Knowledge and Information Systems (2024) 66:3855–3881
https://doi.org/10.1007/s10115-024-02141-3

REGULAR PAPER

PatchMix: patch-level mixup for data augmentation in convolutional neural networks

Yichao Hong[1] · **Yuanyuan Chen**[1]

Received: 15 November 2023 / Revised: 6 May 2024 / Accepted: 14 May 2024 /
Published online: 30 May 2024

Abstract
Convolutional neural networks (CNNs) have demonstrated impressive performance in fitting data distribution. However, due to the complexity in learning intricate features from data, networks usually experience overfitting during the training. To address this issue, many data augmentation techniques have been proposed to expand the representation of the training data, thereby improving the generalization ability of CNNs. Inspired by jigsaw puzzles, we propose PatchMix, a novel mixup-based augmentation method that applies mixup to patches within an image to extract abundant and varied information from it. At the input level of CNNs, PatchMix can generate a multitude of reliable training samples through an integrated and controllable approach that encompasses cropping, combining, blurring, and more. Additionally, we propose PatchMix-R to enhance the robustness of the model against perturbations by processing adjacent pixels. Easy to implement, our methods can be integrated with most CNN-based classification models and combined with varying data augmentation techniques. The experiments show that PatchMix and PatchMix-R consistently outperform other state-of-the-art methods in terms of accuracy and robustness. Class activation mappings of the trained model are also investigated to visualize the effectiveness of our approach.

Keywords Data augmentation · Mixup · Classification · Generalization ability · Robustness

1 Introduction

Recent years have witnessed a growing research and application of data augmentation in various domains, such as image classification [1, 2], face recognition [3, 4], moving object detection [5, 6] and text mining [7, 8]. While machine learning techniques excel in various tasks, they often struggle with variations in data distribution [9]. Furthermore, data acquisition and labeling are manual and limited. Relying solely on raw images for training models, many deep neural networks tend to memorize the data, hindering their ability to learn domain invariance and ultimately leading to poor generalization.

✉ Yuanyuan Chen
chenyuanyuan@scu.edu.cn

Yichao Hong
2022223040013@stu.scu.edu.cn

[1] College of Computer Science, Sichuan University, Chengdu 610065, China

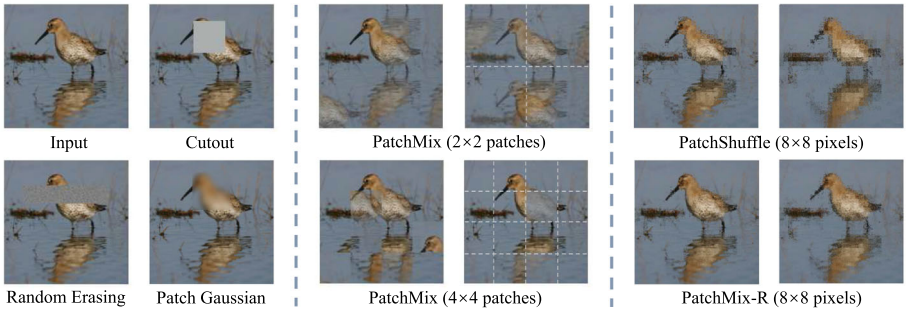

Fig. 1 A visual comparison of the patch-level methods. *Left*: Input and traditional patch-based methods. *Middle*: Our proposed method PatchMix. The notation 2×2 represents dividing an image into 2×2 patches, and each patch is undergone PatchMix. Auxiliary lines are added to the samples in the second column. *Right*: PatchShuffle and our proposed method PatchMix-R. The notation 8×8 means that each non-overlapping patch contains 8×8 pixels, and PatchMix-R is performed on each pixel within the patch. In the left column, only half of patches are processed, while all patches in the images on the right are processed

To address above issues, a range of data augmentation techniques have been proposed. For visual information, traditional data augmentations [1, 10–12] apply various affine transformations, such as horizontal translation, scaling, and squeezing, to enhance the accuracy and robustness of models. In contrast to these global transformations, some patch-level methods [13–15] primarily aim to remove or add noise to patches. However, these operations only perform conservative processing, leaving much of the internal information unexplored. Furthermore, in the field of visual research at the patch-level, there has been a growing trend to consider images as collections of patches. Many visual models are trained directly on patches as input and have demonstrated strong performance [16, 17], highlighting the feasibility of using patches for data representation. Additionally, employing puzzle-like techniques to split and reassemble patches for unsupervised feature learning have shown promising capabilities in knowledge learning and transfer [18]. Therefore, investigating augmentation methods specifically targeting patches holds great potential.

Recently, a line of research known as mixup [19] has been proposed. These methods [20–25] focus on linear or nonlinear mixing of multiple images and labels, allowing the model to better fit multicategory information from different images. Anchor data augmentation [26] and C-mixup [27] have also been proposed to address regression problems. Nevertheless, these derived works introduce additional computational effort and cannot be generalized to unlabeled scenarios such as self-supervised learning [28] due to their strong coupling with labels. Mixup provides a novel approach of linearly combining data, yet in existing research, it has not been applied within an image to enrich the internal information.

In this work, we propose an innovative data augmentation approach called PatchMix. Utilizing patches as the fundamental building blocks, PatchMix develops a blending strategy inspired by mixup to reorganize the image data. As depicted in Fig. 1, PatchMix significantly expands the diversity of the data representation within the domain while preserving its fundamental characteristics. In comparison with existing methods, PatchMix significantly enhances the relative positional information, combination and clarity through cropping, combining and blurring. Label-free and plug-and-play, it can also be seamlessly integrated into existing works and even extended to other domains such as self-supervised learning. To enhance the robustness of the trained model, we extend PatchMix to PatchMix-R, which

alleviates the limitations of PatchShuffle [29] by introducing reasonable perturbations to adjacent pixels.

Our main contributions are three folds:

- We design a lightweight patch-level data augmentation PatchMix for images, which can ensemble with other augmentation methods of varying complexities and visualized to showcase the advantages of training with it.
- We apply PatchMix within patches and propose PatchMix-R. By moderately perturbing the adjacent pixels of the patch, it visibly enhances the robustness of models to noisy samples.
- In the experiments, PatchMix outperforms state-of-the-art methods on CIFAR-10/100 and Tiny-ImageNet. When combined with PatchMix-R, it also improves the robustness of classifiers to noise attacks better than other methods.

2 Related work

2.1 Traditional data augmentation

Traditional data augmentation typically targets individual images, performing basic geometric transformations and color conversions [30]. Based on the content of the augmented images, the traditional approaches can be categorized into image-level and patch-level techniques.

Image-level Random flipping and random cropping, as the most common and effective image-level data augmentation, empirically improve the generalization performance of the neural networks on clean data. Moreover, methods such as sharpness, brightness and Gaussian are also utilized for image augmentation in various works [1, 10, 11]. Subsequently, attention has shifted toward combining different augmentation methods, leading to the proposal of automatic data augmentations [2], such as AutoAugment [31], RandAugment [32], and TrivialAugment [12].

Patch-level Patch-level data augmentation focuses more on the local information of the image. Cutout [13] randomly masks a portion of the image to enhance the accuracy. To achieve a balance between accuracy and robustness, Patch Gaussian [14] adds Gaussian noise to a specific patch. Parameter learning free, Random erasing [15] generates varying levels of occlusion in training images and is easily deployed.

Traditional data augmentation, although simple, does not considerably alter the representation of the data domain. The patch-level methods are specific to one or several patches, conservatively preserving the rough structure of the image, which somewhat limits the diversity of its feature representation.

2.2 Mixup

Linear mixing Mixup [19] trains a neural network to linearly interpolate two training images and their corresponding labels in a random ratio. This regularization-like method has been experimentally shown to improve the accuracy of the model. Manifold Mixup [20] improves upon the Mixup by introducing interpolation at the hidden layer to preserve the manifold structure between input samples. To encourage fair and accurate decision boundaries for all subgroups, Subgroup Mixup [21] develops a pairwise mixup scheme to augment training data.

Nonlinear mixing Instead of processing the entire image, CutMix [22] randomly crops the image and fills the masked patch with another image patch, mixing the labels according to the proportion of patches. PuzzleMix [23] and SaliencyMix [24] incorporate a saliency signal, rendering the selection and mixing of patches non-random. The mixing proportion of the labels is optimized to be determined by the ratio of salient regions. To reduce the loss of saliency inference, AutoMix [25] adaptively generates mixed samples based on mixing ratios and feature maps in an end-to-end manner.

Although many derivative works have improved the reliable and rich representation of samples, the additional computational costs are usually not negligible. Simultaneously, the mixing operation based on multiple samples poses a challenging problem for label calculation. It is worth noting that existing mixup methods are all based on multiple images, and there is no mixup technique available for a single image yet.

