KeepAugment: A Simple Information-Preserving Data Augmentation Approach

Chengyue Gong\textsuperscript{1}, Dilin Wang\textsuperscript{2}, Meng Li\textsuperscript{2}, Vikas Chandra\textsuperscript{2}, Qiang Liu\textsuperscript{1}

\textsuperscript{1}University of Texas, Austin \quad \textsuperscript{2}Facebook Inc

Abstract

Data augmentation (DA) is an essential technique for training state-of-the-art deep learning systems. In this paper, we empirically show data augmentation might introduce noisy augmented examples and consequently hurt the performance on unaugmented data during inference. To alleviate this issue, we propose a simple yet highly effective approach, dubbed KeepAugment, to increase augmented images fidelity. The idea is first to use the saliency map to detect important regions on the original images and then preserve these informative regions during augmentation. This information-preserving strategy allows us to generate more faithful training examples. Empirically, we demonstrate our method significantly improves on 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 proving 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 not resource-consuming, the empirical results on a variety of vision tasks show that we can significantly improve prior art data augmentation (DA) baselines. Specifically, for image classification, we achieve substantial 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].
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 \( x' = A(x) \) that are expected to have the same label as \( x \), where \( A \) 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 \( M_{ij}(S) = 1 \) if \((i, j) \in S\). 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 \( \delta \) 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-
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

\[
\mathcal{I}(S, x, y) = \sum_{(i, j) \in S} g_{ij}(x, y). \tag{1}
\]

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 RGB-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 rectangular 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 \( \tau \). The corresponding augmented example is defined as follows,

\[
\hat{x} = (1 - M(S)) \odot x, \tag{2}
\]

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

**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 \( \tau \), and and paste the region \( S \) of the original image \( x \) to \( x' \), which yields

\[
\hat{x} = M(S) \odot x + (1 - M(S)) \odot x'. \tag{3}
\]

Similarly, \( M_{ij}(S) = I((i, j) \in S) \) 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 \( \tau \)

(a) if use Selective-Cut

repeat randomly select a mask region \( S \) until region score \( \mathcal{I}(S, x, y) < \tau \)

\[
\hat{x} = (1 - M(S)) \odot x \quad (\text{see Eq. 2})
\]

(b) if use Selective-paste

repeat randomly select a mask region \( S \) until region score \( \mathcal{I}(S, x, y) > \tau \)

\[
\hat{x} = M(S) \odot x + (1 - M(S)) \odot x' \quad (\text{see Eq. 3})
\]

**Return** \( \hat{x} \)

**Remark** In practice, we choose our threshold \( \tau \) 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 \( \tau \)-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 \( \tau \) on CIFAR-10, illustrated in Figure 3. Intuitively, for \textit{selective-cut}, it’s more likely to cut out important regions as we use an increasingly larger threshold \( \tau \); on the contrary, a larger \( \tau \) corresponds to copy back more critical regions for \textit{selective-paste}. As we can see from Figure 3, for \textit{Cutout} (Figure 3 (a)), we improve on the standard \textit{Cutout} baseline (dash line) significantly when the threshold \( \tau \) is relatively 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 relatively large threshold \( \tau \) (e.g., \( \tau = 0.8 \)); for \textit{ RandAugment} (Figure 3 (b)), using a lower threshold (e.g., \( \tau = 0.2 \)) tends to yield similar performance as the standard \textit{ RandAugment} baseline (dash line). Increasing the threshold \( \tau \) (\( \tau = 0.6 \) or \( \tau = 0.8 \)) yields better results. We notice that further increasing \( \tau \) (\( \tau = 0.8 \)) may hurt the performance slightly, likely because a large threshold yields too restrictive selection and may miss other informative regions.

### 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 roughly 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 3X computation cost reduction by reducing the resolution from 224 to 112.

**Early 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 3X computation cost reduction when calculate saliency map.

Furthermore, in the experiment part, we show that both approximation strategies do not lead to any performance drop.

![Diagram](image)

**Figure 4.** We demonstrate two different approaches for using KeepAugment with less training time. Using \textit{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 \textit{KeepAugment} significantly improves on existing state-of-the-art data augmentation baselines on a variety of challenging deep learning tasks, including image classification, semi-supervised image classification, multi-view multi-camera tracking, and object detection. For
| 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          |

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

| 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 2. Test accuracy (%) of ResNet-18 on CIFAR-10 across varying cut-out length. All results are averaged over 5 random trials.

| 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 across varying cut-out length. All results are averaged over 5 random trials.

Training Time Cost Table 3 also displays that, the two approaches with less time cost do not hurt the performance on classification accuracy. The accuracy of the low resolution approach is sometimes even better than the original algorithm. As shown in Table 3, the original approach (Keep-Cutout) almost doubles the training cost of Cutout. By using low-resolution image or early loss head to calculate the saliency map, the training cost is only slightly larger than Cutout, while the performance is improved.

4.1. CIFAR-10 Classification

We apply our adaptive selective strategy to improve two state-of-the-art augmentation schemes, Cutout and RandAugment, on the CIFAR-10 [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 decay [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.

