Confidence-Aware Learning for Deep Neural Networks

Jooyoung Moon [*1] Jihyo Kim [*1] Younghak Shin [2] Sangheum Hwang [1 3]

Abstract

Despite the power of deep neural networks for a wide range of tasks, an overconfident prediction issue has limited their practical use in many safety-critical applications. Many recent works have been proposed to mitigate this issue, but most of them require either additional computational costs in training and/or inference phases or customized architectures to output confidence estimates separately. In this paper, we propose a method of training deep neural networks with a novel loss function, named *Correctness Ranking Loss*, which regularizes class probabilities explicitly to be better confidence estimates in terms of ordinal ranking according to confidence. The proposed method is easy to implement and can be applied to the existing architectures without any modification. Also, it has almost the same computational costs for training as conventional deep classifiers and outputs reliable predictions by a single inference. Extensive experimental results on classification benchmark datasets indicate that the proposed method helps networks to produce well-ranked confidence estimates. We also demonstrate that it is effective for the tasks closely related to confidence estimation, out-of-distribution detection and active learning.

1. Introduction

Deep neural networks have shown remarkable performance on a wide spectrum of machine learning tasks for a variety of domains, e.g., image classification (Krizhevsky et al., 2012), speech recognition (Hinton et al., 2012), and medical diagnosis (Nam et al., 2019). They are, however, generally an overconfident estimator that produces predictive probabilities with high confidence even for incorrect predictions (Nguyen et al., 2015; Szegedy et al., 2014).

The overconfident prediction issue makes deep neural network models unreliable, and therefore limits the deployment of the models in safety-critical applications such as autonomous driving and computer-aided diagnosis. For the successful integration of a deep neural network model into real-world systems, the model must not only be accurate but also indicate when it is likely to be wrong. In other words, *the model should know what it does not know*. Hence, a deep neural network model that provides high quality of confidence estimates is required for practical applications.

The quality of confidence estimates associated with a model's prediction can be assessed in two separate perspectives: confidence calibration and ordinal ranking according to confidence values. Confidence calibration is the problem of predicting probability estimates that reflects the true correctness likelihood. Thus, a well-calibrated classifier outputs predictive probabilities that can be directly interpreted as predictions' confidence level. It is known that modern neural networks generate miscalibrated outputs in spite of their high accuracy (Guo et al., 2017). Guo et al. (2017) examined which factors influence calibration performances of deep neural networks and showed that temperature scaling, a simple post-processing technique to learn a single corrective constant, is very effective at calibrating a model's predictions. Obviously, confidence calibration alone is insufficient to evaluate the quality of predictive confidence since it is orthogonal to both classification accuracy and ranking performance (Kumar et al., 2019). For instance, we can have a perfectly calibrated classifier if it outputs the probability of 0.5 on the two-class dataset consisting of 50% positive and 50% negative samples. It means that a well-calibrated model may show lower predictive performances (Guo et al., 2017; Neumann et al., 2018).

Another view is ordinal ranking of predictions according to their confidence. Intuitively, a prediction with higher confidence value should be more likely to be correct than one with lower confidence value. Thus, ordinal ranking aims to estimate confidence values whose ranking among samples are effective to distinguish correct from incorrect predictions. In most previous studies, this problem has been

[*]Equal contribution [1]Department of Data Science, Seoul National University of Science and Technology, Seoul, Republic of Korea [2]LG CNS, Seoul, Republic of Korea [3]Department of Industrial & Information Systems Engineering, Seoul National University of Science and Technology, Seoul, Republic of Korea. Correspondence to: Sangheum Hwang <shwang@seoultech.ac.kr>.

Proceedings of the 37th International Conference on Machine Learning, Online, PMLR 119, 2020. Copyright 2020 by the author(s).

casted into different tasks such as failure prediction (Hecker et al., 2018; Jiang et al., 2018; Corbière et al., 2019), selective classification (El-Yaniv & Wiener, 2010; Geifman & El-Yaniv, 2017), and out-of-distribution detection (DeVries & Taylor, 2018; Liang et al., 2018; Lee et al., 2018; Hendrycks & Gimpel, 2017; Roady et al., 2019), although they have tried to solve fundamentally similar problem under slightly different settings. A model that outputs well-ranked confidence estimates should work well on all of these tasks. In many practical settings, ordinal ranking performance is important since it is very closely related to measure whether the model knows what it knows. In this work, we focus on how to obtain good predictions in terms of ordinal ranking of confidence estimates.

Our goal is to learn a deep neural network for classification that outputs better predictive probabilities to quantify confidence values. In a classification problem, predictive probabilities by themselves must represent confidence estimates of predictions since a conditional distribution of classes given an input is assumed to be a multinomial distribution. With these probabilities, confidence estimates associated with them can be naturally computed by basic metrics including the maximum class probability (i.e., the largest softmax score), entropy, margin, etc., as commonly used to estimate uncertainty (Settles, 2009). In other words, predictive probabilities from a *well-trained* classifier are the essential ingredients to obtain confidence estimates of high quality.

To build such a well-trained model, we propose a simple but effective regularization method that enforces a model to learn an ordinal ranking relationship. The proposed regularization method can be simply implemented via a ranking loss named *Correctness Ranking Loss* (CRL) that incorporates a comparison of randomly selected a pair of samples. It is minimized when confidence estimates of samples with high probabilities of being correct are greater than those of samples with low probabilities of being correct. The main advantage of the proposed method is its computational efficiency, i.e., we need to compute just an additional loss value during training and can obtain high quality of confidence estimates by a single inference. Therefore, it can be universally applied to any architecture with little increase in computational costs.[1]

We validate the proposed method through extensive experiments over various benchmark datasets for image classification and several popular architectures. The experimental results demonstrate that training with CRL is very effective to obtain well-ranked confidence estimates compared with existing methods specially designed to estimate them. With these well-ranked confidence estimates, it is also shown that a classifier alone works surprisingly well on other complicated tasks such as out-of-distribution (OOD) detection and active learning in which ordinal ranking of confidence is important.

[1]In practice, the amount of computation for calculating loss can be completely negligible.

2. Related Work

Confidence (or its opposite, uncertainty) estimation in predictions with modern neural networks becomes actively studied in the machine learning community. Bayesian approach provides a natural representation of uncertainty in a neural network by allowing rich probabilistic interpretations for a model's predictions. With a prior distribution over model parameters of a neural network, several approximate Bayesian methods can be employed to infer the posterior distribution over the parameters which accounts for predictive uncertainty, for instance, Laplace approximation (MacKay, 1992), Markov Chain Monte Carlo (MCMC) (Neal, 1996) and variational inference (Graves, 2011). While these methods are effective for small neural networks, it is computationally expensive for modern deep neural networks. In the study of Gal & Ghahramani (2016), they proposed Monte Carlo dropout (MCdropout) that uses dropout (Srivastava et al., 2014) at test time to estimate predictive uncertainty by sampling several stochastic predictions. It has gained attention as a practical approximate Bayesian method for uncertainty estimation (Gurau et al., 2018; Zhang et al., 2019). Kendall & Gal (2017) presented a framework to decompose the uncertainty into aleatoric one capturing noise inherent in the data and epistemic one accounting for the model's uncertainty. Although they greatly reduce computational costs for estimating uncertainty, it still requires multiple forward passes for inference.