3 Proposed method

In this section, we present the general procedure of the proposed method, PatchMix, and examine the mechanisms for controlling the degree of mixing between patches. Moreover, we extend PatchMix to enhance the robustness of the network by building upon the idea of PatchShuffle [29].

3.1 PatchMix

At present, data augmentation techniques predominantly concentrate on operations that involve merging or stitching multiple images [33, 34]. These methods frequently disregard the diversity of feature representations within individual images, depending instead on basic preprocessing steps such as horizontal flipping or random cropping.

In the context of jigsaw puzzle, an interesting observation is that a picture, when disassembled and reassembled, can be orderly reconstructed due to its key features. Inspired by this concept, PatchMix initially segments an image into distinct patches and applies random shuffling. This approach overcomes the limitations associated with relative positional constraints of patches, thereby creating opportunities for diverse information expression. However, PatchMix does not merely substitute the original patch; instead, it utilizes the linear method of mixup to blend patches, mitigating the structural disruptions caused by the shuffling. In this section, we provide a detailed explanation of PatchMix and how it works.

3.1.1 Formulations

Consider a matrix $\boldsymbol{X}$ of dimensions $H \times W$. Divide $\boldsymbol{X}$ into non-overlapping patches of $h \times w$ elements, represented as

$$\boldsymbol{X} = \begin{pmatrix} P_1 & P_2 & \cdots & P_{\frac{W}{w}} \\ P_{\left(\frac{W}{w}\right)+1} & P_{\left(\frac{W}{w}\right)+2} & \cdots & P_{2\times\left(\frac{W}{w}\right)} \\ \vdots & \vdots & \ddots & \vdots \\ P_{\left(\frac{H}{h}-1\right)\times\left(\frac{W}{w}\right)+1} & P_{\left(\frac{H}{h}-1\right)\times\left(\frac{W}{w}\right)+2} & \cdots & P_{\left(\frac{H}{h}\right)\times\left(\frac{W}{w}\right)} \end{pmatrix}, \tag{1}$$

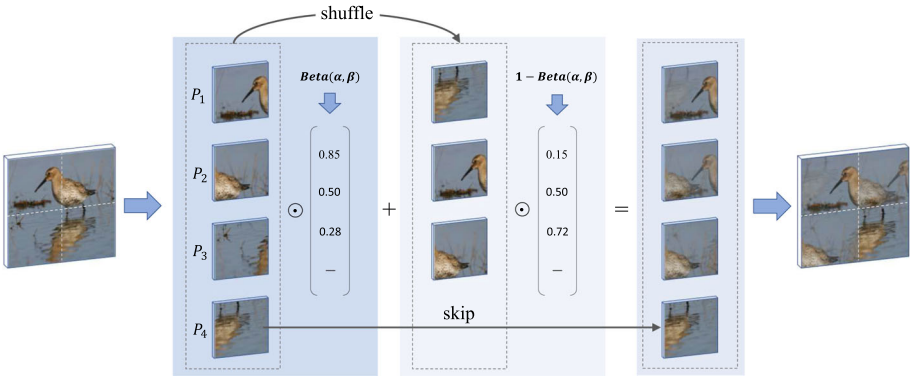

Fig. 2 An illustration of PatchMix on one image ($H \times W$) divided into four non-overlapping patches ($(H/2) \times (W/2)$). The notation $\odot$ indicates that the pixel values of each patch are multiplied by the corresponding weight values in the right column. The shuffled patches are independent of the original patches. Note that there is a possibility that several patches will directly skip to the final, as illustrated in the bottom-most patch

where P_i represents the i-th patch after being split. A random binary switch r determines whether the matrix P_i undergoes the PatchMix transformation. Let the random variable r follow a Bernoulli distribution, $r \sim Bernoulli(\epsilon)$, such that $r = 1$ with probability ϵ and $r = 0$ with probability $1 - \epsilon$. The resulting matrix $\tilde{P}_i$ can be expressed as

$$\tilde{P}_i = (1 - r)P_i + rT(P_i), \tag{2}$$

where $T(\cdot)$ denotes the PatchMix transformation and is formulated as

$$T(P_i) = \lambda P_i + (1 - \lambda)P_{\text{index}[i]}, \tag{3}$$

where index is a randomly generated sequence number obtained by shuffling $\left[1, (H/h) \times (W/w)\right]$, and the mixing ratio λ is randomly sampled from a beta distribution.

3.1.2 PatchMix on images

In the process of visual perception, fragments of an object can supply valuable and sufficient information for classification, without the need to consider the entire object or rely on absolute positional relationships [35]. Similarly, in many computer vision tasks, input images can be treated as matrices and subjected to the PatchMix transformation. As depicted in Fig. 2, The image is divided into P_1, P_2, P_3, P_4 in accordance with Eq. (1). P_4 retains the original patch values, while P_1, P_2, P_3 are mixed based on Eq. (3). After mixing, all patches are reassembled into the image according to the original sequence. The specific processing details are provided in Algorithm 1.

This paper simultaneously elucidates the benefits of our method in adjusting the classification boundary. In Fig. 3, when the training data are limited, classification boundary is prone to overfitting during the training. In such cases, test data are typically not well distinguished. Traditional data augmentation methods (random flipping and random cropping) can to some extent expand the input space of the data, thereby reducing the occurrence of overfitting. PatchMix further expands the representation of image data by manipulating the information through cropping, combining, blurring, etc. The generated data are typically more complex

Algorithm 1 PatchMix Procedure

Input: Input image: I with spatial sizes $H \times W$; patch-mixed probability: p; Patch height: h, Patch width: w; Parameters of beta distribution: α, β. (For simplicity, H and W can be divided by h and w, respectively.)
Output: Mixed image: I^*.

$N \leftarrow (H/h) \times (W/w)$
Divide I into N non-overlapping $h \times w$ patches: $P_1, P_2, \cdots, P_N$
$P_1^*, P_2^*, \cdots, P_N^* \leftarrow P_1, P_2, \cdots, P_N$
index $\leftarrow$ *Randperm* (N)
for $i \leftarrow 1$ to N **do**
  if $Rand\ (0, 1) \geq p$ **then**
    $P_i^* \leftarrow P_i$
  else
    $\lambda \leftarrow Beta\ (\alpha, \beta)$
    $P_i^* \leftarrow \lambda\ P_i + (1\ -\ \lambda)\ P_{index[i]}$
  end if
end for
$I^* \leftarrow P_1^*, P_2^*, \cdots, P_N^*$
return I^*

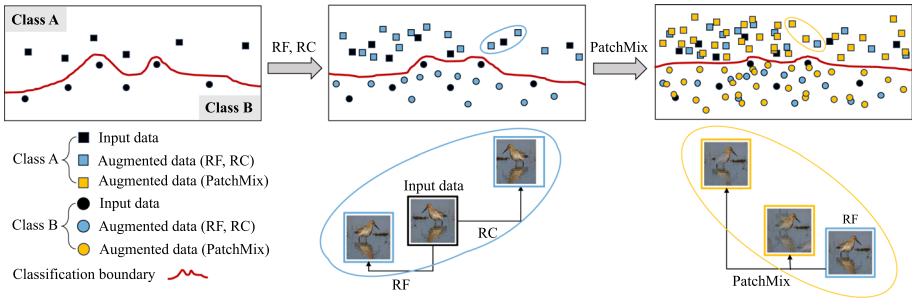

Fig. 3 Classification boundary adjustment by applying data augmentation. *Left*: Classification boundary learned only by input data. *Middle*: Classification boundary adjusted by adding augmented data with traditional methods, random flipping (RF) and random cropping (RC). *Right*: Classification boundary adjusted by adding augmented data with RF, RC and PatchMix. Partial augmentation processes are visualized in the bottom

and diverse. Thus, the trained model is able to better fit the features of different classes of data, thereby adjusting the classification boundary for improved discrimination.

3.1.3 Mixing control

In PatchMix, the blending of patches is controlled by associated hyperparameters, including the patch-mixed probability p, the size of divided patch $h \times w$ and the parameters of the beta distribution α, β. Modifying these parameters enables the control of the mixing ratio, size, and degree, thus helping to mitigate the problem of excessive mixing. While PatchMix serves to enhance the richness of image information, it frequently engenders heightened training complexity for the network. Therefore, we believe that for complex datasets, such as those sensitive to spatial information, it is often necessary to reduce the mixing ratio or degree to ensure that augmentation remains within an appropriate range. In this paper conducts experimental comparisons on different datasets, and the considerations for selecting hyperparameters were mentioned in the section of ablation study. In the following, we primarily investigate the impact of the beta distribution with varying parameter values on the mixture.

