KeepAugment: A Simple Information-Preserving Data Augmentation Approach

Chengyue Gong[1], Dilin Wang[2], Meng Li[2], Vikas Chandra[2], Qiang Liu[1]
[1] University of Texas at Austin [2] Facebook
{cygong, lqiang}@cs.utexas.edu, {wdilin, meng.li, vchandra}@fb.com

Abstract

Data augmentation (DA) is an essential technique for training state-of-the-art deep learning systems. In this paper, we empirically show that the standard data augmentation methods may introduce distribution shift and consequently hurt the performance on unaugmented data during inference. To alleviate this issue, we propose a simple yet effective approach, dubbed KeepAugment*, to increase the fidelity of augmented images. The idea is to use the saliency map to detect important regions on the original images and preserve these informative regions during augmentation. This information-preserving strategy allows us to generate more faithful training examples. Empirically, we demonstrate that our method significantly improves upon a number of prior art data augmentation schemes, e.g. AutoAugment, Cutout, random erasing, achieving promising results on image classification, semi-supervised image classification, multi-view multi-camera tracking and object detection.*

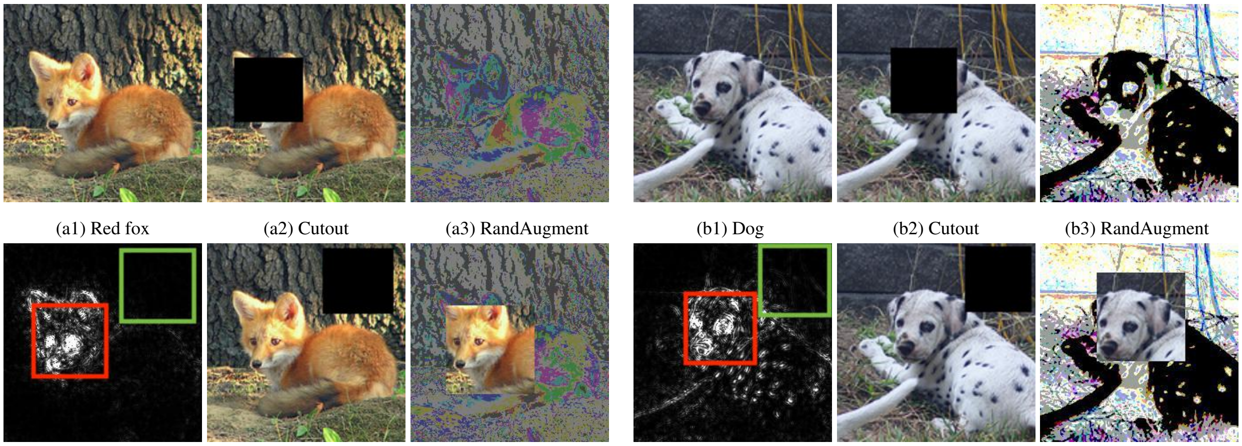
1. Introduction

Recently, data augmentation is proven to be a crucial technique for solving various challenging deep learning tasks, including image classification [e.g. 8, 39, 4, 5], natural language understanding [e.g. 7], speech recognition [25] and semi-supervised learning [e.g. 36, 29, 1]. Notable examples include regional-level augmentation methods, such as Cutout [8] and CutMix [39], which mask or modify randomly selected rectangular regions of the images; and image-level augmentation approaches, such as AutoAugment [4] and Fast Augmentation [18]), which leverage reinforcement learning to find optimal policies for selecting and combining different label-invariant transforms (e.g., rotation, color-inverting, flipping).

Although data augmentation increases the effective data size and promotes diversity in training examples, it inevitably introduces noise and ambiguity into the training process. Hence the overall performance would deteriorate if the augmentation is not properly modulated. For example, as shown in Figure 1, random `Cutout` (Figure 1 (a2) and (b2)) or `RandAugment` (Figure 1 (a3) and (b3)) may destroy the key characteristic information of original images that is responsible for classification, creating augmented images to have wrong or ambiguous labels.

In this work, we propose *KeepAugment*, a simple yet powerful adaptive data augmentation approach that aims to increase the fidelity of data augmentation by *always keeping important regions untouched* during augmentation. The idea is very simple: at each training step, we first score the importance of different regions of the original images using attribution methods such as saliency-map [28]; then we perform data augmentation in an adaptive way, such that regions with high importance scores always remain intact. This is achieved by either avoiding cutting critical high-score areas (see Figure 1(a5) and (b5)), or pasting the patches with high importance scores to the augmented images (see Figure 1(a6) and (b6)).

Although KeepAugment is very simple and computationally efficient, the empirical results on a variety of vision tasks show that it can significantly improve the prior art data augmentation (DA) baselines. Specifically, for image classification, we achieve improvements on existing DA techniques, including `Cutout` [8], `AutoAugment` [4], and `CutMix` [39], boosting the performance on CIFAR-10 and ImageNet across various neural architectures. In particular, we achieve 98.7% test accuracy on CIFAR-10 using PyramidNet-ShakeDrop [38] by applying our method on top of `AutoAugment`. When applied to multi-view multi-camera tracking, we improve upon the recent state-of-the-art results on the Market1501 [44] dataset. In addition, we demonstrate that our method can be applied to semi-supervised learning and the model trained on ImageNet using our method can be transferred to COCO 2017 objective detection tasks [21] and allows us to improve the strong `Detectron2` baselines [35].



