PuzzleShuffle: Undesirable Feature Learning for Semantic Shift Detection

Yusuke Kanebako(✉)* and Kazuki Tsukamoto*

Ricoh Company, Ltd.
{yuusuke.kanebako, kazuki.tsukamoto}@jp.ricoh.com

Abstract. When running a machine learning system, it is difficult to guarantee performance when the data distribution is different between training and production operations. Deep neural networks have attained remarkable performance in various tasks when the data distribution is consistent between training and operation phases, but performance significantly drops when they are not. The challenge of detecting Out-of-Distribution (OoD) data from a model that only trained In-Distribution (ID) data is important to ensure the robustness of the system and the model. In this paper, we have experimentally shown that conventional perturbation-based OoD detection methods can accurately detect non-semantic shift with different domain, but have difficulty detecting semantic shift in which objects different from ID are captured. Based on this experiment, we propose a simple and effective augmentation method for detecting semantic shift. The proposed method consists of the following two components: (1) PuzzleShuffle, which deliberately corrupts semantic information by dividing an image into multiple patches and randomly rearranging them to learn the image as OoD data. (2) Adaptive Label Smoothing, which changes labels adaptively according to the patch size in PuzzleShuffle. We show that our proposed method outperforms the conventional augmentation methods in both ID classification performance and OoD detection performance under semantic shift conditions.

Keywords: Semantic Shift Detection · Data Augmentation · Out-of-Distribution Detection.

1 Introduction

When running a machine learning system, it is difficult to guarantee performance when the data distribution is different between training and production operations. It is important to detect such data not included in the training data or build a model that can make predictions with low confidence for untrained data to ensure the reliability and safety of machine learning systems. Deep neural networks (DNNs) have attained remarkable performance in various tasks when the data distribution is consistent between training and running phases. However, it is difficult to guarantee robustness when the domain changes between

[*] Equal contribution

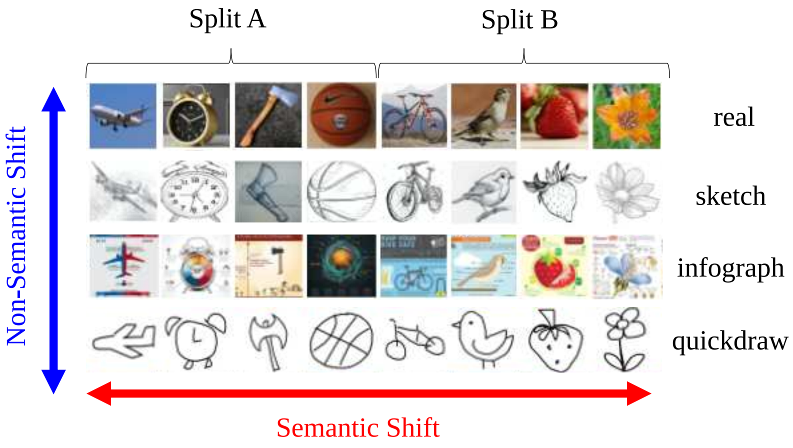

Fig. 1. Domainnet [17]

training and operation or when unexpected objects are captured. This challenge has been formulated as learning only In-Distribution (ID) data and detecting Out-of-Distribution (OoD) data [8], and many methods have been proposed in recent years [10, 14–16, 22].

The cause of the factor difference in distribution between ID and OoD does not distinguish in previous studies on OoD detection. As shown in Fig. 1, GenelizedODIN [10] uses the DomainNet Dataset [17] to separate the problem of OoD detection into two categories: Semantic Shift, in which the class of the object is different between ID and OoD in the same domain, and Non-Semantic Shift, in which the class of the object is the same, but the domain is different. The results showed that the previous OoD detection methods perform excellently to non-semantic shift detection but could not outperform the baseline MaxSoftmax-based method [8] in semantic shift detection.

Semantic shift detection is one of the most critical issues in the operation of machine learning systems. It is necessary to reject the prediction results or consider adding them as untrained data by lowering the confidence level of the prediction for unexpected objects. However, the prediction of DNNs is known to be high-confidence, and calibration by temperature scaling and adversarial training is reported to be effective for this problem [5, 7]. In the framework of Bayesian DNNs, a learning method that theoretically guarantees the uncertainty of the prediction has been proposed [2, 23]. Besides, some data augmentation methods show to improve the uncertainty and robustness of the DNNs model [9, 24, 27, 28].

In this paper, we focus on the semantic shift in OoD detection. In the problem setting where the domain of image data is the same, but the classes of ID and OoD are different, the goal is to detect OoD data from a model that only trained ID data. To address this problem, we propose a new data augmentation method named PuzzleShuffle. Our method was inspired by [14]. The key concept is to

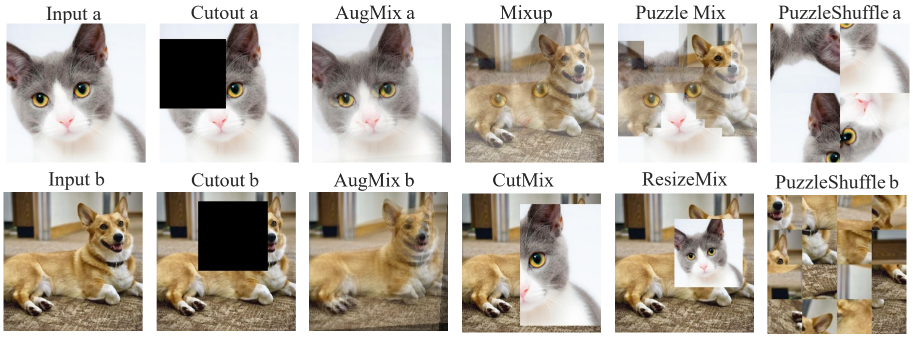

Fig. 2. A visual comparison of Cutout [1], AugMix [9], Mixup [28], CutMix [27], ResizeMix [19], Puzzle Mix [12] , and our PuzzleShuffle