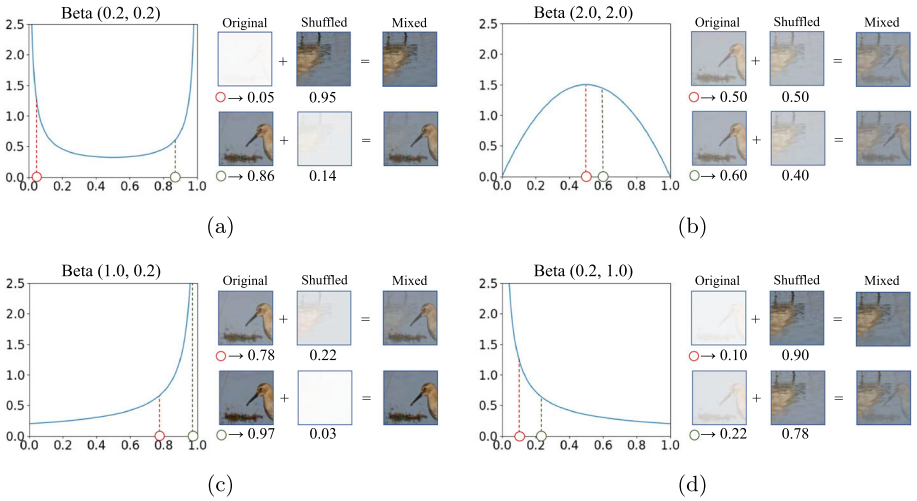

Fig. 4 The mixing effects of PatchMix under different beta distributions. In each subfigure, the left displays the curve shape of the probability density function under different α and β, while the right shows the process of mixing two patches after determining the mixing ratio. **a** $\alpha = 0.2$, $\beta = 0.2$: the mixed patch leans toward either the original patch or the shuffled patch; **b** $\alpha = 2.0$, $\beta = 2.0$: the mixed patch leans toward a fusion of the original patch and the shuffled patch; **c** $\alpha = 1.0$, $\beta = 0.2$: the mixed patch leans toward the shuffled patch, indicating the addition of significant perturbation; **d** $\alpha = 0.2$, $\beta = 1.0$: the mixed patch leans toward the original patch, signifying minimal perturbation

To ensure a wide-ranging set of training samples, it is crucial to strike a balance between blending information from different patches and preserving the integrity of certain patches. The beta distribution, a continuous probability distribution ranging from 0 to 1, serves as a useful tool for this purpose. As illustrated in Fig. 4, the probability density function of the beta distribution is governed by parameters α, β, which control the mixing proportion between different patches. To generate differing training samples, an effective strategy is to increase the perturbation of the original patch. In contrast, if preserving the overall structure is of paramount importance, assigning a larger weight to the original patch may be more appropriate. Consequently, the optimal values for α, β can be chosen based on the specific augmentation task at hand.

3.2 PatchMix-R

In relation to data augmentation, robustness against perturbations is of particular importance [36, 37]. PatchShuffle [29] has been demonstrated to be an effective method for enhancing robustness by swapping the positions of adjacent pixels. Inspired by this idea, we consider applying PatchMix within patches, replacing the original random replacement with a more reasonable mixing method.

3.2.1 Formulations

PatchMix-R, similar to Eqs. (1–2), also carries out block operations initially. However, unlike Eq. (3), it mixes nearby pixels within the patch. To accomplish this, each pixel in the patch is sequentially numbered from 1 to $h \times w$. Let p_j^i represents the j-th pixel in patch P_i.

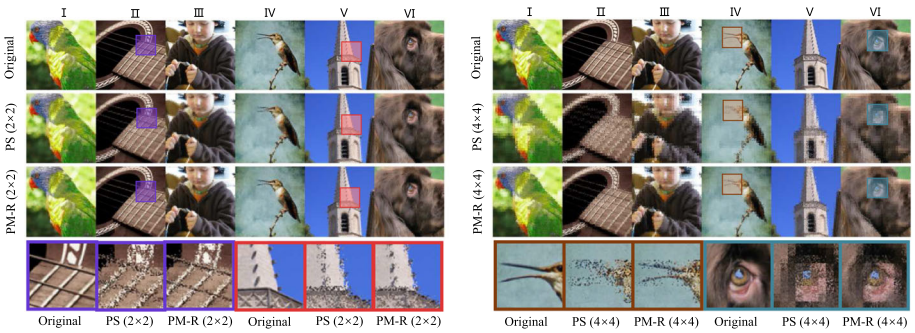

Fig. 5 Examples of PatchShuffle (PS) and PatchMix-R (PM-R). The patch sizes are 2 × 2 pixels (left) and 4 × 4 pixels (right). Image selection was based on overall contrast (I, III), detail (IV, VI) and texture (II, V). Images on the bottom row are zoomed-in regions

PatchMix-R can be expressed as follows,

$$\tilde{p}_j^i = \lambda p_j^i + (1-\lambda) p_{\text{index}[j]}^i, \tag{4}$$

where index is a randomly generated sequence number obtained by shuffling $\left[1, h \times w\right]$, and the mixing ratio λ is also sampled from a beta distribution.

3.2.2 PatchMix-R on images

Assuming the image and patch sizes are 224 × 224 pixels and 4 × 4 pixels, respectively, the original image is divided into 56 × 56 patches. For each patch, PatchMix-R shuffles and mixes 16 pixels according to Eq. (4). Figure 5 illustrates the effects of PatchMix-R and PatchShuffle on different patch sizes. In comparison with PatchShuffle, PatchMix-R enhances the image through mixup, resulting in a more balanced and reasonable pixel transition. In terms of preserving the overall structure, as well as the handling of details and textures, PatchMix-R perturbs the image in a manner that better conforms to the original pixel distribution. PatchMix-R has been proven in the experimental part to effectively improve the robustness.

4 Experiment

4.1 Implementation details

Datasets Several open-source image classification datasets are used in our experiment, including CIFAR-10/100 [38] and Tiny-ImageNet [39]. The CIFAR-10 dataset contains 6000 images with a resolution of 32 × 32, featuring different classes such as airplane, automobile, bird and more. The CIFAR-100 dataset expands the number of classes to 100, with each class having 600 images. The Tiny-ImageNet, another popular dataset, consists of 200 classes, each with 500 training images, 50 validation images and 50 testing images. The resolution of each sample is 64 × 64. Compared to the extensive and challenging dataset ImageNet, Tiny-ImageNet serves as a smaller and more manageable version that is often utilized for benchmarking and evaluating new machine learning methods.

Architectures and settings Four architectures are adopted on above datasets: PreActResNet [40] (PreActResNet-18, PreActResNet-34, PreActResNet-50) and WideResNet [41]

Table 1 Test errors (%) of PatchMix, random flipping and random cropping on CIFAR-10 with PreActResNet-18

Random flipping [1]		✓			✓	✓		✓
Random cropping [1]			✓		✓		✓	✓
PatchMix (ours)				✓		✓	✓	✓
Test errors (%)	10.57	7.55	5.88	9.16	4.83	6.26	4.45	**3.66**

Bold value indicates the optimal results, represented by the smallest test errors (%)

(WideResNet-16-8, WideResNet-28-10), DenseNet [42] (DenseNet-100-BC), MobileNet [43] (MobileNetV2). We follow the overall training protocol in [23]. Differently, this paper trains PreActResNet and MobileNet for 300 epochs, WideResNet and DenseNet for 200 epochs. On CIFAR-10/100, initial data augmentations involve random flipping and random cropping with 4-pixel padding for 32×32 resolution. The training settings include SGD optimizer with a weight decay of 0.0001, momentum of 0.9, and batch size of 100. The initial learning rate is 0.2, decaying by a factor of 0.1 at epochs 100 and 200 for PreActResNet and MobileNet, 120 and 170 for WideResNet and DenseNet. On Tiny-ImageNet, basic augmentations encompass random flipping and random cropping for 64×64 resolution, and we use the similar training ingredients as CIFAR.

Mix-related hyperparameters In PatchMix, patch size h and w are set to half of H and W. Empirically, the patch-mixed probability p is set to 0.5 to preserve the original information of patches. For the beta distribution, both α and β are set to 0.2. In PatchMix-R, We primarily adopt the settings from PatchShuffle [29], where only 5% of the training data are randomly augmented, and the patch size is set to 4×4 pixels. Similar to PatchMix, the patch-mixed probability p is set to 0.5 and the mixing values are randomly selected from the beta distribution with $\alpha = 0.2$, $\beta = 0.2$.

4.2 Experimental results and analysis

4.2.1 Comparison with image-level methods

Random flipping (RF) and random cropping (RC). A comparison of our method with RF and RC is presented in Table 1. When applied individually, RC outperforms the other two methods with an error rate of 5.88%. Combining PatchMix with RF and RC reduces error rates by 2.72% and 1.43%, respectively. Therefore, PatchMix can serve as a supplement to existing regularization techniques. The ensemble of these three methods yields an error rate of 3.66%, 6.91% improvement over the baseline without any augmentation. In subsequent experiments, we adopt RF and RC as initial data transformations.