(a1) Red fox (a2) Cutout (a3) RandAugment (b1) Dog (b2) Cutout (b3) RandAugment

(a4) Saliency map (a5) Keep+Cutout (a6) Keep+RandAugment (b4) Saliency map (b5) Keep+Cutout (b6) Keep+RandAugment

Figure 1. KeepAugment improves existing data augmentation by always keeping the important regions (measured using saliency map) of the image untouched during augmentation. This is achieved by either avoiding to cut important regions (see KeepCutout), or pasting important regions on top of the transformed images (see KeepRandAugment). Images are from ImageNet [6].

2. Data Augmentation

In this work, we focus on label-invariant data augmentation due to their popularity and significance in boosting empirical performance in practice. Let x be an input image, data augmentation techniques allow us to generate new images $x' = \mathcal{A}(x)$ that are expected to have the same label as x, where $\mathcal{A}$ denotes a label-invariant image transform, which is typically a stochastic function. Two classes of augmentation techniques are widely used for achieving state-of-the-art results on computer vision tasks:

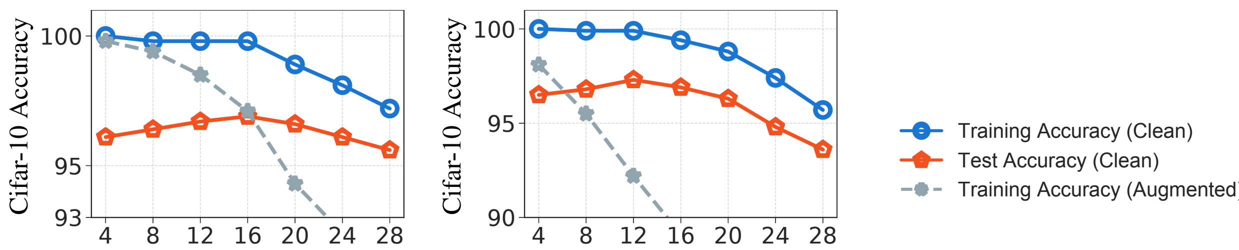
Region-Level Augmentation Region-level augmentation schemes, including Cutout [8] and random erasing [45], work by randomly masking out or modifying rectangular regions of the input images, thus creating partially occluded data examples outside the span of the training data. This procedure could be conveniently formulated as applying randomly generated binary masks to the original inputs. Precisely, consider an input image x of size $H \times W$, and a rectangular region S of the image domain. Let $M(S) = [M_{ij}(S)]_{ij}$ be the binary mask of S with $M_{ij}(S) = \mathbb{I}((i, j) \in S)$. Then the augmented data can be generated by modifying the image on region S, yielding images of form $x' = (1 - M(S)) \odot x + M(S) \odot \delta$, where $\odot$ is element-wise multiplication, and δ can be either zeros (for Cutout) or random numbers (for random erasing). See Figure 1(a2) and (b2) for examples.

Image-Level Augmentation Exploiting the invariance properties of natural images, image-level augmentation methods apply label-invariant transformations on the whole image, such as solarization, sharpness, posterization, and color normalization. Traditionally, image-level transformations are often manually designed and heuristically chosen. Recently, AutoAugment [4] applies reinforcement learning to automatically search optimal compositions of transformations. Several subsequent works, including RandAugment [5], Fast AutoAugment [18], alleviate the heavy computational burden of searching on the space of transformation policies by designing more compact search spaces. See Figure 1(b3) and Figure 1(a3) for examples of transforms used by RandAugment.

Data Augmentation and its Trade-offs Although data augmentation increases the effective size of data, it may inevitably cause loss of information and introduce noise and ambiguity if the augmentation is not controlled properly [e.g. 34, 12]. To study this phenomenon empirically, we plot the train and testing accuracy on CIFAR-10 [16] when we apply `Cutout` with increasingly large cutout length in Figure 2(a), and RandAugment with increasing distortion magnitude (see [5] for the definition) in Figure 2(b). As typically expected, the generalization (the gap between the training and testing accuracy on clean data) improves as the magnitude of the transform increases in both cases. However, when the magnitudes of the transform are too large ($\geq$ 16 for `Cutout` and $\geq$ 12 for `RandAugment`), the training accuracy (blue line), and hence the testing accuracy (red line), starts to degenerate, indicating that augmented data no longer faithfully represent the clean training data in this case, such that the training loss on augmented data no longer forms a good surrogate of the training loss on the clean data.

3. Our Method

We introduce our method for controlling the fidelity of data augmentation and hence decreasing harmful misinfor-



(a) Cutout length in CutOut (b) Distortion magnitude in RandAugment

Figure 2. The training and testing accuracy of Wide ResNet-28-10 trained on CIFAR-10 with Cutout and RandAugment, when we vary the cutout length of Cutout (a), and the distortion magnitude of RandAugment (b). We follow the same implementation details as in [8] and [5]. For RandAugment, we fix the number of transformations to be 3 as suggested in [5].

mation. Our idea is to measure the importance of the rectangular regions in the image by saliency map, and ensure that the regions with the highest scores are always presented after the data augmentation: for `Cutout` , we achieve this by avoiding to cut the important regions (see Figure 1(a5) and (b5)); for image-level transforms such as RandAugment, we achieve this by *pasting* the important regions on the top of the transformed images (see Figure 1 (a6) and (b6)).

