Evolving Image Compositions for Feature Representation Learning

Paola Cascante-Bonilla[1]
pc9za@virginia.edu
Arshdeep Sekhon[1]
as5cu@virginia.edu
Yanjun Qi[1]
yq2h@virginia.edu
Vicente Ordonez[2]
vicenteor@rice.edu

[1] University of Virginia
Charlottesville, Virginia, USA

[2] Rice University
Houston, Texas, USA

Abstract

Convolutional neural networks for visual recognition require large amounts of training samples and usually benefit from data augmentation. This paper proposes PatchMix, a data augmentation method that creates new samples by composing patches from pairs of images in a grid-like pattern. These new samples are assigned label scores that are proportional to the number of patches borrowed from each image. We then add a set of additional losses at the patch-level to regularize and to encourage good representations at both the patch and image levels. A ResNet-50 model trained on ImageNet using PatchMix exhibits superior transfer learning capabilities across a wide array of benchmarks. Although PatchMix can rely on random pairings and random grid-like patterns for mixing, we explore evolutionary search as a guiding strategy to jointly discover optimal grid-like patterns and image pairings. For this purpose, we conceive a fitness function that bypasses the need to re-train a model to evaluate each possible choice. In this way, PatchMix outperforms a base model on CIFAR-10 (+1.91), CIFAR-100 (+5.31), Tiny Imagenet (+3.52), and ImageNet (+1.16).

1 Introduction

Deep convolutional neural networks (CNNs) have pushed forward significant progress in many computer vision tasks [19, 20, 30, 31, 39]. These high-capacity models tend to memorise their training data to some extent, therefore, they might lead to suboptimal generalization. Recent work has proposed various data augmentation techniques to alleviate this issue by smoothing out the input space, the output space, or both. Relevant literature falls roughly into two groups: (1) Data augmentation from individual input samples e.g. [27, 29, 46], and (2) Data augmentation that creates new samples by interpolating pairs of samples e.g. [14, 28, 43]. Our paper focuses on the second line of work and proposes to interpolate two samples via patch-level compositions in a grid pattern. Figure 1 shows examples of using data augmentation strategies to create samples.

© 2021. The copyright of this document resides with its authors.
It may be distributed unchanged freely in print or electronic forms.

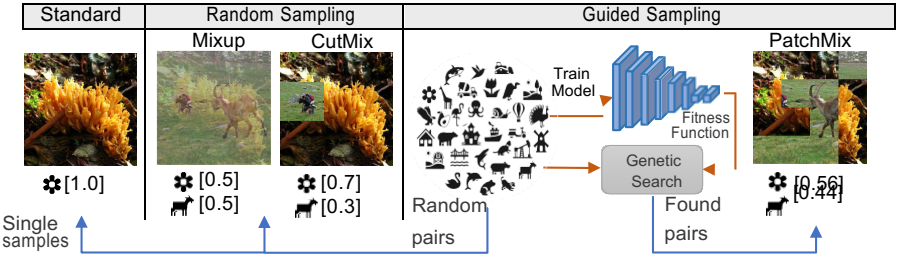

Figure 1: Examples of different data augmentation techniques. Our proposed method exploits patch-based image compositions that allows flexible combinations for data augmentation. PatchMix allows training a model that can be used as a fitness function of an evolutionary search pipeline to find optimum mask configurations and sample pairs. The numbers below each image correspond to the new labels, which are associated with the proportion in which each class has been mixed.

Multiple successful strategies have been proposed for combining pairs of samples to regularize deep learning models such as Mixup [43], Cutmix [42], and Cutout [3, 12, 33]. These studies have shown patch-level perturbations and augmentation strategies to be beneficial for robust feature representation learning in vision. Our method, we call PatchMix, provides two novel designs when compared to the recent works. First, PatchMix uses a grid mask to decompose the image space into a regular grid of patches. Image compositions via a grid mask allows for greater diversity in making the augmented samples (as shown in Figure 1). We also add an auxiliary patch-level supervision on top of the image-level supervision to encourage better and more robust representation learning (refer to Figure 2).

Second, different from the previous methods relying on heuristics to find patches to combine [1], we introduce a guided search strategy via genetic search to find the best set of category pairs to mix and to search for optimal category-dependent grid masks for combining pairs of images. Our search aims to find a set of category pairs (and corresponding masks) for interpolating pairs of image samples to create new samples, hoping to achieve improved model training and generalization. One main challenge when using genetic search for pairwise sample interpolation is the expensive computation cost. This is because evaluating the fitness of each interpolation configuration requires training and evaluating a new model. We, instead, propose a computationally feasible approximation to calculate such a fitness calculation, avoiding the bottleneck of retraining the model for each potential configuration. While genetic search has been used for exploring single sample data augmentation, to the best of our knowledge, our work is the first to explore evolutionary techniques to find the best configurations over the space of interpolations between samples.