As another line of study, there are also several works based on non-Bayesian approach to obtain confidence estimates. In standard deep classifiers, predictive class probabilities that can be used for confidence estimation are naturally appeared as softmax outputs. Hendrycks & Gimpel (2017) showed a simple threshold-based method utilizing confidence estimates from softmax scores are quite effective for both ordinal ranking and OOD detection tasks. Liang et al. (2018) introduced an OOD detector named ODIN to improve the OOD detection performances by applying temperature scaling and adding small perturbations to inputs, and Lee et al. (2018) proposed a confidence estimation method using the Mahalanobis distance on feature spaces of deep classifiers which can be further enhanced in conjunction with input perturbations and feature ensembling. Except for Hendrycks & Gimpel (2017), they are designed specifically for the OOD detection task, and none of the ordinal ranking performances was reported.

Some recent studies have tried to directly learn confidence estimates with deep classifiers by augmenting a network's

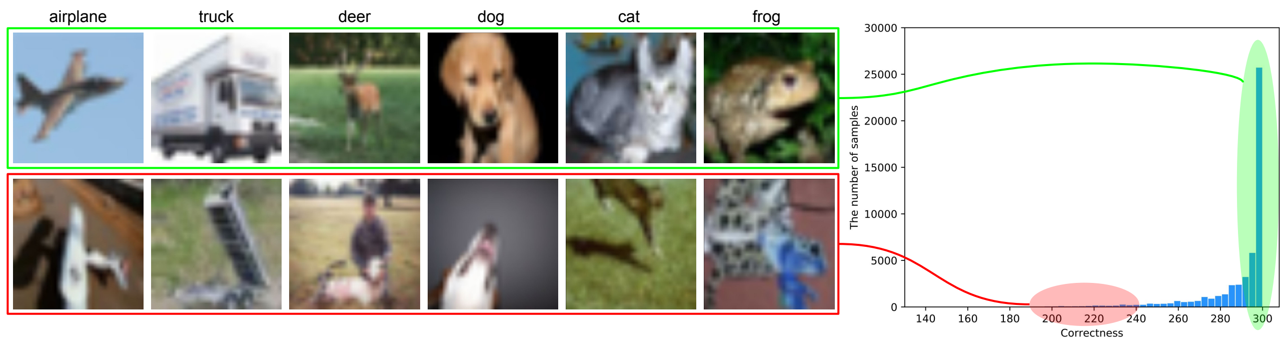


Figure 1. Examples from CIFAR-10 whose number of correct prediction events are high (*green group, top*) and low (*red group, bottom*). The green group consists of easy-to-classify examples while the examples in the red group are hard to recognize.

architecture (DeVries & Taylor, 2018; Corbière et al., 2019). Specifically, they have an additional output node that produces confidence estimates and utilize these estimates for OOD detection (DeVries & Taylor, 2018) or ordinal ranking (Corbière et al., 2019). However, they rely on the predictive performance of confidence estimates from the node since confidence estimates are generated independently of class probabilities.

Ensembles of neural networks, simply called Deep Ensembles, are certainly useful for confidence estimation as can be seen in Lakshminarayanan et al. (2017). Geifman et al. (2019) found that the confidence estimates of easily learnable samples become impaired during training process with a stochastic gradient descent (SGD) based optimizer. To address this issue, they suggest using the Average Early Stopping (AES) algorithm similar to the snapshot ensembles (Huang et al., 2017a) to leverage the quality of confidence estimates in terms of ordinal ranking. However, these approaches are inherently computationally demanding.

Compared to the previous studies, our proposed method neither increases computational costs for training and inference nor augments architectures of standard deep classifiers to have good confidence estimates. With the proposed method, a standard classification network such as examined in Hendrycks & Gimpel (2017) can become a very strong baseline that yields much better confidence estimates.

3. Confidence-Aware Learning

In this section, we introduce the ordinal ranking problem and empirical findings that motivates our work. Then, we provide in-depth descriptions of the proposed *Correctness Ranking Loss* with implementation details.

3.1. Problem Statement

In this work, we address a multi-class classification problem with a deep neural network that utilizes a standard softmax layer to output predicted class probabilities.

Let $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$ be a dataset consisting of n labeled samples from a joint distribution over $\mathcal{X} \times \mathcal{Y}$ where $\mathcal{X}$ is an input space and $\mathcal{Y} = \{1, 2, 3, .., K\}$ is a label set for the classes. A deep neural classifier f is a function $f : \mathcal{X} \rightarrow \mathcal{Y}$ that produces the predicted class probabilities $\mathbf{p}_i = P\,(y|\mathbf{x}_i, \mathbf{w})$ for a sample i where $\mathbf{w}$ is a set of model parameters of the network. With these probabilities, the predicted class $\hat{y}_i$ of an input $\mathbf{x}_i$ is determined as $\hat{y}_i = \mathrm{argmax}_{y\in\mathcal{Y}}\, P(y|\mathbf{x}_i, \mathbf{w})$.

From the predicted class probabilities computed by a softmax layer, we can have several confidence estimates: a class probability associated with $\hat{y}_i$ (i.e., the maximum class probability), negative entropy[2], and margin. Margin is defined as the difference between the predicted probabilities of the first and second most probable classes (Settles, 2009).

Ordinal ranking, also known as failure prediction (Corbière et al., 2019) or error detection (Hendrycks & Gimpel, 2017), is the problem about ranking among samples to distinguish correct from incorrect predictions according to their confidence estimates. In case of perfect ordinal ranking, every pair of $(\mathbf{x}_i, y_i)$ and $(\mathbf{x}_j, y_j)$ from the true joint distribution should hold the following relationship:

$$\begin{aligned} &\kappa(\mathbf{p}_i|\mathbf{x}_i, \mathbf{w}) \leq \kappa(\mathbf{p}_j|\mathbf{x}_j, \mathbf{w}) \\ \Longleftrightarrow\ &P(\hat{y}_i = y_i|\mathbf{x}_i) \leq P(\hat{y}_j = y_j|\mathbf{x}_j) \end{aligned} \tag{1}$$

where κ denotes a confidence function (e.g., the maximum class probability, negative entropy, and margin). Note that $P(\hat{y}_i = y_i|\mathbf{x}_i)$ represents the true probability of being correct for a sample i. It is desirable for a model to learn the relationship in Eq. (1) during training.

3.2. Motivation

Ideally we expect that a model can learn the relationship in Eq. (1) directly during training. However, estimating the

[2]For entropy, confidence should be the negative of entropy.

true probability of getting a correct prediction is the major obstacle. It is generally impractical since we do not know the true joint distribution over $\mathcal{X} \times \mathcal{Y}$ and a classifier f is gradually biased towards the training dataset as training proceeds.