Specifically, let $g_{ij}(x, y)$ be saliency map of an image x on pixel (i, j) with the given label y. For a region S on the image, its importance score is defined by

$$\mathcal{I}(S, x, y) = \sum_{(ij) \in S} g_{ij}(x, y). \tag{1}$$

In our work, we use the standard saliency map based on vanilla gradient [28]. Specifically, given an image x and its corresponding label logit value $\ell_y(x)$, we take $g_{ij}(x, y)$ to be the absolute value of vanilla gradients $|\nabla_x \ell_y(x)|$. For RBG-images, we take channel-wise maximum to get a single saliency value for each pixel (i, j).

Selective-Cut For region-level (e.g. cutout-based) augmentation that masks or modifies randomly selected rectangle regions, we control the fidelity of data augmentation by ensuring that the regions being cut can not have large importance scores. This is achieved in practice by Algorithm 1(a), in which we randomly sample regions S to be cut until its importance score $\mathcal{I}(S, x, y)$ is smaller than a given threshold τ. The corresponding augmented example is defined as follows,

$$\tilde{x} = (1 - M(S)) \odot x, \tag{2}$$

where $M(S) = [M_{ij}(S)]_{ij}$ is the binary mask for S, with $M_{ij} = \mathbb{I}((i, j) \in S)$.

Selective-Paste Because image-level transforms modify the whole images jointly, we ensure the fidelity of the transform by pasting a random region with high importance score (see Figure 1(a6) and (b6) for an example). Algorithm 1(b) shows how we achieve this in practice, in which we draw an image-level augmented data $x' = \mathcal{A}(x)$, uniformly sample a region S that satisfies $\mathcal{I}(S, x, y) > \tau$ for a threshold τ, and and paste the region S of the original image x to x', which yields

$$\tilde{x} = M(S) \odot x + (1 - M(S)) \odot x'. \tag{3}$$

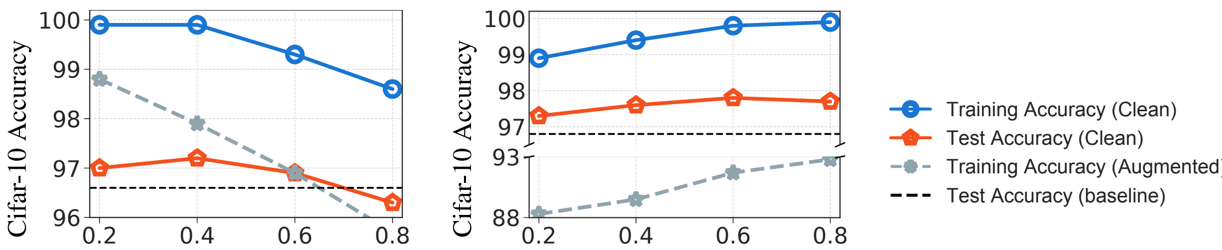
Similarly, $M_{ij}(S) = \mathbb{I}((i, j) \in S)$ is the binary mask of region S.

Algorithm 1 KeepAugment: An information-preserving data augmentation approach

Input: given a network, an input image and label pair (x, y), threshold τ
(a) if use Selective-Cut
 repeat randomly select a mask region S **until** region score $\mathcal{I}(S, x, y) < \tau$
 $\tilde{x} = (1 - M(S)) \odot x$ (see Eq. 2)
(b) if use Selective-paste
 $x' = \mathcal{A}(x)$ //apply data augmentation
 repeat randomly select a mask region S **until** region score $\mathcal{I}(S, x, y) > \tau$
 $\tilde{x} = M(S) \odot x + (1 - M(S)) \odot x'$ (see Eq. 3)
Return $\tilde{x}$

Remark In practice, we choose our threshold τ in an adaptive way. Technically, given an image and consider an region size $h \times w$ of interest, we first calculate the importance scores of all possible candidate regions, following Eq. 1; then we set our threshold to be the τ-quantile value of all the importance scores $\mathcal{I}(S, x, y)$ of all candidate regions. For *selective-cut*, we uniformly keep sampling a mask region S until its corresponding score $\mathcal{I}(S, x, y)$ is smaller than the threshold. For *selective-paste*, we uniformly sample a region S with importance score is greater than the threshold.

We empirically study the effect of our threshold τ on CIFAR-10, illustrated in Figure 3. Intuitively, for *selective-*



(a) threshold τ in CutOut (b) threshold τ in RandAugment

Figure 3. Analysis of the effect of threshold τ of our algorithm for Cutout (a) and RandAugment (b). In (a), we fix the cutout length 20. In (b), We fix the number of transformation to be 3 and distortion magnitude to be 15 and the paste back region size to be 8×8. We plot how the accuracy changes with respect to different choices of τ. We use Wide ResNet-28-10 and train on CIFAR-10. The dash line *(baseline)* in (a) represents test accuracy achieved by CutOut without *selective-cut*; the dash line *baseline* in (b) is the test accuracy achieved by RandAugment without *selective-paste*.

cut, it's more likely to cut out important regions as we use an increasingly larger threshold τ; on the contrary, a larger τ corresponds to copy back more critical regions for *selective-paste*. As we can see from Figure 3, for `Cutout` (Figure 3 (a)), we improve on the standard `Cutout` baseline (dash line) significantly when the threshold τ is relative small (e.g. $\tau \leq 0.6$) since we would always avoid cutting important regions. As expected, the performance drops sharply when important regions are removed with a relative large threshold τ (e.g. $\tau = 0.8$); for `RandAugment` (Figure 3 (b)), using a lower threshold (e.g., $\tau = 0.2$) tends to yield similar performance as the standard `RandAugment` baseline (dash line). Increasing the threshold τ (τ= 0.6 or 0.8) yields better results. We notice that further increasing τ ($\tau = 0.8$) may hurt the performance slightly, likely because a large threshold yields too restrictive selection and may miss other informative regions. we further evaluated $\tau = 0$, such that the saliency map information would be ignored. With $\tau = 0$, we achieved 97.3% accuracy, which is worse compared to the result of our default setting (i.e., 97.8% accuracy with $\tau = 0.6$).