Empirically, we validate the effectiveness of PatchMix on the regular image classification task (via CIFAR [21, 22], Tiny Imagenet [23] and ImageNet [11]), on the weakly-supervised localization (WSOL) task (via CUB-200-2011 [37]), the object detection task (via Pascal VOC [13]), on the transfer learning task (via CUB-200-2011, SUN397 [40] and multi-label datasets: Pascal VOC and MS-COCO [25] and NUS [8]), and on the image captioning task (via MS-COCO). Finally, we show consistent robustness results on a model trained with PatchMix when tested against adversarial examples using the Fast Gradient Sign Method (FGSM) [16] white box attack.

2 Related Work

2.1 Pairwise data augmentation

Mixup [43] was the first work that proposed the idea of interpolating two images, and their ground truth labels to augment the training data. Mixup and its variants may suffer from the issue of object local ambiguity, also called *manifold intrusion* [17]. This occurs when the objects inside two image samples are interpolated in such a way that introduce visual confusion, and the true labels contradict the synthetic labels of the generated mixed sample. However, this method has proved effective and general, almost always providing some improvement over a baseline that relies only on single sample data augmentation strategies.

Recent literature has tried a set of mechanisms to deal with the *manifold intrusion* problem [17] by proposing different data interpolation alternatives to Mixup. For example, ManifoldMixup [35] and PatchUp [14] interpolate the hidden states instead of the input space. MetaMixUp [28] proposes to use meta-learning to learn a mixing coefficient that could avoid a high frequency of cases of manifold intrusion. [10] train an extra neural network to anticipate whether a particular combination of two images may suppress information or add manifold intrusions. [7] propose to force a balanced sampling from the training set for selecting the images to be interpolated. CutMix and variants create random binary masks to sample a patch and to apply the corresponding image interpolation [18, 42] only on a subregion. Our paper proposes a new strategy, PatchMix, that allows for sampling multiple patches from an image to interpolate with a second image. Moreover, we select optimal interpolations that are on average more challenging than random interpolations using genetic search.

More recently, other methods [14, 38] such as GridMix [1], have explored patch-like masks to enable input samples interpolations along with their corresponding labels. In our proposed method, we take the last layer of the CNN and divide it as a matrix where each patch also corresponds to the Patch-Mask we use to mix the input samples. In our case, our patch-loss is equally weighted into the whole pipeline and we try to solve the *manifold intrusion* problem during training. This issue happens when synthetic samples generated from interpolating two real samples are assigned a label that contradicts the individual samples. Additional analysis about the manifold intrusion could be found in the Appendix, showing its effect in the decision boundary for a three-way classifier on synthetic data when using different interpolations such as Mixup, CutMix, and our proposed PatchMix.

2.2 Samplewise data augmentation

Another popular group of data augmentation research explores ways to augment samples via individual one-to-one sample transformations. Multiple recent works apply random transformations over an image to augment training data. These transformations range from random cropping, flipping, or rotating an image [34], to random erasing [12, 44], and even more complex random transformations [41]. More recently, researchers have proposed methods to automatically search for data augmentation policies with Reinforcement Learning (RL) or Evolutionary Algorithms [27, 29, 46]. This idea also relates to using RL systems to find state-of-the-art model architectures for image classification [47] using policy gradient optimization methods [32]. This setup is typically expensive due to the need to retrain the model for evaluating all sub-policies or configurations [9, 24]. Differently, PatchMix generates augmented images from pairs of samples and uses genetic search that is guided by a novel fitness criteria based on the difficulty of the chosen configurations.

3 Our Method

PatchMix includes three components: (1) A patch mask that enables a grid-level composition involving all patches from two images. We decompose the image space into a grid of regular-sized patches and design a binary mask on each patch position controlling the composition. (2) A new loss function that enforces patch-level label supervision, in addition to the global image supervision. This loss enhances the regularization provided by our patch-based sample augmentation and provides a useful fitness function for evaluating what candidate patches to use. (3) A search strategy based on genetic search to find the best patch mask for combining pairs of image categories.