make the model explicitly train with data that has an undesirable feature. Fig. 2 shows a comparison with conventional augmentation methods. PuzzleShuffle divides the image into some patches. And the patches are randomly rearranging to intentionally destroy the semantic information of the image, and then the models are trained with images that have undesirable features. The labels of the data to which PuzzleShuffle is applied are adaptively smoothed according to the patch size. For images with large patch size, we give labels close to one-hot distribution because we believe that there is still much semantic information, and for images with small patch size, we give labels close to uniform distribution because we believe that the semantic information is strongly corrupted. In this way, DNNs can learn to predict the semantic information with lower confidence as they move away from the ID. To verify our proposed method's effectiveness, we evaluated OoD detection's performance under the semantic shift using various datasets. As a result, we show that our proposed method outperforms the conventional augmentation methods in both the performance of ID classification accuracy and OoD detection performance under the semantic shift conditions.

In summary, our paper makes the following contributions:

- We show that the existing perturbation-based OoD detection methods cannot outperform the baseline method's OoD detection performance in semantic shift conditions.
- We show that adversarially trains the data with features not included in ID data and effectively improves OoD detection performance under the semantic shift conditions.
- We proposed a new simple and effective augmentation method to improve OoD detection accuracy under the semantic shift conditions.
- We show that the proposed method improves OoD detection performance in combination with any conventional augmentation methods.

2 Related Work

2.1 Out-of-Distribution Detection

The problem of OoD detection was formulated by [8], and the proposed method of separating ID and OoD using the maximum softmax value of DNNs is widely used as a baseline. ODIN [16] performs OoD detection by applying a perturbation to the input image such that the maximum softmax value increases. Similarly, Mahalanobis [15] also detect OoD using perturbation but assumes that the feature map's intermediate output follows a multivariate gaussian distribution and calculates the distance between the distributions during training and testing using the mahalanobis distance, and uses that value as the threshold for OoD detection. [10,22] does not use perturbation and calculates the logit using cosine similarity instead of the linear transformation before the softmax function. [14] uses generative adversarial nets (GANs) [4] to generate boundary data ID and OoD and training generated data for confidence calibration and improvement of OoD detection. In any research, the problem of OoD detection under the semantic shift has not been solved. Besides, GeneralizedODIN [10] shows that the existing OoD methods cannot outperform the baseline method's [8] OoD detection performance in semantic shift conditions.

2.2 Data Augmentation

Data augmentation can improve the generalization performance of the model and uncertainty and robustness [9,24,27,28]. CutMix [27] randomly cuts a portion of an image and pastes it at a position corresponding to the cut position in another image to improve performance, confidence calibration, and OoD detection. ResizeMix [19] pointed out that CutMix may not capture the intended object in the cropped image and clarified the importance of capturing the object and, ReizeMix outperforms CutMix by pasting a resized image instead of cropping the image. Mixup [28] proposes a method to compute convex combination of two images pixel by pixel, and Puzzle Mix [12] achieves effective mixup by using saliency information and graph cut. AugMix [9] improves the robustness and uncertainty evaluation by applying multiple augmentations to a single image and training the weighted combined image and the original image to be close in distribution. All the methods have improved test accuracy, OoD detection accuracy, robustness against distortion images, and uncertainty evaluation, but OoD detection under semantic shift has not been verified. Our method is novel in that it learns undesirable features as ID, and we propose that it can improve the OoD detection performance by giving appropriate soft labels to the data.

2.3 Uncertainty Calibration

In order to achieve high accuracy in the perturbation-based OoD detection described above, the confidence of the DNNs prediction must be properly calibrated. The confidence level of DNNs prediction is known to be high-confidence

[5, 7], which means that the confidence level of DNNs is high even though the prediction results are wrong. Some Bayesian DNNs methods provide theoretical guarantees on uncertainty estimation [2, 23]. In these methods, the estimation of uncertainty is theoretically guaranteed by using Dropout and Batch Normalization. However, although both of these methods achieve confidence calibration, they have not been reported as effective OoD detection methods.

In contrast to these works, we develop a new augmentation method for semantic shift detection. Inspired by [14], our method proposes a simple and effective augmentation method that can improve OoD detection performance under the Semantic Shift by explicitly training data with features not found in ID. First, we experimentally demonstrate the possibility of improving OoD detection performance by adversarial training data with uniformly distributed labels that have features not found in ID. Based on the results, we propose an augmentation method that intentionally corrupts the semantic information of ID data and learns the data as undesirable ID data.

3 Preliminaries

In this chapter, we conduct two preliminary verifications to propose a method to improve semantic shift detection performance. Section 3.1 discusses why the perturbation-based OoD detection method fails to detect OoD data under semantic shift conditions. In Section 3.2, inspired by [14], we investigate adversarial training using explicitly semantic shift data that can improve the detection performance of semantic shift. The experimental settings are the same as those described in Section 5.

3.1 The Effects by Perturbation

We investigate the effectiveness of perturbation-based methods, which have been reported to be effective as OoD detection methods for semantic shift and non-semantic shift. We use MaxSoftmax [8], a baseline method, as the OoD detection method without perturbation, and ODIN [16] as the method with perturbation. To compare them, we choose four domains (real, sketch, infograph, and quickdraw) from the Domainnet [17] dataset and divide them into two classes: class labels 0-172 as A, and class labels 173-344 as B, for a total of eight datasets. The A group in the real domain use as ID, and the other groups evaluate as OoD. The results show in Table 1. The perturbation-based method detects OoD with higher accuracy than the method without perturbation in the non-semantic shift detection. However, the perturbation-based method is inferior to without perturbation in the semantic shift detection. The reason for this may be that the more similar the OoD features are to the ID features, the more they are embedded in the similar features by perturbation. ODIN tries to separate ID and OoD by adding perturbations to the image to increase the softmax value, so when similar features are obtained, OoD is also perturbed similarly to ID, making separation difficult. Fig. 3 shows the result of plotting the intermediate features of semantic shift (real-B) and non-semantic shift (quickdraw-A) in two dimensions by tSNE. It can be seen that the non-semantic shift, which dynamically changes the trend of the image, produces features that are not similar to ID, while the semantic shift, which is in the same domain, produces features that are similar. Therefore, we hypothesize that it is important to explicitly learn undesirable features in order to improve the detection performance of semantic shift.

