MISCLASSIFICATION DETECTION VIA CLASS AUGMENTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Despite the impressive performance in various pattern recognition tasks, deep neural networks (DNNs) are typically *overconfident* in their predictions, making it difficult to determine whether a test example is misclassified. In this paper, we propose a simple yet effective method of *class augmentation (classAug)* to address the challenge of misclassification detection in DNNs. Specifically, we increase the number of classes during training by assigning new classes to the samples generated using between-class interpolation. In spite of the simplicity, extensive experiments demonstrate that the misclassification detection performance of DNNs can be significantly improved by seeing more generated pseudo-classes during training. Additionally, we observe that DNNs trained with classAug are more robust on out-of-distribution examples. Finally, as a general regularization strategy, classAug can also enhance the original classification accuracy and few-shot learning performance.

1 INTRODUCTION

Deep neural networks (DNNs) have recently demonstrated state-of-the-art performance in various applications ranging from computer vision (He et al., 2016; Deng et al., 2019) to natural language processing (Kenton & Toutanova, 2019; Yao et al., 2019). However, there are other aspects of DNNs which are undesirable and less well understood. Recent works (Guo et al., 2017; Hendrycks & Gimpel, 2017) have shown that DNNs are typically *overconfident* in their predictions (shown in Figure 1), which brings great concerns when DNNs being deployed in various real-world applications (Janai et al., 2017; Miotto et al., 2016; Amodei et al., 2016). In safety-critical systems, one crucial requirement is to assign low confidence when a prediction is likely to be incorrect. Equipped with such ability, a system could quantify the predictive confidence to make much safer decision. For instance, an autonomous driving car should rely more on other sensors for braking or trigger an alarm when the detection network is unable to confidently predict obstructions (Janai et al., 2017). Alternatively, the control should be handed over to human doctors when the confidence of a disease diagnosis network is low (Miotto et al., 2016). In conclusion, being able to provide *reliable confidence* is practically important for real-world machine learning systems.

Reliable confidence is crucial for two related problems: *out-of-distribution (OOD) detection* and *misclassification detection (MD)* (Hendrycks & Gimpel, 2017). The purpose of OOD detection is to determine whether an input is outside of the distribution (unseen class) of the training data. Differently, MD focuses on whether the model will make an erroneous prediction on a particular test example (seen class). Recently, the machine learning community has paid significant attention to the OOD detection (Liang et al., 2018; Lee et al., 2018; DeVries & Taylor, 2018; Malinin & Gales, 2018; Yu & Aizawa, 2019; Hendrycks et al., 2019b;a). However, MD remains far less explored. To the best of our knowledge, there is no method that can significantly outperform the baseline (using softmax probability as confidence) (Hendrycks & Gimpel, 2017) on MD task.

Related Work. Earlier works (Fumera & Roli, 2002; Grandvalet et al., 2009; Cortes et al., 2016) studied the problem of MD with a reject option to improve the reliability of machine learning systems in real-world tasks. Typically such methods jointly learn a classifier to perform classification and a rejection function to decide whether the classifiers decision should be accepted. However, those methods are mainly based on support vector machines. For DNNs, Hendrycks & Gimpel (2017) firstly established a standard baseline for detecting misclassified samples by using maximum

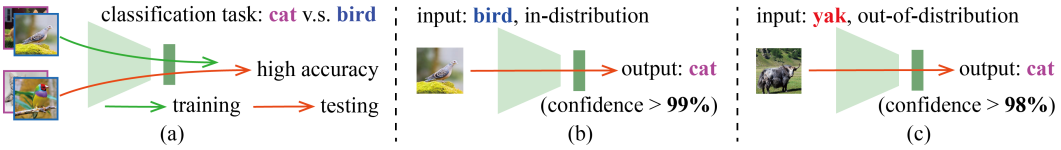


Figure 1: An illustration of the overconfident problem in DNNs. (a) A high-accuracy model typically (b) assigns high confidence for misclassified samples from seen classes, and (c) classifies out-of-distribution samples (from unseen classes) as seen classes confidently. We mainly focus on misclassification detection problem, that is, detecting the misclassified samples from seen classes.

softmax probability. However, DNNs tend to overconfident in their false predictions, which contributes to the main drawback of the baseline approach. "Trust Score" (Jiang et al., 2018), the ratio between the distance from the sample to the nearest class different from the predicted class and the distance to the predicted class, was adopted as a confidence measure. However, computing nearest neighbors in large dataset and high dimensional spaces is expensive and unstable, therefore degrades the practicality and extensibility of this method. Recently, Corbière et al. (2019) addressed MD problem by learning the true class probability (TCP) of misclassified examples with another DNN called ConfidNet. Nevertheless, it turns out that when the model is well trained and has a high training accuracy, few (or even none) misclassified examples will exist in training set, and the TCP of a training example is just the same as the highest softmax probability. Outlier Exposure (Hendrycks et al., 2019a) leveraged diverse, realistic datasets to improve OOD detection performance. However, this method has no or even negative effects on MD task. More recently, some works (Haldimann et al., 2019; Xia et al., 2020) leveraged generative models such as generative adversarial network (Goodfellow et al., 2014) to detect wrongly segmented instances by comparing the difference between the generated image and the original input. However, it is inefficient to train generative models and the quality of generated synthetic images is highly affected by generative models.