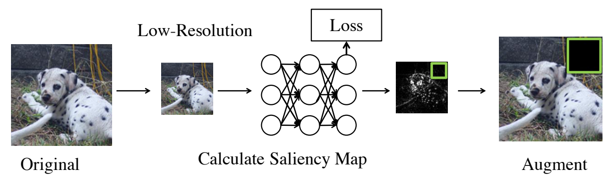
3.1. Efficient Implementation of KeepAugment

Note that our KeepAugment requires to calculate the saliency maps via back-propagation at each training step. Naive implementation leads to roughy twice of the computational cost. In this part, we propose two computational efficient strategies for calculating saliency maps that overcome this weakness.

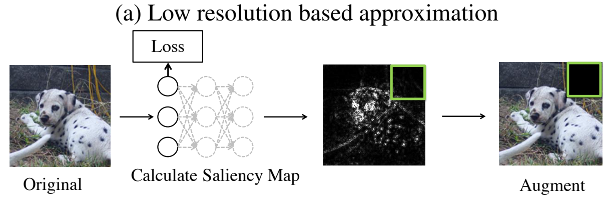
Low resolution based approximation we proceed as follows: a) for a given image x, we first generate a low-resolution copy and then calculate its saliency map; b) we map the low-resolution saliency maps to their corresponding original resolution. This allows us to speed up the saliency maps calculation significantly, e.g., on ImageNet, we achieve roughly $3\times$ computation cost reduction by reducing the resolution from 224 to 112.

Early classification head based approximation Our second idea is to introduce an early loss head in the network, then we approximate saliency maps with this loss. In practice, we add an additional average pooling layer and a linear head after the first block of our networks evaluated. Our training objective is the same as the Inception Network [31]. The neural network is trained with the standard loss together with the auxiliary loss. We achieve about $3\times$ computation cost reduction in computing saliency maps.

Furthermore, in section 4, we show that both approximation strategies do not lead to any performance drop. In the following, we denote our low resolution based approximation as *low resolution* and early classification head based approximation as *early loss* for presentation clarity.



(a) Low resolution based approximation



(b) Early classification head based approximation

Figure 4. We demonstrate two different approaches for using KeepAugment with less training time. Using `Cutout` as an example, Figure (a) shows that we can use a low resolution copy to calculate the saliency map, and then generate the augmented image. Figure (b) shows that when calculating the saliency map, we can use an additional loss at early layer of a given neural network.

4. Experiments

In this section, we show our adaptive augmentation strategy `KeepAugment` significantly improves on existing state-of-the-art data augmentation baselines on a va-

Model	ResNet-18	ResNet-110	Wide ResNet-28-10
Cutout	95.6±0.1	94.8±0.1	96.9±0.1
KeepCutout	96.1±0.1	**95.5±0.1**	**97.3±0.1**
KeepCutout (*low resolution*)	**96.2±0.1**	**95.5±0.1**	**97.3±0.1**
KeepCutout (*early loss*)	96.0±0.1	95.3±0.1	97.2±0.1
Model	Wide ResNet-28-10	Shake-Shake	PyramidNet+ShakeDrop
AutoAugment	97.3±0.1	97.4±0.1	98.5
KeepAutoAugment	**97.8±0.1**	97.8±0.1	**98.7±0.0**
KeepAutoAugment (*low resolution*)	**97.8±0.1**	**97.9±0.1**	**98.7±0.0**
KeepAutoAugment (*early loss*)	**97.8±0.1**	97.7±0.1	98.6±0.0

Table 1. Test accuracy (%) on CIFAR-10 using various models architectures.

riety of challenging deep learning tasks, including image classification, semi-supervised image classification, multi-view multi-camera tracking, and object detection. For semi-supervised image classification and multi-view multi-camera tracking, we use *low resolution* images to calculate saliency maps as discussed above.

Settings We apply our method to improve prior art region-level augmentation methods, including [8], CutMix [39], Random Erasing [45] and image-level augmentation approach, such as AutoAugment [4]. To sample the region of interest, for each image, we rank the absolute saliency values measured on all candidate regions and take our threshold to be the τ-th percentile value. We set τ to 0.6 for all our experiments. Additionally, we set the cutout *paste-back* length to be 16 on CIFAR-10 and 40 on ImageNet, which is the default setting used by `Cutout` [8]. For our *low resolution* based efficient training strategy, we reduce the image width and height by half with bicubic interpolation. For the *early loss* based approach, we use an additional head (linear transform and loss) with a coefficient of 0.3 after the first block of each network.