3.2 Adversarial Undesirable Feature Learning

In this section, we verify the hypothesis that semantic shift detection performance can be improved by explicitly learning undesirable features. We use the

Table 1. The OoD detection performance to semantic shift and non-semantic shift by perturbation

OoD	Shift		AUROC
	S	NS	Baseline / ODIN*
real-B	✓		**68.2** / 65.0
sketch-A		✓	70.6 / **75.0**
sketch-B	✓	✓	75.5 / **78.8**
infograph-A		✓	75.6 / **80.2**
infograph-B	✓	✓	76.65 / **81.7**
quickdraw-A		✓	70.3 / **96.0**
quickdraw-B	✓	✓	71.7 / **96.7**

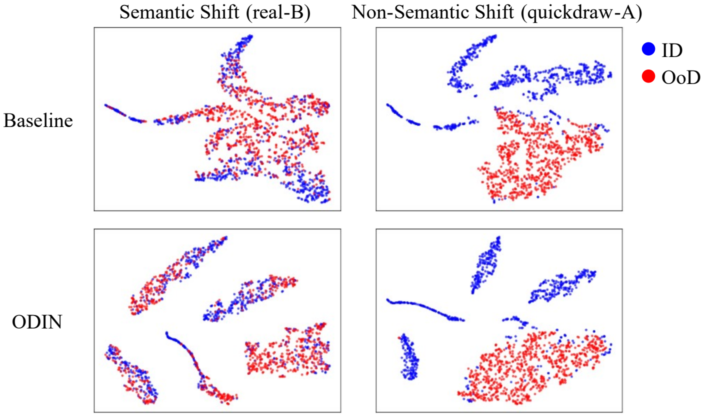

Fig. 3. Results of tSNE visualization of features from conv layer output in semantic shift and non-semantic shift data. The blue points indicate ID, and the red points indicate OoD. The semantic shift tends to extract features similar to ID, and the distribution of ID becomes closer to the semantic shift when the perturbation is applied. On the contrary, the non-semantic shift tends to extract features different from ID, and the ID distribution does not overlap with the non-semantic shift even after perturbation.

CIFAR-10 dataset as ID and the CIFAR-100 dataset as OoD for adversarial training to verify this hypothesis. In adversarial training, the ID data is training with one-hot labels, and the OoD data is training with uniform distribution labels. We explicitly train the OoD data as undesirable features by training the OoD data with uniformly distributed labels. We split the 100 classes of CIFAR-100 into five types, from split1 to split5, based on 20 superclasses, and observe the effect of increasing the variation of OoD classes step by step. The ratio of the number of ID and OoD data included in a minibatch during training is 1:1. Table 2 shows the results. The results indicate that training the data that has undesirable features with uniform labels improves ID accuracy and OoD detection.

Fig. 4 is the extracted features from the convolution layer of the model trained with OoD and without one. The results show that adversarial training of undesirable feature embeds the unobserved data to the non ID space, significantly improving the semantic shift detection performance. The hypothesis that adversarial training of undesirable features improves semantic shift detection by

Table 2. ID accuracy and AUROC of each OoD split when adversarial learning with adding OoD step by step. The OoD split used for training has the AUROC value in bold for each testing OoD split.

train OoD	ID Acc.	AUROC				
		split1	split2	split3	split4	split5
none	86.65	82.41	80.40	79.87	81.61	78.85
split1	88.47	**96.94**	92.54	85.76	83.36	83.65
split1˜2	88.94	**96.93**	**98.64**	89.41	83.48	86.32
split1˜3	89.40	**97.39**	**98.30**	**97.20**	83.33	85.72
split1˜4	89.65	**96.49**	**98.07**	**96.54**	**91.51**	84.10
split1˜5	90.06	**96.27**	**97.83**	**96.54**	**92.77**	**92.86**

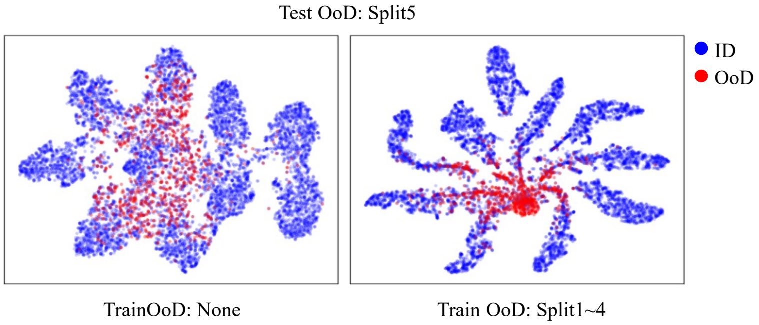

Fig. 4. Results of tSNE visualization of features from conv layer output (Left: train OoD is none, Right: train OoD is split1-4). By explicitly training OoD data as undesirable features, we show that unobserved OoD data are embedded in places that are not ID regions.

using OoD data is revealed. However, in a machine learning system operation, OoD data cannot be accessed in advance. Therefore, it is necessary to learn undesirable features using only ID data. To solve this problem, we propose a new augmentation method that destroys the semantic structure and intentionally induces semantic shift by shuffling the patches in the image like a puzzle.