We hypothesis that the probability of being correct is *roughly* proportional to the frequency of correct predictions during training with SGD-based optimizers. The empirical findings in Toneva et al. (2019) and Geifman et al. (2019) support our hypothesis. Toneva et al. (2019) investigated the number of forgetting events for each sample and showed that samples being frequently forgotten are relatively more difficult to classify. Similarly, Geifman et al. (2019) observed that easy-to-classify samples are learned earlier during training compared to hard-to-classify samples. Motivated by these findings, we expect that the frequency of correct prediction events for each sample examined on SGD-based optimizer's trajectory can be used as a good proxy for the probability estimates of being correct for it.[3]

Figure 1 shows the distribution of correct prediction events for training data and examples sampled according to their number of correct prediction events. For this visual inspection, we trained PreAct-ResNet110 (He et al., 2016) on CIFAR-10 dataset (Krizhevsky & Hinton, 2009) for 300 epochs. To count the correct prediction events of each sample, we consider only samples in the current mini-batch, and therefore all samples are examined once per epoch. The top green box contains the images that are correctly classified with high frequency and the bottom red box consists of less correctly classified images. Examples in the green box contain the complete shape of objects with a clearly distinguishable background, and therefore they are easy to recognize. On the other hand, examples in the red box are hard to classify into true classes since the objects appear as a part or with other unrelated objects. Based on this observation, we suppose that it is able to estimate the probability of being classified correctly by the frequency of correct prediction events.

3.3. Correctness Ranking Loss (CRL)

It is enabled to design a loss function to reflect the desirable ordinal ranking of confidence estimates in Eq. (1) if the true class probability is estimated by how many times a sample is classified correctly during training. The loss function should be affected by whether the ranking of a pair of samples is right or not, and the loss will be incurred when the relationship in Eq. (1) is violated.

We propose CRL so that a classifier learns the ordinal ranking relationship. For a pair of $\mathbf{x}_i$ and $\mathbf{x}_j$, it is defined as

$$\mathcal{L}_{\text{CR}}(\mathbf{x}_i, \mathbf{x}_j) = \max(0, -g(c_i, c_j)(\kappa_i - \kappa_j) + |c_i - c_j|) \quad (2)$$

where c_i is the proportion of correct prediction events of $\mathbf{x}_i$ over the total number of examinations (i.e., $c_i \in [0, 1]$), κ_i represents $\kappa(\mathbf{p}_i|\mathbf{x}_i, \mathbf{w})$ and

$$g(c_i, c_j) = \begin{cases} 1, & \text{if } c_i > c_j \\ 0, & \text{if } c_i = c_j \\ -1, & \text{otherwise} \end{cases}$$

As can be seen in Figure 1, in general, the distribution of correct prediction events is highly skewed to the left especially for modern neural networks showing high performance. It means that most training samples are correctly classified during the whole course of training. For those samples, learning the ranking relationship is meaningless. Moreover, our model should learn more from a pair of samples with a large difference in c values rather than those with a small difference. To this end, we introduce some margin $|c_i - c_j|$ to CRL. As a result, CRL enforces a model to output well-ranked κ_i's. For example, for a pair with $c_i > c_j$, CRL will be zero when κ_i is larger than $\kappa_j + |c_i - c_j|$, but otherwise a loss will be incurred. Given a labeled dataset $\mathcal{D}$, the total loss function $\mathcal{L}$ is a weighted sum of a cross-entropy loss $\mathcal{L}_{\text{CE}}$ and a CRL $\mathcal{L}_{\text{CR}}$:

$$\mathcal{L} = \sum_{(\mathbf{x}_i, y_i) \in \mathcal{D}} \mathcal{L}_{\text{CE}}(\mathbf{p}_i, y_i) + \lambda \sum_{(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{D}_C} \mathcal{L}_{\text{CR}}(\mathbf{x}_i, \mathbf{x}_j) \quad (3)$$

where λ is a constant controlling the influence of $\mathcal{L}_{\text{CR}}$ and $\mathcal{D}_C$ denotes a set of all possible sample pairs from $\mathcal{D}$.

Implementation details. With a mini-batch of size b, $\{(\mathbf{x}_{[i]}, y_{[i]})\}_{i=1}^{b}$, $\mathcal{L}_{\text{CR}}$ should be computed over all possible sample pairs at each model update. However, it is computationally expensive, so we employ a few approximation schemes following to Toneva et al. (2019) for reducing the costs. First, only samples in the current mini-batch are examined to determine whether each sample is correctly classified or not as done in Section 3.2. Note that it can be judged by softmax outputs with no costs. Second, since the number of all possible pairs within a mini-batch is too large, we consider only b pairs to include as many pairs of samples as the computational cost is manageable. Specifically, for $i = 1, \dots, b-1$, $\mathbf{x}_{[i]}$ is paired with $\mathbf{x}_{[i+1]}$ and the last sample $\mathbf{x}_{[b]}$ is paired with $\mathbf{x}_{[1]}$.

For the confidence function κ, we consider simple and popular three estimators: the maximum class probability, negative entropy, and margin. Confidence estimates from the maximum class probability and margin as well as c_i naturally lies in $[0, 1]$ while those from negative entropy does not. Therefore, confidence estimates obtained from negative entropy are normalized by using the min-max scaling.

[3]Strictly speaking, this is not an appropriate estimator of probability in a statistical sense since correct prediction events are not *i.i.d.* observations.

Table 1. Comparison of the quality of confidence estimates on various datasets and architectures. The means and standard deviations over five runs are reported. ↓ and ↑ indicate that lower and higher values are better respectively. For each experiment, the best result is shown in boldface. AURC and E-AURC values are multiplied by 10^3, and NLL are multiplied by 10 for clarity. All remaining values are percentage.