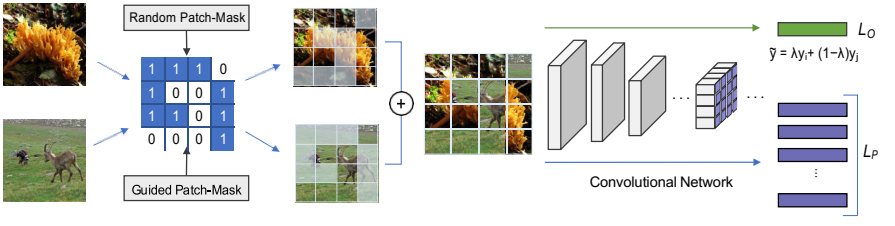

Figure 2: Overview of PatchMix. We first create a binary mask M with PxP number of patches using our proposed PatchMix strategies: Guided PatchMix and its variation Random PatchMix (see Section 3 for details on how to obtain the mask). This mask is then used to interpolate two random images which will create a new image sample. This generated sample is used to train the model, a CNN with the last convolutional layer modified to create PxP patches of equal size. In this way, we are able to output the values corresponding to the input patches corresponding to each image, and the mixed output of the whole new image.

3.1 Binary Patch-Mask M

Let $\mathbf{x} \in \mathbb{R}^{W \times H \times C}$ denote a training image and $\mathbf{y}$ be its corresponding label. The goal of PatchMix is to synthesize additional training samples by interpolating pairs of inputs. For example, for samples $\mathbf{x}_i$, $\mathbf{x}_j$ and their corresponding labels $\mathbf{y}_i$, $\mathbf{y}_j$, we use a patch mask matrix $\mathbf{M} \in \{0,1\}^{W \times H}$ to create a new sample $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}})$:

$$\tilde{\mathbf{x}} = \mathbf{M} \oslash \mathbf{x}_i + (1 - \mathbf{M}) \oslash \mathbf{x}_j, \tag{1}$$

$$\tilde{\mathbf{y}} = \lambda \mathbf{y}_i + (1 - \lambda) \mathbf{y}_j, \tag{2}$$

where $\lambda = \sum_{s=1}^{W} \sum_{t=1}^{H} \mathbf{M}(s,t) / (W \times H)$. More specifically, we divide the mask $\mathbf{M}$ into P^2 patches, resulting in each patch region of size $W/P \times H/P$. We additionally constrain the values $\mathbf{M}(s,t)$ in each patch region to be the same values. In this way, we force the two input images to be interpolated using patches of the same size, with $2^{P \times P}$ possible configurations. This process is illustrated in Figure 2 (on the left side) for a given Patch-Mask with $P = 4$.

3.2 Patch-level Supervision

The key intuition of PatchMix is that image patches provide strong supervisory signals. We, therefore, design mechanisms to exploit such weak supervisions for each of the image patches. We design a Convolutional Neural Network, that takes as input $\tilde{x}$, so that its last convolutional layer produces a set of feature region vectors corresponding to each of

the input P^2 patch regions. In this way, we force the network to output P^2 additional class predictions corresponding to labels for each individual patch. Thus, we adopt the following objectives for each generated sample:

$$L_O = -\sum_{i=1}^{C} \tilde{y_i} \log(\hat{y_i}), \tag{3}$$

$$L_P = -\sum_{n=1}^{P^2} \sum_{i=1}^{C} \tilde{y}_{in} \log(\hat{y}_{in}), \tag{4}$$

$$L_T = (L_O + (L_P/P^2))/2, \tag{5}$$

where C is the number of classes, L_O corresponds to the cross entropy loss over the image-level label vector $\tilde{\mathbf{y}}$, L_P corresponds to a sum of the cross entropy losses for each patch with respect to patch-level (pseudo) labels. Here we assume for the n-th patch, its label $\tilde{\mathbf{y}}_n$ is the same as its image label. L_T is the combined loss using both image-level and patch-level supervision. Figure 2 shows the full process of combining two images given a fixed patch interpolation mask for a pair of input images.

3.3 Evolutionary Search over Interpolations

In order to train a model well under PatchMix, we need to define the patch masks $\mathbf{M}$ for combining a pair of samples. In its most basic form we can train a model by selecting random pairs of images from two arbitrary categories under a mask $\mathbf{M}$ such that the entries for each patch region are sampled from a beta distribution $B(\alpha, \alpha)$. We refer to this basic formulation as **Random PatchMix**. Further, we propose a better strategy to search for the optimal masks $\mathbf{M}$ that help us interpolate pairs of samples optimally. We refer to this as **Guided PatchMix**. In Guided PatchMix, we search for a specific mask $\mathbf{M}_{i\ j}$ for a pair of image categories (c_i, c_j). We propose a novel genetic search optimization to automatically identify a set of category pairs (c_i, c_j) that are good to interpolate, and the set of mask $\mathbf{M}_{\cdot,\cdot}$ that determine how their images mix to generate new samples.

