吉林大学

学习汇报

汇 报 人： 高金彤

2023.1.6

2022 NeurIPS

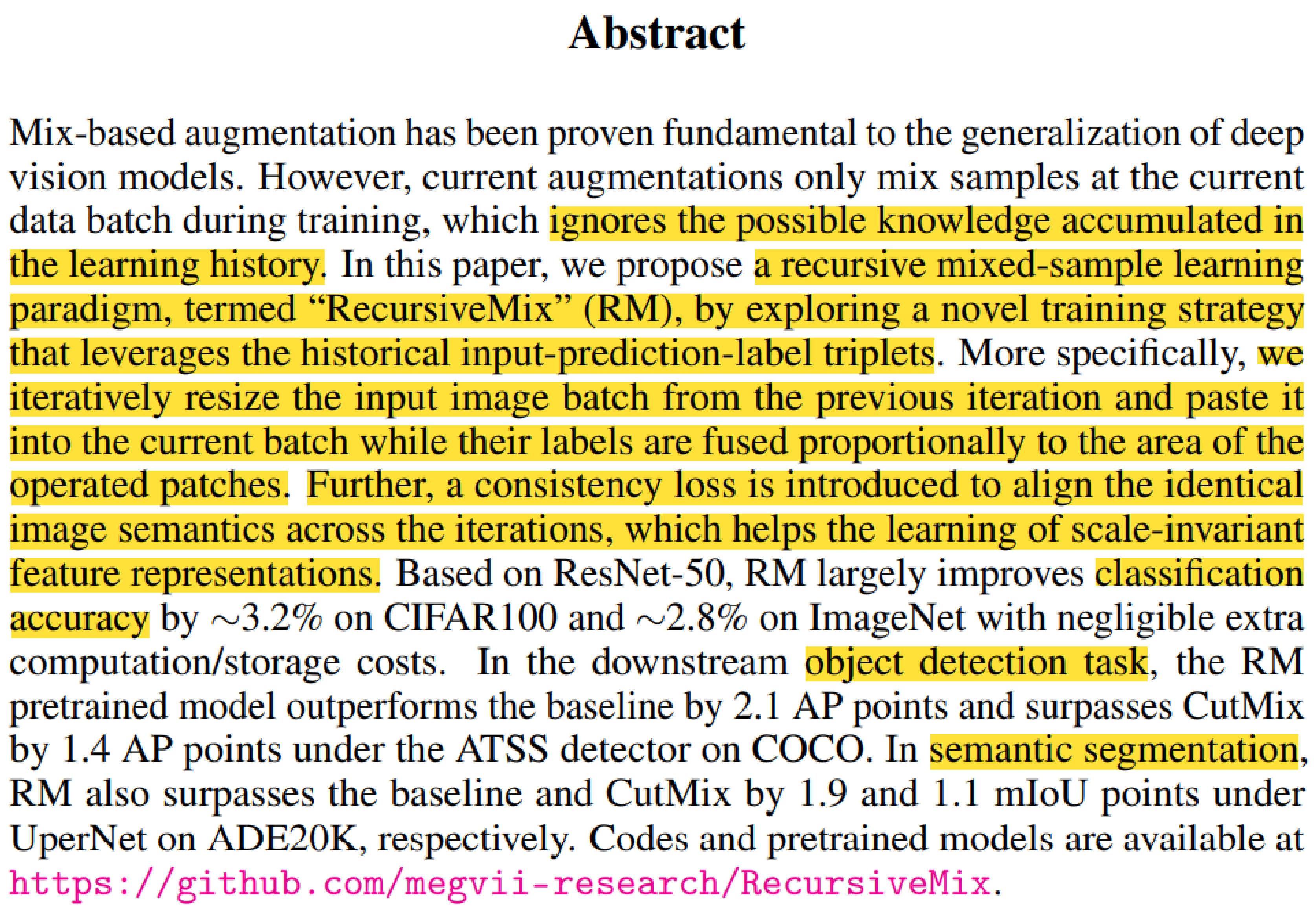
RecursiveMix: Mixed Learning with History

Lingfeng Yang[1#], Xiang Li[1#], Borui Zhao[2], Renjie Song[2], Jian Yang[1*]

[1]Nanjing University of Science and Technology, [2]Megvii Technology

`{yanglfnjust, xiang.li.implus, csjyang}@njust.edu.cn`
`zhaoborui.gm@gmail.com, songrenjie@megvii.com`

Abstract



Abstract

Mix-based augmentation has been proven fundamental to the generalization of deep vision models. However, current augmentations only mix samples at the current data batch during training, which ignores the possible knowledge accumulated in the learning history. In this paper, we propose a recursive mixed-sample learning paradigm, termed "RecursiveMix" (RM), by exploring a novel training strategy that leverages the historical input-prediction-label triplets. More specifically, we iteratively resize the input image batch from the previous iteration and paste it into the current batch while their labels are fused proportionally to the area of the operated patches. Further, a consistency loss is introduced to align the identical image semantics across the iterations, which helps the learning of scale-invariant feature representations. Based on ResNet-50, RM largely improves classification accuracy by $\sim$3.2% on CIFAR100 and $\sim$2.8% on ImageNet with negligible extra computation/storage costs. In the downstream object detection task, the RM pretrained model outperforms the baseline by 2.1 AP points and surpasses CutMix by 1.4 AP points under the ATSS detector on COCO. In semantic segmentation, RM also surpasses the baseline and CutMix by 1.9 and 1.1 mIoU points under UperNet on ADE20K, respectively. Codes and pretrained models are available at `https://github.com/megvii-research/RecursiveMix`.