Automatic augmentation techniques. TrivialAugment [12], an automatic augmentation technique, integrates multiple data augmentation methods such as rotation, scaling, color adjustment, and noise addition. Table 2 compares our method with TrivialAugment. When applied alone, PatchMix outperforms TrivialAugment, reducing the error by 0.31%. Evidently, PatchMix is more effective in enhancing accuracy compared to basic augmentation techniques and their ensembles in TrivialAugment. Combining PatchMix and TrivialAugment achieves a 3.35% error rate, 1.48% improvement over the baseline.

Table 2 Test errors (%) of PatchMix and TrivialAugment on CIFAR-10 with PreActResNet-18

TrivialAugment [12]		✓		✓
PatchMix (ours)			✓	✓
Test errors (%)	4.83	3.94	3.66	**3.35**

Bold value indicates the optimal results, represented by the smallest test errors (%)

4.2.2 Comparison with patch-level methods

Traditional patch-level methods [13–15] primarily focus on simple noise removal or addition to patches. In order to comprehensively compare the classification performance and robustness, this paper further introduce PatchShuffle and PatchMix-R. PatchMix+PatchMix-R refers to performing the PatchMix on the image first, followed by applying PatchMix-R. As detailed in Table 3, PatchMix+PatchMix-R demonstrates superior performance across all datasets and models, reducing the test error by 0.91–3.05% compared to the vanilla method. Additionally, the individual performance of PatchMix surpasses of other traditional patch-level methods.

4.2.3 Ensemble with mixup-based methods

Mixup-based methods combine or merge features from multiple images. This paper conducts experimental research on the ensemble of PatchMix and various multi-image methods. As shown in Table 4, even when applied to a single image, PatchMix ($\alpha = 0.2$, $\beta = 0.2$) surpasses many multi-image augmentation methods, such as Mixup, Manifold, and CutMix. Additionally, when applying PatchMix ($\alpha = 1.0$, $\beta = 0.2$) as the initial augmentation step before mixup-based methods, the classification performance of the trained models improved by 0.55–3.21%. Taking reference from Fig. 4, this paper infers that PatchMix, when used alone or combined with simple data augmentation techniques, benefits from the richness and diversity of patch mixing (refers to Fig. 4a). Differently, when ensembled with complex methods, aligning the patches closer to the original image and applying moderate perturbations (refers to Fig. 4c), enables the synergistic advantages of different augmentation methods to be fully realized. Therefore, PatchMix can ensemble with various data augmentation methods by simply controlling the beta distribution, thereby enhancing the performance of the model.

4.2.4 Performance on large-scale images

This paper also evaluates the performance of our method on large-scale images. In this experiment, we primarily follow the training protocol outlined in [44] and select the VGG-19 [10] and WideResNet-101-2 models, pre-trained on the ImageNet dataset. Three distinct categories of large-scale datasets are chosen: Caltech-101 [45], which are relatively similar to the source dataset; Describable Texture [46], which differs significantly from the source dataset; and the commonly encountered fine-grained datasets, Stanford Cars [47] and Oxford 102 Flower [48]. Given that the images in the above datasets have large and varying sizes, this paper performs the necessary processing for image resizing. During the training, we also record the classification errors at different epochs. The experimental results are presented in Table 5. Across different types of classification datasets, PatchMix performs best on large-scale datasets and helps the pre-trained model quickly adapt to the new data domain.

Table 3 Test errors (%) of the patch-level methods on various models and datasets that composed of clean data.

Method	CIFAR-10							CIFAR-100							Tiny-ImageNet			
	R-18	R-34	R-50	W16-8	W28-10	DenseNet	MobileNet	R-18	R-34	R-50	W16-8	W28-10	DenseNet	MobileNet	R-18	W16-8	DenseNet	MobileNet
Vanilla	4.83	4.62	4.47	4.42	3.76	4.48	4.57	23.84	23.39	22.78	21.42	19.48	22.39	22.68	42.75	41.89	42.08	42.53
Cutout [13]	3.76	3.56	3.41	3.42	3.11	3.51	3.65	22.16	21.96	21.73	20.87	17.93	21.18	21.45	41.55	40.70	41.46	41.61
Random erasing [15]	3.92	3.63	3.39	3.41	3.01	3.45	3.53	21.86	21.54	21.16	20.07	16.94	20.72	20.91	40.76	39.94	40.72	40.81
Patch Gaussian [14]	4.07	3.88	3.71	3.56	3.13	3.79	3.77	22.15	21.76	21.42	20.61	17.23	20.98	21.31	40.93	40.35	40.85	40.99
PatchShuffle [29]	4.60	4.58	4.51	4.29	3.49	4.49	4.63	23.03	23.11	23.14	21.54	18.77	22.45	22.79	45.61	43.81	44.92	45.07
PM (Ours)	3.66	3.45	3.36	3.28	2.90	3.22	3.41	21.43	21.05	19.97	19.22	16.68	19.83	20.77	40.11	39.23	39.79	40.01
PM-R (ours)	4.72	4.35	4.43	4.51	3.97	4.41	4.59	23.30	23.44	22.61	22.36	18.45	22.09	22.72	42.61	42.08	42.12	42.51
PM+PM-R (ours)	**3.37**	**3.31**	**3.25**	**3.13**	**2.85**	**3.19**	**3.29**	**20.98**	**20.85**	**19.84**	**18.96**	**16.43**	**19.77**	**19.91**	**39.70**	**39.15**	**39.52**	**39.63**

Bold values indicate the optimal results, represented by the smallest test errors (%)
R-18:PreActResNet-18 [40], W16-8: WideResNet-16-8 [41]
PM PatchMix, *PM-R* PatchMix-R

Table 4 Test errors (%) of the mixup-based methods combined with the PatchMix in Tiny-ImageNet classification with PreActResNet-18

	Vanilla	Mixup-based method					
		Mixup [19]	Manifold [20]	CutMix [22]	SaliencyMix [24]	PuzzleMix [23]	AutoMix [25]
No PatchMix	42.75	43.17	41.82	41.35	40.89	36.65	34.12
+ PatchMix ($\alpha = 0.2$, $\beta = 0.2$)	**40.11**	40.57	40.05	39.87	39.57	35.96	33.84
+ PatchMix ($\alpha = 1.0$, $\beta = 0.2$)	40.48	**39.96**	**39.54**	**38.91**	**38.76**	**35.47**	**33.57**

Bold values indicate the optimal results, represented by the smallest test errors (%)

Table 5 Test errors (%) of PatchMix and other augmentation methods on large-scale datasets after different epochs of fine-tuning

Dataset	Input Size	Model	Epoch	Test errors (%)			
				Vanilla	Cutout [13]	Random erasing [15]	PatchMix
Caltech-101	224 × 224	VGG-19	5/10	9.45/7.32	8.73/6.85	7.96/6.32	**6.83/5.82**
		WideResNet-101-2	5/10	3.56/2.91	3.25/2.74	3.08/2.59	**2.45/2.14**
Describable Textures	448 × 448	VGG-19	25/50	35.76/29.83	33.43/29.13	32.71/28.53	**30.15/27.62**
		WideResNet-101-2	25/50	27.32/24.59	26.12/23.77	25.73/23.24	**24.35/22.84**
Stanford Cars	448 × 448	VGG-19	12/25	16.36/12.65	15.95/12.43	15.33/11.97	**14.62/11.35**
		WideResNet-101-2	12/25	8.72/6.73	8.41/6.42	7.86/6.09	**7.12/5.49**
Oxford 102 Flower	448 × 448	VGG-19	12/25	8.56/4.83	8.35/4.75	7.93/4.66	**6.68/4.33**
		WideResNet-101-2	25/50	1.95/0.73	1.82/0.73	1.69/0.71	**1.47/0.65**

Bold values indicate the optimal results, represented by the smallest test errors (%)

Table 6 Test errors (%) of KNN classifier in the self-supervised stage and test errors (%) of the linear classifier in the downstream classification task when using the pre-trained model.

	CIFAR-10		CIFAR-100	
	KNN	Linear	KNN	Linear
Baseline	12.64	8.15	48.22	34.56
Baseline + PatchMix	**11.64**	**7.01**	**45.18**	**31.02**

Bold values indicate the optimal results, represented by the smallest test errors (%)
Baseline: default data augmentation methods in the Simsiam [28], including random cropping, random flipping, color jitter, grayscale, etc.

4.2.5 Performance on self-supervised learning and transfer learning scenarios

Self-supervised learning aims to learn useful representations from scalable unlabeled data without relying on human annotation. The Siamese network [28, 49–51] is one promising approach among many self-supervised learning approaches and outperforms supervised counterparts across numerous visual benchmarks. The Simsiam [28], as a typical Siamese network, aims to learn similar feature representations for different views of the same image, enabling effective transfer to various downstream tasks. Data augmentation techniques in Simsiam include random cropping, flipping, color jittering, etc., which are used as inputs to the encoder. In this paper, we introduce PatchMix into the data augmentation process of Simsiam, to further investigate its advantages in self-supervised learning and transfer scenarios.