4.1. CIFAR-10 Classification

We apply of our adaptive selective strategy to improve two state-of-the-art augmentation schemes, `Cutout` and `RandAugment`, on the CIFAR-10 [1] [15] dataset. We experiment with various of backbone architectures, such as ResNets [13], Wide ResNets [40], PyramidNet ShakeDrop [38] and Shake-Shake [9]. We closely follow the training settings suggested in [8] and [5]. Specifically, we train 1,800 epochs with cosine learning rate deacy [22] for PyramidNet-ShakeDrop and 300 epochs for all other networks, We report the test test accuracy in Table 1. All results are averaged over three random trials, except for PyramidNet-ShakeDrop [38], on which only one random trial is reported.

[1] https://www.cs.toronto.edu/~kriz/cifar.html

From Table 1, we observe a consistent improvement on test accuracy by applying our information-preserving augmentation strategy.

Improve on CutOut We study the relative improvements on `Cutout` across various cutout lengths. We use ResNet-18 and train on CIFAR-10. We experiment with a variety of cutout length from 8 to 24. As shown in Table 2, we observe that our KeepCutout achieves increasingly significant improvements over `Cutout` when the cutout regions become larger. This is likely because that with large cutout length, `Cutout` is more likely to remove the most informative region and hence introducing misinformation, which in turn hurts the network performance. On the other hand, with a small cutout length, e.g. 8, those informative regions are likely to be preserved during augmentation; standard `Cutout` strategy achieves better performance by taking advantage of more diversified training examples.

Cutout length	Cutout	KeepCutout
8	**95.3±0.0**	95.1±0.0
12	95.4±0.0	**95.5±0.0**
16	95.6±0.0	**96.1±0.0**
20	95.5±0.1	**96.0±0.1**
24	94.9±0.1	**95.6±0.1**

Table 2. Test accuracy (%) of ResNet-18 on CIFAR-10. All results are averaged over 5 random trials.

Improve on AutoAugment In this case, we use the AutoAugment policy space, apply our *selective-paste* and study the empirical gain over AutoAugment for four distortion augmentation magnitude (6, 12, 18 and 24). We train Wide ResNet-28-10 on CIFAR-10 and closely follow the training setting suggested in [4]. As we can see from Table 3, our method yields better performance in all settings consistently, and our improvements is more significant when the transformation distortion magnitude is large.

Magnitude	AutoAugment	KeepAutoAugment
6	96.9±0.1	**97.3±0.1**
12	97.1±0.1	**97.5±0.1**
18	97.1±0.1	**97.6±0.1**
24	97.3±0.1	**97.8±0.1**

Table 3. Test accuracy (%) of wide ResNet-28-10 on CIFAR-10 across varying distortion augmentation magnitudes. All results are averaged over 5 random trials.

Wide ResNet-28-10	Accuracy (%)	Time (s)
GridMask	97.5±0.1	92
AugMix	97.5±0.0	92
Attentive CutMix	97.3±0.1	127
KeepAutoAugment+L	**97.8±0.1**	111
ShakeShake	Accuracy (%)	Time (s)
GridMask	97.4±0.1	124
AugMix	97.5±0.0	124
Attentive CutMix	97.4±0.1	166
KeepAutoAugment+L	**97.9±0.1**	142

Table 4. Results on CIFAR-10 using various models architectures and various baselines. 'Time' reports the per epoch training time on one TITAN X GPU. 'Accuracy' reports the accuracy on test set, which is averaged over 5 trials. 'L' denotes *low resolution*. We use *Wide ResNet-28-10*, and the corresponding AutoAugment baseline result is presented above.

Additional Comparisons on CIFAR-10 Recently, some researchers [3, 33, 14] also mix the clean image and augmented image together to achieve higher performance. Girdmask [3], AugMix [14] and Attentive `CutMix` are popular methods among these approaches. Here, we conduct experiments on CIFAR-10 to show the accuracy and training cost of each method. Note that we implement all the baselines by ourselves, and the results of our implementation are comparable or even better than the results reported in the original papers.

Table 4 shows that our proposed algorithm can achieve clear improvements on accuracy over all other baselines. Moreover, Gridmask only implements upon `Cutout` and Attentive `CutMix` only implements upon `CutMix` by pasting the most important region. But our approach is more flexible and can be easily applied to improve a large variety of data augmentation schemes.

4.2. ImageNet Classification

We conduct experiments on large-scale challenging ImageNet dataset, on which our adaptive augmentation algorithm again shows clear advantage over existing methods.

Dataset and Settings We use ILSVRC2012, a subset of ImageNet classification dataset [6], which contains around 1.28 million training images and 50,000 validation images from 1,000 classes. We apply our adaptive data augmentation strategy to improve `CutMix` [39] and `AutoAugment` [4], respectively.

`CutMix` randomly mixes images and labels. To augment an image x with label y, `CutMix` removes a randomly selected region from x and replace it with a patch of the same size copied from another random image x' with label y'. Meanwhile, the new label is mixed as $\lambda y+(1-\lambda)y'$, where λ equals the uncorrupted percentage of image x. We improve on `CutMix` by using *selective-cut*. In practice, we found it is often quite effective to simply avoiding cutting informative region from x. We denote our adaptive CutMix method as `KeepCutMix`. We further improve on `AutoAugment` by pasting-backing randomly selected regions with important score greater than $\tau = 0.6$.

For a fair comparison, we closely follow the training settings in `CutMix` [39] and `AutoAugment` [5]. We test our method on both ResNet50 and ResNet101 [13]. Our models are trained for 300 epochs, and the experiment is implemented based on the open-source code [2].