Three benefits of the proposed RecursiveMix

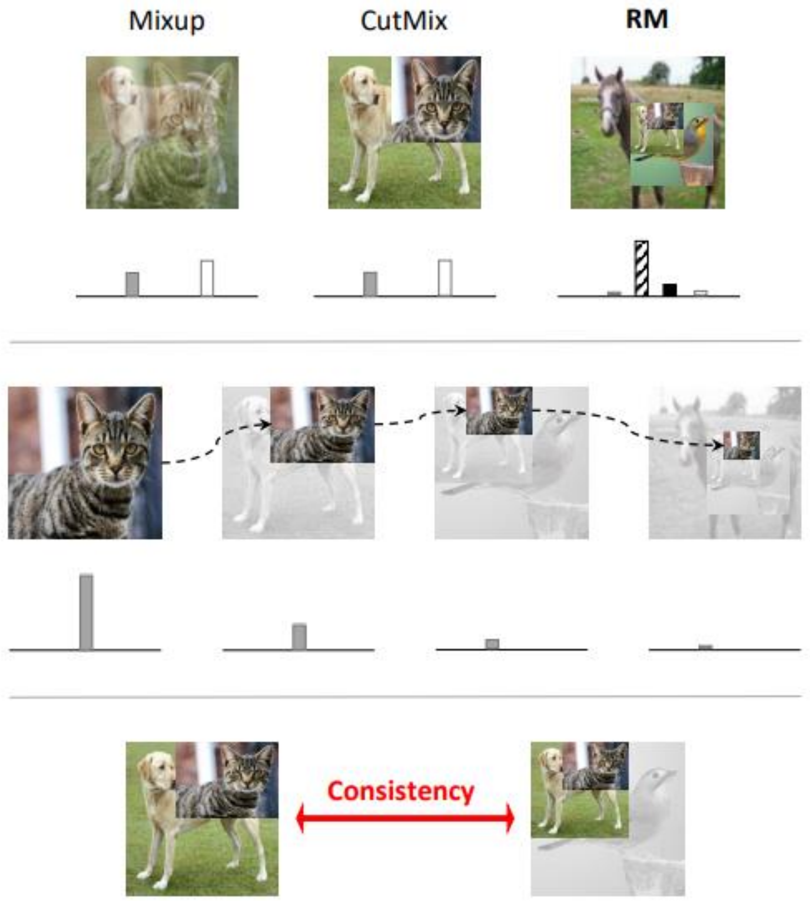
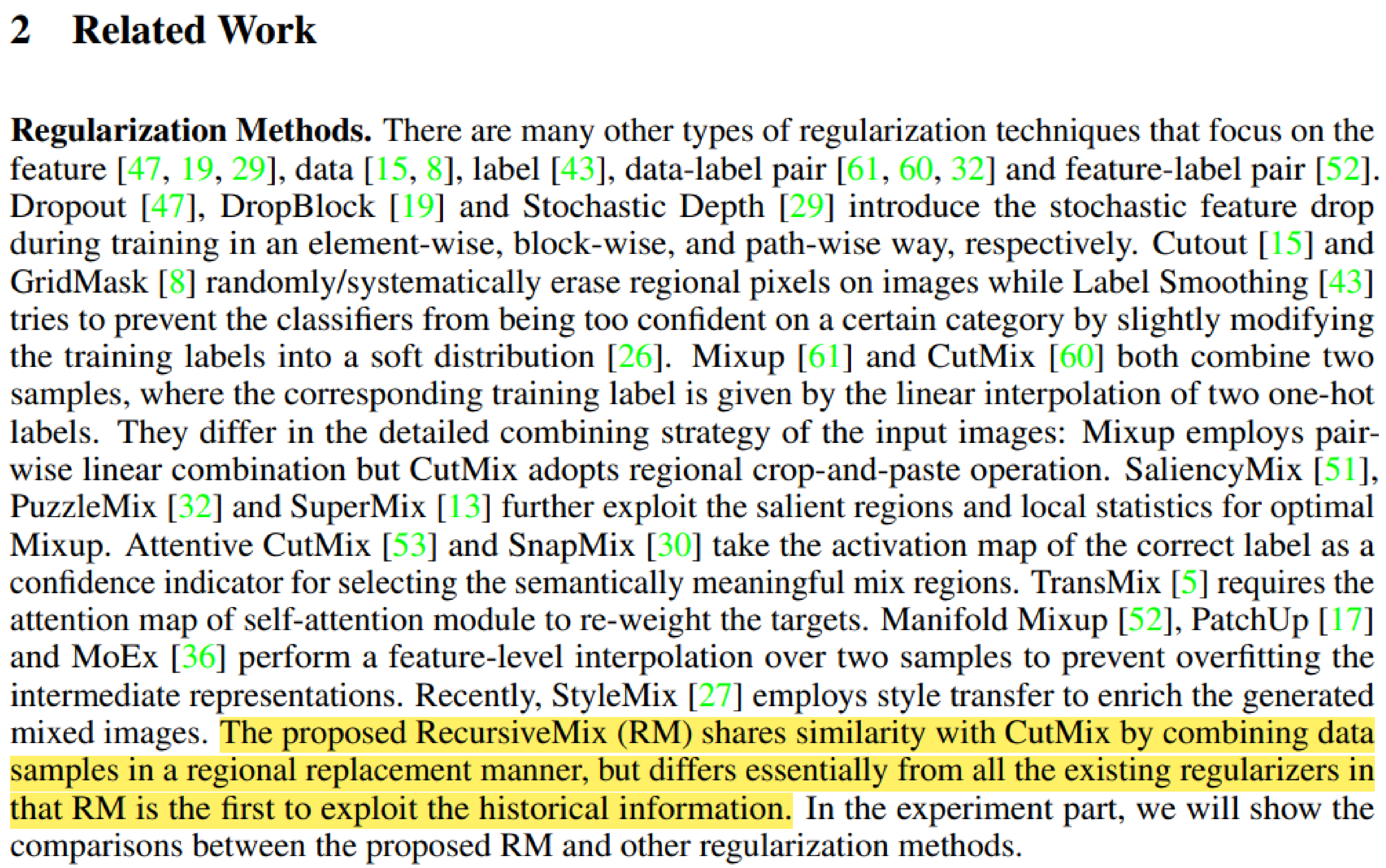


Figure 2: Three benefits of the proposed RecursiveMix, which iteratively leverages the historical input-prediction-label triplets: 1) an enlarged diversity of data space and richer supervisions, 2) adequate training signals for an instance with multi-scale/-space views, and 3) explicit learning on the spatial semantic consistency which can further benefit the downstream tasks.

