and all parameters are identical for all methods. For CIFAR-100, we use DenseNet-161, ResNet-18, and WRN with {16, 16, 32, 64} filters respectively. For Tiny-ImageNet, we use DenseNet-161, ResNet-34, and WRN-50-2. The dimensionality of the latent space of the autoencoder is set to 256 for CIFAR-100 and 1024 for Tiny-ImageNet. We use a pretrained Wikipedia2Vec [28] as our word2vec model \mathcal{V} with a vector of length 100. For histogram-based ECE, we set the number of interval bins $M = 15$ following [7,18]. For KDE-based ECE, we use the Triweight Kernel $K(u) = \frac{35}{32}(1 - u^2)^3$ and bandwidth $h = 1.06\sigma n^{-1/5}$ [31].

Table 1: Histogram and KDE-based ECE (%) results of prediction (P) and output (O) on CIFAR-100.

Model	ECE	One-hot	TS	Uniform	Mixup	DirODIR	ETS	L_1 (Ours)	L_2 (Ours)	AE(Ours)	WE(Ours)
DN Hist P	16.40	2.08	2.41	4.58	2.10	2.04	2.39	2.65	1.96	1.68	
DN Hist O	35.52	3.55	8.11	8.81	3.54	3.48	4.15	4.74	3.32	2.98	
DN KDE P	15.28	1.90	2.68	4.52	1.95	1.88	2.61	2.72	2.08	1.75	
DN KDE O	28.65	7.01	15.08	13.40	7.09	7.00	12.85	12.92	12.44	11.32	
RN Hist P	22.96	1.98	2.35	2.70	1.85	1.71	2.84	2.76	2.02	1.74	
RN Hist O	50.07	3.13	3.80	5.03	2.98	2.90	5.06	5.10	3.34	2.47	
RN KDE P	21.13	2.16	2.60	2.83	1.84	1.65	2.91	2.76	2.19	1.76	
RN KDE O	39.23	9.49	15.17	12.03	10.42	9.14	11.89	11.15	10.13	9.02	
WRN Hist P	12.17	2.34	2.77	2.29	2.07	1.79	1.68	1.56	0.77	0.72	
WRN Hist O	25.71	5.11	5.41	5.02	4.54	3.91	2.91	2.60	1.64	1.55	
WRN KDE P	11.57	2.35	2.83	2.49	1.99	1.83	1.57	1.48	1.15	1.11	
WRN KDE O	21.13	9.08	9.27	9.47	6.87	6.92	7.59	7.01	4.53	3.73	

Table 2: Histogram and KDE-based ECE (%) results of prediction (P) and output (O) on Tiny-ImageNet.

Model	ECE	One-hot	TS	Uniform	Mixup	DirODIR	ETS	L_1 (Ours)	L_2 (Ours)	AE(Ours)	WE(Ours)
DN Hist P	13.98	3.45	3.77	3.70	3.23	2.16	2.93	3.07	2.54	1.19	
DN Hist O	29.76	11.64	13.27	12.13	10.57	8.80	9.94	10.40	7.74	3.00	
DN KDE P	13.38	3.46	3.72	3.76	3.21	2.21	2.95	3.13	2.60	1.23	
DN KDE O	27.64	18.36	21.01	19.06	17.87	17.48	17.51	18.18	17.56	15.67	
RN Hist P	24.95	4.83	5.24	5.87	5.15	3.12	2.95	2.86	2.49	1.23	
RN Hist O	58.01	9.94	10.92	11.76	10.07	6.52	5.78	5.59	5.05	2.12	
RN KDE P	23.55	4.93	5.21	5.80	5.17	3.01	3.02	2.82	2.57	1.26	
RN KDE O	46.51	22.33	27.57	24.99	23.14	13.15	27.21	28.60	29.97	11.84	
WRN Hist P	22.39	10.15	12.92	8.73	9.79	6.71	8.55	8.03	7.41	4.21	
WRN Hist O	52.14	22.19	25.10	17.00	19.17	14.36	16.87	16.59	13.82	7.93	
WRN KDE P	21.37	10.75	12.90	8.74	9.78	6.55	8.50	7.97	7.45	4.13	
WRN KDE O	43.78	28.14	37.64	26.76	27.34	19.44	25.81	24.80	24.33	11.30	

6.2 Confidence Calibration Results

We show confidence calibration results on CIFAR-100 in Table 1 and Tiny-ImageNet in Table 2. We set $\alpha = 0.1$ and $\beta \in [0.5, 6]$. Parameter effects are discussed in detail in Section 6.3.