Method	ResNet-50		ResNet-101	
	Top-1	Top-5	Top-1	Top-5
Vanilla [13]	76.3	92.9	77.4	93.6
Dropout [30]	76.8	93.4	77.7	93.9
DropPath [17]	77.1	93.5	-	-
Manifold Mixup [32]	77.5	93.8	-	-
Mixup [41]	77.9	93.9	79.2	94.4
DropBlock [10]	78.3	94.1	79.0	94.3
RandAugment [5]	77.6	93.8	79.2	94.4
Random Erasing [45]	77.3	93.3	79.6	94.7
AutoAugment [4]	77.6	93.8	79.3	94.4
KeepAutoAugment	78.0	**93.9**	**79.7**	**94.6**
+ *low resolution*	**78.1**	**93.9**	**79.7**	**94.6**
+ *early loss*	77.9	93.8	79.6	94.5
CutMix [39]	78.6	94.0	79.9	94.6
KeepCutMix	79.0	**94.4**	**80.3**	95.1
+ *low resolution*	**79.1**	**94.4**	**80.3**	**95.2**
+ *early loss*	**79.0**	94.3	80.2	95.1

Table 5. Validation accuracy (%) on ImageNet using ResNet-50 and ResNet-101.

Results We report the single-crop top-1 and top-5 accuracy on the validation set in table 5. Compared to `CutMix`, we method `KeepCutMix` achieves 0.5% improvements on top1 accuracy using ResNet-50 and 0.4% higher top1 accuracy using ResNet101; compared to AutoAugment [4], our method improves top-1 accuracy from 77.6% to 78.1% and 79.3% to 79.7% using ResNet-50 and ResNet-101, respectively. Again, we also notice that our accelerated ap-

[2] https://github.com/clovaai/CutMix-PyTorch

Model	R-18	R-110	Wide ResNet
Cutout	19	28	92
KeepCutout	38 +100.0%	54 +92.8%	185 +101.1%
+ *low resolution*	24 +26.3%	35 +25.0%	111 +20.6%
+ *early loss*	23 +13.0%	34 +21.1%	104 +13.0%

Table 6. Per epoch training time (seconds) on CIFAR-10. Here R-18 and R-110 represents ResNet-18 and ResNet-110, respectively.

Model	ResNet-50	ResNet-101
CutMix	41.5	68.6
CutMix + *low resolution*	49.8 +20.0%	83.5 +21.7%
CutMix + *early loss*	49.1 +18.3%	82.7 +20.6 %
AutoAugment	41.1	68.2
AutoAugment + *low resolution*	49.5 +20.4%	83.2 +21.9%
AutoAugment + *early loss*	48.9 +19.0%	82.4 +20.8%

Table 7. Average training time (minutes) per epoch.

proaches do not hurt the performance of the model. We also notice that, similar to the results on CIFAR-10, the proposed accelerating approach can speed up KeepAugment without loss of accuracy on ImageNet.

Training time cost We provide additional training cost comparisons on both CIFAR-10 and ImageNet in Table 6 and Table 7, respectively. On CIFAR-10, all models are trained on one TITAN X GPU with batch size 128; On ImageNet, we train all models on 8 TITAN X GPUs with batch size 384. As we can see from Figure 6, our *low resolution* and *early loss* based approximation significantly accelerates the computation of salience maps. Meanwhile, in general, our *KeepAugment* equipped with *low resolution* or *early loss* based saliency map approximation leads to ∼20% increase in per epoch training time compared to the corresponding baselines. To factor in this training overhead, in Table 8, we increase the training budget of CutMix and AutoAgument for additional 20%, from 300 epochs to 360 epochs. Comparing with our results in Table 5, our method still yields the best performance.

Method	Epochs	Top-1	Top-5
CutMix	360	78.72	94.15
AutoAugment	360	77.63	93.85

Table 8. Validation accuracy (%) on ImageNet using ResNet-50.

4.3. Semi-Supervised Learning

Semi-supervised learning (SSL) is a key approach toward more data-efficient machine learning by jointly leverage both labeled and unlabeled data. Recently, data augmentation has been shown a powerful tool for developing state-of-the-art SSL methods. Here, we apply the proposed method to unsupervised data augmentation [37] (UDA) on CIFAR-10 to verify whether our approach can be applied to more general applications.

	4000 labels	2500 labels
UDA + RandAug	95.1±0.2	91.2±1.0
UDA + KeepRandAug	**95.4±0.2**	**92.4±0.8**

Table 9. Result on CIFAR-10 semi-supervised learning. '4000 labels' denotes that 4,000 images have labels.

UDA minimizes the following loss on unlabelled data: $\mathbb{E}_{x\sim\mathcal{D}_u,\, x'\sim\mathcal{P}_x}\left[KL(p_\theta(\cdot\mid x)\ \|\ p_\theta(\cdot\mid x'))\right]$, where $\mathcal{P}$ denotes the randomized augmentation distribution, x' denotes an augmented image and θ denotes the neural network parameters. Notice that for semi-supervised learning, we do not have labels to calculate the saliency map. Instead, we use the max logit of $p_\theta(\cdot\mid x)$ to calculate the saliency map. We simply replace the RandAug [5] in UDA with our proposed approach, and use the WideResNet-28-2.

In Table 9, we show that our approach improves on RandAug and leads to improved semi-supervised learning performance on CIFAR-10.