We have two concrete search goals in Guided PatchMix.

- (a) To identify what pairs of image categories are suitable for mixing.
- (b) For a specific category pair (c_i, c_j), what is the best mask $\mathbf{M}_{\cdot,\cdot}$ that allows for their images to interpolate well so that they generate new samples resulting in some improved generalization.

We, therefore, represent an **individual** candidate solution in our search as follows: (a) It includes a set of active class combinations $A \subseteq \{(c_i, c_j)\}$ and $|A| <= N$ (here N is a hyperparameter to tune). (b) For each active class pair (c_{i_a}, c_{j_a}) in the active set A, we have a mask matrix $\mathbf{M}_a$ of dimension $P \times P$ to search for, presenting $2^{P \times P}$ possible configurations to combine images from class $(c_{i_a}$ and $c_{j_a})$. Our **population** is initialized with I different individuals. Each individual A_i is built from random pairing between classes and from assigning random binary values in each $\mathbf{M}_a$. These individuals are evolved for approximately G **generations**. We decided to limit the amount of active combinations to size N, in order to narrow down the growth of the search space; this decision helps the algorithm to converge faster and yield better compositions. We show one cycle of our genetic search implementation in Figure 3 in the Supplementary Material.

One key component in using an evolutionary framework is to require a cost-effective **fitness function** for evaluating each "individual" candidate solution. For our search, we

can evaluate whether a specific "individual" (i.e., a set of active class combinations and their interpolation masks) is effective or not by training a model that uses those for data augmentation. The resulting accuracy of such a model serves as a good fitness criterion. However, it is computationally too expensive to train models for each possible "individual" configuration, considering the vast search space for possible set $A = \{(c_i, c_j)\}$ with size N plus the vast configuration space to search for each mask $\mathbf{M}_{c_{ia}, c_{ja}}$. Instead, our fitness criteria uses the average of the patch-level scores (based on Equation 4) from f_T on the validation set to choose "individual" solutions that are challenging and thus yielding potentially more informative interpolations than random patch selections.

After evaluating the fitness scores on each individual in the current population, some of them are discarded. Then some pairs of individuals are combined using a **crossover** function. Our choice of crossover function combines corresponding masks $\mathbf{M}_{i,j}$ from two different surviving individuals by copying the left and right half of each mask. Furthermore, some of these new **offspring** are transformed using a **mutation** operation with a low random probability. We define a set of possible mutation operations. Figure 4 in the Supplementary Materials shows examples of mutation operations. The search algorithm stops after a specific number of generations, or early stops if there is no further improvement. This last condition typically happens when the offspring in the current generation are almost the same as in the previous generation.

3.4 PatchMix Training Workflow

In summary, guided PatchMix trains a model as follows:

- First phase: We train Random PatchMix to define our fitness criteria f_T by optimizing L_T over a dataset of images and their corresponding labels.
- Second phase: We use genetic search to find the best set of masks $\mathbf{M}_{i,j}$ and active category pairs (c_i, c_j) that correspond to each of the discovered class combinations by using the fitness criteria induced by f_T.
- Third phase: We use the best set of masks $\mathbf{M}_{i,j}$ learned by our evolutionary algorithm to create informative augmented training samples based on the class combinations (c_i, c_j) discovered in the second phase;
- Fourth phase: We train a final prediction network f_O using the original training set, a randomly-augmented set based on random masks similar to the one described in the first phase, and the augmented set sampled in the third phase. We train this function by minimizing the sum of losses L_O over these samples.

4 Experimental Setup

4.1 Implementation Details

Evolutionary Search We adopt DEAP [15], an evolutionary computational framework, to work as our base genetic search data structure. This framework allowed us to define each individual in the population as a set of vectors, along with their grid mask configurations. The population is set to 500 individuals, which are evolved for 250 generations. Each individual has a limited number of active combinations, we treat the total number of allowed active combinations (N) as a hyperparameter. In our experiments, we set this as equal to the number of classes in the dataset, along with the same class combinations (c_i, c_i) pairs that are forced to be always active. Since we set $P = 4$ in all our experiments, each combination

has 65, 536 possible configurations. We also set the crossover probability to 50% and the mutation probability to 30%. Since our fitness function is a model trained using Random PatchMix, we spawn 20 processes to work in parallel, each containing the trained network to evaluate each individual. These 20 processes ran on 5 servers, each with 4 NVIDIA GPUs (ranging from GTX1080, GTX1080 Ti and Titan X).

PatchMix Training We train for 400 epochs, using mini-batches of 100 images. All the networks are optimized using Stochastic Gradient Descent (SGD) with Nesterov momentum. We use a weight decay regularization of 0.0005, a momentum factor of 0.9, and an initial learning rate of 0.1 which is updated using cosine annealing [26]. In all our experiments, we set $P = 4$ and $\alpha = 1$.