The models trained with our proposed method generally outperforms other methods. Among the four distance measures, WE performs the best in all scenarios which is expected considering that the distance in the vector space more faithfully reflects class semantic relationships than pixel-wise distances and the latent encoding distance. WE delivers better confidence calibration than all comparison methods in most cases. By comparing AE with WE, we can see the performance gap is more significant on Tiny-ImageNet than on CIFAR-100. As the complexity of a dataset and the number of classes increase, it becomes more difficult for an autoencoder to learn good representations, which leads to the performance difference. The large output ECE difference between WE and Uniform shows that the uniform distribution is not an optimal objective. We note that

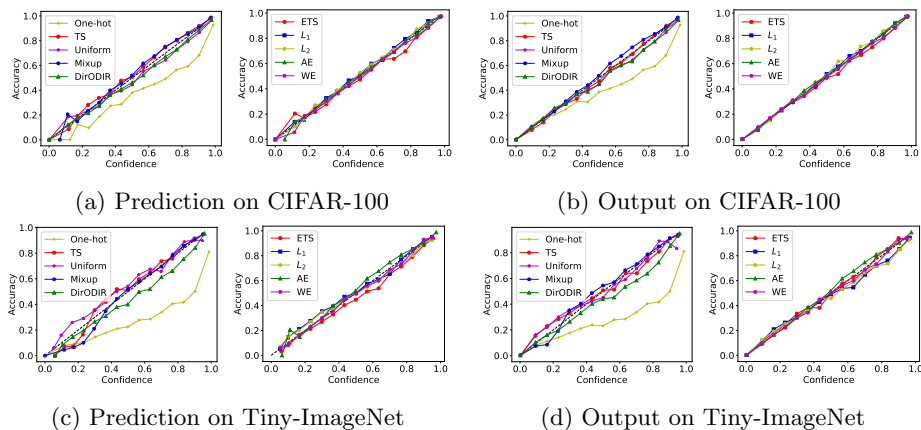


Fig. 1: Reliability diagrams on CIFAR-100 and Tiny-ImageNet dataset.

for output ECE, the KDE estimate is significantly distinct from the histogram estimate. The output probabilities of a model contain a majority of near-zero values, thus in this case the density estimation is not very accurate.

In Figure 1 we show the reliability diagrams for WRN on CIFAR-100 and ResNet-34 on Tiny-ImageNet. The plots confirm our findings of the ECE results. For better readability, we divide the diagrams of the ten techniques into two figures. In all the plots, the dashed black diagonal line represents perfect calibration for which the confidence matches the accuracy. The model trained using one-hot labels is clearly overconfident since the accuracy is always below the confidence. While all other methods have a better performance, diagrams from our proposed smooth labels almost identically match the diagonal line.

6.3 Effects of Parameters

In this section, we perform a series of experiments on Tiny-ImageNet to explore the effects of two important hyper-parameters α and β .

First, we test different α values that determine the strength of label smoothing. We compare WE to uniform LS and set $\beta = 2$. The results are presented in Figure 2a and 2b. We observe that our proposed smooth label generally outperforms uniform smooth label for all α values, and the best results for both methods are achieved at $\alpha = 0.1$ which is the commonly value used in practice. When α is very small, the labels are only weakly smoothed and the model is still over-confident. When α becomes large, the labels are too noisy and the model is not well-calibrated because of the excessive smoothing.