4.4. Multi-View Multi-Camera Tracking

We apply our adaptive data augmentation to improve a state-of-the-art multi-view multi-camera tracking approach [23]. Recent works [e.g. 23, 45, 44] have shown that data augmentation is an effective technique for improving the performance on this task.

Settings [23] builds a strong baseline based on `Random Erasing` [45] data augmentation. `Random Erasing` is similar to `Cutout` , except filling the region dropped with random values instead of zeros. We improve over [23] by only cutting out regions with importance score smaller than $\tau = 0.6$. We denote the widely-used open-source baseline *open-ReID* [3] as the standard baseline in table 10. To ablate the role of our selective cutting-out strategy, we pursue minima changes made to the baseline code base. We follow all the training settings reported in [23], except using our adaptive data augmentation strategy. We use ResNet-101 as the backbone network.

We evaluate our method on a benchmark dataset, Market1501 [44]. Market1501 contains 32,668 bounding boxes of 1,501 identities, in which images of each identity are captured by at most six cameras.

Results We report test accuracy and mean average precision (mAP) of different methods in Table 10. Our method achieves the best performance. In particular, we achieve a 95.0% accuracy and 87.4 mAP on Market1501.

[3] https://github.com/Cysu/open-reid

Method	Market1501	
	Accuracy	mAP
Standard Baseline	88.1±0.2	74.6±0.2
+ Bag of Tricks [23]	94.5±0.1	87.1±0.0
+ Ours	**95.0±0.1**	**87.4±0.0**

Table 10. We compare our method with the standard and [23] on two benchmark datasets. *mAP* represents mean average precision.

Model	Backbone	Detectron2	Ours
Faster R-CNN	ResNet50-C4	38.4	**39.5**
Faster R-CNN	ResNet50-FPN	40.2	**40.7**
RetinaNet	ResNet50-FPN	37.9	**39.1**
Faster R-CNN	ResNet101-C4	41.1	**42.2**
Faster R-CNN	ResNet101-FPN	42.0	**42.9**
RetinaNet	ResNet101-FPN	39.9	**41.2**

Table 11. Detection mean Average Precision (mAP) Results on COCO 2017 validation set. (mAP%) is reported for comparsion.

4.5. Transfer Learning: Object Detection

We demonstrate the transferability of our ImageNet pre-trained models on the COCO 2017 [21] object detection task, on which we observe significant improvements over strong `Detectron2` [35] baselines by simply applying our pre-trained models as backbones.

Dataset and Settings COCO 2017 consists of 118,000 training images and 5,000 validation images. To verify that our trained models can be widely useful for different detector systems, we test several popular structures, including Faster RCNN [26], feature pyramid networks [19] (FPN) and RetinaNet [20]. We use the codebase provided by `Detectron2` [35], follow almost all the hyper-parameters except changing the backbone networks from `PyTorch` provided models to our models. For our method, we test the ResNet-50 and ResNet-101 models trained with our `KeepCutMix`.

Results We report mean average precision (mAP) on the COCO 2017 validation set [21]. As we can see from Table 11, our method consistently improves over baseline approaches. Simply replacing the backbone network with our pre-trained model gives performance gains for the COCO 2017 object detection tasks with no additional cost. In particular, on the single-stage detector RetinaNet, we improve the 37.9 mAP to 39.1, and 39.9 mAP to 41.2 for ResNet-50 and ResNet-101, respectively.

5. Related Works

Our work is most related to [12], which studies the impact of affinity (or fidelity) and diversity of data augmentation empirically, and finds out that a good data augmentation strategy should jointly optimize these two aspects. Recently, many other works also show the importance of balancing between fidelity and diversity. For example, [11] and [43] show that optimize the worst case or choose the most difficult augmentation policy is helpful, which indicates the importance of diversity. [34] considers to correct the *label* of noisy augmented examples by using a teacher network, thus increasing fidelity. This approach also needs additional supervision and only focus on one typical data augmentation method. Compared to these works, our augmentation improves on stronger data augmentation by preserving informative regions, thus naturally achieve fidelity and diversity. It allows us to train better models by leveraging more diversified faithful examples.

Our work focus on improving label-invariant data augmentation. Another line of data augmentation schemes create augmented examples by mixing both images and their corresponding labels, exemplified by *mixup* [41], Manifold Mixup [32], CutMix [39]. It is not clear how to quantify noisy examples for label-mixing augmentation since labels are also mixed, nevertheless we show empirically that our *selective-cut* also improves on `CutMix` and leave further extensions as our future work.

The idea of using saliency map for improving computer vision systems have been widely explored in the literature. Saliency map can be applied to object detection [42], segmentation [24], knowledge distillation [2] and many more [e.g. 2, 27]. We propose to use the saliency map to measure the relative importance of different regions, thus improving regional-level cutting-based data augmentation by avoiding informative regions; or improving image-level augmentation techniques by pasting-back discriminative regions.

6. Conclusion

In this work, we empirically show that prior art data augmentation schemes might introduce noisy training examples and hence limit their ability in boosting the overall performance. Thus we use saliency map to measure the importance of each region, and propose to avoid cutting important regions for region-level data augmentation approaches, such as `Cutout` ; or pasting back critical areas from the clean data for image-level data augmentation, like `RandAugment` and `AutoAugment` . Throughout an extensive evaluation, we have demonstrated that our adaptive augmentation approach helps to significantly improve the performance of image classification, multi-view multi-camera tracking and object detection.