Baselines For both Mixup [43] and Cutmix [42], we use $\alpha = 1.0$. A cropping region of $16 * 6$ is used for Cutmix which is sampled from a Gaussian distribution with mean at the image centre. We also train for 400 epochs, using mini-batches of 100 images, SGD with nesterov momentum, a weight decay regularization of 0.0005, a momentum factor of 0.9, and an initial learning rate of 0.1 which is updated using cosine annealing.

5 Experimental Results

5.1 Supervised Image Classification

We evaluate PatchMix using both the random patch selections and our guided sampling strategy found using genetic search. Table 1 shows the top-1 accuracy and comparison against Mixup, Manifold Mixup and Cutmix, which are now standard techniques for data-augmentation and regularization on CIFAR-10 and CIFAR-100. Table 2 shows the top-1 accuracy and comparison against Mixup and Cutmix on Tiny-Imagenet and ImageNet. Random PatchMix outperforms a model trained without any data augmentation in all scenarios and is comparable to Mixup and Cutmix. Guided PatchMix outperforms all models trained using the other regularization approaches.

Model	Base	Mixup	Manifold Mixup	Cutmix	Rand PatchMix	Guided PatchMix
CIFAR-10						
MobileNetV2	90.55 ± 0.04	91.39 ± 0.02	91.79 ± 0.11	91.93 ± 0.04	92.64 ± 0.02	**93.85 ± 0.07**
ResNet32	92.61 ± 0.03	93.40 ± 0.02	94.14 ± 0.05	93.92 ± 0.06	94.13 ± 0.07	**94.93 ± 0.03**
ResNet50	93.70 ± 0.06	94.75 ± 0.03	95.24 ± 0.06	94.89 ± 0.05	95.04 ± 0.08	**95.48 ± 0.02**
ResNet56*	93.95 ± 0.04	94.42 ± 0.06	94.15 ± 0.03	93.92 ± 0.07	94.62 ± 0.09	**94.80 ± 0.07**
ResNet164*	94.06 ± 0.07	95.12 ± 0.03	95.55 ± 0.08	95.72 ± 0.07	95.81 ± 0.12	**96.06 ± 0.04**
CIFAR-100						
MobileNetV2	66.55 ± 0.21	68.45 ± 0.38	68.97 ± 0.41	69.14 ± 0.39	69.18 ± 0.38	**70.05 ± 0.37**
ResNet32	68.52 ± 0.38	69.12 ± 0.31	70.82 ± 0.39	71.32 ± 0.30	71.09 ± 0.49	**72.83 ± 0.26**
ResNet50	71.37 ± 0.27	71.99 ± 0.31	72.65 ± 0.40	72.91 ± 0.36	73.02 ± 0.48	**73.63 ± 0.31**
ResNet56*	71.60 ± 0.37	72.43 ± 0.29	73.21 ± 0.47	74.02 ± 0.35	74.56 ± 0.32	**75.26 ± 0.38**
ResNet164*	72.43 ± 0.22	74.14 ± 0.34	75.07 ± 0.49	76.97 ± 0.31	76.39 ± 0.44	**78.16 ± 0.47**

Table 1: Results on supervised classification datasets. Base refers to each model trained without any interpolation technique. The asterisk (*) refers to PreAct-ResNet. All experiments were run 3 times, we report their mean and standard deviation.

5.2 Weakly Supervised Object Localization and Object Detection

We also evaluate PatchMix on the weakly supervised localization task, which aims to find a target object using only the image-level label as supervision. In particular, we use the

	Tiny-Imagenet					ImageNet				
Model	Base	Mixup	Cutmix	Random PatchMix	Guided PatchMix	Base	Mixup	Cutmix	Random PatchMix	Guided PatchMix
ResNet50	61.18	63.04	63.36	62.94	**64.70**	76.27	77.01	77.41	77.38	**77.43**

Table 2: Results on supervised classification on ImageNet and Tiny-Imagenet. Base refers to each model trained without any interpolation technique.

Class Activation Mapping (CAM) [6] to extract the attention maps, and then we compute the maximal box accuracy, which is the bounding box accuracy and the Intersection over Union (IoU) of the proposed boxes, following the WSOL framework and evaluation benchmark recently proposed in [5] referred to as MaxBoxAccV2 [1].