Method. In this paper, we propose a conjecture that *the overconfident phenomenon of DNNs is partially due to the fixed and limited number of classes used for training*. Actually, for human beings, *the more we learn, the more we realize how much we do not know*. Therefore, we propose a simple approach of *class augmentation* to address the overconfident problem by letting the DNNs to learn more classes during training. Intuitively, class augmentation enables the model to leverage more robust and generalizable features by seeing more classes during training. New pseudo-classes are generated by between-class interpolation, e.g., random interpolations of two samples drawn from different classes. In the training process, an extend $(K+M)$-class problem is optimized by applying softmax over all classes and using cross-entropy loss. In the inference process, however, we only care about the original K classes, and softmax is applied on the first K outputs. As usual, the class with maximum softmax probability is the predicted class, and the maximum probability is viewed as the confidence.

The proposed method is related to the *mixup* (Zhang et al., 2018) which also apply random interpolation on a pair of training samples and the respective one-hot labels. However, mixup is a *data augmentation* approach, and therefore, the interpolated samples are near original data. Differently, our approach is aimed at *class augmentation*, therefore, the interpolated samples are among the confusable region. In mixup, the number of classes is not changed, but in our method it is greatly increased. As shown later, our approach can outperform mixup significantly, especially for misclassification detection, due to the expanded classes. Moreover, our approach is also effective for other tasks which are not achievable by mixup.

Contribution. In our approach, the augmented classes are actually laying in the confusing region of two classes. By treating them as newly-added classes for the original problem, lower confidence will be assigned for the samples which are likely to be erroneously classified, thus giving us possibility to distinguish correct and misclassified samples. In spite of the simplicity, extensive experiments demonstrate the effectiveness of *class augmentation* on improving the reliability of the confidence outputted by DNNs, and we have achieved new benchmark results on misclassification detection task. Moreover, we find that DNNs trained with class augmentation are more robust on out-of-distribution examples. Finally, *class augmentation* can also be viewed as a general regularization strategy for improving the generalization performance of the original classification task and the few-shot learning (Snell et al., 2017; Bertinetto et al., 2018) task.

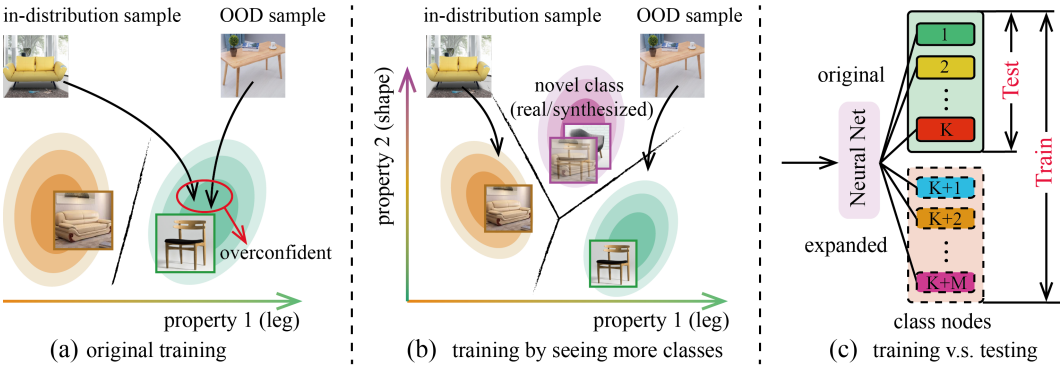


Figure 2: Motivation. (a) For instance, real data is described by latent characteristics like w/o legs and shape. Only the most discriminative characteristic (e.g., w/o legs) are learned by original training scheme. (b) Class augmentation: more classes (realistic or synthetic samples) are introduced during training, which enables the model to leverage more robust and generalizable features (e.g. shape) to overcome overconfident problem. (c) A $(K + M)$-class problem is optimized in training and only the original K classes are evaluated in testing.

2 Motivation and Methodology

2.1 Motivation

To build machine learning models, it is normal to predefine and fix the number of classes, mainly for two reasons. First, this will make the data collection process easier and straightforward, which are reasonable in laboratory environment. Second, this is a requirement for most classification models, for example, the last layer of DNNs is usually viewed as class nodes, and the number of them need to be fixed for making the training process to work. However, the closed-world setting, i.e., there are a fixed number of classes for both training and test, has potential influence on the overconfident phenomenon of DNNs (Zhang et al., 2020). Since modern DNNs have very strong fitting capacity, it is common to achieve near perfect accuracy on training set, resulting in a sharp posterior distribution over classes which leads to the overconfident problem.

Our motivation is illustrated in in Figure 2. For example, a pattern recognition task is to classify sofa and chair, which could be easily solved by determining whether the input has legs. However, when deploying the system in open-world, inputs from OOD (e.g., desk) or in-distribution may be misclassified with very high confidence. By introducing novel classes during training, the model may learn other features that are less discriminative for current task but useful for alleviating overconfident problem.