Acknowledgement The work is supported in part by NSF CAREER-1846421, SenSE-2037267, EAGER-2041327 and NSF AI Institute for Foundations of Machine Learning (IFML). We would like to thank the reviewers for their thoughtful comments and efforts towards improving our manuscript.

References

[1] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *NeurIPS*, pages 5050–5060, 2019. 1

[2] Alvin Chan, Yi Tay, and Yew-Soon Ong. What it thinks is important is important: Robustness transfers through input gradients. In *CVPR*, 2020. 8

[3] Pengguang Chen. Gridmask data augmentation. *arXiv preprint arXiv:2001.04086*, 2020. 6

[4] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. In *CVPR*, 2019. 1, 2, 5, 6

[5] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical data augmentation with no separate search. In *CVPR*, 2020. 1, 2, 3, 5, 6, 7

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. IEEE, 2009. 2, 6

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019. 1

[8] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 1, 2, 3, 5

[9] Xavier Gastaldi. Shake-shake regularization. *arXiv preprint arXiv:1705.07485*, 2017. 5

[10] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V Le. Dropblock: A regularization method for convolutional networks. In *NeurIPS*, pages 10727–10737, 2018. 6

[11] Chengyue Gong, Tongzheng Ren, Mao Ye, and Qiang Liu. Maxup: A simple way to improve generalization of neural network training. *arXiv preprint arXiv:2002.09024*, 2020. 8

[12] Raphael Gontijo-Lopes, Sylvia J Smullin, Ekin D Cubuk, and Ethan Dyer. Affinity and diversity: Quantifying mechanisms of data augmentation. *arXiv preprint arXiv:2002.08973*, 2020. 2, 8

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 5, 6

[14] Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. In *ICLR*, 2020. 6

[15] Alex Krizhevsky and Geoff Hinton. Convolutional deep belief networks on cifar-10. 40(7):1–9, 2010. 5

[16] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 2

[17] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Fractalnet: Ultra-deep neural networks without residuals. In *ICLR*, 2017. 6

[18] Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast autoaugment. In *NeurIPS*, pages 6662–6672, 2019. 1, 2

[19] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 2117–2125, 2017. 8

[20] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pages 2980–2988, 2017. 8

[21] 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 *ECCV*, pages 740–755. Springer, 2014. 1, 8

[22] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *ICLR*, 2016. 5

[23] Hao Luo, Youzhi Gu, Xingyu Liao, Shenqi Lai, and Wei Jiang. Bag of tricks and a strong baseline for deep person re-identification. In *CVPRW*, pages 0–0, 2019. 7, 8

[24] Prerana Mukherjee, Brejesh Lall, and Archit Shah. Saliency map based improved segmentation. In *ICIP*, pages 1290–1294. IEEE, 2015. 8

[25] Daniel S Park, Yu Zhang, Ye Jia, Wei Han, Chung-Cheng Chiu, Bo Li, Yonghui Wu, and Quoc V Le. Improved noisy student training for automatic speech recognition. *arXiv preprint arXiv:2005.09629*, 2020. 1

[26] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, pages 91–99, 2015. 8

[27] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. 8

[28] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *NIPS*, 2013. 1, 3

[29] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *arXiv preprint arXiv:2001.07685*, 2020. 1

[30] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, pages 1929–1958, 2014. 6

[31] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, pages 1–9, 2015. 4

[32] Vikas Verma, Alex Lamb, Christopher Beckham, Aaron Courville, Ioannis Mitliagkis, and Yoshua Bengio. Manifold mixup: Encouraging meaningful on-manifold interpolation as a regularizer. *ICML*, 2019. 6, 8

[33] Devesh Walawalkar, Zhiqiang Shen, Zechun Liu, and Marios Savvides. Attentive cutmix: An enhanced data augmentation approach for deep learning based image classification. In *ICASSP*, pages 3642–3646. IEEE, 2020. 6

[34] Longhui Wei, An Xiao, Lingxi Xie, Xin Chen, Xiaopeng Zhang, and Qi Tian. Circumventing outliers of autoaugment with knowledge distillation. *arXiv preprint arXiv:2003.11342*, 2020. 2, 8

[35] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2, 2019. 1, 8

[36] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Unsupervised data augmentation. *arXiv*

preprint arXiv:1904.12848, 2019. 1

[37] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Unsupervised data augmentation. *arXiv preprint arXiv:1904.12848*, 2019. 7

[38] Yoshihiro Yamada, Masakazu Iwamura, Takuya Akiba, and Koichi Kise. Shakedrop regularization for deep residual learning. *arXiv preprint arXiv:1802.02375*, 2018. 1, 5

[39] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, 2019. 1, 5, 6, 8

[40] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016. 5

[41] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 6, 8

[42] Jianming Zhang and Stan Sclaroff. Saliency detection: A boolean map approach. In *CVPR*, pages 153–160, 2013. 8

[43] Xinyu Zhang, Qiang Wang, Jian Zhang, and Zhao Zhong. Adversarial autoaugment. In *ICLR*, 2020. 8

[44] Liang Zheng, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, and Qi Tian. Scalable person re-identification: A benchmark. In *ICCV*, pages 1116–1124, 2015. 1, 7

[45] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. *arXiv preprint arXiv:1708.04896*, 2017. 2, 5, 6, 7