PatchMix excels in this setting due to the patch-level supervision, which seems to give additional cues to the be considered by the scoring function. We show results on the CUB-200-2011 dataset trained on ResNet-50, VGG-16 and Inceptionv3 backbones. Random PatchMix outperforms other interpolation techniques as well as the baseline CAM. We show our results on Table 3, and qualitative results in the supplementary materials (section A.2).

Method	ResNet50	VGG16	Inceptionv3
Baseline (CAM) [45]	63.0	55.6	56.7
Mixup	55.8	51.9	53.4
Cutmix	62.8	54.9	57.4
PatchMix	**63.9**	**57.3**	**57.7**

Table 3: Results for the Weakly Supervised Object Localization task on the CUB-200-2011 dataset using three different backbones. The baseline is using the class activation mapping (CAM) without any data augmentation. We then apply Mixup, Cutmix and PatchMix, and report the MaxBoxAccV2 [5].

5.3 Transfer Learning Capacity

Table 4 presents results for various models pretrained on the ImageNet ILSVRC dataset and finetuned on seven different downstream tasks, including food recognition (Food-101[2]), bird classification (CUB-200-2011 [37]), scene recognition (SUN397 [40]), multi-label object classification (Pascal VOC [13], COCO [25], and NUS [8]), and image captioning (COCO Captions [4]). Our results include the performance for a base ResNet-50 model, a ResNet-50 model trained with CutMix and a ResNet-50 model trained with Random PatchMix. PatchMix shows the largest transferability across these tasks with the best performance scores in 7 out of 8 tasks.

5.4 Robustness

We perform studies on the FGSM [16] white box attack on ImageNet using $\varepsilon = 0.1, 0.2, 0.3$. The aim of this test is to create adversarial samples by fixing the perturbation on a pixel to be of a fixed size (i.e. ε). As shown in Table 5, PatchMix consistently outperforms previous methods in 2 out of 3 attacks.

[1] https://github.com/clovaai/wsolevaluation

ImageNet Pretrained	Food-101 top-1 acc	CUB-200 top-1 acc	SUN397 top-1 acc	VOC mAP	COCO mAP	NUS mAP	MS-COCO NIC [36] BLEU-1	BLEU-4
RN	87.70	76.30	60.41	92.13	79.64	80.19	61.4	22.9
RN+M	87.82	78.91	60.16	91.80	81.20	81.72	61.6	23.2
RN+CM	**88.02**	77.77	58.48	92.41	79.68	80.53	64.8	24.9
RN+PM	87.95	79.17	**61.08**	**92.42**	81.27	82.45	**66.8**	**26.3**
RN+GPM	87.50	**79.50**	60.99	92.39	**83.21**	**82.95**	65.5	25.5

Table 4: Transfer learning results on different datasets. We use a ResNet-50 model pretrained on ImageNet via four different training strategies. The first row corresponds to normal training without data-pair interpolations. RN+M, RN+CM, RN+PM and RN+GPM refers to finetuning a ResNet-50 model with Mixup, CutMix, PatchMix and Guided PatchMix respectively.

ε	Base	Mixup	Cutmix	Random PatchMix	Guided PatchMix
0.1	15.96	28.42	29.26	30.62	**31.88**
0.2	9.12	20.45	19.92	21.07	**21.68**
0.3	5.87	**15.31**	13.65	14.29	14.57

Table 5: Results on the FGSM white box attach on ImageNet: we report the top-1 accuracy a ResNet-50 model trained with ImageNet using all techniques. The ε indicates the perturbation level of the adversarial images generated.

5.5 Ablation Studies

Given the flexible capabilities our grid-mask design allows, we conduct a thorough set of experiments to determine whether the patch-level loss is helpful or not, and to what extent the number of patches impact the overall performance. To examine all of these possibilities, we run a set of experiments on CIFAR-10 using ResNet-32 as the base network. For these experiments we keep the same hyperparameter selections we use to report our results in section 5. We vary the size of P by a factor of 2, which affects the Grid Mask and the patch-level loss L_P. We also investigate the effect of activating or deactivating the full image supervision L_O on each possible combination.

We report the results of these experiments in Table 6. Our proposed patch-level loss gives the additional supervision that is necessary for the network to stabilize and converge, mitigating the noise from the data interpolations. In addition, this patch-level supervision enables a form of visual representation learning, and the combination of it along with the image-level supervision yields the best performance. Furthermore, when evaluating the value of P, we found that a grid of 4×4 yield the best performance. We note that incrementing the value of P hurts the performance dramatically. This may occur due to the significant level of freedom added by a 8×8 grid-mask, where the patch-level supervision is not able to mitigate the noise introduced. We also experimented on adding a hyperparameter to balance both losses but found out that giving the same weight performs better.