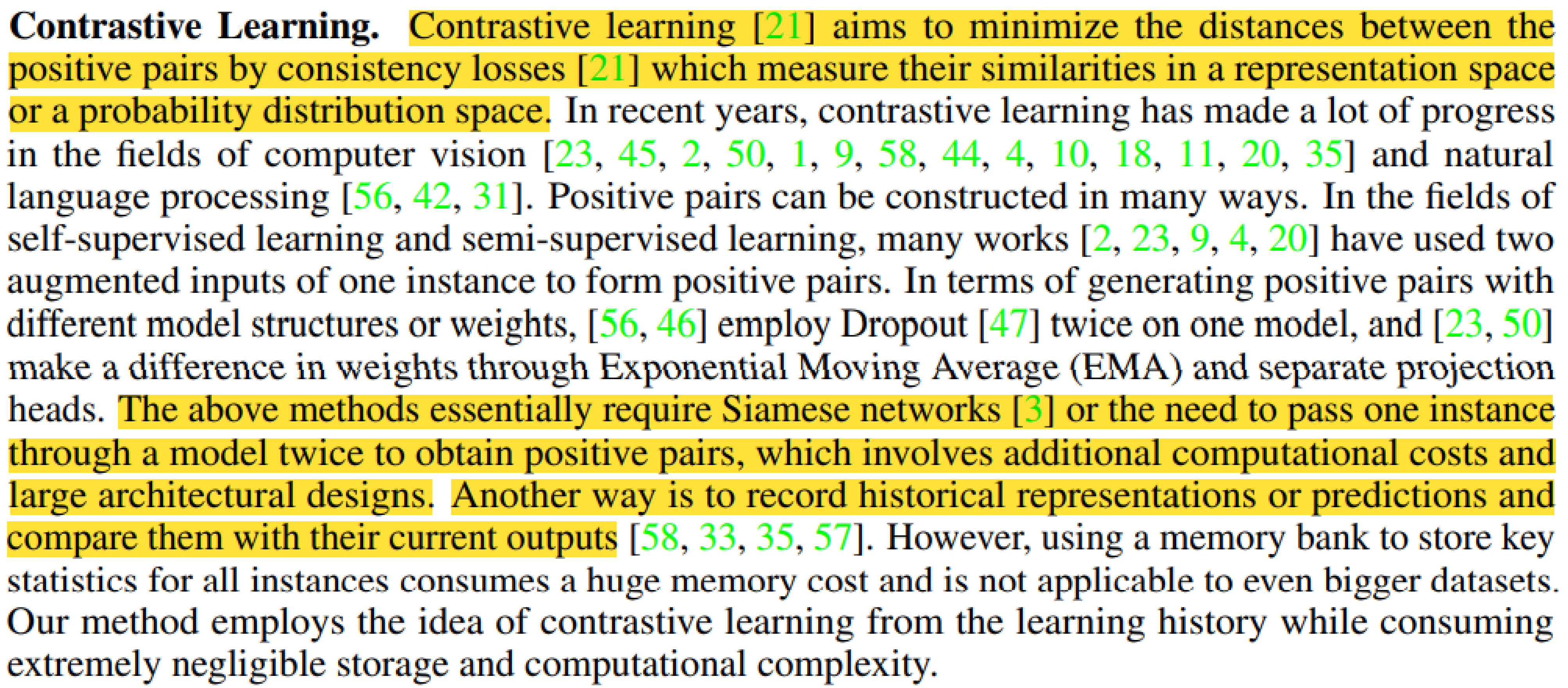
2 Related Work



2 Related Work

Regularization Methods. There are many other types of regularization techniques that focus on the feature [47, 19, 29], data [15, 8], label [43], data-label pair [61, 60, 32] and feature-label pair [52]. Dropout [47], DropBlock [19] and Stochastic Depth [29] introduce the stochastic feature drop during training in an element-wise, block-wise, and path-wise way, respectively. Cutout [15] and GridMask [8] randomly/systematically erase regional pixels on images while Label Smoothing [43] tries to prevent the classifiers from being too confident on a certain category by slightly modifying the training labels into a soft distribution [26]. Mixup [61] and CutMix [60] both combine two samples, where the corresponding training label is given by the linear interpolation of two one-hot labels. They differ in the detailed combining strategy of the input images: Mixup employs pair-wise linear combination but CutMix adopts regional crop-and-paste operation. SaliencyMix [51], PuzzleMix [32] and SuperMix [13] further exploit the salient regions and local statistics for optimal Mixup. Attentive CutMix [53] and SnapMix [30] take the activation map of the correct label as a confidence indicator for selecting the semantically meaningful mix regions. TransMix [5] requires the attention map of self-attention module to re-weight the targets. Manifold Mixup [52], PatchUp [17] and MoEx [36] perform a feature-level interpolation over two samples to prevent overfitting the intermediate representations. Recently, StyleMix [27] employs style transfer to enrich the generated mixed images. The proposed RecursiveMix (RM) shares similarity with CutMix by combining data samples in a regional replacement manner, but differs essentially from all the existing regularizers in that RM is the first to exploit the historical information. In the experiment part, we will show the comparisons between the proposed RM and other regularization methods.

2 Related Work



Contrastive Learning. Contrastive learning [21] aims to minimize the distances between the positive pairs by consistency losses [21] which measure their similarities in a representation space or a probability distribution space. In recent years, contrastive learning has made a lot of progress in the fields of computer vision [23, 45, 2, 50, 1, 9, 58, 44, 4, 10, 18, 11, 20, 35] and natural language processing [56, 42, 31]. Positive pairs can be constructed in many ways. In the fields of self-supervised learning and semi-supervised learning, many works [2, 23, 9, 4, 20] have used two augmented inputs of one instance to form positive pairs. In terms of generating positive pairs with different model structures or weights, [56, 46] employ Dropout [47] twice on one model, and [23, 50] make a difference in weights through Exponential Moving Average (EMA) and separate projection heads. The above methods essentially require Siamese networks [3] or the need to pass one instance through a model twice to obtain positive pairs, which involves additional computational costs and large architectural designs. Another way is to record historical representations or predictions and compare them with their current outputs [58, 33, 35, 57]. However, using a memory bank to store key statistics for all instances consumes a huge memory cost and is not applicable to even bigger datasets. Our method employs the idea of contrastive learning from the learning history while consuming extremely negligible storage and computational complexity.