Dataset Model	Method	ACC (↑)	AURC (↓)	E-AURC (↓)	AUPR-Err (↑)	FPR-95% TPR (↓)	ECE (↓)	NLL (↓)	Brier (↓)
CIFAR-10 VGG-16	Baseline	93.74±0.14	7.10±0.31	5.10±0.26	44.19±0.34	41.43±0.38	5.20±0.11	3.79±0.11	11.30±0.21
	AES(k=10)	**93.97±0.12**	7.15±0.25	5.30±0.25	44.47±1.00	41.01±1.75	1.61±0.27	2.06±0.04	9.26±0.15
	MCdropout	93.78±0.27	6.72±0.28	4.72±0.19	45.08±2.14	41.52±2.83	1.11±0.19	1.93±0.05	9.34±0.39
	Aleatoric+MC	93.91±0.13	**6.57±0.29**	**4.68±0.22**	44.67±1.76	41.68±1.86	**0.86±0.12**	**1.89±0.05**	**9.08±0.24**
	CRL-softmax	93.82±0.18	6.78±0.18	4.83±0.08	**46.79±1.75**	**40.21±2.18**	1.24±0.20	2.09±0.04	9.33±0.21
CIFAR-10 ResNet110	Baseline	94.11±0.20	9.11±0.44	7.34±0.39	42.70±1.59	40.42±2.30	4.46±0.16	3.34±0.13	10.19±0.32
	AES(k=10)	94.22±0.22	6.71±0.54	5.00±0.44	44.31±2.00	39.80±2.35	1.38±0.15	1.94±0.05	8.82±0.32
	MCdropout	94.25±0.00	**5.48±0.19**	**3.80±0.16**	45.21±2.19	**36.74±3.06**	1.45±0.15	1.88±0.05	8.48±0.13
	Aleatoric+MC	**94.33±0.09**	6.02±0.33	4.38±0.30	**45.55±0.87**	38.72±1.82	1.25±0.07	**1.80±0.03**	**8.36±0.12**
	CRL-softmax	94.00±0.12	6.02±0.26	4.21±0.19	45.20±1.15	38.81±1.59	**1.23±0.18**	1.81±0.04	8.85±0.20
CIFAR-10 DenseNet	Baseline	94.87±0.23	5.15±0.35	3.82±0.30	44.21±2.21	36.35±2.02	3.20±0.20	2.23±0.09	8.33±0.37
	AES(k=10)	**95.00±0.14**	5.31±0.32	4.04±0.26	43.29±1.83	37.13±2.69	1.00±0.10	1.66±0.04	**7.65±0.27**
	MCdropout	94.69±0.25	5.30±0.38	3.85±0.28	45.64±2.65	36.61±2.38	1.20±0.09	1.73±0.05	7.92±0.28
	Aleatoric+MC	94.73±0.19	5.17±0.20	3.76±0.14	**45.67±3.18**	**34.69±1.03**	1.25±0.06	1.72±0.04	7.80±0.16
	CRL-softmax	94.71±0.09	**4.92±0.14**	**3.49±0.94**	45.16±2.12	36.13±3.35	**0.87±0.07**	**1.60±0.02**	7.84±0.17
CIFAR-100 VGG-16	Baseline	73.49±0.34	77.33±1.15	38.61±0.66	68.59±0.64	62.01±0.39	19.81±0.33	17.77±0.37	44.85±0.51
	AES(k=10)	**74.68±0.25**	72.25±1.13	37.09±0.58	67.69±0.76	60.88±0.92	7.42±0.26	10.02±0.11	**35.83±0.36**
	MCdropout	73.06±0.42	77.36±1.15	37.85±0.51	67.68±0.95	62.39±2.16	3.37±0.37	10.05±0.02	36.59±0.29
	Aleatoric+MC	73.12±0.28	77.31±1.00	37.43±0.42	67.67±0.53	63.53±0.81	**3.22±0.19**	**10.02±0.04**	36.63±0.21
	CRL-softmax	74.06±0.18	**71.83±0.47**	**34.84±0.57**	**69.60±1.11**	**59.47±1.01**	13.86±0.27	13.10±0.12	39.42±0.19
CIFAR-100 ResNet110	Baseline	72.85±0.30	87.24±1.21	46.50±1.09	66.01±0.43	66.03±1.52	16.58±0.16	15.09±0.14	42.83±0.38
	AES(k=10)	73.65±0.29	79.12±1.07	40.88±0.49	66.72±0.74	63.81±1.40	8.90±0.15	10.67±0.13	37.67±0.37
	MCdropout	74.08±0.00	75.47±1.07	38.53±1.13	66.14±1.68	64.59±1.46	5.35±0.32	10.06±0.15	36.06±0.38
	Aleatoric+MC	**74.50±0.24**	**73.26±0.83**	37.56±0.95	65.65±0.91	63.53±1.78	**2.68±0.25**	**9.24±0.13**	**34.96±0.20**
	CRL-softmax	74.16±0.32	73.59±1.39	**36.90±1.08**	**67.23±1.13**	**62.56±1.26**	11.52±0.36	10.87±0.05	37.71±0.44
CIFAR-100 DenseNet	Baseline	75.39±0.29	71.75±0.89	38.63±0.72	65.18±1.71	63.30±1.93	12.67±0.25	11.54±0.08	37.26±0.21
	AES(k=10)	76.10±0.16	67.18±0.37	36.04±0.18	64.82±0.83	62.59±0.69	6.78±0.37	9.39±0.04	34.04±0.14
	MCdropout	75.80±0.36	66.92±1.45	34.97±0.46	65.11±1.10	63.27±1.47	**5.59±0.33**	9.49±0.14	34.02±0.38
	Aleatoric+MC	75.50±0.39	67.87±1.55	35.05±0.65	**65.92±1.38**	**61.69±1.79**	6.01±0.22	9.45±0.13	34.25±0.47
	CRL-softmax	**76.82±0.26**	**61.77±1.07**	**32.57±0.81**	65.22±1.40	61.79±2.20	8.59±0.17	**9.11±0.09**	**33.39±0.28**
SVHN VGG-16	Baseline	96.20±0.10	5.97±0.28	5.24±0.28	41.15±0.95	32.08±0.56	3.15±0.11	2.69±0.05	6.86±0.17
	AES(k=10)	96.54±0.09	4.59±0.10	3.98±0.11	**43.48±0.86**	**27.40±0.99**	0.54±0.09	1.34±0.01	5.31±0.06
	MCdropout	96.79±0.05	4.64±0.34	4.12±0.31	41.62±1.21	27.46±0.95	**0.36±0.02**	**1.25±0.03**	**4.96±0.11**
	Aleatoric+MC	**96.80±0.01**	4.86±0.26	4.34±0.26	41.14±0.60	27.60±1.45	0.38±0.07	1.26±0.01	4.99±0.02
	CRL-softmax	96.55±0.07	**4.47±0.10**	**3.86±0.08**	42.82±1.35	29.82±1.42	0.88±0.12	1.52±0.03	5.44±0.10
SVHN ResNet110	Baseline	96.45±0.06	8.02±0.76	7.38±0.75	38.83±1.79	35.78±1.45	2.79±0.06	2.38±0.04	6.25±0.12
	AES(k=10)	96.77±0.05	4.41±0.17	3.89±0.16	**43.56±2.51**	**27.39±1.34**	**0.43±0.11**	1.26±0.01	4.97±0.05
	MCdropout	97.00±0.00	4.99±0.35	4.53±0.34	39.10±0.94	28.69±2.22	0.65±0.07	1.29±0.01	4.73±0.13
	Aleatoric+MC	**97.01±0.04**	5.54±0.24	5.09±0.23	38.71±1.08	31.60±0.50	0.54±0.05	**1.25±0.01**	**4.69±0.05**
	CRL-softmax	96.81±0.09	**4.25±0.12**	**3.74±0.14**	43.46±1.78	27.71±0.56	0.85±0.09	1.31±0.02	4.97±0.12
SVHN DenseNet	Baseline	96.40±0.08	7.70±0.41	7.05±0.39	39.43±0.78	34.23±1.21	2.51±0.07	2.10±0.05	6.13±0.15
	AES(k=10)	96.78±0.08	4.50±0.16	3.98±0.15	**43.43±1.39**	**26.16±1.17**	**0.41±0.09**	**1.24±0.02**	**4.96±0.10**
	MCdropout	96.82±0.04	5.10±0.52	4.59±0.51	39.57±2.58	31.04±1.67	0.42±0.06	1.29±0.03	4.97±0.11
	Aleatoric+MC	**96.86±0.14**	5.68±1.19	5.18±1.15	39.09±2.28	31.43±3.61	0.79±0.87	1.44±0.35	5.05±0.41
	CRL-softmax	96.61±0.12	**4.47±0.14**	**3.89±0.13**	43.35±0.81	28.35±1.62	0.85±0.06	1.38±0.04	5.26±0.18

4. Experiments

First, we evaluate our method on the ordinal ranking task with image classification benchmark datasets. Then, the performances on out-of-distribution detection and active learning tasks are presented. The subsequent subsections provide about experimental settings and results of each task. More details on datasets, evaluation metrics, and methods for comparison are available in the supplementary material. Our code is available at https://github.com/daintlab/confidence-aware-learning.

4.1. Ordinal Ranking

In this section, we examine how well confidence estimates obtained from a deep classifier trained with CRL is ranked according to the correctness of predictions. This is our primary goal in order to build a classifier being immune to the overconfident prediction issue.