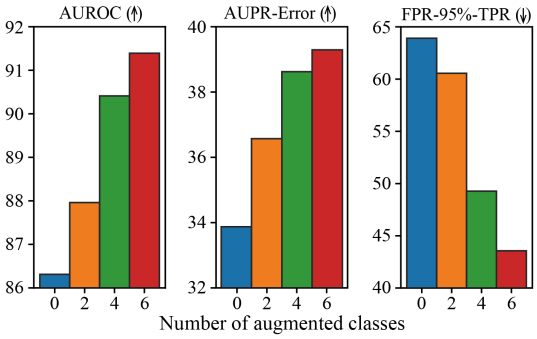


Figure 3: Misclassification detection can be improved by seeing more real classes during training. The base class number is $K = 4$, and the augmented class number is $M = 0, 2, 4, 6$.

To verify our motivation, we use real data as novel classes and carry out an experiment on the CIFAR-10 dataset (Krizhevsky et al., 2009). There are totally 10 classes in CIFAR-10, and we view the first 4 of them as the base classes ($K = 4$). The remaining classes hence give us possibility to view the benefit of adding classes for training. We set the augmented class number as $M = 0, 2, 4, 6$ respectively. As shown in Figure 2 (c), a $(K + M)$-class model is optimized in training and only the original K-class are evaluated in test. The used model is ResNet-18 (He et al., 2016). The used metrics are AUROC, AUPR-Error and FPR at 95%-TPR (Hendrycks & Gimpel, 2017) for misclassification detection. As show in Figure 3, we can find that: seeing more additional real classes during training substantially improves the performance of misclassification detection. However, it is unrealistic to always have access to more real classes in practice. In addition, it is also

an open question to decide which dataset to use. Therefore, a natural question is: can we use some algorithms for *synthesizing classes*, and will it also be effective like adding real classes?

2.2 METHODOLOGY

Here we consider the case that using the samples coming from K classes to synthesize more classes. Actually, this is a difficult task due to the missing of description and also the large freedom of new classes. This means the synthesized classes may not have any actual meanings. This is different from the *data augmentation* approaches for which the generated samples still have clear meanings of belonging to a known class. Moreover, the purpose of synthesizing classes is to improve the confidence estimation when adding them to the training process. Therefore, it is better for them to be representative for some low-confidence samples, because the original samples usually have high confidence after standard training.

In light of this, we propose a class synthesis method named *between-class interpolation* (BCI). As shown in Figure 4, the central idea is to *randomly* interpolate two samples from two different classes for generating a new sample representing a new class. In this way, the synthesized class will lay in the confusing area (low-confidence region) of two classes, which is beneficial for improving confidence estimation. Considering that we have two real classes A and B, we use x_a to denote a sample randomly drawn from class A and x_b to represent a sample randomly drawn from class B. Since we focus on vision tasks, here x_a and x_b are images. To efficiently and effectively realize BCI, we propose two different methods.

BCI-1. The first method to synthesize novel class is BCI-1, which is based on linear interpolation of two images:

$$x_{ab}^{\text{new}} = \lambda x_a + (1-\lambda)x_b, \tag{1}$$

where λ is a *random* number drawn from $[0, 1]$. Suppose class A has n_1 samples and class B has n_2 samples, there are actually $n_1 \times n_2$ kinds of sampling for x_a and x_b. Moreover, the interpolation coefficient λ is also a random number. This actually gives us possibility of generating infinite new samples. However, to reduce the overlap of the new class and the old classes (A and B), we restrict λ to be sampled from the interval of $[0.4, 0.6]$. In this way, the new class is actually laying in the confusing area between two base classes.

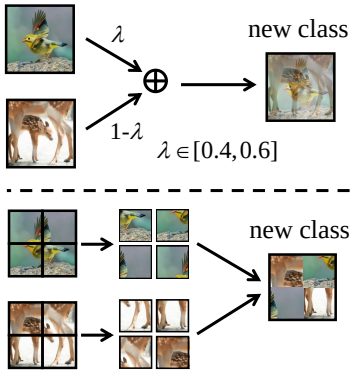


Figure 4: Illustrations of BCI-1 (top) and BCI-2 (bottom).

BCI-2. Besides linear interpolation, we also consider a non-linear approach of BCI-2 to generate synthesize samples for novel class. Firstly, both the images of x_a and x_b are divided into a 2×2 grids with equal sizes, and then 4 grids (2 grids from each example) are selected and reorganized *randomly* to produce a new image. We define the combining operation as:

$$x_{ab}^{\text{new}} = \mathbf{M} \odot x_a + (\mathbf{1} - \mathbf{M}) \odot x_b, \tag{2}$$

where $\mathbf{M}$ denotes a binary mask, and $\odot$ is element-wise multiplication. BCI-2 is builds upon cutmix (Yun et al., 2019), who cuts and pastes patches among training images. Different from cutmix, we focus on the confusing area (low-confidence region) of two classes, and assign new class label to the generated samples.

The illustrations of BCI-1 and BCI-2 are shown in Figure 4. Via between-class interpolation, the data in this new class have large probability of being wrongly classified to class A and B. However, we view them as a new class during training, for making the model learn how to assign confusable samples to another class, thus reducing its confidence on original classes (A and B). For a K-class problem, we can generate $K(K-1)/2$ new classes using BCI. As shown in Figure 2(c), we merge the generated new classes to M auxiliary classes, and the original K-class problem is therefore extended to a $(K+M)$-class problem. Softmax is applied on the $K+M$ nodes and cross-entropy loss is used for training. In the inference process, we only apply softmax on the original K classes.