From Table 1, we observe a consistent improvement on test accuracy by applying our information-preserving augmentation strategy. In particular, to the best of our knowledge, our results establish a new state-of-the-art test accuracy 98.7% using PyramidNet+Shakedrop among baselines without using extra training data (e.g. ImageNet pretraining).

Improve on CutOut We study the relative improvements on Cutout across various cutout lengths. We choose 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.

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

**Additional Comparisons on CIFAR-10** Recently, some researchers [3, 33, 14] also mix the clean image and augmented image together to achieve higher performance. Gridmask [3], AugMix [14] and Attentive CutMix are popular methods among these approaches. (Please refer to Section 5 for more details about CutMix.) Here, we do 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 5 shows that the proposed algorithm can achieve clear improvement on accuracy over all the baselines. Using the low resolution image to calculate the saliency map, our method does not introduce additional time cost to the training process. On the other hand, while Gridmask only implements upon Cutout and Attentive CutMix only implements upon CutMix by pasting the most important region, our approach is more flexible and can be applied to other data augmentation approaches.

### 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.

| Method                  | ResNet-50 |       |       |
|-------------------------|-----------|-------|-------|
|                         | 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  |
| Mixup [41]              | 77.9      | 93.9  | 79.2  | 94.4  |
| ShakeShake [10]         | 78.3      | 94.1  | 79.0  | 94.3  |
| RandAugment [5]         | 77.6      | 93.8  | 79.2  | 94.4  |
| AutoAugment [4]         | 77.6      | 93.8  | 79.3  | 94.4  |
| + 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. 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.

| Method                  | ResNet-50 |       |       |
|-------------------------|-----------|-------|-------|
|                         | 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  |
| 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  |
| ShakeShake [10]         | 78.3      | 94.1  | 79.0  | 94.3  |
| DropPath [17]           | 77.1      | 93.5  | -     | -     |
| Mixup [41]              | 77.5      | 93.8  | -     | -     |
| + Low Resolution        | 77.9      | 93.9  | 79.2  | 94.4  |
| + 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 6. Validation set Top-1 and Top-5 accuracy (%) on ImageNet using ResNet-50 and ResNet-101.
improve on CutMix by using selective-cut. In practice, we found it is often quite effective to simply avoiding cutting informative region from x, as we observe the mixing rate $\lambda$ sampled is often close to 1. 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.

Results We report the single-crop top-1 and top-5 accuracy on the validation set in table 6. 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 approaches 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.

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.

UDA minimizes the following loss on unlabelled data:

$$\mathbb{E}_{x \sim D_u, x' \sim \mathcal{P}} \left[ KL(p_{\theta}(\cdot | x) \parallel p_{\theta}(\cdot | x')) \right],$$

where $\mathcal{P}$ denotes the randomized augmentation distribution, $x'$ denotes an augmented image and $\theta$ 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 $p_{\theta}(\cdot | x)$ to calculate the saliency map. We simply replace the RandAugment [5] in UDA with our proposed approach, and use the WideResNet-28-2.

Table 7 shows that our approach can consistently improve the UDA with different number of labelled image 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 3.

Table 7. Result on CIFAR-10 semi-supervised learning. '4000 labels' denotes that 4,000 images have labels while the other 4,6000 do not.

| Method                      | 4000 labels | 2500 labels |
|-----------------------------|-------------|-------------|
| UDA + RandAug               | 95.1±0.2    | 91.2±1.0    |
| UDA + KeepRandAug           | 95.4±0.4    | 92.4±0.8    |

Table 8. We compare our method with the standard and [23] on two benchmark datasets. mAP represents mean average precision.

| 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 8. We compare our method with the standard and [23] on two benchmark datasets. mAP represents mean average precision.

Dataset We evaluate our method on a widely used 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 the accuracy and mean average precision (mAP) of different methods in Table 8. Our method achieves the best performance on both datasets. In particular, we achieve a 95.0% accuracy and 87.4 mAP on Market1501.

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.

https://github.com/clovaai/CutMix-PyTorch

https://github.com/Cysu/open-reid
### Dataset and Settings
We use the COCO 2017 challenge, which 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 9, 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, while the pretrained model processed by CutMix cannot even match the performance of the Detectron2 baselines. ResNet-50 + CutMix achieves 23.7, 27.1 and 25.4 (mAP) in our C4, FPN and RetinaNet settings, respectively. These results are much worse compared to our results.

### 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 optimizing the worst case or choose the most difficult augmentation policy is helpful, which indicates the importance of diversity.

[33] also considers to extract the most informative region to reduce noise in CutMix. Their proposed method, Attentive CutMix needs an additional pretrained classification model to serve as a teacher, while our approach does not need additional supervision. Moreover, our approach is much more general and can be applied to more augmentation methods and tasks (e.g. semi-supervision) which Attentive CutMix cannot. [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 to measure the relative importance of different regions of the input, 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 state-of-the-art architectures on CIFAR-10, ImageNet, multi-view multi-camera tracking and object detection.
[37] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V. Le. Unsupervised data augmentation. *arXiv preprint arXiv:1904.12848*, 2019.

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

[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.

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

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

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

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

[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.

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