In the self-supervised learning phase, the Simsiam network is trained on the CIFAR dataset using PreActResNet-18. We use the KNN [52] (k=1) classifier as a monitor of the training progress. By comparing the predictions of the KNN classifier with ground-truth labels, the performance of the model can be evaluated. In the downstream classification task, this paper employs the pre-trained model with frozen weights to train a supervised linear classifier on the corresponding CIFAR dataset. The classification performance is quantified by the accuracy of the linear classifier. Importantly, it should be noted that data augmentation is exclusively employed during the self-supervised learning phase. The specific results are shown in Table 6. Evidently, PatchMix empowers the self-supervised model with better representation capabilities, resulting in superior performance when transferred to the downstream tasks.

4.3 Robustness against corruption

4.3.1 Performance on the CIFAR-C dataset

CIFAR-10-C and CIFAR-100-C [53] are two prevalent datasets for evaluating the robustness of computer vision models. Both datasets consist of the original CIFAR test images that poisoned by 15 distinct distortion types. Each distortion has five intensity levels when injected into images. Comparing with different patch-level methods on various datasets and models, this paper presents the experimental results in Table 7. Our proposed PatchMix-R outperforms the previous approaches in terms of robustness against unseen corruptions, reducing the test error by 8.79–13.92%. As for PatchMix, it exhibits better robustness than other methods. Notably, the robustness of PatchMix-R decreases slightly by 0.19–1.59% after incorporating PatchMix. One plausible interpretation is that the amalgamation of information from varying

Table 7 Test errors (%) of patch-level methods on various models and datasets that composed of corrupted data.

Method	CIFAR-10-C							CIFAR-100-C						
	R-18	R-34	R-50	W16-8	W28-10	DenseNet	MobileNet	R-18	R-34	R-50	W16-8	W28-10	DenseNet	MobileNet
Vanilla	25.38	25.24	25.32	25.35	25.27	25.26	25.31	54.91	53.41	53.12	52.81	52.84	52.89	53.34
Cutout [13]	25.45	25.35	25.33	25.46	25.43	25.39	25.38	54.07	53.09	52.56	52.35	52.07	52.51	53.05
Random erasing [15]	25.07	25.32	25.15	25.21	24.81	25.11	25.17	54.20	53.15	52.48	50.75	47.94	51.61	52.17
Patch Gaussian [14]	21.91	21.46	20.94	21.03	20.39	23.89	24.16	50.44	49.64	49.05	48.27	46.19	49.47	50.75
PatchShuffle [29]	19.30	19.15	18.99	19.21	19.01	18.97	19.03	49.45	47.58	46.46	44.59	42.67	45.41	45.21
PM (Ours)	24.48	24.17	24.13	23.87	23.28	23.94	24.11	51.77	51.33	50.98	49.56	47.68	45.41	50.51
PM-R (Ours)	**16.59**	**16.43**	**16.41**	**16.25**	**15.85**	**16.19**	**16.27**	**42.91**	**41.56**	**41.12**	**39.25**	**38.92**	**40.33**	**40.79**
PM+PM-R (Ours)	17.26	17.25	17.14	16.79	16.04	16.77	17.01	44.50	42.15	41.54	40.23	39.45	41.09	41.23

Bold values indicate the optimal results, represented by the smallest test errors (%)
R-18: PreActResNet-18 [40], W16-8: WideResNet-16-8 [41]
PM PatchMix, *PM-R* PatchMix-R

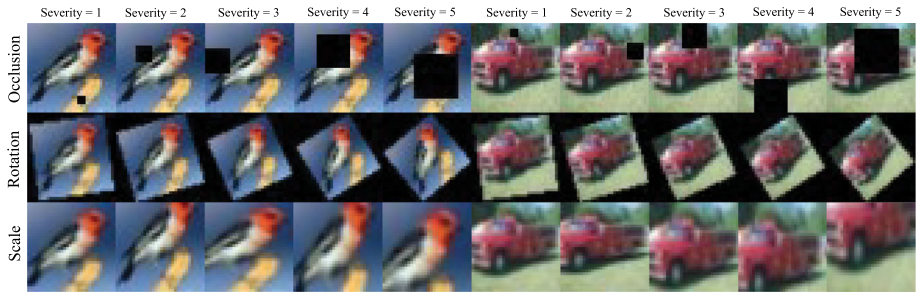

Fig. 6 The CIFAR-ORS dataset consists of 3 types of algorithmically generated corruptions from occlusions, rotations and scale variations. Each type of corruption has five levels of severity, resulting in 15 distinct corruptions

patches by PatchMix partially interferes with the intended perturbations of PatchMix-R. Therefore, how to effectively integrate these two facets will be an intriguing research topic in future work. Meanwhile, in order to demonstrate intuitively the advantages of our approach in adversarial perturbation, this paper also presents the errors of different corruptions and methods on CIFAR-10-C. In Table 8, our method exhibits the best or second-best performance in adversarial perturbation of various types. Specifically, PatchMix-R performs exceptionally best in noise, weather, and digital perturbations.

4.3.2 Performance on proposed CIFAR-ORS dataset

Specifically for the common challenges such as occlusions, rotations, and scale variations, this paper proposes a new perturbation dataset called CIFAR-ORS. Following the design methodology in [53], we incorporate the above three types of perturbations into the test sets of CIFAR-10 and CIFAR-100 through an algorithmic approach. Each perturbation is also represented at five different severity levels, as shown in Fig. 6. After applying the 15 transformations to each image, this paper establishes the CIFAR-ORS dataset, including CIFAR-10-ORS and CIFAR-100-ORS.

The performance of different patch-level methods on CIFAR-ORS is compared, which are presented in Table 9. PatchMix achieves the optimal or near-optimal performance when dealing with patch-level perturbations, such as occlusion or scale. In addition, PatchMix-R excels in scenarios involving rotation. Based on previous experiments, we infer that PatchMix-R focuses more on enhancing the positional relationships between adjacent pixels which can effectively alleviate interference in terms of details. On the other hand, PatchMix enhances interactions between patches, making it notably efficacious for patch-level perturbations.

4.4 Class activation map (CAM) analysis

Class activation map (CAM) [54] identifies the regions in an input image where the model concentrates its attention to recognize an object. In the experiments, this paper computes CAMs for a vanilla WideResNet-28-10 model equipped with various patch-level and mixup-based data augmentation methods on the small-scale dataset CIFAR-10. Figure 7 demonstrates that most existing state-of-the-art (SOTA) techniques, such as Cutout and PuzzleMix, often concentrate on specific representative parts of the content, such as the head of a bird, the wheels of a car, or the legs of a horse. The proposed PatchMix effectively directs the model attention

Table 8 Clean error (%), mCE (%), and corruption error (%) of different corruptions and methods on CIFAR-10-C.

Method	Error	mCE	Noise			Blur				Weather				Digital			
			Gauss	Shot	Impulse	Defocus	Glass	Motion	Zoom	Snow	Frost	Fog	Bright	Contrast	Elastic	Pixel	JPEG
Vanilla	4.83	25.38	54.57	40.98	47.60	16.28	43.80	19.70	20.71	17.00	21.78	11.00	6.07	23.10	14.79	23.51	19.74
Cutout [13]	3.76	25.45	60.38	48.68	49.21	16.44	41.06	18.65	22.79	14.64	22.14	10.10	5.42	19.35	13.04	20.51	19.26
Random erasing [15]	3.92	25.07	57.10	44.03	50.59	17.22	34.05	22.76	22.04	14.27	18.20	11.08	5.55	19.67	13.73	25.65	20.14
Patch Gaussian [14]	4.07	21.91	**29.09**	30.93	39.12	18.39	37.35	23.40	23.33	15.59	19.09	11.51	5.60	19.69	14.54	21.89	19.06
PatchShuffle [29]	4.60	19.30	35.34	28.02	43.54	15.35	15.66	19.04	19.74	12.39	12.85	11.46	6.42	23.62	13.64	14.81	**17.67**
PM (Ours)	3.66	24.48	59.96	49.31	53.37	15.04	36.08	**17.41**	**19.03**	12.95	20.14	9.30	5.22	14.36	12.82	21.95	20.26
PM-R (Ours)	4.72	**16.59**	32.67	**26.50**	**33.76**	**14.62**	11.94	17.70	19.75	**9.45**	**9.07**	9.42	**5.00**	15.02	**11.57**	**14.55**	17.85
PM+PM-R (Ours)	**3.37**	17.26	36.67	29.87	35.29	14.68	**11.75**	17.94	19.94	9.48	9.53	**9.03**	5.06	**13.72**	12.07	14.58	19.35

Bold values indicate the optimal results, represented by the smallest test errors (%)
The mCE value is the mean corruption error of the corruptions in noise, blur, weather, and digital columns. Models are trained only on clean CIFAR-10 with PreActResNet-18
PM PatchMix, *PM-R* PatchMix-R

Table 9 Test errors (%) of patch-level methods on CIFAR-ORS dataset and various models.