Table 1: Results on misclassification detection. All values are percentages. ↑ indicates larger value is better, and ↓ indicates lower value is better.

Dataset	Method	AUROC ↑	AUPR-S ↑	AUPR-E ↑	FPR95 ↓	Prob-W ↓	Accuracy ↑
Fashion-MNIST model C	Baseline	86.91	98.88	39.02	52.15	92.43	92.79
	MCDropout	89.77	99.09	41.88	49.87	87.55	92.54
	ConfidNet	86.97	98.91	41.26	50.47	88.18	92.79
	Mixup	90.17	99.15	40.90	48.70	68.55	93.45
	ClassAug (ours)	**92.45**	**99.37**	**42.81**	**46.77**	**66.93**	**93.72**
STL-10 VGG-16	Baseline	79.51	90.09	58.48	77.71	80.02	70.00
	MCDropout	81.09	91.54	59.61	76.22	84.21	71.04
	ConfidNet	79.86	90.68	59.54	75.38	75.68	70.00
	Mixup	76.27	89.10	53.57	74.90	65.83	73.75
	ClassAug (ours)	**83.88**	**93.16**	**60.15**	**71.22**	**50.36**	**74.39**
STL-10 ResNet-18	Baseline	80.89	91.58	58.47	80.08	82.19	70.50
	MCDropout	81.07	90.96	61.59	77.06	84.90	70.61
	ConfidNet	81.20	91.81	60.24	76.47	78.95	70.50
	Mixup	83.04	93.18	59.13	70.38	57.58	73.88
	ClassAug (ours)	**84.67**	**93.98**	**61.92**	**69.99**	**52.11**	**74.97**
CIFAR-10 ResNet-18	Baseline	91.35	99.27	42.95	47.75	88.95	92.88
	MCDropout	92.22	99.26	47.20	45.90	87.60	91.96
	ConfidNet	91.96	99.53	45.03	45.68	83.46	92.88
	Mixup	91.00	98.91	47.70	42.27	69.74	93.91
	ClassAug (ours)	**93.80**	**99.60**	**49.75**	**36.86**	**63.57**	**94.46**
CIFAR-10 ResNet-34	Baseline	92.68	99.37	46.77	42.46	87.56	92.77
	MCDropout	91.90	99.25	45.34	46.46	87.41	92.23
	ConfidNet	92.96	99.60	46.81	40.22	84.38	92.77
	Mixup	88.01	98.65	45.72	40.25	**63.87**	93.59
	ClassAug (ours)	**94.51**	**99.60**	**47.71**	**34.46**	74.80	**95.13**
CIFAR-100 ResNet-34	Baseline	85.48	93.79	68.69	66.06	80.51	69.98
	MCDropout	85.54	93.63	68.65	70.35	75.59	68.62
	ConfidNet	86.03	93.96	69.02	64.53	73.78	69.98
	Mixup	85.33	92.65	68.52	64.42	67.12	72.45
	ClassAug (ours)	**87.16**	**95.02**	**69.43**	**63.90**	**65.38**	**72.80**

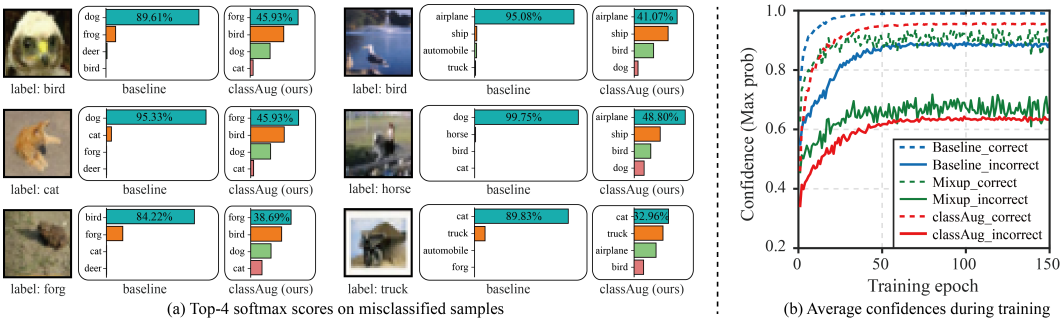
3 EXPERIMENTS

In this section, we firstly compare our method, class augmentation (classAug), with several recent baselines on misclassification detection (MD) of DNNs. We conduct experiments to demonstrate the effectiveness of classAug, and results show that classAug outperforms other approaches on MD by a large margin on a variety of datasets and network architectures. Specifically, we show that the more class, the better performance of MD. Then we demonstrate that using real data from other datasets does not work as well as using synthetic samples generated by BCI. Additionally, we compare our method with OE (Hendrycks et al., 2019a) for MD task. Results show that classAug, which is a new way to leverage synthetic samples, is more effective than OE for MD task. Furthermore, in Section 3.3, we show that DNNs trained with classAug are more robust on out-of-distribution examples, and the accuracy of few-shot learning could be improved by classAug. Note that we mainly used BCI-1 for experiments, and compared BCI-1 and BCI-2 for MD in Section 3.2. Implementation details are described in Appendix. Our code will be publicly available upon the publish of this paper.