5.6 Fitness Function Analysis.

We also evaluate how the fitness function impacts the performance of the combinations discovered by our genetic algorithm. After a network is trained using our Random PatchMix approach, it can be used as a function approximation of the underlying distribution generated by the image-pair interpolation along with their corresponding new labels. Thus, we can use

Grid	Image-level loss L_O	Patch-level loss L_P	Top-1 Acc
2 × 2	✓	✓	93.78
2 × 2	✓		93.30
2 × 2		✓	92.80
4 × 4	✓	✓	94.10
4 × 4	✓		92.73
4 × 4		✓	92.02
8 × 8	✓	✓	92.50
8 × 8	✓		91.94
8 × 8		✓	92.02

Table 6: Ablation analysis: Top-1 accuracy on CIFAR-10 when varying the grid size, and the effect of using image and patch level supervision using ResNet-32 as the base network.

this network to assess the patch mask $M_{i,j}$ configurations and class activations (c_i, c_j), by computing the L_P loss over these masks and image pairs. We show our results in Table 7. We evaluate the how the patch-level accuracy of the validation set affects our genetic search. First, we show the result of using the configurations that yield the highest patch-level scores. Then we show the results of using the configurations that yield the lowest patch-level scores. We observe that using the configurations with the lowest patch-level accuracy yield better results. This means that the genetic algorithm is able to find challenging configurations for the model trained with random masks. Thus, our guided version allows the model to benefit from this information, leading to better results.

Fitness Function	Allow Same Class Pairs?	
	Yes	No
max L_P	94.80	95.42
min L_P	95.97	**96.32**

Table 7: Top-1 accuracy on CIFAR-10 when applying different fitness functions and the effect of using the same class combinations. We use PreAct-ResNet-164 as the backbone network architecture.

6 Conclusion

Our paper introduces PatchMix a novel interpolation method for augmenting the available number of samples during training by combining pairs of samples. Unlike previous methods that rely on patch-level interpolations our method allows for a more significant degree of flexibility regarding possible combinations by using a grid-like pattern. Moreover, an evolutionary search method for optimally selecting combinations that lead to increased exploration of critical areas of the input space was devised. We also found a fitness criteria that requires no model training by leveraging a pretrained PatchMix model that is trained by selecting random patches. We posit that PatchMix can serve as a regularizer that can complement other single sample data augmentation methods.

7 Acknowledgements

The authors would like to thank the anonymous reviewers for their insightful comments. This work is supported by the National Science Foundation under awards No #2045773 and #2040961.

References

[1] Kyungjune Baek, Duhyeon Bang, and Hyunjung Shim. Gridmix: Strong regularization through local context mapping. *Pattern Recognition*, 109:107594, 2021. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2020.107594. URL https://www.sciencedirect.com/science/article/pii/S0031320320303976.

[2] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014.

[3] Pengguang Chen, Shu Liu, Hengshuang Zhao, and Jiaya Jia. Gridmask data augmentation, 2020.

[4] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015.

[5] Junsuk Choe, Seong Joon Oh, Seungho Lee, Sanghyuk Chun, Zeynep Akata, and Hyunjung Shim. Evaluating weakly supervised object localization methods right. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. to appear.

[6] Junsuk Choe, Seong Joon Oh, Seungho Lee, Sanghyuk Chun, Zeynep Akata, and Hyunjung Shim. Evaluating weakly supervised object localization methods right. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3133–3142, 2020.

[7] Hsin-Ping Chou, S. Chang, J. Pan, Wei Wei, and D. Juan. Remix: Rebalanced mixup. *ArXiv*, abs/2007.03943, 2020.

[8] Tat-Seng Chua, Jinhui Tang, Richang Hong, Haojie Li, Zhiping Luo, and Yantao Zheng. Nus-wide: A real-world web image database from national university of singapore. In *Proceedings of the ACM International Conference on Image and Video Retrieval*, CIVR '09, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605584805. doi: 10.1145/1646396.1646452. URL https://doi.org/10.1145/1646396.1646452.

[9] E. Cubuk, Barret Zoph, Dandelion Mané, V. Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation strategies from data. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 113–123, 2019.

[10] Ali Dabouei, Sobhan Soleymani, Fariborz Taherkhani, and Nasser M. Nasrabadi. Supermix: Supervising the mixing data augmentation. *ArXiv*, abs/2003.05034, 2020.

[11] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[12] Terrance Devries and Graham W. Taylor. Improved regularization of convolutional neural networks with cutout. *ArXiv*, abs/1708.04552, 2017.

[13] Mark Everingham, Luc Gool, Christopher K. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vision*, 88 (2):303–338, June 2010. ISSN 0920-5691. doi: 10.1007/s11263-009-0275-4. URL https://doi.org/10.1007/s11263-009-0275-4.