Next, we perform experiments on different β which determines how uniform the similarity distribution is. We choose the WE model and set $\alpha = 0.1$. The results are shown in Figure 2c. When β is too small, the similarity distribution is close to uniform and the class relation is not well represented, therefore the

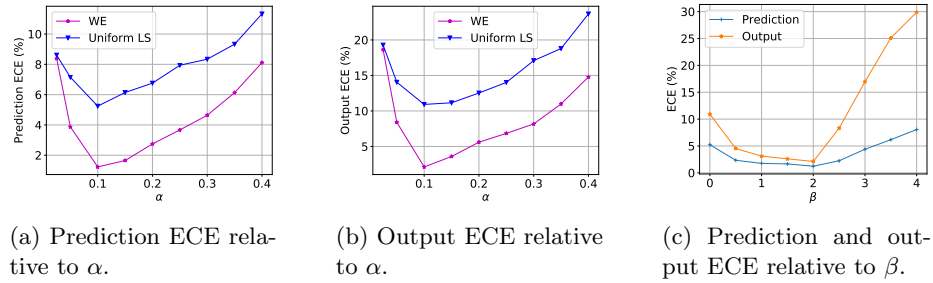


Fig. 2: Effects of parameters.

model is not well calibrated. As β keeps increasing when it becomes too large, the softmax function will produce extreme similarity values that are concentrated in only a few classes that do not represent the optimal distribution either.

6.4 Evaluation on Out-of-Distribution Data

DNNs are not only overconfident on the data they are trained on but also on unseen out-of-distribution data [9,25]. In this section, we evaluate the methods on two types of out-of-distribution data: another unseen dataset and random noise. We test the WRN models trained on CIFAR-100 in Section 6.2. We use the validation set of Tiny-ImageNet as the unseen dataset and generate uniformly distributed random samples as the second type of out-of-distribution data. We show the distributions of the prediction confidence values in Figure 3.

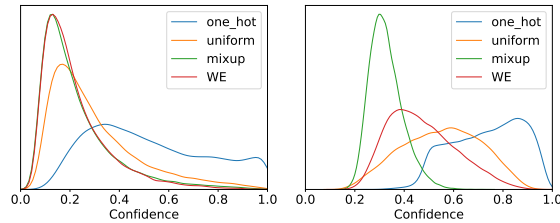


Fig. 3: Distribution of the prediction confidences on the validation set of Tiny-ImageNet (left) and uniform random noise (right).

From the left plot, we can see that our method and mixup perform the best at refraining to produce high confidence on unseen out-of-distribution data. On the random noise samples, all three methods significantly outperform the one-hot training. Note that our method is based on class similarities which is not well-suited for random noise samples where the notion of similarity does not exist. Although this application is not the main focus of our method, we note that it still significantly outperforms the uniform label smoothing in both scenarios.

7 Conclusion

In this paper, we address the confidence calibration problem in a more holistic framework. Motivated by directly optimizing the objective of confidence calibration, we propose class-similarity based label smoothing. We adopt several similarity metrics, including those that capture feature based similarities or semantic similarity. We demonstrate through extensive experiments that our method significantly outperforms state-of-the-art techniques.