	CIFAR-10-ORS						CIFAR-100-ORS					
	Occlusion		Rotation		Scale		Occlusion		Rotation		Scale	
	R-18	W16-8	R-18	W16-8	R-18	W16-8	R-18	W16-8	R-18	W16-8	R-18	W16-8
Vanilla	8.81	7.88	52.13	48.10	18.25	15.60	31.53	27.98	73.37	66.30	45.96	41.46
Cutout [13]	**4.53***	**3.74***	51.62	47.86	18.28	16.01	**23.53***	**19.53***	72.55	65.58	45.44	41.61
Random erasing [15]	5.88	4.54	51.44	47.43	17.52	15.21	25.59	20.14	72.06	65.17	44.19	39.94
Patch Gaussian [14]	6.13	4.68	54.08	48.91	17.66	15.83	25.76	20.76	74.85	67.61	44.70	39.90
PatchShuffle [29]	7.37	5.12	48.49	45.95	17.13	14.91	27.45	22.99	71.16	64.73	42.74	38.63
PM (Ours)	4.81	4.02	49.87	46.12	**15.78**	**12.89**	23.91	19.69	71.53	64.82	**40.94**	**36.30**
PM-R (Ours)	6.19	4.72	**45.82**	**43.32**	16.05	13.58	25.83	21.01	**70.07**	**62.62**	41.73	37.31
PM+PM-R (Ours)	4.98	4.15	46.32	43.84	15.81	13.01	24.23	19.96	70.20	63.12	41.05	36.55

Bold values indicate the optimal results, represented by the smallest test errors (%)
*Cutout itself achieves data augmentation through occlusions, resulting in optimal performance
PM PatchMix, *PM-R* PatchMix-R

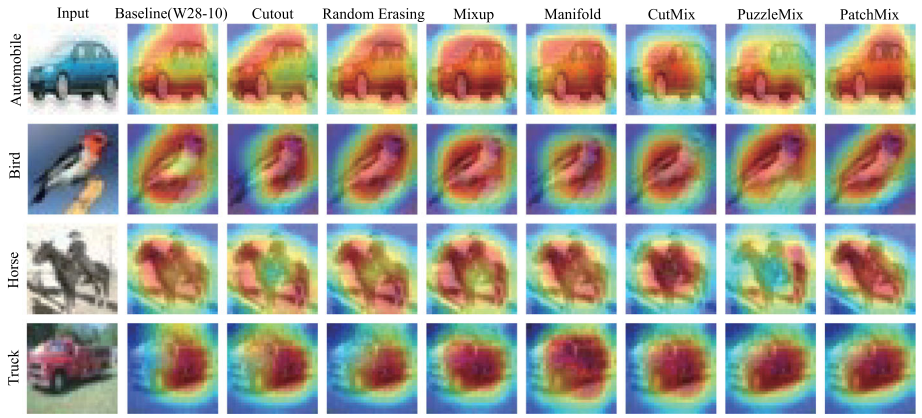

Fig. 7 CAM visualizations on images from CIFAR-10. The proposed data augmentation method guides the model to precisely focus on target object

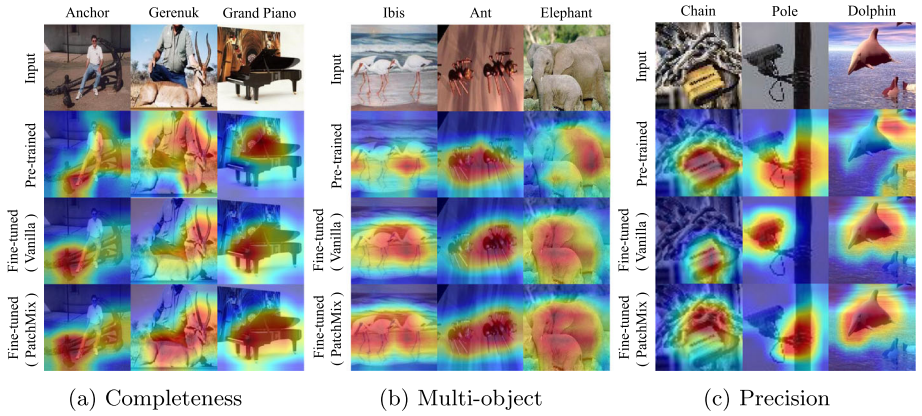

Fig. 8 CAM visualizations on large-scale images. The proposed data augmentation method has multiple advantages in assisting model fine-tuning. Pre-trained (the second row): CAMs from the model pre-trained on the ImageNet. Fine-tuned (the third and forth rows): CAMs from the pre-trained model after fine-tuning

toward the target object with higher precision compared to other methods, which indicates that PatchMix enable the network to learn comprehensive information of the classes, rather than merely memorizing key features.

Images from CIFAR-10 typically contain only one object. In order to explore the advantages of PatchMix on large-scale datasets that may contain multiple objects, this paper also visualizes the CAMs computed from the fine-tuned model from Table 5. We primarily utilize the Caltech-101 dataset and the pre-trained WideResNet-101-2 model. Figure 8 illustrates the similar effect when searching for a specific object in a scene with multiple objects. In Fig 8a, PatchMix captures comprehensive features of the target object, including the head of the anchor, the bodies of the deer and the grand piano. In Fig 8b, PatchMix effectively recognizes multiple target categories, e.g., scattered ibises and ants, overlapping elephants. In Fig 8c, PatchMix enables the model to accurately identify the target chain and pole. Also, the most representative feature regions of target are focused on, e.g., the significant areas in recognizing the dolphin.

Table 10 Test errors (%) with different patch size on CIFAR-10/100 and Tiny-ImageNet

Patch size	CIFAR-10 (32 × 32)	CIFAR-100 (32 × 32)	Tiny-ImageNet (64 × 64)
64 × 64	–	–	42.75
32 × 32	4.83	23.84	**40.11**
16 × 16	**3.66**	**21.43**	41.06
8 × 8	3.78	22.18	41.58
4 × 4	4.05	22.73	–

Bold values indicate the optimal results, represented by the smallest test errors (%)

4.5 Ablation study

When implementing PatchMix on CNN training, the evaluation of hyperparameters becomes crucial. To demonstrate the impact of these hyperparameters on the model performance, experiments are performed on the CIFAR-10/100 and Tiny-ImageNet datasets using the PreActResNet-18 network under varying hyperparameter settings.

4.5.1 The effect of patch size

In this section, we verify the effect of patch size that determines the range of mixing. Referred to [29, 55] for image chunking, non-overlapping square-shaped patches are adopted, as opposed to irregular shapes or overlapping sampling. This approach is considered the simplest yet most effective way to validate the feasibility of PatchMix. As illustrated in Table 10, PatchMix exhibits optimal performance when the patch height (h) and width (w) are configured to half of the image height (H) and width (W). However, when h and w are reduced to 1/4 and 1/8 of H and W, the test error gradually increases and the classification performance of PatchMix weakens. The ablation results underscore the importance of patch size.

Analogous to the jigsaw puzzle, if the image is subdivided into a greater number of smaller fragments, it will become increasingly challenging to establish correspondences, which leads to the loss of vital information. Consequently, PatchMix adopts a straightforward strategy of setting the patch size to half of the image size.

4.5.2 The effect of patch-mixed probability

The patch-mixed probability p determines the mixing ratio of internal information within an image. The higher of p, the richer and more complex the information represented in the augmented image. In this section, we further investigate the impact of p on the model's performance, as presented in Table 11. It can be observed that the performance improves in the presence of patch mixing and the performance is generally optimal within the range of 0.4–0.7 for p. When the value of p is too low, it is posited that the advantages of PatchMix may not be fully realized. Conversely, if all patches undergo mixing operations, it may lead to excessively high complexity in the combinations. Therefore, in our experiments, p is chosen to be 0.5.

Table 11 Test errors (%) with different patch-mixed probability on CIFAR-10/100 and Tiny-ImageNet

p	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
CIFAR-10	4.83	4.33	4.21	4.05	3.74	**3.66**	3.72	3.68	3.79	3.91	4.12
CIFAR-100	23.84	23.51	23.15	22.84	21.47	**21.43**	21.45	21.53	21.51	21.74	22.03
Tiny-ImageNet	42.75	42.11	41.89	41.05	40.39	40.11	**40.08**	40.24	40.51	40.89	41.61

Bold values indicate the optimal results, represented by the smallest test errors (%)

4.5.3 The effect of beta distribution

In Mixup [19], both α and β in beta distribution are set to 0.2, yielding optimal experimental outcomes. However, the specific reasons behind this choice remain scantily explained. Hence, this section delves into this aspect in the context of PatchMix.