4 Proposed Method

Fig. 5 illustrates the proposed method. This method consists of two steps: (a) applying augmentation to the image and (b) adaptively changing the label according to the augmentation result. Details are described below.

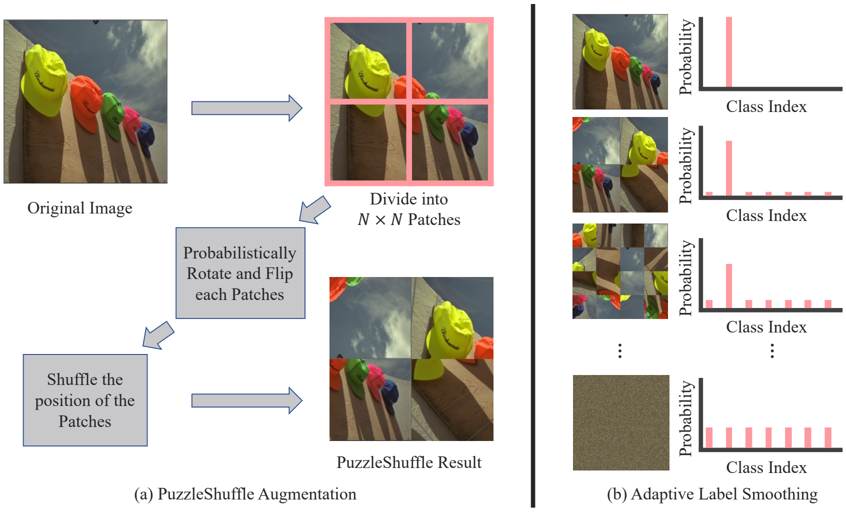

Fig. 5. Proposed Method

4.1 PuzzleShuffle Augmentation

Algorithm 1 describes the proposed method named Puzzleshuffle. PuzzleShuffle is a simple augmentation method that divides an image into patches of arbitrary size and applies probabilistic rotate or flip augmentation to each patch. After then, the patch positions rearrange randomly, and shuffled images use as training data. In this method, the size of the input image and the patch size are limited to be square. The number of divisions is randomly selected from a divisor of the size of the image when creating a mini-batch during training. A similar method is PatchShuffle regularization [11] method, which randomly shuffles the pixels in a local patch in the image or feature map. Our method differs in that the patch

Algorithm 1: PuzzleShuffle Augmentation

```
Input: Dataset $\mathcal{D}$, probability $p$ of applying PuzzleShuffle, Operations
  $\mathcal{O}$ ={rotate, flip}
Output: A puzzle shuffled image $\tilde{x}$ and its adaptively changed label $\tilde{y}$
$divisors$ = CaluclateDivisor($image\_size$)
Sample $(x, y) \sim \mathcal{D}$
$\beta$ ~Bernouli($p$)
if $\beta = 1$ then
    Sample $patch\_size$ ~ RandomSelect(divisors)
    $divided\_images$ = DivideImage($x$, $patch\_size$)
    for $i = 1, ..., patch\_size \times patch\_size$ do
        $divided\_image_i \leftarrow \mathcal{O}(divided\_image_i)$
    ShufflePatchPosition($divided\_image$)
    $\tilde{x} \leftarrow divided\_image$
    Set $label$ according to Algorithm 2
    $\tilde{y} \leftarrow label$
else
    # Original data is returned.
    $\tilde{x} \leftarrow x$
    $\tilde{y} \leftarrow y$
```

size is variable and the shuffle is performed while preserving the global features, and the labels are changed adaptively according to the patch size as described below.

4.2 Adaptive Label Smoothing

If the number of divisions in PuzzleShuffle is small, the structural information of the image remains, and if the number of divisions is large, the structural information is collapsed. Since images with many patches in PuzzleShuffle are like random noise, it is inappropriate to train them with one-hot labels. We propose a method to adaptively change the distribution of labels according to the number of divisions. When the number of divisions is large, we give label information with a distribution close to one-hot labels. When small, we give label information with a distribution close to the uniform distribution. Algorithm 2 has described Adaptive Label Smoothing. The basic idea is the same as that of Label Smoothing. The target class value discounts from the one-hot label and distributes the discounted value to other class labels. Label Smoothing has attained remarkable improvement of generalization performance as a regularization method for DNNs [18, 21]. We prepare a lookup table, a list of values from the inverse number of classes to 1.0, equally divided by image size and sorted in descending order. We select values from the lookup table using the selected divisor number at PuzzleShuffle Augmentation as an index and use the selected values as the target class values for Label Smoothing. In this way, the labels of PuzzleShuffle image assign according to the patch size.

Algorithm 2: Adaptive Label Smoothing

```
Input:
$C \cdots$ the number of classes
$LUT \cdots$ A lookup table of numbers from the inverse of $C$ to 1.0, equally
  divided by image size and sorted in descending order.
$index \cdots$ Selected divisor by Altorithm 1
Output: Smoothed label $\hat{y}$
$score = LUT(index)$
$residual = (1 - score)/(C - 1)$
$$\hat{y}[i] = \begin{cases} score & \text{(if } i = y\text{)} \\ residual & \text{(otherwise)} \end{cases} \quad (1)$$
```

4.3 Motivation

The motivation for PuzzleShuffle is the effect of learning undesirable features, as shown in Section 3. In the semantic shift problem, the structural information of data is different between ID and OoD. Thus, we believe that it is important to learn structural information not available in ID data explicitly for semantic shift detection. In situations where OoD data is not available, it is necessary to create it from ID data. In [14], the boundary between ID and OoD is generated by GANs. However, GANs are generally expensive and difficult to learn stably, so we divided the image into patches and randomly rearranged the patches' positions. Convolutional neural networks tend to make decisions based on texture information rather than structural information of images [3]. Therefore, to detect semantic shift, we thought it is essential to give appropriate labels to images with broken structural information when learning structural information not present in ID data.