3.1 RecursiveMix

We attempt to exploit the historical input-prediction-label triplets of a learning instance to improve the generalization of vision models. A practical idea is to reuse the past input-prediction-label triplets to construct diversely mixed training samples with the current batch (see Fig. 1). To be specific, the historical input images are resized and then filled into the current ones where the labels are fused proportionally to the area of the mixed patches. Interestingly, such an operation formulates an exact recursive paradigm where each instance can have a gradually shrunken size of view during training, thus we term the method "RecursiveMix" (RM). Specifically at iteration t, the preliminary of RM is to generate a new training sample $(\widetilde{x}^t, \widetilde{y}^t)$ by combining the current sample (x^t, y^t) and the past historical one (x^h, y^h). The new training sample $(\widetilde{x}^t, \widetilde{y}^t)$ participates in training with the original

loss objectives $\mathcal{L}_{CE}$ and then updates the historical pair (x^h, y^h) iteratively:

$$\begin{aligned}
\widetilde{x}^t &= \mathbf{M} \odot \text{ResizeFill}(x^h, \mathbf{M}) + (\mathbf{1} - \mathbf{M}) \odot x^t, \\
\widetilde{y}^t &= \lambda^t y^h + (1 - \lambda^t) y^t, \\
x^h &= \widetilde{x}^t, \\
y^h &= \widetilde{y}^t,
\end{aligned} \tag{1}$$

The illustration of the proposed RecursiveMix

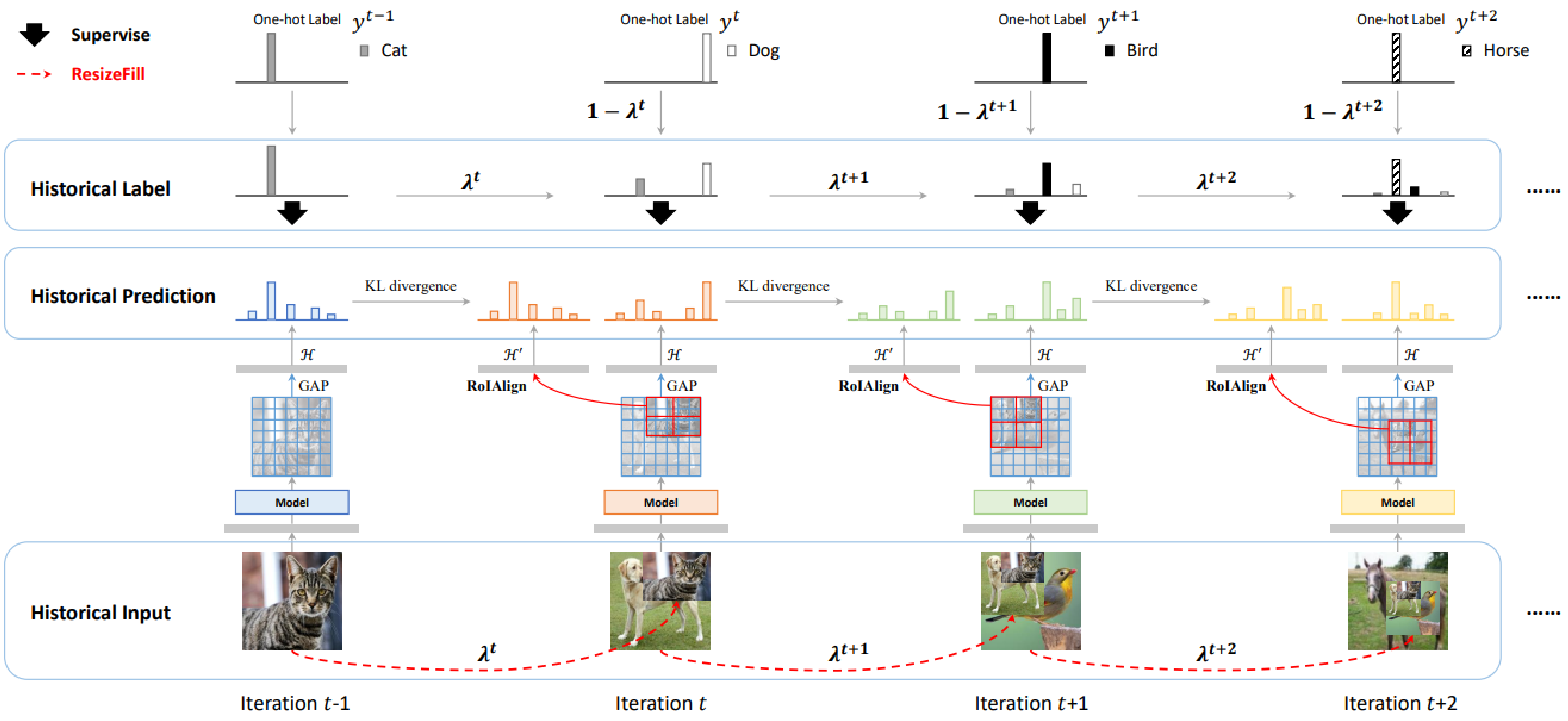


Figure 1: The illustration of the proposed RecursiveMix which leverages the historical input-prediction-label triplets. The historical input images are resized and then mixed into the current ones where the labels are fused proportionally (i.e., λ and $1-\lambda$) to the area of the operated patches, formulating a recursive paradigm.