Experimental settings. We evaluate our method on benchmark datasets for image classification: SVHN (Netzer et al., 2011) and CIFAR-10/100 (Krizhevsky & Hinton, 2009). For models to compare, we consider popular deep neural network architectures: VGG-16 (Simonyan & Zisserman, 2015), PreAct-ResNet110 (He et al., 2016) and DenseNet-BC ($k = 12, d = 100$) (Huang et al., 2017b). All models are trained using SGD with a momentum of 0.9, an initial learning rate of 0.1, and a weight decay of 0.0001 for 300 epochs with the mini-batch size of 128. The learning rate is reduced by a factor of 10 at 150 epochs and 250 epochs. We employ a standard data augmentation scheme, i.e., random horizontal flip and 32×32 random crop after padding with 4 pixels on each side.

Table 2. Comparison of ensembles of five classifiers. For each experiment, the best result is shown in boldface. AURC and E-AURC values are multiplied by 10^3, and NLL are multiplied by 10 for clarity. All remaining values are percentage.

Dataset Model	**Method**	**ACC (↑)**	**AURC (↓)**	**E-AURC (↓)**	**AUPR-Err (↑)**	**FPR-95% TPR (↓)**	**ECE (↓)**	**NLL (↓)**	**Brier (↓)**
CIFAR-10	Baseline	95.02	4.45	3.19	**46.45**	**33.73**	1.52	1.92	7.65
VGG-16	CRL-softmax	**95.09**	**4.32**	**3.09**	45.27	37.88	**1.32**	**1.78**	**7.51**
CIFAR-10	Baseline	95.42	4.01	2.95	**44.14**	**29.03**	1.12	1.63	6.86
ResNet110	CRL-softmax	**95.55**	**3.72**	**2.72**	44.01	29.88	**0.84**	**1.50**	**6.60**
CIFAR-10	Baseline	**96.03**	**3.02**	**2.22**	44.17	30.73	**0.79**	1.29	**5.97**
DenseNet	CRL-softmax	95.97	3.17	2.35	**45.25**	**29.77**	0.85	**1.27**	5.99
CIFAR-100	Baseline	78.34	54.53	29.16	64.99	58.44	4.07	9.53	31.05
VGG-16	CRL-softmax	**78.53**	**52.53**	**27.63**	**66.53**	**57.89**	**3.80**	**9.11**	**30.47**
CIFAR-100	Baseline	78.83	54.91	30.72	64.42	58.99	2.39	8.63	30.19
ResNet110	CRL-softmax	**79.08**	**52.87**	**29.27**	**64.88**	**57.74**	**2.11**	**8.06**	**29.59**
CIFAR-100	Baseline	80.34	47.43	26.70	**63.83**	**56.10**	1.87	7.43	27.74
DenseNet	CRL-softmax	**80.85**	**45.63**	**25.99**	61.46	57.33	**1.79**	**7.13**	**27.34**
SVHN	Baseline	96.91	4.48	4.00	**40.66**	**28.64**	1.09	1.60	4.93
VGG-16	CRL-softmax	**96.95**	**4.07**	**3.60**	40.52	29.25	**1.02**	**1.53**	**4.92**
SVHN	Baseline	97.13	4.33	3.91	**42.52**	**26.30**	0.92	1.38	4.47
ResNet110	CRL-softmax	**97.29**	**3.80**	**3.43**	40.75	26.80	**0.88**	**1.23**	**4.26**
SVHN	Baseline	**97.24**	4.93	4.55	36.49	30.54	**0.83**	1.34	4.51
DenseNet	CRL-softmax	97.18	**4.10**	**3.70**	**43.31**	**29.05**	0.87	**1.25**	**4.46**

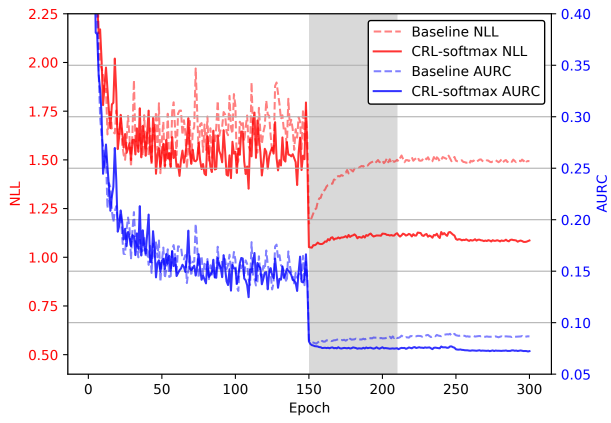


Figure 2. Comparison of AURC (blue) and NLL (red) curves of Baseline and CRL model with PreAct-ResNet110 on CIFAR-100. Dashed and solid line represents the curves from Baseline and CRL model respectively.

For learning with CRL (CRL model), we set λ in Eq. (3) to 1.0 without the hyperparameter search process. Note that when estimating confidence from a model trained with CRL, we use the κ that is utilized for training. For example, when we set κ as the maximum class probability, the confidence function used to evaluate metrics is also the maximum class probability. We compare the performance of CRL model with a standard deep classifier trained with only $\mathcal{L}_{\text{CE}}$ (hereafter referred to as Baseline), MCdropout (Gal & Ghahramani, 2016), Aleatoric+MCdropout (Kendall & Gal, 2017) and AES (Geifman et al., 2019) with 10 and 30 snapshot models. For MCdropout and Aleatoric+MCdropout, entropy on the predicted class probabilities averaged over 50 stochastic predictions is used as uncertainty estimates (Kendall & Gal, 2017; Corbière et al., 2019). The maximum of the averaged class probabilities from snapshot models is used to measure confidence for AES (Geifman et al., 2019).

Evaluation metrics. We evaluate the quality of confidence estimates in terms of both ordinal ranking and calibration. To measure the ordinal ranking performance, commonly used metrics are employed: the area under the risk-coverage curve (AURC) that is defined to be risk (i.e., error rate) as a function of coverage, *Excess*-AURC (E-AURC) that is a normalized AURC (Geifman et al., 2019), the area under the precision-recall curve using errors as the positive class (AUPR-Error) (Corbière et al., 2019), and the false positive rate at 95% true positive rate (FPR-95%-TPR). For calibration, we use the expected calibration error (ECE) (Naeini et al., 2015), the Brier score (Brier, 1950) and negative log likelihood (NLL).

Results.[4] Comparative results are summarized in Table 1. CRL-softmax in the table means CRL model using the maximum class probability as a confidence estimator. From the results, we observe that a standard deep classifier trained with CRL improves both classification accuracy and the quality of confidence estimates of Baseline. For example, in case of DenseNet on CIFAR-100, CRL-softmax has 1.43% higher accuracy than Baseline and shows greatly improved confidence estimates evaluated over all performance metrics. It implies that CRL is an effective regularizer encouraging a classifier to produce good probabilistic predictions. Surprisingly, we observe that CRL model outputs comparable or better confidence estimates compared to MCdropout, Aleatoric+MCdropout and AES which require multiple stochastic predictions or snapshot models. For instance, CRL model outperforms the competing methods in 7 cases among all 9 experiments in terms of AURC. The results demonstrate that training with CRL is very effective to build a reliable and strong baseline being comparable to such methods.