5 Experiments

5.1 Experimental Settings

Networks and Training Details: We use ResNet-34 [6] for all experiments. It is trained with batch size 128 for 200 epochs with and weight decay 0.0005. The optimizer is SGD with momentum 0.9, and the initial learning rate set to 0.1. The learning rate decreases by factor 0.1 at 50% and 75% of the training epochs.

Datasets: In the experiments, we use CIFAR-10/100 [13], Tiny ImageNet [20] (cropped and resized), LSUN [26] (cropped and resized), iSUN [25], Uniform noise, Gaussian noise and DomainNet [17]. If one of the CIFAR-10/100 use as ID, the other is evaluated as OoD. Tiny ImageNet, LSUN, iSUN, Uniform noise, and Gaussian noise are all used as OoD. The experiments using DomainNet follow the experimental method of GeneralizedODIN [10]. We divide the images in each domain into two groups: A for class labels 0-172 and B for class labels 173-344. The A group in the real domain use as ID, and the other groups evaluate as OoD.

Evaluation Metrics: Following previous OoD detection studies [8, 10, 14–16, 22], we use the area under the receiver operating characteristic curve (AUROC) and true negative rate at 95% true positive rate (TNR@TPR95) as the evaluation metrics. We also evaluate the classification performance of ID data. For all of these metrics, a higher value indicates better performance.

5.2 Compared Methods

We use Cutout [1], AugMix [9], Mixup [28], CutMix [27], ResizeMix [19], and Puzzle Mix [12] to compare augmentation methods. We evaluate these augmentation methods performance alone and in combination with standard augmentation (i.e., crop, horizontal flip) and the proposed methods. In comparison with the OoD detection method, we evaluate the performance of combining the proposed methods on the DomainNet dataset. We employed the MaxSoftmax-based method [8] as Baseline and compared it with ODIN [16] and Scaled Cosine [10].

5.3 Results

Comparison of augmentation method Table 3 shows the results when each augmentation applies by itself and standard augmentations are not in use. In many experiments, our method has shown high OoD detection performance. In particular, we achieve high detection performance on datasets where the image is resized instead of cropped and Uniform and Gaussian noise datasets. This is because the proposed method can learn to focus on the image structure information and the minimum patch size is one pixel.

Combination of augmentation method Table 4 shows the results of combining standard augmentation methods such as crop and horizontal flip with existing augmentation methods and our proposed method. The results show that for many augmentation methods, the combination of our proposed method can improve the performance of OoD detection. Table 5 shows the classification performance of ID data when using each augmentation method combined with our proposed method. In all cases, the performance does not degrade significantly. Therefore, from Tables 4 and 5, we can see that our method can improve the OoD detection performance while maintaining the ID data classification performance.

Table 3. Performance comparison of each augmentation methods.

ID	OoD	Method (AUROC/TNR@TPR95)							
		Baseline	Cutout	Mixup	CutMix	AugMix	ResizeMix	Pazzle Mix	Our
CIFAR-10	C100	80.5/19.8	82.6/23.0	81.1/24.3	81.4/22.8	82.7/21.1	68.8/27.3	80.7/26.3	**86.5/30.1**
	TINc	80.6/18.3	84.6/26.1	82.7/25.5	89.8/**39.5**	84.2/25.1	77.9/36.1	87.9/35.8	**89.9**/36.3
	TINr	76.0/16.6	80.9/21.7	80.6/23.7	89.0/39.2	84.2/24.4	75.4/42.3	92.5/49.8	**94.9/63.3**
	LSUNc	80.7/14.6	80.5/20.0	81.9/23.2	87.3/32.1	89.0/**38.9**	68.5/27.5	80.1/26.7	**90.1**/36.8
	LSUNr	80.8/19.3	86.3/29.5	84.6/30.2	92.3/49.1	86.8/28.1	88.5/60.1	94.0/55.8	**95.8/69.4**
	iSUN	79.8/20.1	85.4/27.6	83.5/28.7	91.8/47.6	86.3/28.0	85.6/54.9	94.1/56.8	**96.0/71.2**
	Uniform	86.0/19.3	88.8/31.0	81.0/11.3	83.8/19.3	97.7/83.7	92.7/51.9	96.2/72.4	**100.0/100.0**
	Gaussian	97.9/85.3	90.6/35.6	85.6/12.6	83.2/19.0	98.6/92.0	51.2/15.0	93.6/53.3	**100.0/99.9**
CIFAR-100	C10	66.6/9.7	69.4/12.2	70.3/11.7	70.0/11.3	69.6/12.1	**74.1/15.3**	71.3/71.3	73.6/14.7
	TINc	75.0/19.0	74.5/19.5	78.6/23.0	73.7/11.2	64.5/6.0	**82.1/26.7**	76.3/16.1	79.5/22.1
	TINr	69.2/12.5	62.9/9.1	67.5/9.2	45.4/1.7	73.1/14.3	78.7/21.5	54.3/4.7	**88.3/49.2**
	LSUNc	66.5/9.7	67.0/10.5	63.2/10.7	68.8/9.0	53.9/4.1	**77.4/21.2**	69.9/9.4	75.2/12.6
	LSUNr	71.5/13.8	63.8/8.2	69.2/8.7	45.3/1.3	73.2/14.6	80.4/22.6	54.6/3.6	**88.6/50.1**
	iSUN	69.4/12.1	62.1/7.1	67.6/7.7	44.4/1.1	70.0/12.2	78.2/20.0	54.2/3.6	**88.1/50.9**
	Uniform	57.5/0.9	35.4/0.1	29.4/0.0	60.0/0.7	33.5/0.0	20.0/0.0	91.0/49.3	**100.0/100.0**
	Gaussian	36.7/0.0	73.4/5.2	38.4/0.0	64.7/1.7	54.4/0.0	94.1/58.7	61.7/0.5	**100.0/100.0**

Table 4. The results of the combinations of augmentation methods. SA indicate using standard augmentation (i.e. crop and horizontal flip). The numbers in parentheses indicate the performance when the proposed method is combined, and the bold type indicates the improvement of performance by the proposed method.