References

1. Bengio, Y., Courville, A., Vincent, P.: Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence* **35**(8), 1798–1828 (2013)
2. Brundage, M., Avin, S., Clark, J., Toner, H., Eckersley, P., Garfinkel, B., Dafoe, A., Scharre, P., Zeitsoff, T., Filar, B., et al.: The malicious use of artificial intelligence: Forecasting, prevention, and mitigation. *arXiv preprint arXiv:1802.07228* (2018)
3. Deselaers, T., Ferrari, V.: Visual and semantic similarity in imagenet. In: *CVPR 2011*. pp. 1777–1784. IEEE (2011)
4. Dosovitskiy, A., Brox, T.: Generating images with perceptual similarity metrics based on deep networks. In: *Advances in neural information processing systems*. pp. 658–666 (2016)
5. Gal, Y., Ghahramani, Z.: Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In: *international conference on machine learning*. pp. 1050–1059 (2016)
6. Goodfellow, I., Bengio, Y., Courville, A.: *Deep learning* (2016)
7. Guo, C., Pleiss, G., Sun, Y., Weinberger, K.Q.: On calibration of modern neural networks. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. pp. 1321–1330. JMLR. org (2017)
8. Gupta, K., Rahimi, A., Ajanthan, T., Mensink, T., Sminchisescu, C., Hartley, R.: Calibration of neural networks using splines. *arXiv preprint arXiv:2006.12800* (2020)
9. Hendrycks, D., Gimpel, K.: A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136* (2016)
10. Kull, M., Nieto, M.P., Kängsepp, M., Silva Filho, T., Song, H., Flach, P.: Beyond temperature scaling: Obtaining well-calibrated multi-class probabilities with dirichlet calibration. In: *Advances in Neural Information Processing Systems*. pp. 12316–12326 (2019)
11. Kull, M., Silva Filho, T., Flach, P.: Beta calibration: a well-founded and easily implemented improvement on logistic calibration for binary classifiers. In: *Artificial Intelligence and Statistics*. pp. 623–631. PMLR (2017)
12. Kumar, A., Liang, P.S., Ma, T.: Verified uncertainty calibration. In: *Advances in Neural Information Processing Systems*. pp. 3792–3803 (2019)
13. Lee, W.N., Shah, N., Sundlass, K., Musen, M.: Comparison of ontology-based semantic-similarity measures. In: *AMIA annual symposium proceedings*. vol. 2008, p. 384. American Medical Informatics Association (2008)
14. Maddox, W.J., Izmailov, P., Garipov, T., Vetrov, D.P., Wilson, A.G.: A simple baseline for bayesian uncertainty in deep learning. In: *Advances in Neural Information Processing Systems*. pp. 13153–13164 (2019)

15. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
16. Miller, G.A.: Wordnet: a lexical database for english. *Communications of the ACM* **38**(11), 39–41 (1995)
17. Mukhoti, J., Kulharia, V., Sanyal, A., Golodetz, S., Torr, P.H., Dokania, P.K.: Calibrating deep neural networks using focal loss. arXiv preprint arXiv:2002.09437 (2020)
18. Müller, R., Kornblith, S., Hinton, G.: When does label smoothing help? arXiv preprint arXiv:1906.02629 (2019)
19. Naeini, M.P., Cooper, G., Hauskrecht, M.: Obtaining well calibrated probabilities using bayesian binning. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. vol. 29 (2015)
20. Park, W., Kim, D., Lu, Y., Cho, M.: Relational knowledge distillation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 3967–3976 (2019)
21. Patel, K., Beluch, W., Yang, B., Pfeiffer, M., Zhang, D.: Multi-class uncertainty calibration via mutual information maximization-based binning. arXiv preprint arXiv:2006.13092 (2020)
22. Platt, J., et al.: Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers* **10**(3), 61–74 (1999)
23. Ritter, H., Botev, A., Barber, D.: A scalable laplace approximation for neural networks. In: *6th International Conference on Learning Representations, ICLR 2018- Conference Track Proceedings*. vol. 6. *International Conference on Representation Learning* (2018)
24. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. pp. 2818–2826 (2016)
25. Thulasidasan, S., Chennupati, G., Bilmes, J.A., Bhattacharya, T., Michalak, S.: On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In: *Advances in Neural Information Processing Systems*. pp. 13888–13899 (2019)
26. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing* **13**(4), 600–612 (2004)
27. Xing, C., Arik, S., Zhang, Z., Pfister, T.: Distance-based learning from errors for confidence calibration. arXiv preprint arXiv:1912.01730 (2019)
28. Yamada, I., Asai, A., Shindo, H., Takeda, H., Takefuji, Y.: Wikipedia2vec: an optimized tool for learning embeddings of words and entities from wikipedia. arXiv preprint arXiv:1812.06280 (2018)
29. Yun, S., Han, D., Oh, S.J., Chun, S., Choe, J., Yoo, Y.: Cutmix: Regularization strategy to train strong classifiers with localizable features. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. pp. 6023–6032 (2019)
30. Zadrozny, B., Elkan, C.: Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In: *Icml*. vol. 1, pp. 609–616. Citeseer (2001)
31. Zhang, J., Kailkhura, B., Han, T.: Mix-n-match: Ensemble and compositional methods for uncertainty calibration in deep learning. arXiv preprint arXiv:2003.07329 (2020)
32. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 586–595 (2018)