We also examine whether Deep Ensembles (Lakshminarayanan et al., 2017) benefits from CRL. Table 2 presents the comparison results of ensembles based on five Baseline and five CRL models. For these experiments, we set λ to 0.5 for CRL models since we empirically found that the ensemble of CRL models with $\lambda = 1$ does not improve Baseline ensemble on CIFAR-10 except other datasets (refer to Table S5 in the supplementary material). We infer that it is because CRL acts as a strong regularizer so the trained CRL models with a large λ from random initial points lose their diversity. Thus, we use smaller λ to address this diversity issue. With $\lambda = 0.5$, it is observed that CRL also improves Deep Ensembles. One notable point from the results is that although CRL is designed to learn better confidence estimates in terms of ordinal ranking, it is also beneficial to calibration performance.

[4]Due to the space limitation, we present a subset of results. Complete results can be found in the supplementary material.

Table 3. Performances of CRL model on out-of-distribution detection task. The means and standard deviations are computed from five models presented in Section 4.1. For each comparison, better result is shown in boldface. All values are percentage.

In-dist Model	Out-of-dist	FPR-95%TPR(↓)	Detection Err(↓)	AUROC(↑)	AUPR-In(↑)	AUPR-Out(↑)
				Baseline / CRL		
				Baseline+ODIN / CRL+ODIN		
				Baseline+Mahalanobis / CRL+Mahalanobis		
SVHN ResNet110	TinyImageNet	29.65±2.40 / **5.89±0.70**	12.11±0.96 / **5.05±0.23**	93.00±1.06 / **98.83±0.13**	96.31±0.91 / **99.56±0.04**	84.95±1.41 / **96.72±0.50**
		27.50±3.09 / **2.17±0.26**	13.14±1.29 / **3.41±0.23**	92.32±1.38 / **99.39±0.08**	95.63±1.17 / **99.76±0.03**	85.61±1.85 / **98.41±0.30**
		0.24±0.08 / 0.30±0.12	**1.15±0.16** / 1.39±0.27	**99.88±0.03** / 99.82±0.06	**99.96±0.01** / 99.93±0.02	**99.39±0.19** / 98.99±0.28
	LSUN	32.37±2.78 / **7.48±0.91**	13.01±1.17 / **5.50±0.20**	92.19±1.39 / **98.62±0.17**	95.82±1.30 / **99.49±0.06**	83.48±1.74 / **96.14±0.62**
		29.57±3.98 / **2.92±0.51**	14.06±1.23 / **3.88±0.30**	91.56±1.78 / **99.28±0.11**	95.19±1.65 / **99.72±0.03**	84.42±2.38 / **98.06±0.42**
		0.08±0.05 / **0.06±0.07**	0.88±0.14 / **0.85±0.32**	**99.91±0.03** / 99.89±0.06	**99.97±0.01** / 99.96±0.02	**99.45±0.25** / 99.03±0.38
SVHN DenseNet	TinyImageNet	26.32±5.55 / **7.99±2.49**	11.49±1.61 / **5.75±0.71**	93.75±1.43 / **98.53±0.41**	96.62±0.94 / **99.43±0.16**	87.30±2.74 / **96.15±1.16**
		19.93±4.43 / **3.39±1.34**	11.46±1.80 / **4.04±0.80**	94.06±1.47 / **99.17±0.28**	96.56±0.94 / **99.65±0.12**	90.03±2.39 / **98.00±0.67**
		1.44±1.62 / **1.03±1.41**	2.42±1.15 / **1.86±0.92**	99.37±0.91 / **99.48±0.79**	99.52±1.08 / **99.62±0.99**	98.45±1.04 / **98.48±0.80**
	LSUN	28.95±5.80 / **11.05±3.09**	12.39±1.84 / **6.58±0.74**	92.95±1.76 / **98.12±0.49**	96.11±1.23 / **99.29±0.19**	85.93±3.12 / **95.06±1.40**
		22.22±4.86 / **4.63±1.84**	12.35±1.98 / **4.68±0.94**	93.32±1.80 / **98.93±0.36**	96.15±1.22 / **99.56±0.15**	88.83±2.73 / **97.45±0.89**
		0.41±0.84 / 0.44±0.46	**1.23±0.63** / 1.23±0.66	99.73±0.60 / **99.75±0.17**	**99.86±0.88** / 99.79±0.13	**98.97±0.60** / 98.70±0.62
CIFAR-10 ResNet110	TinyImageNet	66.09±2.86 / **53.17±5.60**	22.59±1.81 / **22.06±2.35**	82.59±2.91 / **86.25±2.76**	79.63±5.39 / **86.56±3.27**	82.07±2.00 / **85.61±2.50**
		49.33±4.19 / **43.08±5.15**	22.08±2.28 / **17.69±2.02**	84.31±3.22 / **90.40±1.91**	80.73±5.07 / **90.77±2.13**	86.01±2.30 / **90.03±1.77**
		8.46±2.12 / 9.44±2.25	**6.39±0.87** / 7.02±0.91	**98.34±0.41** / 97.92±0.41	**98.40±0.37** / 97.85±0.35	**98.22±0.49** / 98.02±0.44
	LSUN	57.65±2.89 / **44.53±6.57**	**17.78±1.21** / 17.89±1.87	88.25±1.54 / **90.46±1.93**	87.73±2.59 / **91.37±1.95**	87.06±1.29 / **89.60±1.97**
		34.72±5.75 / **32.10±5.29**	16.29±1.76 / **13.50±1.64**	90.63±1.97 / **93.90±1.23**	88.98±2.79 / **94.48±1.21**	91.47±1.66 / **93.30±1.25**
		6.33±2.54 / **5.52±1.39**	5.51±1.25 / **5.16±0.76**	98.66±0.51 / **98.71±0.29**	**98.79±0.48** / 98.76±0.25	98.49±0.61 / **98.62±0.31**
CIFAR-10 DenseNet	TinyImageNet	45.81±3.95 / **29.87±4.09**	13.15±1.41 / **12.99±1.03**	93.25±1.04 / **94.50±0.84**	94.53±0.94 / **95.17±0.71**	91.82±1.24 / **93.87±1.03**
		10.73±6.24 / **10.41±3.09**	7.09±2.02 / **6.89±1.10**	97.86±1.09 / **97.97±0.56**	97.90±1.04 / **98.16±0.48**	**97.84±1.13** / 97.78±0.65
		6.99±1.13 / **6.28±3.18**	5.92±0.58 / **5.61±1.54**	98.37±0.50 / **98.52±1.17**	**98.22±1.29** / 98.09±2.07	98.49±0.38 / **98.57±0.82**
	LSUN	36.31±3.64 / **21.22±2.73**	10.60±0.88 / **10.59±0.55**	95.18±0.64 / **96.34±0.44**	96.16±0.50 / **96.80±0.35**	94.14±0.86 / **95.94±0.57**
		4.32±2.55 / 5.29±1.53	**4.46±1.15** / 5.03±0.71	**99.04±0.46** / 98.81±0.29	**99.10±0.41** / 98.92±0.24	**98.99±0.50** / 98.70±0.35
		5.27±1.15 / **3.86±2.15**	5.08±0.57 / **4.26±1.11**	98.73±0.50 / **98.89±0.66**	**98.68±1.56** / 98.67±1.07	98.71±0.36 / **98.91±0.50**

To further understand the effect of CRL, NLL and AURC curves on CIFAR-100 test set are shown in Figure 2. NLL curves from Baseline and CRL model show that CRL effectively regularizes a classifier. Also, in Baseline model, we can observe a natural trend that overfitting to NLL leads to poor ordinal ranking as can be seen in the shaded area. Remarkably, training with CRL, however, further improves the ranking performance even when the model slightly overfits to NLL. This observation supports the regularization effect on training a classifier with CRL.