3.1 EXPERIMENTAL SETUP

Datasets and network architectures. We thoroughly validate our method on several well-known classification datasets: Fashion-MNIST (Xiao et al., 2017), STL-10 Coates et al. (2011), CIFAR-10 and CIFAR-100 (Krizhevsky et al., 2009). We use a wide range of network architectures to demonstrate the effectiveness and robustness of our approach. A small convolution network named model C in (Springenberg et al., 2014), VGG-16 (Simonyan & Zisserman, 2015), ResNet-18 and ResNet-34 (He et al., 2016) are mainly adopted in our experiments.

Evaluation metrics. We adopt the standard metrics as Hendrycks & Gimpel (2017) to measure the quality of misclassification detection: **FPR at 95%-TPR**, **AUPR-S**, **AUPR-E** and **AUROC**. Specifically, **FPR at 95%-TPR** can be interpreted as the probability that a negative (misclassified)



(a) Top-4 softmax scores on misclassified samples

(b) Average confidences during training

Figure 5: (a) Predictive distributions on misclassified samples. For misclassified samples, the predictive distribution is smoother, and the softmax scores of the ground-truth class are increased by training with classAug. (b) Average confidences (the max softmax scores) of correctly classified samples and misclassified samples during training. We use ResNet-18 on CIFAR-10 dataset.

example is misclassified as a correct prediction when the true positive rate (TPR) is as high as 95%. The metric **AUPR-S** and **AUPR-E** indicate the area under the precision-recall curve where correct predictions and errors are specified as positives, respectively. **AUROC** is the area under the receiver operating characteristic curve. As described in Hendrycks & Gimpel (2017), we also list the mean predicted class probability of wrongly classified examples (**Prob-W**). In addition, we emphasize that the **classification accuracy** of the neural network are also important since we can not sacrifice the original task performance to improve the quality of confidence estimation. Further details about these metrics are in Appendix.

Comparison Methods. We compare our method with several approaches: baseline (Hendrycks & Gimpel, 2017), Minte-Carlo Dropout (MCDropout) (Gal & Ghahramani, 2016), ConfidNet (Corbière et al., 2019), and Mixup Training (Thulasidasan et al., 2019).

3.2 Results and Discussion

Comparative results on MD are summarized in Table 1. We observe that our method consistently outperforms the baseline approach by a large margin, which confirms that classAug can effectively improve the reliability of confidence estimation. Intuitively, by explicitly modeling the samples laying in the confusing region of two classes, the network encourages samples which are prone to be misclassified to have lower confidence over original class nodes. As shown in Figure 5(a), classAug not only reduces the confidence over false prediction but also enhances the prediction values of the ground-truth classes. This implies that classAug induces meaningful predictions by forcing the model to leverage more generalizable features to recognize confusion introduced by BCI. Figure 5(b) shows that the average confidence of misclassified samples can be reduced from 88.95% to 63.57% by our approach.

Furthermore, it is also shown that mixup can improve baseline results on some datasets and network architectures such as Fashion-MNIST and STL-10 with ResNet-18, while classAug consistently performs well on larger category tasks and more complex datasets. Actually, mixup training can successfully reduce the confidence of misclassified samples. However, as shown in Figure 5(b), it also ruins the confidence of correct classified samples, which can not improve the separability between erroneous and correct predictions, resulting in no effectiveness or even negative influence on misclassification detection performance. As a conclusion, our approach outperforms other methods for MD significantly and consistently, and also achieves better generalization performance.

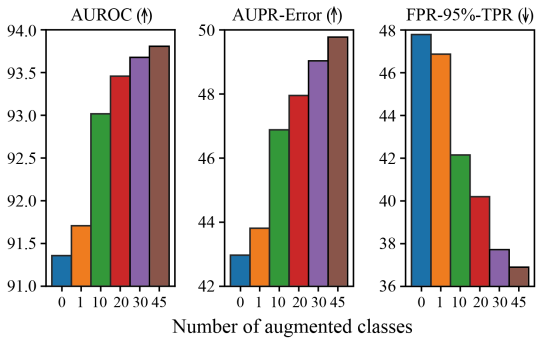


Figure 6: MD can be improved by seeing more synthetic classes during training.

Novel class: the more the better? We empirically study how the number of new classes affect MD performance. We fix the total number of generated samples, then increase the number of novel classes. Specifically, we merge the novel samples into different number of classes, starting by merging all the new data to a single auxiliary class (similar in some ways to a reject option). From Figure 6, we observe that increasing the number of new classes can improve the MD performance. Specifically, the performance are improved significantly when the number of new classes rises from 1 to 10. Then, the growth moves slower against class number increasing. Intuitively, when merging all new data to a single class, on the one hand, the variance of the features in this single class would be big, which might be harmful for representation learning; on the other hand, the model would learn less discriminative features since it has no need to perform classificaiton among those new data.

Real Class v.s. BCI. Actually, the class augmentation is a general idea. In this paper, we use BCI to generate synthetic samples and classes. However, it is also possible to use real classes from other datasets for class augmentation. Table 2 shows such comparisons. For the task of CIFAR-10, we use SVHN, CIFAR-100, TinyImageNet, BCI-1 and BCI-2 for class augmentation. It is shown that BCI-1 significantly outperforms real classes from other datasets on MD task. This is because the real classes from other datasets have a domain gap to the base classes, while generated classes by BCI are still in the same domain of original class (note that the purpose of MD task is to detect these misclassified in-distribution samples). Lastly, we find that BCI-1 significantly outperforms BCI-2, indicating linear interpolation of two images is an effective strategy in producing new classes.