3.1 RecursiveMix

where $\mathbf{M} \in \{0,1\}^{W \times H}$ is the binary mask indicating the place for the historical input to fill, $\mathbf{1}$ is the binary mask filled with ones and $\odot$ is the element-wise product. The combination ratio λ (λ^t indicates λ at moment t) is sampled from the uniform distribution $U[0, \alpha]$ where α defaults to 0.5. This setting is different from the symmetric beta distribution of Mixup [61] and CutMix [60]. Because in our case, λ denotes the proportion of the historical images, which is not suggested to be quite large. Otherwise, the information of the new data points has a risk of being discarded. The function $\text{ResizeFill}(x^h, \mathbf{M})$ means resizing the image x^h and filling it back exactly into its rectangle sub-region where $\mathbf{M} = 1$. The resized height and width are also determined by the rectangle area. This design is to make the recursive mechanism reasonable since the original cut operation in CutMix [60] could possibly lose former information, which would cause the inconsistency problem with the label (see Fig. 3).

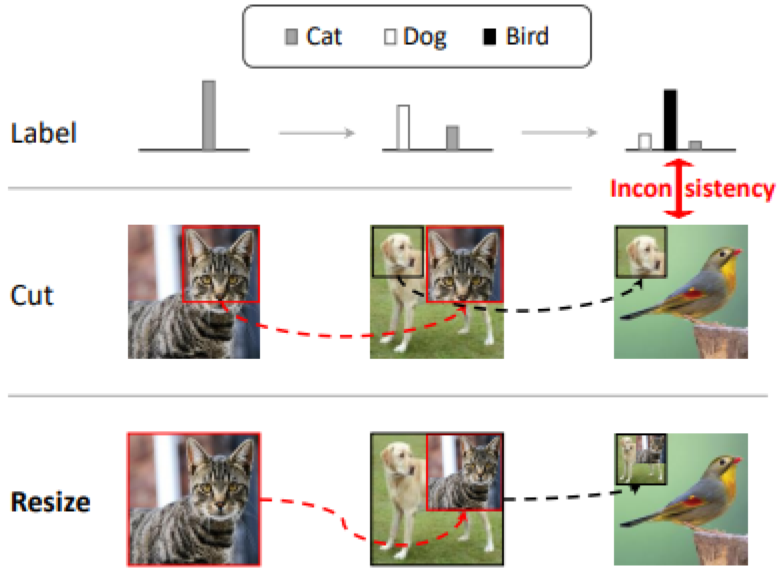
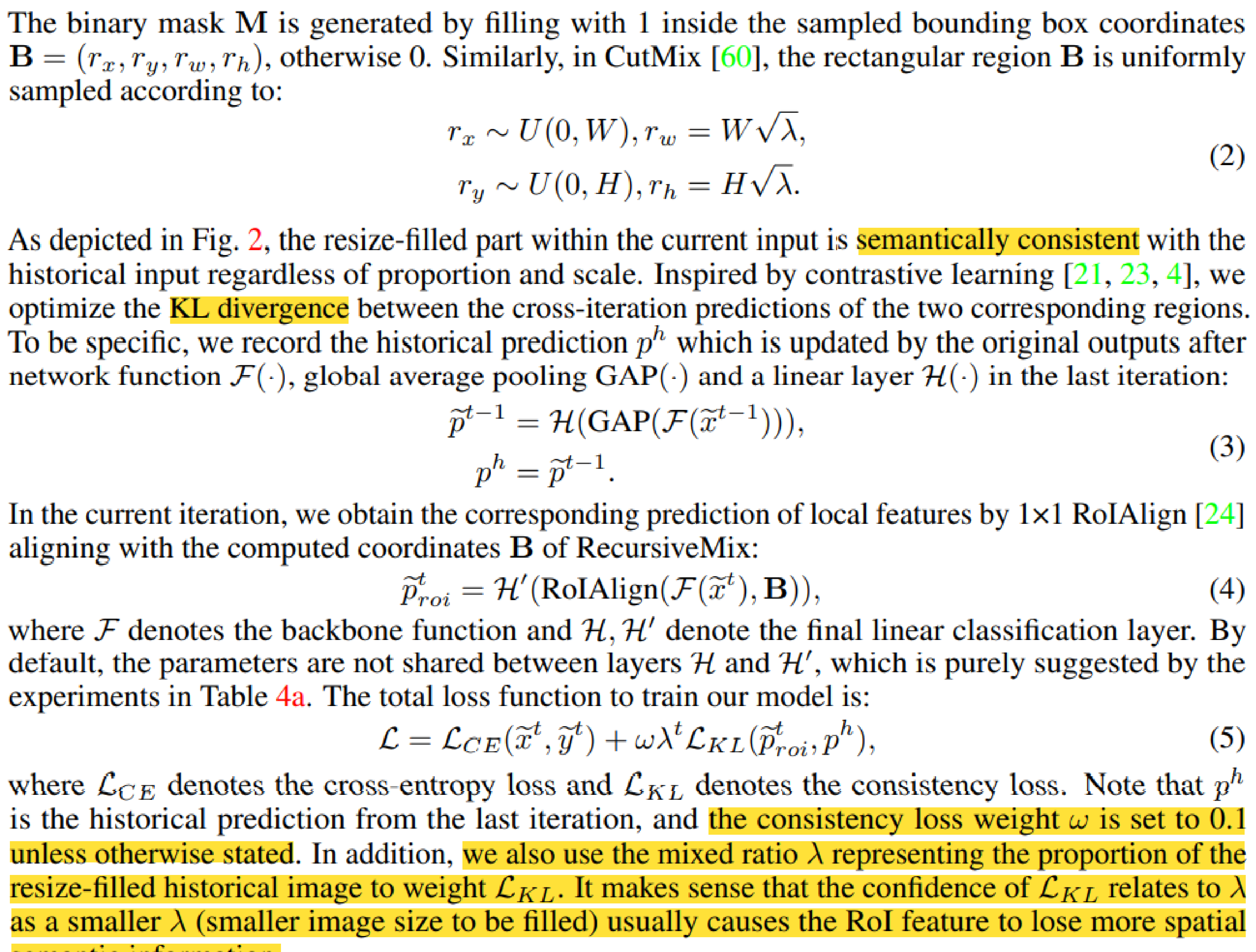


Figure 3: "Cut" operation may lead to the inconsistency between input and label under the proposed historical operation, but "Resize" can correctly preserve the consistency.