This study initially compares four typical beta distributions with distinct shapes shown in Fig. 4. In each distribution, three pairs of values are taken to represent different levels of intensity. From Table 12, the best outcome is achieved when setting α and β to the same value within the range of 0–1, which is considered to be the optimal mixing range for the image classification. Simply put, the beta distribution in Fig. 4c tends to retain the features of original patches, providing stability, while the beta distribution in Fig. 4d increases regional diversity. The aim of PatchMix is to balance these tendencies; hence, the parameters range as demonstrated in Fig. 4a.

To further probe this particular distribution, different α and β within the range of 0–1 are investigated. From Table 13, it can be concluded that the classification performance does not differ significantly under this distribution and the optimal results are obtained with both α and β set to around 0.2. Therefore, the parameters of the beta distribution in our experiment are uniformly chosen as 0.2, excluding other complex values.

4.6 Discussion

Extensions and variations PatchMix offers several avenues for further exploration and research. It is potential to investigate the impact of using patches with different or irregular sizes, as it may lead to more comprehensive representations. Similar to the concept of Manifold Mixup [20], exploring the integration of PatchMix into intermediate layers of the model would be valuable. Furthermore, previous studies have demonstrated the effectiveness of replacing patches from pairs of images in improving network performance in visual tasks [22, 56]. Building upon this foundation, the idea of mixing patches could potentially be introduced to further enhance the feature extraction capabilities.

In weakly supervised object detection and segmentation tasks, many CAM-based pseudo-label generation methods often suffer from the problem of focusing only on partial discriminative foreground regions [57, 58]. PatchMix provides a promising solution by accurately attending to the holistic characteristics of the classes and holds great potential for application. In addition to CNNs, current research has also begun exploring image augmentation techniques suitable for the vision transformer (ViT) architecture [59]. The investigation of PatchMix, which based on rich relative positional relationships, is worth considering in terms of its potential impact on encoding positional information.

Computational overhead The implementation of PatchMix itself is not inherently complex and is not constrained by the dataset or network architecture. This section focuses on analyzing the computational overhead of PatchMix to evaluate its scalability. The experiments are conducted on a server equipped with an Intel Xeon Silver 4216 CPU running at 2.10 GHz. Each set of control experiments is performed exclusively on a same NVIDIA Tesla T4 graphics card. We statistically measure and compare the average epoch durations of various methods during training, and the findings are showcased in Table 14. PatchMix exhibits remarkable efficiency compared to other data augmentation methods, with a mere 0.54-s computational overhead per epoch. Additionally, the integration of PatchMix with other methods imposes minimal overhead, requiring a mere 2% additional time investment. Consequently, within the

Table 12 Test errors (%) with four typical types of beta distributions controlled by different α and β on CIFAR-10/100 and Tiny-ImageNet.

(α, β)	Type 1				Type 2				Type 3				Type 4			
	(0.2, 0.2)	(0.5, 0.5)	(0.8, 0.8)	Average	(2.0, 2.0)	(5.0, 0.5)	(8.0, 8.0)	Average	(1.0, 0.2)	(1.0, 0.5)	(1.0, 0.8)	Average	(0.2, 1.0)	(0.5, 1.0)	(0.8, 1.0)	Average
CIFAR-10	3.66	3.78	3.87	**3.77**	4.03	4.33	4.71	4.36	4.57	4.32	4.25	4.38	3.95	4.04	4.15	4.05
CIFAR-100	21.43	21.52	21.64	**21.53**	22.25	22.89	23.42	22.85	23.26	22.71	22.51	22.83	21.91	22.13	22.44	22.16
Tiny-ImageNet	40.11	40.28	40.61	**40.33**	41.67	42.14	42.59	42.13	42.41	42.15	41.62	42.06	41.13	41.22	41.35	41.23

Bold values indicate the optimal results, represented by the smallest test errors (%)
Type 1 to type 4 correspond to Fig. 4a–d, respectively

Table 13 Test errors (%) with different α and β of beta distribution on CIFAR-10/100 and Tiny-ImageNet.

$\alpha = \beta$	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
CIFAR-10	3.68	**3.66**	3.71	3.68	3.78	3.71	3.79	3.87	3.82
CIFAR-100	21.50	21.43	**21.36**	21.47	21.52	21.45	21.53	21.64	21.68
Tiny-ImageNet	40.15	**40.11**	40.34	40.26	40.28	40.36	40.33	40.61	40.53

Bold values indicate the optimal results, represented by the smallest test errors (%)
The values of α and β are equal and range between 0 and 1

Table 14 Epoch duration (s) with the incorporation of PatchMix for different methods in CIFAR10 with PreActResNet18

Method	Vanilla	Mixup [19]	Manifold [20]	CutMix [22]	SaliencyMix [24]	PuzzleMix [23]	AutoMix [25]
No PatchMix	41.57	42.12	42.75	44.56	46.51	102.38	189.86
+ PatchMix	42.11 (+0.54)	43.07 (+0.95)	43.21 (+0.46)	45.49 (+0.93)	47.13 (+0.62)	103.83 (+1.45)	191.64 (+1.78)

realm of expansive datasets and complex networks, PatchMix emerges as a versatile plug-and-play data augmentation solution, offering the advantage of controllable computational overhead.

5 Conclusion

In this paper, we propose two data augmentation techniques, PatchMix and PatchMix-R, with the goal of improving the generalization and robustness of classification models. PatchMix introduces mixup into traditional patch-level augmentation, generating a large amount of new training data through a random but controllable processing approach. This method can seamlessly integrate into other data augmentation methods of varying complexities under different beta distribution. CAMs visualize the advantages of PatchMix when applied to both small-scale and large-scale images. PatchMix-R is an extension of PatchMix that mixes pixels within each patch instead of across different patches. This modification significantly enhances the ability of neural networks to resist adversarial perturbations. Extensive experiments on the widely used classification datasets and networks are conducted to verify the feasibility and effectiveness of our methods.

Author Contributions Yichao Hong was involved in the conceptualization, formal analysis, investigation, and writing—original draft. Yuanyuan Chen contributed to the methodology, validation, resources, writing—review and editing, supervision, and funding acquisition.

Funding This work was supported by National Natural Science Foundation of China (NSFC) (No. 62376173).

Data availability Data sharing is not applicable to this article as no datasets were generated during the current study.

Declarations

Conflict of interest The authors declare that they have no conflict of interest to this work.

References

1. Krizhevsky A, Sutskever I, Hinton GE (2017) Imagenet classification with deep convolutional neural networks. Commun ACM 60(6):84–90
2. Yang Z, Sinnott RO, Bailey J, Ke Q (2023) A survey of automated data augmentation algorithms for deep learning-based image classification tasks. Knowl Inf Syst 65(7):2805–2861
3. Ammar S, Bouwmans T, Zaghden N, Neji M (2020) Towards an effective approach for face recognition with DCGANs data augmentation. In: Advances in visual computing: 15th International symposium, ISVC 2020, San Diego, USA, October 5–7, 2020, proceedings, part I 15, pp 463–475
4. Bae G, La Gorce M, Baltrušaitis T, Hewitt C, Chen D, Valentin J, Cipolla R, Shen J (2023) Digiface-1m: 1 million digital face images for face recognition. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision, pp 3526–3535
5. Sakkos D, Shum HP, Ho ES (2019) Illumination-based data augmentation for robust background subtraction. In: 2019 13th international conference on software, knowledge, information management and applications (SKIMA). Island of Ulkulhas, Maldives, pp 1–8
6. Cauli N, Reforgiato Recupero D (2022) Survey on videos data augmentation for deep learning models. Future Internet 14(3):93
7. Silva L, Barbosa L (2023) Matching news articles and wikipedia tables for news augmentation. Knowl Inf Syst 65(4):1713–1734
8. Iosifidis V, Ntoutsi E (2020) Sentiment analysis on big sparse data streams with limited labels. Knowl Inf Syst 62(4):1393–1432
9. Zhang C, Bengio S, Hardt M, Recht B, Vinyals O (2021) Understanding deep learning (still) requires rethinking generalization. Commun ACM 64(3):107–115
10. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556
11. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
12. Müller SG, Hutter F (2021) Trivialaugment: tuning-free yet state-of-the-art data augmentation. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 774–782
13. DeVries T, Taylor GW (2017) Improved regularization of convolutional neural networks with cutout. arXiv preprint arXiv:1708.04552
14. Lopes RG, Yin D, Poole B, Gilmer J, Cubuk ED (2019) Improving robustness without sacrificing accuracy with patch gaussian augmentation. arXiv preprint arXiv:1906.02611
15. Zhong Z, Zheng L, Kang G, Li S, Yang Y (2020) Random erasing data augmentation. In: Proceedings of the AAAI conference on artificial intelligence, vol 34, pp 13001–13008
16. Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S, Uszkoreit J, Houlsby N (2021) An image is worth 16×16 words: transformers for image recognition at scale. In: International conference on learning representations, online
17. Trockman A, Kolter JZ (2022) Patches are all you need? arXiv preprint arXiv:2201.09792
18. Wei C, Xie L, Ren X, Xia Y, Su C, Liu J, Tian Q, Yuille AL (2019) Iterative reorganization with weak spatial constraints: solving arbitrary jigsaw puzzles for unsupervised representation learning. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 1910–1919
19. Zhang H, Cisse M, Dauphin YN, Lopez-Paz D (2018) Mixup: beyond empirical risk minimization. In: International conference on learning representations. Vancouver, Canada
20. Verma V, Lamb A, Beckham C, Najafi A, Mitliagkas I, Lopez-Paz D, Bengio Y (2019) Manifold mixup: better representations by interpolating hidden states. In: International conference on machine learning, pp 6438–6447
21. Navarro M, Little C, Allen GI, Segarra S (2024) Data augmentation via subgroup mixup for improving fairness. In ICASSP 2024-2024 IEEE international conference on acoustics, speech and signal processing (ICASSP), pp 7350–7354
22. Yun S, Han D, Oh SJ, Chun S, Choe J, Yoo Y (2019) Cutmix: regularization strategy to train strong classifiers with localizable features. In: Proceedings of the IEEE/CVF International conference on computer vision, pp 6023–6032
23. Kim J-H, Choo W, Song HO (2020) Puzzle mix: exploiting saliency and local statistics for optimal mixup. In: International conference on machine learning, pp 5275–5285
24. Uddin AFMS, Monira MS, Shin W, Chung T, Bae S-H (2021) Saliencymix: a saliency guided data augmentation strategy for better regularization. In: International conference on learning representations, online