Table 2: Comparative results on MD (Real Class v.s. BCI; OE v.s. classAug). All values are percentages. ↑ indicates larger value is better, and ↓ indicates lower value is better.

New classes	Real	AUROC ↑		AUPR-S ↑		AUPR-E ↑		FPR95 ↓	
		OE	classAug	OE	classAug	OE	classAug	OE	classAug
SVHN	✓	91.39	**92.54**	**99.31**	99.30	44.05	**46.93**	51.36	**46.63**
CIFAR-100	✓	90.84	**92.50**	99.15	**99.26**	42.49	**47.26**	50.33	**46.62**
TinyImageNet	✓	90.99	**92.88**	99.05	**99.37**	45.59	**48.20**	53.36	**44.35**
BCI-2	✗	88.06	**92.95**	98.88	**99.33**	34.59	**47.76**	58.87	**44.59**
BCI-1	✗	89.01	**93.80**	98.88	**99.60**	34.43	**49.75**	59.11	**36.86**

OE v.s. classAug. Outlier Exposure (OE) (Hendrycks et al., 2019a) is a recently proposed method to improve OOD detection performance by leveraging diverse, realistic datasets. Specifically, OE minimize the Kullback-Leibler (KL) divergence between the predicted probability of OOD samples and the uniform distribution. While our method, classAug, is a new and different way to use auxiliary data. Here, we compare the effect of OE and classAug for MD task in Table 2. we can observe that: (1) OE is less effective than classAug (on both real and synthetic data) for MD. (2) For OE, real data is more effective than synthetic data. The reason why OE is useless for MD is that the unreasonableness of forcing the model to output uniform distribution over all classes for samples near the local class boundaries, especially for data generated by BCI.

Table 3: Comparative results on OOD detection. All values are percentages.

In-dist	OOD	AUROC ↑			AUPR-In ↑			AUPR-Out ↑		
		Baseline	Mixup	classAug	Baseline	Mixup	classAug	Baseline	Mixup	classAug
CIFAR-10 ResNet-18	MNIST	87.02	92.46	**94.99**	79.89	89.00	**93.05**	92.26	95.48	**97.20**
	Fashion-MNIST	90.28	93.37	**94.40**	86.18	89.11	**92.43**	94.26	96.19	**96.78**
	LSUN	88.50	88.80	**93.90**	83.48	74.71	**91.08**	92.92	94.09	**96.73**
	TinyImageNet	88.49	84.96	**93.92**	83.84	64.02	**91.77**	92.70	92.19	**96.55**
	Mean	88.57	89.90	**94.30**	83.35	79.21	**92.08**	93.04	94.49	**96.81**
CIFAR-10 ResNet-34	MNIST	72.46	69.73	**83.78**	77.67	69.97	**86.96**	67.22	64.36	**80.05**
	Fashion-MNIST	81.53	82.29	**86.59**	81.53	83.48	**89.09**	77.45	78.90	**83.89**
	LSUN	74.07	54.73	**74.92**	74.07	50.84	**79.13**	69.07	54.18	**69.77**
	TinyImageNet	71.92	54.25	**72.05**	72.03	49.53	**75.98**	67.04	54.53	**67.27**
	Mean	75.00	65.25	**79.33**	75.03	63.45	**82.79**	70.19	62.99	**75.24**

Computational cost. ClassAug adds extra training time complexity compared with vanilla training. However, we find that increasing the number of classes is more effective than increasing the amount of data. Therefore, we keep the same amount of total new samples as that of vanilla training to speedup training. In fact, training time with classAug is less than twice of baseline since classAug converges faster. Similar to mixup, we apply BCI between one minibatch and its random shuffled counterpart to efficiently realize random sampling. We found this strategy works equally well but reduces I/O requirements significantly compared with producing all classes and samples beforehand.

3.3 OTHER BENEFITS FROM CLASSAUG

Although classAug is mainly motivated and proposed for MD task, we find that classAug also improves OOD detection performance. Moreover, as a regularization strategy, it can enhance the accuracy of few-shot learning task.

OOD detection. We use AUROC, AUPR-In and AUPR-Out in Hendrycks & Gimpel (2017) as metrics to evaluate the quality of OOD detection. As shown in Table 3, classAug noticeably improves the OOD detection performance of baseline method. By recognize synthetic samples in the confusing area, DNNs could learn more robust and transferable features which could be generalized to OOD (illustrated in Figure 2), thus improving the OOD detection performance. Moreover, as shown in Table 3, mixup sometimes damage the performance of OOD detection, which further demonstrates that *class augmentation* is better than *data augmentation* for confidence estimation.

Few-shot learning. Building on the ProtoNet (Snell et al., 2017) and the R2D2 (Bertinetto et al., 2018) methods, we demonstrate that classAug classAug can be utilized in few-shot learning tasks, and achieve state-of-the-art performance on FC100 (Oreshkin et al., 2018) and CIFAR-FS (Bertinetto et al., 2018) benchmarks. Intuitively, a larger amount of novel classes are introduced by classAug, thus more diverse tasks could be sampled during training, which would be beneficial to learn more discriminative feature and generalize to unseen classes. Moreover, we can see that the boost in few-shot accuracy from mixup training is not conclusive, which is also demonstrated by Mangla et al. (2020). See Appendix for details about datasets, architectures, and implementation.