3.1 RecursiveMix



The binary mask $\mathbf{M}$ is generated by filling with 1 inside the sampled bounding box coordinates $\mathbf{B} = (r_x, r_y, r_w, r_h)$, otherwise 0. Similarly, in CutMix [60], the rectangular region $\mathbf{B}$ is uniformly sampled according to:

$$\begin{aligned} r_x \sim U(0, W), r_w = W\sqrt{\lambda}, \\ r_y \sim U(0, H), r_h = H\sqrt{\lambda}. \end{aligned} \tag{2}$$

As depicted in Fig. 2, the resize-filled part within the current input is semantically consistent with the historical input regardless of proportion and scale. Inspired by contrastive learning [21, 23, 4], we optimize the KL divergence between the cross-iteration predictions of the two corresponding regions. To be specific, we record the historical prediction p^h which is updated by the original outputs after network function $\mathcal{F}(\cdot)$, global average pooling $\mathrm{GAP}(\cdot)$ and a linear layer $\mathcal{H}(\cdot)$ in the last iteration:

$$\begin{aligned} \widetilde{p}^{t-1} &= \mathcal{H}(\mathrm{GAP}(\mathcal{F}(\widetilde{x}^{t-1}))), \\ p^h &= \widetilde{p}^{t-1}. \end{aligned} \tag{3}$$

In the current iteration, we obtain the corresponding prediction of local features by 1×1 RoIAlign [24] aligning with the computed coordinates $\mathbf{B}$ of RecursiveMix:

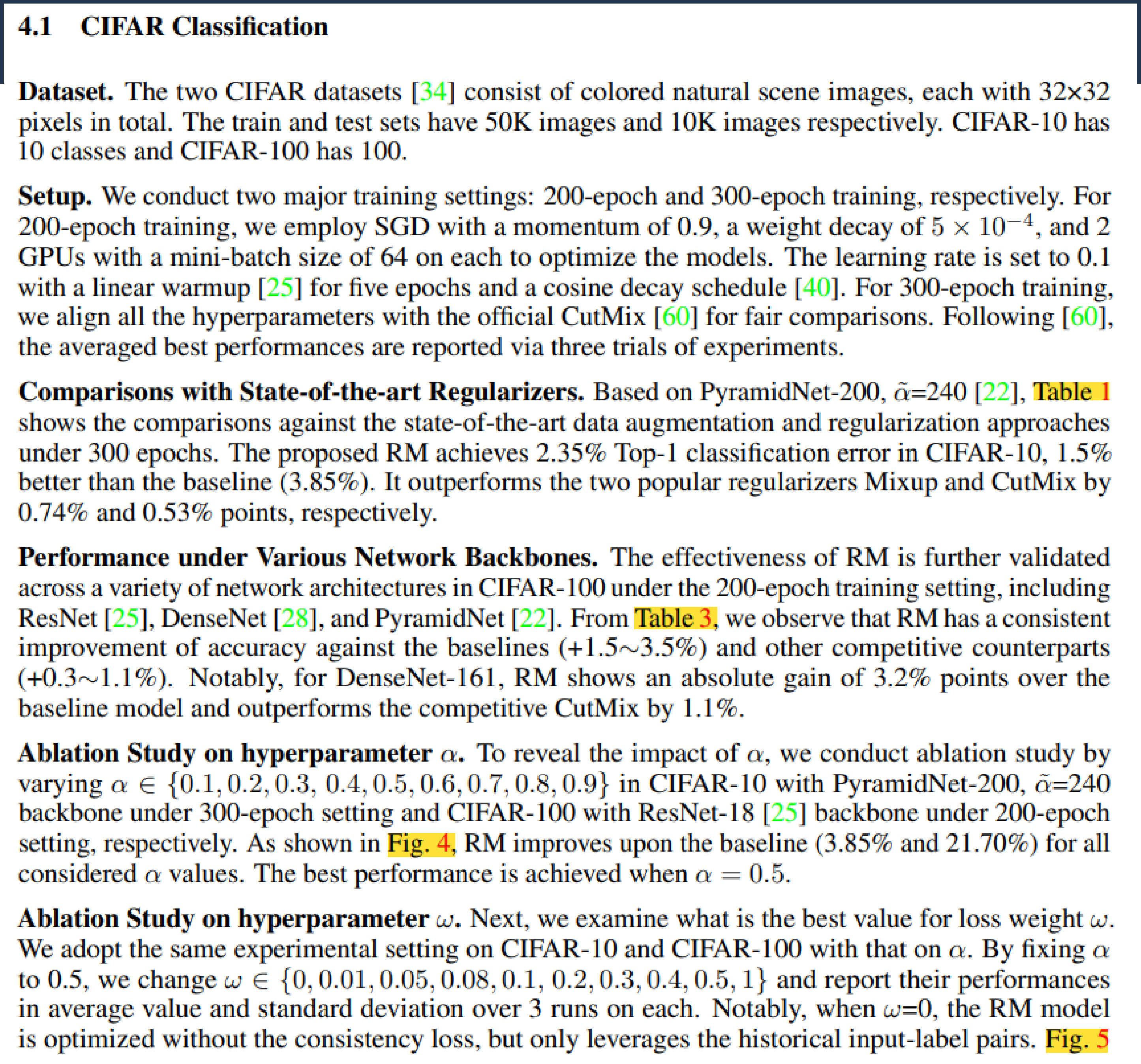
$$\widetilde{p}^t_{roi} = \mathcal{H}'(\mathrm{RoIAlign}(\mathcal{F}(\widetilde{x}^t), \mathbf{B})), \tag{4}$$

where $\mathcal{F}$ denotes the backbone function and $\mathcal{H}, \mathcal{H}'$ denote the final linear classification layer. By default, the parameters are not shared between layers $\mathcal{H}$ and $\mathcal{H}'$, which is purely suggested by the experiments in Table 4a. The total loss function to train our model is:

$$\mathcal{L} = \mathcal{L}_{CE}(\widetilde{x}^t, \widetilde{y}^t) + \omega\lambda^t \mathcal{L}_{KL}(\widetilde{p}^t_{roi}, p^h), \tag{5}$$

where $\mathcal{L}_{CE}$ denotes the cross-entropy loss and $\mathcal{L}_{KL}$ denotes the consistency loss. Note that p^h is the historical prediction from the last iteration, and the consistency loss weight ω is set to 0.1 unless otherwise stated. In addition, we also use the mixed ratio λ representing the proportion of the resize-filled historical image to weight $\mathcal{L}_{KL}$. It makes sense that the confidence of $\mathcal{L}_{KL}$ relates to λ as a smaller λ (smaller image size to be filled) usually causes the RoI feature to lose more spatial

4 Experiment



4.1 CIFAR Classification

Dataset. The two CIFAR datasets [34] consist of colored natural scene images, each with 32×32 pixels in total. The train and test sets have 50K images and 10K images respectively. CIFAR-10 has 10 classes and CIFAR-100 has 100.