4.2. Out-of-Distribution Detection (OOD)

OOD detection is the problem of identifying inputs that come from the distribution (i.e., out-of-distribution) sufficiently different from the training distribution (i.e., in-distribution). Through the experiments, we demonstrate that a classifier trained with CRL separate well in- and out-of-distribution samples.

Experimental settings. Following DeVries & Taylor (2018), we use two in-distribution datasets: SVHN and CIFAR-10. For the out-of-distribution datasets, we use TinyImageNet[5], LSUN (Yu et al., 2015), and iSUN (Xu et al., 2015). Also, we utilize five Baseline and CRL-softmax models that are trained previously for Section 4.1.

First, we compare the OOD detection performance of Baseline models with CRL-softmax models. Then, we investigate whether ODIN (Liang et al., 2018) and the Mahalanobis detector (Lee et al., 2018) combined with CRL models can improve the detection performance further. ODIN and Mahalanobis are post-processing methods that boost the OOD detection performance of a pre-trained classifier significantly, which have the hyperparameters: a temperature T for ODIN, and a perturbation magnitude ϵ for both ODIN and Mahalanobis. To determine the hyperparameter values, we employ the procedure described in Lee et al. (2018).[6]

Evaluation metrics. We employ five metrics commonly used for the task (Hendrycks & Gimpel, 2017; DeVries & Taylor, 2018): FPR-at-95%-TPR, detection error that measures the minimum classification error over all possible thresholds, the area under the receiver operating characteristic curve (AUROC), the area under the precision-recall curve using in-distribution samples as the positives (AUPR-In), and AUPR using out-of-distribution samples as the positives (AUPR-Out).

Results. Comparing the performance of Baseline and CRL-

[5]https://tiny-imagenet.herokuapp.com/

[6]For the experiment, we used the code publicly available at https://github.com/pokaxpoka/deep_Mahalanobis_detector.

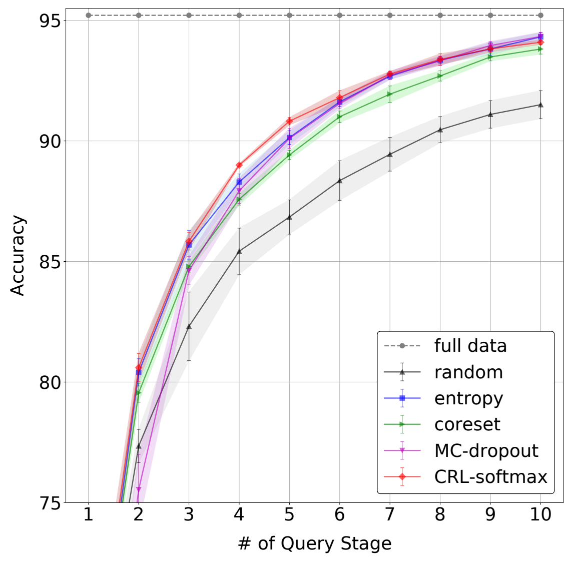
softmax models, CRL models perform better in most cases with a large margin as shown in Table 3.[7] This means that the CRL model provides good confidence estimates that distinguish OOD samples from in-distribution ones much more easily. We also observe that ODIN indeed becomes a more reliable detector when combined with CRL model. It outperforms ODIN with Baseline in all experiments with the exception of DenseNet with CIFAR-10 on LSUN, the OOD dataset. Interestingly, CRL model by itself performs even better than ODIN where SVHN dataset is the in-distribution dataset. For example, in case of FPR-95%-TPR for Tiny-ImageNet OOD dataset with DenseNet, the values from CRL-softmax (i.e., 7.99) is significantly lower than those from Baseline ODIN (i.e., 19.93), and we find similar results for the remaining metrics. The Mahalanobis detector is already a strong OOD detector on the datasets we consider, but it also slightly benefits from CRL models although the performance improvements are marginal compared to ODIN. Note that the conventional experimental setting for OOD detection is disadvantageous to CRL models since our models are trained to produce low confidence even for in-distribution samples if they are misclassified. Nevertheless, our experimental results show that deep classifiers trained with CRL perform well on the OOD detection task under that setting.

4.3. Active Learning

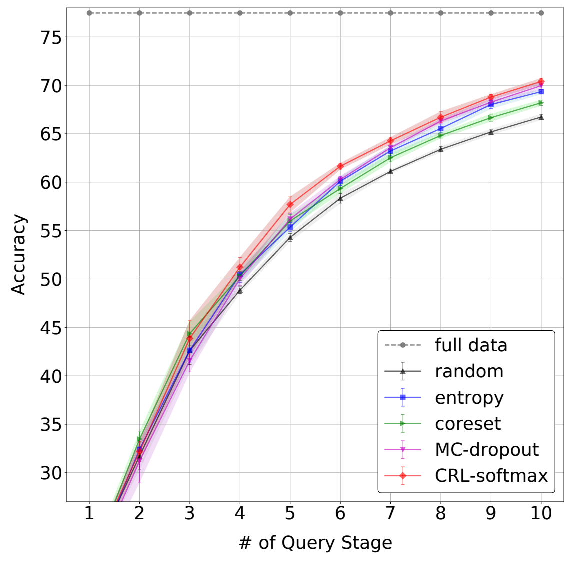
The key hypothesis of active learning lies that we can build a good predictive model with less labeled samples if a model knows which samples should be labeled to improve predictive performance. Thus, the goal of active learning is to achieve greater accuracy with fewer training labels (Settles, 2009).

Experimental settings. We evaluate the active learning performance of CRL model with ResNet18 architecture[8] by using CIFAR-10 and CIFAR-100 datasets. In this experiment, we train the model during 200 epochs, and decay the learning rate with a factor of 10 at 120 and 160 epochs. Other hyperparameters involved in training are same as in Section 4.1. For a comparison, we consider a CRL-softmax model associated with the least confidence-based sampling, MCdropout with entropy-based sampling, and Baseline with core-set sampling (Sener & Savarese, 2018) designed specifically for active learning to query representative samples. As other baselines commonly employed in active learning, we also use Baseline with random sampling and entropy-based sampling.

For this task, we follow a typical process to evaluate the performance of sampling strategy for active learning (Sener & Savarese, 2018; Yoo & Kweon, 2019). Given a unlabeled dataset $\mathcal{D}_U^0$ (i.e., the whole 50,000 images without labels), the labeled dataset $\mathcal{D}_L^1$ at the first stage consists of 2,000 samples that are randomly sampled without replacement from $\mathcal{D}_U^0$. With $\mathcal{D}_L^1$, we train an initial model. According to uncertainty estimates from the model, additional 2,000 samples are added to the labeled dataset for the next stage, and this $\mathcal{D}_L^2$ is used to update the current model. We proceed a total of 10 stages for a single trial. To rigorously compare the performances, we repeat this trial five times.



(a) CIFAR-10



(b) CIFAR-100

Figure 3. Active learning performance on (a) CIFAR-10 and (b) CIFAR-100 with various sampling methods. Curves are averages over five runs, and shaded areas denote ± one standard deviation.