ID	OoD	Method (AUROC/TNR@TPR95)					
		SA (+**Our**)	SA+Cutout (+**Our**)	SA+Mixup (+**Our**)	SA+CutMix (+**Our**)	SA+ResizeMix (+**Our**)	SA+Puzzle Mix (+**Our**)
CIFAR-10	C100	86.7/36.2 (**89.0/40.7**)	89.9/43.5 (**90.3/44.5**)	74.9/37.1 (**82.7/37.9**)	85.7/38.6 (**88.8/46.4**)	83.6/43.7 (**88.1/46.6**)	83.8/44.7 (**87.6/46.1**)
	TINc	91.7/49.5 (**92.2**/48.3)	94.1/58.6 (**94.3/60.0**)	83.2/58.1 (86.3/50.1)	96.1/73.1 (**96.7/77.2**)	91.2/54.4 (**93.1/56.6**)	97.1/84.0 (96.3/74.7)
	TINr	88.6/39.6 (**96.9/77.7**)	93.6/56.7 (**97.5/83.0**)	84.5/39.7 (**97.4/82.8**)	97.1/84.3 (**98.9/95.6**)	76.6/53.8 (**95.1/71.5**)	97.2/82.9 (**98.8/93.8**)
	LSUNc	93.6/57.5 (93.3/53.3)	93.9/57.7 (**95.4/66.6**)	84.8/68.4 (**89.0**/60.5)	94.4/61.1 (**97.1/81.0**)	91.2/53.5 (**93.8/58.4**)	95.3/77.1 (**96.1**/76.5)
	LSUNr	90.7/46.5 (**97.6/83.4**)	94.9/63.8 (**97.6/83.9**)	88.2/49.2 (**98.0/87.6**)	98.2/92.5 (**99.2/98.3**)	86.2/68.3 (**97.1/80.3**)	98.2/92.2 (**99.2/96.6**)
	iSUN	89.9/44.3 (**97.4/82.0**)	94.8/62.8 (**97.6/83.8**)	88.0/46.9 (**97.8/86.3**)	97.9/89.8 (**99.2/97.7**)	83.6/63.7 (**97.1/80.5**)	98.0/90.4 (**99.1/95.5**)
	Uniform	90.0/21.9 (**100.0/100.0**)	87.7/6.2 (**100.0/100.0**)	90.5/19.8 (**100.0/100.0**)	4.5/0.0 (**100.0/100.0**)	92.7/51.9 (**100.0/100.0**)	80.0/5.3 (**100.0/100.0**)
	Gaussian	98.1/89.1 (**100.0/100.0**)	97.2/84.9 (**100.0/100.0**)	98.0/92.9 (97.9/**95.1**)	78.0/3.2 (**97.0/99.8**)	84.5/7.0 (**100.0/100.0**	59.0/0.0 (**100.0/100.0**)
CIFAR-100	C10	75.6/16.5 (**76.5/19.3**)	76.0/16.7 (75.2/16.4)	73.9/18.7 (**74.8/20.5**)	77.7/21.0 (75.9/**21.2**)	75.7/18.6 (**76.3/21.1**)	78.4/21.0 (77.2/20.5)
	TINc	82.3/26.5 (**83.8/32.0**)	80.5/22.7 (**82.6/29.0**)	84.7/39.6 (82.0/31.5)	86.6/36.5 (84.6/34.6)	82.5/30.4 (**85.0/33.9**)	88.9/40.8 (87.2/39.6)
	TINr	76.5/18.5 (**89.9/52.2**)	74.0/17.0 (**92.6/64.3**)	76.1/21.6 (**91.5/60.1**)	84.1/31.8 (**90.4/52.7**)	79.2/25.8 (**88.4/44.4**)	77.2/19.8 (**94.9/72.7**)
	LSUNc	79.5/21.3 (**82.7/29.3**)	75.7/16.4 (**83.5/29.9**)	83.5/38.1 (79.8/26.2)	84.1/31.2 (**84.6/34.4**)	79.7/26.5 (**82.1/27.7**)	85.4/32.1 (85.3/**33.1**)
	LSUNr	78.5/20.3 (**89.6/51.8**)	73.6/14.8 (**92.8/64.1**)	77.2/21.3 (**91.8/59.2**)	86.5/35.1 (**90.5/51.7**)	80.3/26.7 (**88.3/42.5**)	76.8/18.4 (**95.4/74.6**)
	iSUN	77.2/18.6 (**89.0/50.4**)	73.7/14.9 (**92.2/63.8**)	76.0/20.0 (**90.6/56.9**)	84.7/31.0 (**90.2/51.3**)	79.1/24.9 (**88.3/44.0**)	75.1/17.0 (**93.8/68.9**)
	Uniform	74.4/1.0 (**100.0/100.0**)	97.5/86.3 (**100.0/100.0**)	78.2/1.3 (**100.0/100.0**)	90.8/41.8 (**100.0/100.0**)	45.9/0.0 (**100.0/100.0**)	68.8/0.4 (**100.0/100.0**)
	Gaussian	52.8/0.0 (**98.8/97.7**)	80.7/0.0 (**100.0/100.0**)	60.1/0.0 (**100.0/100.0**)	89.6/24.3 (**100.0/100.0**)	35.9/0.0 (**99.7/98.9**)	61.0/0.0 (**99.8/100.0**)

Table 5. Comparison of ID classification accuracy. In all cases, we use the standard augmentation of crop and horizontal flip.