Setup. We conduct two major training settings: 200-epoch and 300-epoch training, respectively. For 200-epoch training, we employ SGD with a momentum of 0.9, a weight decay of 5×10^{-4}, and 2 GPUs with a mini-batch size of 64 on each to optimize the models. The learning rate is set to 0.1 with a linear warmup [25] for five epochs and a cosine decay schedule [40]. For 300-epoch training, we align all the hyperparameters with the official CutMix [60] for fair comparisons. Following [60], the averaged best performances are reported via three trials of experiments.

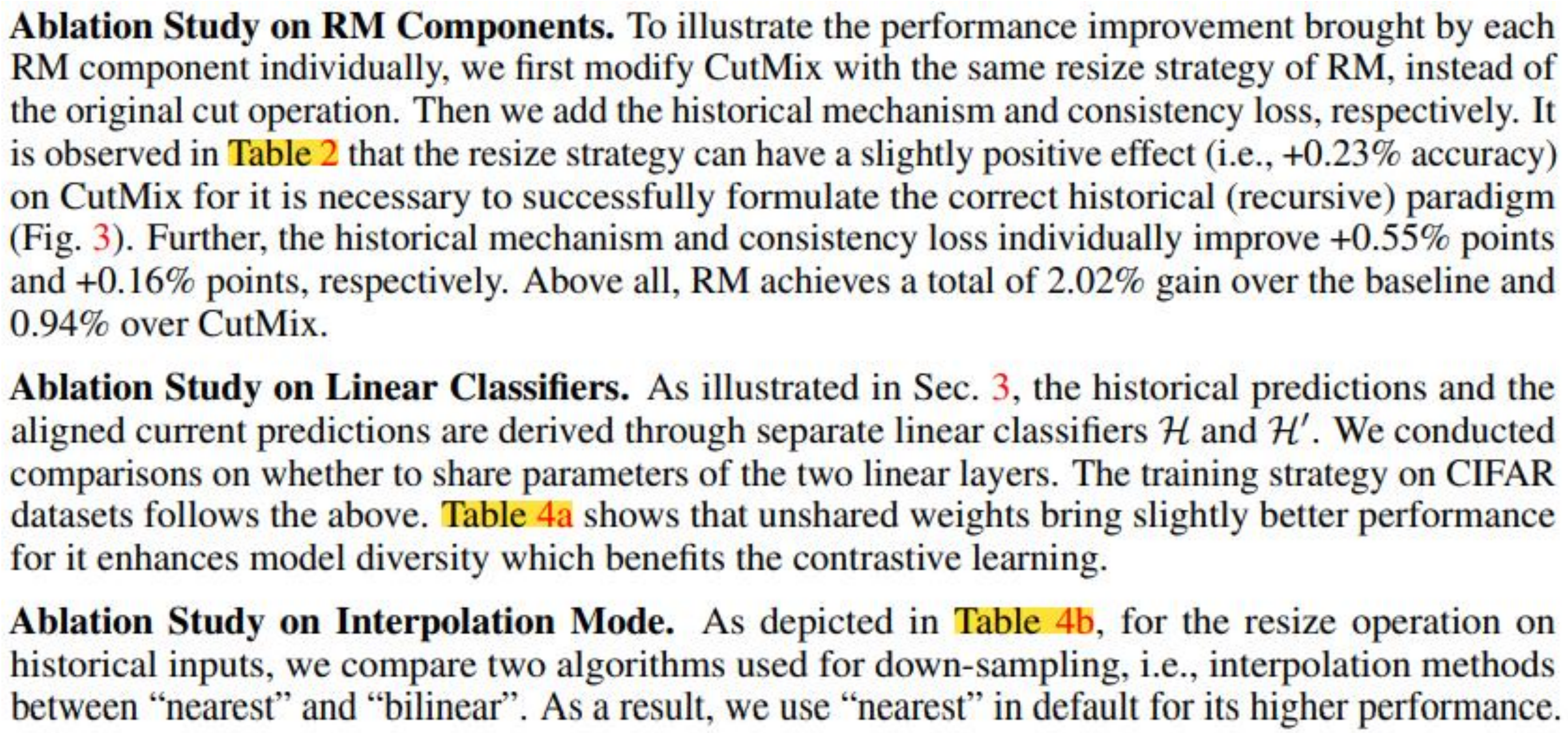
Comparisons with State-of-the-art Regularizers. Based on PyramidNet-200, $\tilde{\alpha}$=240 [22], Table 1 shows the comparisons against the state-of-the-art data augmentation and regularization approaches under 300 epochs. The proposed RM achieves 2.35% Top-1 classification error in CIFAR-10, 1.5% better than the baseline (3.85%). It outperforms the two popular regularizers Mixup and CutMix by 0.74% and 0.53% points, respectively.

Performance under Various Network Backbones. The effectiveness of RM is further validated across a variety of network architectures in CIFAR-100 under the 200-epoch training setting, including ResNet [25], DenseNet [28], and PyramidNet [22]. From Table 3, we observe that RM has a consistent improvement of accuracy against the baselines (+1.5∼3.5%) and other competitive counterparts (+0.3∼1.1%). Notably, for DenseNet-161, RM shows an absolute gain of 3.2% points over the baseline model and outperforms the competitive CutMix by 1.1%.