Table 4: Comparative results on few-shot learning. All values are percentages.

Dataset	Method	ProtoNet			R2D2		
		5-shot	3-shot	1-shot	5-shot	3-shot	1-shot
FC100	Original model	54.61	50.62	40.20	55.78	51.49	41.70
	+ Mixup	55.32	51.08	41.30	56.92	53.04	42.93
	+ ClassAug (ours)	**57.25**	**53.30**	**43.02**	**56.96**	**53.64**	**43.68**
CIFAR_FS	Original model	85.59	83.62	73.62	85.99	83.55	75.02
	+ Mixup	85.55	83.60	73.36	85.96	83.88	74.13
	+ ClassAug (ours)	**87.25**	**85.59**	**76.41**	**87.34**	**84.79**	**76.10**

4 CONCLUSION

DNNs are powerful and accurate models, however, they are often overconfident about themselves. Recently, the overconfident problem has drawn significant attention form machine learning community. However, they mainly focus on OOD detection and confidence calibration. Misclassification detection, which is also important for many safety-critical applications, is far less explored. This paper proposes a simple method of *class augmentation* to address miclassification detection problem by letting DNNs to learn more classes synthesized by between-class interpolation. Comprehensive experimental results show that class augmentation can improve the performance for misclassification detection. As a regularization strategy, it is also effective to learn better representations for improving generalization (accuracy and few-shot) performance.

Class augmentation is a simple but not studied topic in machine learning community. Compared with data augmentation, there are still many unknowns and potentials worth further exploration in class augmentation. Important future works include theoretic analysis of class augmentation and other class synthesis method besides between-class interpolation.

REFERENCES

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

Luca Bertinetto, Joao F Henriques, Philip Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. In *International Conference on Learning Representations*, 2018.

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 215–223, 2011.

Charles Corbière, Nicolas Thome, Avner Bar-Hen, Matthieu Cord, and Patrick Pérez. Addressing failure prediction by learning model confidence. In *Advances in Neural Information Processing Systems*, pp. 2898–2909, 2019.

Corinna Cortes, Giulia DeSalvo, and Mehryar Mohri. Learning with rejection. In *International Conference on Algorithmic Learning Theory*, pp. 67–82. Springer, 2016.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4690–4699, 2019.

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

Giorgio Fumera and Fabio Roli. Support vector machines with embedded reject option. In *International Workshop on Support Vector Machines*, pp. 68–82. Springer, 2002.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International conference on machine learning*, pp. 1050–1059, 2016.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Yves Grandvalet, Alain Rakotomamonjy, Joseph Keshet, and Stéphane Canu. Support vector machines with a reject option. In *Advances in neural information processing systems*, pp. 537–544, 2009.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, pp. 1321–1330, 2017.

David Haldimann, Hermann Blum, Roland Siegwart, and Cesar Cadena. This is not what i imagined: Error detection for semantic segmentation through visual dissimilarity. *ArXiv*, abs/1909.00676, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

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

Dan Hendrycks, Mantas Mazeika, and Thomas G Dietterich. Deep anomaly detection with outlier exposure. In *International Conference on Learning Representations*, 2019a.

Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. Using self-supervised learning can improve model robustness and uncertainty. In *Advances in Neural Information Processing Systems*, pp. 15663–15674, 2019b.

Joel Janai, Fatma Güney, Aseem Behl, and Andreas Geiger. Computer vision for autonomous vehicles: Problems, datasets and state-of-the-art. *arXiv preprint arXiv:1704.05519*, 2017.

Heinrich Jiang, Been Kim, Melody Guan, and Maya Gupta. To trust or not to trust a classifier. In *Advances in neural information processing systems*, pp. 5541–5552, 2018.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pp. 4171–4186, 2019.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

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

Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Advances in Neural Information Processing Systems*, pp. 7167–7177, 2018.

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

Andrey Malinin and Mark Gales. Predictive uncertainty estimation via prior networks. In *Advances in Neural Information Processing Systems*, pp. 7047–7058, 2018.

Puneet Mangla, Nupur Kumari, Abhishek Sinha, Mayank Singh, Balaji Krishnamurthy, and Vineeth N Balasubramanian. Charting the right manifold: Manifold mixup for few-shot learning. In *The IEEE Winter Conference on Applications of Computer Vision*, pp. 2218–2227, 2020.

Riccardo Miotto, Li Li, Brian A Kidd, and Joel T Dudley. Deep patient: an unsupervised representation to predict the future of patients from the electronic health records. *Scientific reports*, 6:26094, 2016.

Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A simple neural attentive meta-learner. In *International Conference on Learning Representations*, 2018.

Boris Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In *Advances in Neural Information Processing Systems*, pp. 721–731, 2018.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2015.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pp. 4077–4087, 2017.

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

Sunil Thulasidasan, Gopinath Chennupati, Jeff Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In *Advances in Neural Information Processing Systems*, pp. 13888–13899, 2019.

Yingda Xia, Yu Zhang, Fengze Liu, Wei Shen, and Alan L. Yuille. Synthesize then compare: Detecting failures and anomalies for semantic segmentation. *ArXiv*, abs/2003.08440, 2020.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Liang Yao, Chengsheng Mao, and Yuan Luo. Graph convolutional networks for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 7370–7377, 2019.

Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *ArXiv*, abs/1506.03365, 2015.

Qing Yu and Kiyoharu Aizawa. Unsupervised out-of-distribution detection by maximum classifier discrepancy. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 9518–9526, 2019.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 6023–6032, 2019.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

Xu-Yao Zhang, Cheng-Lin Liu, and Ching Y Suen. Towards robust pattern recognition: A review. *Proceedings of the IEEE*, 108(6):894–922, 2020.

A Appendix

A.1 Implementation Details

The number of new classes and the number of samples per novel class are two important hyper-parameters for the implementation of classAug. In our experiments, we introduces 45 novel classes for each dataset including Fashion-MNIST, STL-10 as well as CIFAR-10. For CIFAR-100, we generate 1000 novel classes in addition to original 100 classes. We train all the models for 250 epochs with batch size 64 and Adam (Kingma & Ba, 2015) optimizer with 0.001 initial learning rate. We drops the learning rate by 0.85 when validation error is not decreased. In each dataset, we keep 10% of training samples as a validation dataset for model selection. All the experiments are repeated five times and the average performances are reported.

A.2 Evaluation Metrics for Misclassification Detection

We use the standard metrics in Hendrycks & Gimpel (2017) to measure the quality of misclassification detection: **FPR at 95%-TPR**, **AUPR-S**, **AUPR-E** and **AUROC**.

(1) **FPR at 95%-TPR** can be interpreted as the probability that a negative (misclassified) example is misclassified as a correct prediction when the true positive rate (TPR) is as high as 95%. True positive rate can be computed by TPR=TP/(TP+FN), where TP and FN denote the number of true positives and false negatives respectively. The false positive rate (FPR) can be computed by FPR=FP/(FP+TN), where FP and TN denote the number of false positives and true negatives respectively.

(2) **AUPR** is the Area under the Precision-Recall (PR) curve. The PR curve is a graph showing the precision=TP/(TP+FP) versus recall=TP/(TP+FN). The metric **AUPR-S** and **AUPR-E** indicate the area under the precision-recall curve where correct predictions and errors are specified as positives, respectively. Specifically, AUPR-Error constitutes the primary metrics in our tests, since we want to detect misclassified samples.

(3) **AUROC** measures the Area Under the Receiver Operating Characteristic curve (AUROC). The ROC curve depicts the relationship between True Positive Rate and False Positive Rate. This metric is a threshold-independent performance evaluation. The AUROC can be interpreted as the probability that a positive example is assigned a higher prediction score than a negative example.

A.3 Out-of-distribution Detection

A.3.1 Evaluation Metrics

Similarly to the metrics of misclassification detection, **AUROC**, **AUPR-In**, and **AUPR-Out** are the widely used metrics for OOD detection. The metric **AUPR-In** and **AUPR-Out** indicate the area

under the precision-recall curve where in-distribution samples and out-of-distribution samples are specified as positives, respectively.

A.3.2 OOD DATASETS

The test set from CIFAR-10 (CIFAR-100) dataset can be viewed as the in-distribution (positive) examples. For out-of-distribution (negative) examples, we test on MNIST LeCun (1998), Fashion-MNIST (Xiao et al., 2017), LSUN (resize) Yu et al. (2015) and TinyImageNet (resize).

(1) **LSUN.** The Large-scale Scene UNderstanding dataset (LSUN) contains 10,000 test images from 10 different scenes (Yu et al., 2015). We downsample each image to size 32×32 to construct LSUN (resize).

(2) **TinyImageNet.** The Tiny ImageNet dataset consists of contains 10,000 test images from 200 different classes, which is a subset of ImageNet images dataset (Deng et al., 2009). Similar to LSUN, We downsample each image to size 32×32 to construct TinyImageNet (resize).

A.4 FEW-SHOT LEARNING

Datasets and network architectures. To verify the effectiveness of classAug for few-shot learning task, we build on the ProtoNet (Snell et al., 2017) and the Ridge Regression Differentiable Discriminator (R2D2) (Bertinetto et al., 2018) methods, and evaluate on FC100 (Oreshkin et al., 2018) and CIFAR-FS (Bertinetto et al., 2018) datasets.

(1) **FC100** is derived from CIFAR-100, and the 100 classes are further grouped into 20 superclasses. The train split contains 60 classes from 12 superclasses, the validation and test contain 20 classes from 5 superclasses each.

(2) **CIFAR-FS** is randomly sampled from CIFAR-100, and divided into training 64 classes, 16 validation classes and 20 test classes, espectively.

Implementation details. Following Mishra et al. (2018); Oreshkin et al. (2018), we employ ResNet-12 as the backbone feature extractor. We set both training way and shot to 5 for all methods and datasets. Specifically, for each task, we firstly sample 3 classes uniformly and 5 training samples per class, then classAug is applied to generate additional 2 classes with 5 training samples per class. Moreover, classAug is also applied to query set during training, and each class contains 6 query examples. During testing, we use 15 query images per task for 5-shot, 3-shot and 1-shot. In meta-learning stage, the model was trained for 70 epochs. In order to further boosting the performance, we save the model at the end of each epoch and use the average prediction of saved models as final results during testing.