ID	Method	Classification Accuracy
CIFAR-10	Baseline (+**Our**)	94.8(94.8)
	Cutout (+**Our**)	95.4(95.6)
	Mixup (+**Our**)	94.2(94.2)
	CutMix (+**Our**)	96.3(96.2)
	ResizeMix (+**Our**)	96.7(96.3)
	Puzzle Mix (+**Our**)	96.4(95.4)
CIFAR-100	Baseline (+ **Our**)	74.0(76.0)
	Cutout (+**Our**)	74.0(75.3)
	Mixup (+**Our**)	75.4(76.8)
	CutMix (+**Our**)	79.9(80.0)
	ResizeMix (+**Our**)	79.0(80.3)
	Puzzle Mix (+**Our**)	80.4(80.0)

Table 6. Results of combining the proposed method with the OoD detection method using DomainNet.

OoD	Shift		AUROC	TNR@TPR95
	S	NS	Baseline(+Our) / ODIN(+Our) / Cosine(+Our)	
real-B	✓		68.2(**71.5**)/65.0(**69.4**)/66.2(**69.9**)	9.7(**11.5**)/10.1(**11.8**)/8.6(**10.9**)
clipart-A		✓	67.6(**71.0**)/80.1(**81.5**)/70.2(**77.5**)	13.3(**15.5**)/30.5(**33.8**)/13.8(**21.3**)
clipart-B	✓	✓	74.8(**78.1**)/86.5(**87.5**)/77.0(**83.2**)	17.0(**19.4**)/38.2(**42.4**)/16.5(**24.6**)
infograph-A		✓	75.6(**77.6**)/80.2(**81.9**)/79.8(**85.4**)	16.9(**19.2**)/17.8(**23.0**)/20.7(**31.7**)
infograph-B	✓	✓	76.6(**79.2**)/81.7(**83.6**)/80.6(**86.6**)	16.9(**20.2**)/19.9(**25.9**)/20.5(**33.0**)
painting-A		✓	67.1(**71.1**)/55.7(**63.1**)/68.8(**75.4**)	11.0(**12.8**)/ 3.2(**4.2**)/11.9(**17.8**)
painting-B	✓	✓	73.3(**77.1**)/61.2(**69.4**)/74.3(**80.5**)	14.0(**16.7**)/ 4.6(**6.6**)/13.9(**20.7**)
quickdraw-A		✓	70.3(**77.2**)/96.0(**97.1**)/72.6(**78.9**)	12.1(**14.0**)/80.3(**84.4**)/**11.0**(10.8)
quickdraw-B	✓	✓	71.7(**78.7**)/96.7(**97.6**)/74.0(**80.2**)	12.2(**15.4**)/82.8(**87.4**)/**11.8**(11.3)
sketch-A		✓	70.6(**76.1**)/75.0(**80.8**)/72.9(**81.5**)	13.7(**18.2**)/22.1(**29.7**)/13.3(**22.7**)
sketch-B	✓	✓	75.5(**79.4**)/78.8(**83.8**)/77.4(**84.4**)	16.4(**21.1**)/24.0(**32.0**)/15.1(**25.3**)

Comparison of OoD method Table 6 shows the OoD detection results using the DomainNet dataset. The proposed method can improve the OoD detection performance in both cases of semantic shift and non-semantic shift. These results indicate that the proposed method learns features that only exist in ID data (i.e., real-A), thus improving the detection of semantic shifts and the detection of non-semantic shifts. Besides, the proposed method improves the performance of Baseline and existing OoD methods.

5.4 Analysis

Effect of network architecture Table 7 shows the performance of the proposed method for different network architectures. It show that our proposed method improves the performance of all network architectures.

Impact of multiple patch sizes Table 8 shows the results when PuzzleShuffle is performed with single patch size and with multiple sizes. On average, both ID classification performance and OoD detection performance are higher when multiple scales are combined. It shows that it is important to perform PuzzleShuffle with multiple sizes to learn more diverse undesirable features.

Impact of labeling method Our proposed method changed the label according to the patch size, but a method to calculate the image similarity by images before and after applying PuzzleShuffle is also possible. We use two image similarity metrics, SSIM [29] and the cosine similarity of feature vectors obtained from the models trained by ImageNet [20]. The calculated image similarity is applied to the score of Algorithm 2 to give a label. We also compare the results with one-hot and uniform labels for all patch sizes. The results show in Table 9. The image similarity obtained from the pre-trained model shows superior performance in ID classification and OoD detection. These results indicate that it is important to appropriately reflect the similarity to the original image in the label, which is a future challenge.

Table 7. Performance evaluation of various network architectures. We used standard augmentation and combined our proposed method.

ID	OoD	Network	ID Acc.	AUROC	TNR@TPR95
C-10	C-100	ResNet-34	94.8(**94.8**)	86.7(**89.0**)	36.2(**40.7**)
		WideResNet-28-10	95.3(**95.9**)	89.3(**89.7**)	43.1(**43.6**)
		DenseNet-100	94.4(**94.6**)	88.2(**89.0**)	35.2(**38.4**)
C-100	C-10	ResNet-34	74.0(**76.0**)	75.6(**76.5**)	16.5(**19.3**)
		WideResNet-28-10	79.8(79.3)	79.2(**80.1**)	20.8(**22.2**)
		DenseNet-100	75.1(**76.64**)	75.5(**75.7**)	17.8(**18.2**)

Table 8. Performance evaluation for various patch sizes.

ID	OoD	Num. of Div.	ID Acc.	AUROC	TNR@TPR95
C-10	C-100	1×1	94.8	89.0	36.2
		2×2	**95.3**	87.2	**41.0**
		4×4	94.5	86.7	37.1
		8×8	94.9	88.4	39.3
		16×16	93.5	84.8	35.2
		Multi-scale	94.8	**89.9**	40.7
C-100	C-10	1×1	74.0	75.6	16.5
		2×2	75.1	75.7	17.1
		4×4	73.5	74.8	16.3
		8×8	73.4	75.0	16.1
		16×16	74.4	75.5	16.7
		Multi-scale	**76.0**	**76.5**	**19.3**

Table 9. Performance evaluation using various labeling methods. * indicates using pre-trained model.