[14] M. Faramarzi, M. Amini, Akilesh Badrinaaraayanan, Vikas Verma, and A. Chandar. Patchup: A regularization technique for convolutional neural networks. *ArXiv*, abs/2006.07794, 2020.

[15] Félix-Antoine Fortin, François-Michel De Rainville, Marc-André Gardner, Marc Parizeau, and Christian Gagné. DEAP: Evolutionary algorithms made easy. *Journal of Machine Learning Research*, 13:2171–2175, jul 2012.

[16] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015. URL http://arxiv.org/abs/1412.6572.

[17] Hongyu Guo, Yongyi Mao, and Richong Zhang. Mixup as locally linear out-of-manifold regularization. In *AAAI*, 2019.

[18] E. Harris, A. Marcu, Matthew Painter, M. Niranjan, Adam Prugel-Bennett, and Jonathon S. Hare. Fmix: Enhancing mixed sample data augmentation. 2020.

[19] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[20] A. Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *CACM*, 2017.

[21] Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2012.

[22] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-100 (canadian institute for advanced research). URL http://www.cs.toronto.edu/~kriz/cifar.html.

[23] Y. Le and X. Yang. Tiny imagenet visual recognition challenge. 2015.

[24] Sungbin Lim, Ildoo Kim, Taesup Kim, C. Kim, and S. Kim. Fast autoaugment. *ArXiv*, abs/1905.00397, 2019.

[25] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 740–755, Cham, 2014. Springer International Publishing. ISBN 978-3-319-10602-1.

[26] Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with restarts. *CoRR*, abs/1608.03983, 2016. URL http://arxiv.org/abs/1608.03983.

[27] Zhichao Lu, Ian Whalen, V. Boddeti, Yashesh D. Dhebar, K. Deb, E. Goodman, and W. Banzhaf. Nsga-net: neural architecture search using multi-objective genetic algorithm. *Proceedings of the Genetic and Evolutionary Computation Conference*, 2019.

[28] Zhijun Mai, Guosheng Hu, Dexiong Chen, F. Shen, and H. Shen. Metamixup: Learning adaptive interpolation policy of mixup with meta-learning. *ArXiv*, abs/1908.10059, 2019.

[29] Hieu Pham, Melody Y. Guan, Barret Zoph, Quoc V. Le, and Jeff Dean. Efficient neural architecture search via parameter sharing. In *ICML*, 2018.

[30] Shaoqing Ren, Kaiming He, Ross B. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:1137–1149, 2015.

[31] Olga Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Zhiheng Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211–252, 2015.

[32] John Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347, 2017.

[33] Krishna Kumar Singh, Hao Yu, Aron Sarmasi, Gautam Pradeep, and Yong Jae Lee. Hide-and-seek: A data augmentation technique for weakly-supervised localization and beyond, 2018.

[34] Ryo Takahashi, Takashi Matsubara, and Kuniaki Uehara. Data augmentation using random image cropping and patching for deep cnns. *ArXiv*, abs/1811.09030, 2018.

[35] Vikas Verma, A. Lamb, C. Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *ICML*, 2019.

[36] Oriol Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3156–3164, 2015.

[37] C. Wah, S. Branson, P. Welinder, P. Perona, and Serge J. Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

[38] Devesh Walawalkar, Zhiqiang Shen, Zechun Liu, and M. Savvides. Attentive cutmix: An enhanced data augmentation approach for deep learning based image classification. *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3642–3646, 2020.

[39] Shaoru Wang, Yongchao Gong, Junliang Xing, Lichao Huang, C. Huang, and Weiming Hu. Rdsnet: A new deep architecture for reciprocal object detection and instance segmentation. In *AAAI*, 2020.

[40] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3485–3492, 2010. doi: 10.1109/CVPR.2010.5539970.

[41] Qizhe Xie, Zihang Dai, E. Hovy, Minh-Thang Luong, and Quoc V. Le. Unsupervised data augmentation. *ArXiv*, abs/1904.12848, 2019.

[42] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6022–6031, 2019.

[43] Hongyi Zhang, M. Cissé, Yann Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *ArXiv*, abs/1710.09412, 2018.

[44] Z. Zhong, L. Zheng, Guoliang Kang, Shaozi Li, and Y. Yang. Random erasing data augmentation. In *AAAI*, 2020.

[45] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[46] Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning. *ArXiv*, abs/1611.01578, 2017.

[47] Barret Zoph, V. Vasudevan, Jonathon Shlens, and Quoc V. Le. Learning transferable architectures for scalable image recognition. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8697–8710, 2018.