25. Liu Z, Li S, Wu D, Liu Z, Chen Z, Wu L, Li, SZ (2022) Automix: unveiling the power of mixup for stronger classifiers. In: Computer vision–ECCV 2022: 17th European conference, Tel Aviv, Israel, October 23–27, 2022, proceedings, part XXIV, pp 441–458
26. Schneider N, Goshtasbpour S, Perez-Cruz F (2023) Anchor data augmentation. In: Thirty-seventh Conference on neural information processing systems
27. Yao H, Wang Y, Zhang L, Zou JY, Finn C (2022) C-mixup: improving generalization in regression. Adv Neural Inf Process Syst 35:3361–3376
28. Chen X, He K (2021) Exploring simple siamese representation learning. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 15750–15758
29. Kang G, Dong X, Zheng L, Yang Y (2017) Patchshuffle regularization. arXiv preprint arXiv:1707.07103
30. Shorten C, Khoshgoftaar TM (2019) A survey on image data augmentation for deep learning. J Big Data 6(1):1–48
31. Cubuk ED, Zoph B, Mane D, Vasudevan V, Le QV (2019) Autoaugment: learning augmentation strategies from data. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 113–123
32. Cubuk ED, Zoph B, Shlens J, Le QV (2020) Randaugment: practical automated data augmentation with a reduced search space. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, pp 702–703
33. Kim J, Choo W, Jeong H, Song HO (2021) Co-mixup: saliency guided joint mixup with supermodular diversity. In: International conference on learning representations, Online
34. Venkataramanan S, Kijak E, Amsaleg L, Avrithis Y (2022) Alignmixup: improving representations by interpolating aligned features. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 19174–19183
35. Cheng Y-C, Chen S-Y (2003) Image classification using color, texture and regions. Image Vis Comput 21(9):759–776
36. Lee K, Lee K, Shin J, Lee H (2019) Network randomization: a simple technique for generalization in deep reinforcement learning. arXiv preprint arXiv:1910.05396
37. Eghbal-zadeh H, Zellinger W, Pintor M, Grosse K, Koutini K, Moser BA, Biggio B, Widmer G (2024) Rethinking data augmentation for adversarial robustness. Inf Sci 654:119838
38. Krizhevsky A, Hinton G (2009) Learning multiple layers of features from tiny images. Technical report, University of Toronto
39. Chrabaszcz P, Loshchilov I, Hutter F (2017) A downsampled variant of imagenet as an alternative to the cifar datasets. arXiv preprint arXiv:1707.08819
40. He K, Zhang X, Ren S, Sun J (2016) Identity mappings in deep residual networks. In: Computer vision–ECCV 2016: 14th European conference, Amsterdam, The Netherlands, October 11–14, 2016, proceedings, part IV 14, pp 630–645
41. Zagoruyko S, Komodakis N (2016) Wide residual networks. arXiv preprint arXiv:1605.07146
42. Huang G, Liu Z, Van Der Maaten L, Weinberger KQ (2017) Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4700–4708
43. Sandler M, Howard A, Zhu M, Zhmoginov A, Chen L-C (2018) Mobilenetv2: inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4510–4520
44. Kabir HD, Abdar M, Khosravi A, Jalali SMJ, Atiya AF, Nahavandi S, Srinivasan D (2022) Spinalnet: deep neural network with gradual input. IEEE Trans Artif Intell. https://doi.org/10.1109/TAI.2022.3185179
45. Li F-F, Andreeto M, Ranzato M, Perona P (2022) Caltech 101. CaltechDATA. https://doi.org/10.22002/D1.20086
46. Cimpoi M, Maji S, Kokkinos I, Mohamed S, Vedaldi A (2014) Describing textures in the wild. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3606–3613
47. Krause J, Stark M, Deng J, Fei-Fei L (2013) 3d object representations for fine-grained categorization. In: Proceedings of the IEEE international conference on computer vision workshops, pp 554–561
48. Nilsback M-E, Zisserman A (2008) Automated flower classification over a large number of classes. In: 2008 Sixth Indian conference on computer vision, graphics & image processing, pp 722–729
49. Bromley J, Guyon I, LeCun Y, Säckinger E, Shah R (1993) Signature verification using a "Siamese" time delay neural network. In: Proceedings of the 6th international conference on neural information processing systems. Morgan Kaufmann Publishers Inc., San Francisco, pp 737–744
50. Chen T, Kornblith S, Norouzi M, Hinton G (2020) A simple framework for contrastive learning of visual representations. In: International conference on machine learning, pp 1597–1607
51. He K, Fan H, Wu Y, Xie S, Girshick R (2020) Momentum contrast for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 9729–9738

52. Cover T, Hart P (1967) Nearest neighbor pattern classification. IEEE Trans Inf Theory 13(1):21–27
53. Hendrycks D, Dietterich T (2019) Benchmarking neural network robustness to common corruptions and perturbations. arXiv preprint arXiv:1903.12261
54. Zhou B, Khosla A, Lapedriza A, Oliva A, Torralba A (2016) Learning deep features for discriminative localization. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2921–2929
55. Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S et al (2020) An image is worth 16×16 words: transformers for image recognition at scale. arXiv preprint arXiv:2010.11929
56. Cascante-Bonilla P, Sekhon A, Qi Y, Ordonez V (2021) Evolving image compositions for feature representation learning. arXiv preprint arXiv:2106.09011
57. Xu J, Xie H, Xu H, Wang Y, Liu S-A, Zhang Y (2022) Boat in the sky: background decoupling and object-aware pooling for weakly supervised semantic segmentation. In: Proceedings of the 30th ACM international conference on multimedia, pp 5783–5792
58. Zhu L, She Q, Chen Q, Meng X, Geng M, Jin L, Zhang Y, Ren Q, Lu Y (2023) Background-aware classification activation map for weakly supervised object localization. IEEE Trans Pattern Anal Mach Intell. https://doi.org/10.1109/TPAMI.2023.3309621
59. Zhu J, Bai H, Wang L (2023) Patch-mix transformer for unsupervised domain adaptation: a game perspective. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 3561–3571

Yichao Hong received his BS degree in Computer Science and Technology from Sichuan University in 2022. He is currently pursuing a MS degree at the College of Computer Science, Sichuan University, China. His research interests include deep learning, image analysis, and data augmentation.

Yuanyuan Chen received her MS in pattern recognition and intelligent systems from Shanghai Jiaotong University in 2007, and then she became a lecturer for teaching and research in the Department of Computer Science at Sichuan University. She received her PhD in machine intelligence from Sichuan University in 2015. Currently, she is an associate professor and master supervisor of Sichuan University. Yuanyuan Chen is a director of Sichuan artificial intelligence society. Her research chiefly focuses on artificial intelligence. She has always been dedicated to the research of the theory and applications of neural networks and medical image analysis. She has published more than 30 journal articles in this field.