ID	OoD	Method	ID Acc.	AUROC	TNR@TPR95
C-10	C-100	One-hot	94.8	**89.2**	39.0
		Uniform	94.5	85.6	44.2
		SSIM	94.5	85.4	42.7
		Cosine*	**95.5**	85.9	**49.8**
		Alg. 2	94.8	89.0	40.7
C-100	C-10	One-hot	73.3	74.7	16.9
		Uniform	76.8	75.7	16.1
		SSIM	**77.7**	75.1	16.6
		Cosine*	77.4	**77.2**	**20.1**
		Alg. 2	76.0	76.5	19.3

6 Conclusion

This paper focuses on OoD detection under semantic shift and shows that conventional OoD detection methods cannot detect semantic shift. Our proposed method improves the performance of OoD detection without degrading the performance of ID classification. In the future, we will study OoD detection that can detect not only the semantic shift but also the non-semantic shift and investigate more robust model construction and running machine learning systems.

References

1. DeVries, T., Taylor, G.W.: Improved regularization of convolutional neural networks with cutout. arXiv preprint arXiv:1708.04552 (2017)
2. Gal, Y., Ghahramani, Z.: Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In: Proceedings of The 33rd International Conference on Machine Learning. pp. 1050–1059 (2016)
3. Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F.A., Brendel, W.: Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In: International Conference on Learning Representations (2019)
4. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in Neural Information Processing Systems. vol. 27 (2014)
5. Guo, C., Pleiss, G., Sun, Y., Weinberger, K.Q.: On calibration of modern neural networks. In: Proceedings of the 34th International Conference on Machine Learning. pp. 1321–1330 (2017)
6. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778 (2016)
7. Hein, M., Andriushchenko, M., Bitterwolf, J.: Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 41–50 (2019)
8. Hendrycks, D., Gimpel, K.: A baseline for detecting misclassified and out-of-distribution examples in neural networks. Proceedings of International Conference on Learning Representations (2017)
9. Hendrycks*, D., Mu*, N., Cubuk, E.D., Zoph, B., Gilmer, J., Lakshminarayanan, B.: Augmix: A simple method to improve robustness and uncertainty under data shift. In: International Conference on Learning Representations (2020)
10. Hsu, Y.C., Shen, Y., Jin, H., Kira, Z.: Generalized odin: Detecting out-of-distribution image without learning from out-of-distribution data. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10951–10960 (2020)
11. Kang, G., Dong, X., Zheng, L., Yang, Y.: Patchshuffle regularization. arXiv preprint arXiv:1707.07103 (2017)
12. Kim, J.H., Choo, W., Song, H.O.: Puzzle mix: Exploiting saliency and local statistics for optimal mixup. In: Proceedings of the 37th International Conference on Machine Learning. pp. 5275–5285 (2020)

13. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images. Master's thesis, Department of Computer Science, University of Toronto (2009)
14. Lee, K., Lee, H., Lee, K., Shin, J.: Training confidence-calibrated classifiers for detecting out-of-distribution samples. In: International Conference on Learning Representations (2018)
15. Lee, K., Lee, K., Lee, H., Shin, J.: A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In: Advances in Neural Information Processing Systems. vol. 31 (2018)
16. Liang, S., Li, Y., Srikant, R.: Enhancing the reliability of out-of-distribution image detection in neural networks. In: International Conference on Learning Representations (2018)
17. Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., Wang, B.: Moment matching for multi-source domain adaptation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 1406–1415 (2019)
18. Pereyra, G., Tucker, G., Chorowski, J., Kaiser, Ł., Hinton, G.: Regularizing neural networks by penalizing confident output distributions. arXiv preprint arXiv:1701.06548 (2017)
19. Qin, J., Fang, J., Zhang, Q., Liu, W., Wang, X., Wang, X.: Resizemix: Mixing data with preserved object information and true labels. arXiv preprint arXiv:2012.11101 (2020)
20. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision **115**(3), 211–252 (2015)
21. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2818–2826 (2016)
22. Techapanurak, E., Suganuma, M., Okatani, T.: Hyperparameter-free out-of-distribution detection using cosine similarity. In: Proceedings of the Asian Conference on Computer Vision (2020)
23. Teye, M., Azizpour, H., Smith, K.: Bayesian uncertainty estimation for batch normalized deep networks. In: Proceedings of the 35th International Conference on Machine Learning. pp. 4907–4916 (2018)
24. Verma, V., Lamb, A., Beckham, C., Najafi, A., Mitliagkas, I., Lopez-Paz, D., Bengio, Y.: Manifold mixup: Better representations by interpolating hidden states. In: Proceedings of the 36th International Conference on Machine Learning. pp. 6438–6447 (2019)
25. Xu, P., Ehinger, K.A., Zhang, Y., Finkelstein, A., Kulkarni, S.R., Xiao, J.: Turkergaze: Crowdsourcing saliency with webcam based eye tracking. arXiv preprint arXiv:1504.06755 (2015)
26. Yu, F., Seff, A., Zhang, Y., Song, S., Funkhouser, T., Xiao, J.: Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. arXiv preprint arXiv:1506.03365 (2015)
27. Yun, S., Han, D., Oh, S.J., Chun, S., Choe, J., Yoo, Y.: Cutmix: Regularization strategy to train strong classifiers with localizable features. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6023–6032 (2019)
28. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. In: International Conference on Learning Representations (2018)
29. Zhou Wang, Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing **13**(4), 600–612 (2004)