[7]We omit the results where the OOD dataset is iSUN since it shows similar results with LSUN. Please refer to Table S6 in the supplementary material.

[8]https://github.com/kuangliu/pytorch-cifar

Results. Figure 3(a) shows the performance improvement over the stages on CIFAR-10. Obviously, Baseline with random sampling is inferior to other methods. CRL-softmax with the least confidence sampling shows better performance than the competing methods for most of stages. At the second stage, CRL-softmax has 80.6% of accuracy while entropy-based, core-set, and MCdropout have 80.40%, 79.55%, and 75.53% accuracy respectively. Our method also shows the highest performance compared to others at the 6-th stage. It reaches to 91.8% accuracy at this stage while entropy-based, core-set, MCdropout sampling methods show 0.2%, 0.8%, and 0.25% lower accuracy than our model.

The performance curves on CIFAR-100 can be found in Figure 3(b). Since CIFAR-100 is a more challenging dataset than CIFAR-10, it is comparatively hard to learn with small labeled dataset at early stages. Nevertheless, CRL model selects most of the informative samples that should be labeled, thereby showing better performance for all stages after the 4-th one. Finally, CRL model is the only one that achieves over 70% (i.e., 70.4%) accuracy. It shows the 0.43% accuracy gap with MCdropout, the second-best performing model.

Apart from CRL model, it is observed that Baseline with entropy-based sampling performs quite well on both datasets even if it is one of the most simple approaches, as similarly reported in Yoo & Kweon (2019). It should be mentioned that the core-set sampling is a query strategy to enhance active learning performance, and MCdropout method needs multiple stochastic forward paths to estimate uncertainty. Through the experimental results, we demonstrate that good confidence estimates naturally obtained from the CRL model are indeed effective for active learning.

5. Conclusion

In this paper, we introduce a simple but effective regularization method that can be employed for training deep neural networks to alleviate the well-known overconfident prediction issue. Our method is motivated by the observation regarding the correct prediction events during training with the SGD-based optimizer. Based on that, the proposed regularization method is implemented by the ranking loss CRL, which greatly improves confidence ranking performance of deep classifiers. We have demonstrated that deep neural networks trained with CRL produce well-ranked confidence estimates that are particularly important to the tasks related to what the model does not know such as OOD detection and active learning. Although we apply the proposed method to image classification tasks in the experiments, it can be extended to other classification tasks in natural language processing. It would be also interesting to investigate other properties of the proposed method such as its robustness to adversarial samples.

Acknowledgements

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(NRF-2018R1D1A1A02086017).

References

Brier, G. W. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1–3, 1950.

Corbière, C., THOME, N., Bar-Hen, A., Cord, M., and Pérez, P. Addressing failure prediction by learning model confidence. In *Advances in Neural Information Processing Systems*. 2019.

DeVries, T. and Taylor, G. W. Learning confidence for out-of-distribution detection in neural networks. *arXiv preprint arXiv:1802.04865*, 2018.

El-Yaniv, R. and Wiener, Y. On the foundations of noise-free selective classification. *The Journal of Machine Learning Research*, 11:1605–1641, 2010.

Gal, Y. and Ghahramani, Z. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, 2016.

Geifman, Y. and El-Yaniv, R. Selective classification for deep neural networks. In *Advances in Neural Information Processing Systems*. 2017.

Geifman, Y., Uziel, G., and El-Yaniv, R. Bias-reduced uncertainty estimation for deep neural classifiers. In *International Conference on Learning Representations*, 2019.

Graves, A. Practical variational inference for neural networks. In *Advances in Neural Information Processing Systems*. 2011.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. In *International Conference on Machine Learning*, 2017.

Gurau, C., Bewley, A., and Posner, I. Dropout distillation for efficiently estimating model confidence. *arXiv.org*, 2018.

He, K., Zhang, X., Ren, S., and Sun, J. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, 2016.

Hecker, S., Dai, D., and Van Gool, L. Failure prediction for autonomous driving. In *IEEE Intelligent Vehicles Symposium*, 2018.

Hendrycks, D. and Gimpel, K. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference on Learning Representations*, 2017.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., and Kingsbury, B. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

Huang, G., Li, Y., Pleiss, G., Liu, Z., Hopcroft, J. E., and Weinberger, K. Q. Snapshot ensembles: Train 1, get M for free. In *International Conference on Learning Representations*, 2017a.

Huang, G., Liu, Z., van der Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017b.

Jiang, H., Kim, B., Guan, M., and Gupta, M. To trust or not to trust a classifier. In *Advances in Neural Information Processing Systems*. 2018.

Kendall, A. and Gal, Y. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems*. 2017.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*. 2012.

Kumar, A., Liang, P. S., and Ma, T. Verified uncertainty calibration. In *Advances in Neural Information Processing Systems*. 2019.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pp. 6402–6413. 2017.

Lee, K., Lee, K., Lee, H., and Shin, J. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Advances in Neural Information Processing Systems*. 2018.

Liang, S., Li, Y., and Srikant, R. Enhancing the reliability of out-of-distribution image detection in neural networks. In *International Conference on Learning Representations*, 2018.

MacKay, D. J. C. A practical Bayesian framework for backpropagation networks. *Neural Computation*, 4(3): 448–472, 1992.

Naeini, M. P., Cooper, G. F., and Hauskrecht, M. Obtaining well calibrated probabilities using Bayesian binning. In *AAAI Conference on Artificial Intelligence*, 2015.

Nam, J. G., Park, S., Hwang, E. J., Lee, J. H., Jin, K.-N., Lim, K. Y., Vu, T. H., Sohn, J. H., Hwang, S., Goo, J. M., and Park, C. M. Development and validation of deep learning–based automatic detection algorithm for malignant pulmonary nodules on chest radiographs. *Radiology*, 290(1):218–228, 2019.

Neal, R. M. *Bayesian Learning for Neural Networks*. Springer-Verlag, 1996. ISBN 0387947248.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Neumann, L., Zisserman, A., and Vedaldi, A. Relaxed softmax: Efficient confidence auto-calibration for safe pedestrian detection. In *Machine Learning for Intelligent Transportation Systems Workshop, NIPS*, 2018.

Nguyen, A., Yosinski, J., and Clune, J. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *The IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

Roady, R., Hayes, T. L., Kemker, R., Gonzales, A., and Kanan, C. Are out-of-distribution detection methods effective on large-scale datasets? *CoRR*, abs/1910.14034, 2019.

Sener, O. and Savarese, S. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018.

Settles, B. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

Toneva, M., Sordoni, A., des Combes, R. T., Trischler, A., Bengio, Y., and Gordon, G. J. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*, 2019.

Xu, P., Ehinger, K. A., Zhang, Y., Finkelstein, A., Kulkarni, S. R., and Xiao, J. Turkergaze: Crowdsourcing saliency with webcam based eye tracking. *CoRR*, abs/1504.06755, 2015.

Yoo, D. and Kweon, I. S. Learning loss for active learning. In *The IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

Yu, F., Zhang, Y., Song, S., Seff, A., and Xiao, J. LSUN: construction of a large-scale image dataset using deep learning with humans in the loop. *CoRR*, abs/1506.03365, 2015.

Zhang, Z., Dalca, A. V., and Sabuncu, M. R. Confidence calibration for convolutional neural networks using structured dropout. *CoRR*, abs/1906.09551, 2019.