Ablation Study on hyperparameter α. To reveal the impact of α, we conduct ablation study by varying $\alpha \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$ in CIFAR-10 with PyramidNet-200, $\tilde{\alpha}$=240 backbone under 300-epoch setting and CIFAR-100 with ResNet-18 [25] backbone under 200-epoch setting, respectively. As shown in Fig. 4, RM improves upon the baseline (3.85% and 21.70%) for all considered α values. The best performance is achieved when $\alpha = 0.5$.

Ablation Study on hyperparameter ω. Next, we examine what is the best value for loss weight ω. We adopt the same experimental setting on CIFAR-10 and CIFAR-100 with that on α. By fixing α to 0.5, we change $\omega \in \{0, 0.01, 0.05, 0.08, 0.1, 0.2, 0.3, 0.4, 0.5, 1\}$ and report their performances in average value and standard deviation over 3 runs on each. Notably, when ω=0, the RM model is optimized without the consistency loss, but only leverages the historical input-label pairs. Fig. 5

4 Experiment



Ablation Study on RM Components. To illustrate the performance improvement brought by each RM component individually, we first modify CutMix with the same resize strategy of RM, instead of the original cut operation. Then we add the historical mechanism and consistency loss, respectively. It is observed in Table 2 that the resize strategy can have a slightly positive effect (i.e., +0.23% accuracy) on CutMix for it is necessary to successfully formulate the correct historical (recursive) paradigm (Fig. 3). Further, the historical mechanism and consistency loss individually improve +0.55% points and +0.16% points, respectively. Above all, RM achieves a total of 2.02% gain over the baseline and 0.94% over CutMix.

Ablation Study on Linear Classifiers. As illustrated in Sec. 3, the historical predictions and the aligned current predictions are derived through separate linear classifiers $\mathcal{H}$ and $\mathcal{H}'$. We conducted comparisons on whether to share parameters of the two linear layers. The training strategy on CIFAR datasets follows the above. Table 4a shows that unshared weights bring slightly better performance for it enhances model diversity which benefits the contrastive learning.

Ablation Study on Interpolation Mode. As depicted in Table 4b, for the resize operation on historical inputs, we compare two algorithms used for down-sampling, i.e., interpolation methods between "nearest" and "bilinear". As a result, we use "nearest" in default for its higher performance.

4 Experiment

吉林大学

PyramidNet-200, $\tilde{\alpha}$=240 (300 epochs)	Top-1 Err (%)
Baseline	3.85
+ Label Smoothing [43]	3.74
+ DropBlock [19]	3.27
+ Stochastic Depth [29]	3.11
+ Cutout [15]	3.10
+ Mixup (α=1.0) [61]	3.09
+ Manifold Mixup (α=1.0) [52]	3.15
+ CutMix [60]	2.88
+ MoEx [36]	3.44
+ StyleCutMix (auto-γ) [27]	2.55
+ RM (ours)	**2.35**

Table 1: Comparison of state-of-the-art regularization methods in CIFAR-10 under 300 epochs settings, the same with that of the CutMix [60] paper. Results are reported as average over 3 runs.

Model	RS	HIS	CL	Top-1 Err (%)
PyramidNet	–	–	–	16.67
+ CutMix [60]				15.59
+ RM (ours)	✓			15.36
	✓	✓		14.81
	✓	✓	✓	**14.65**

Table 2: Ablation studies on the individual gain by each component of RM in CIFAR-100 with PyramidNet-164, $\tilde{\alpha}$=270 under the 200-epoch setting. The results are averaged over 3 runs. "**RS**": Resize strategy. "**HIS**": Historical mix. "**CL**": Consistency loss.

Model (200 epochs)	Type	Top-1 Err (%)	
ResNet-18 [25]	Baseline	21.70	
	+ Mixup [61]	20.99	↓1%
	+ CutMix [60]	19.61	
	+ RM (ours)	**18.64**	
ResNet-34 [25]	Baseline	20.62	
	+ Mixup [61]	19.19	< ↓1%
	+ CutMix [60]	17.89	
	+ RM (ours)	**17.15**	
DenseNet-121 [28]	Baseline	19.51	
	+ Mixup [61]	17.71	↓1%
	+ CutMix [60]	17.21	
	+ RM (ours)	**16.22**	
DenseNet-161 [28]	Baseline	18.78	
	+ Mixup [61]	16.84	<
	+ CutMix [60]	16.64	
	+ RM (ours)	**15.54**	
PyramidNet-164, $\tilde{\alpha}$=270 [22]	Baseline	16.67	
	+ Mixup [61]	16.02	
	+ CutMix [60]	15.59	<
	+ RM (ours)	**14.65**	

Table 3: Performance on various architectures in CIFAR-100 under 200 epochs. The proposed RM (α=0.5, ω=0.1) shows consistent improvements over other competitive approaches. We report an average of 3 runs.

4 Experiment

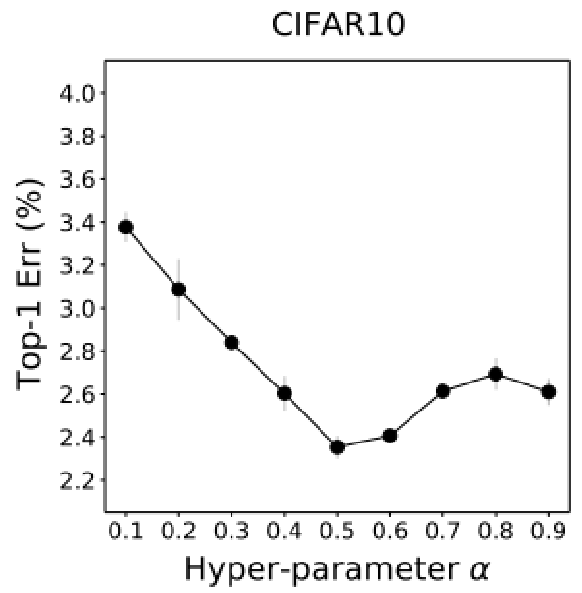
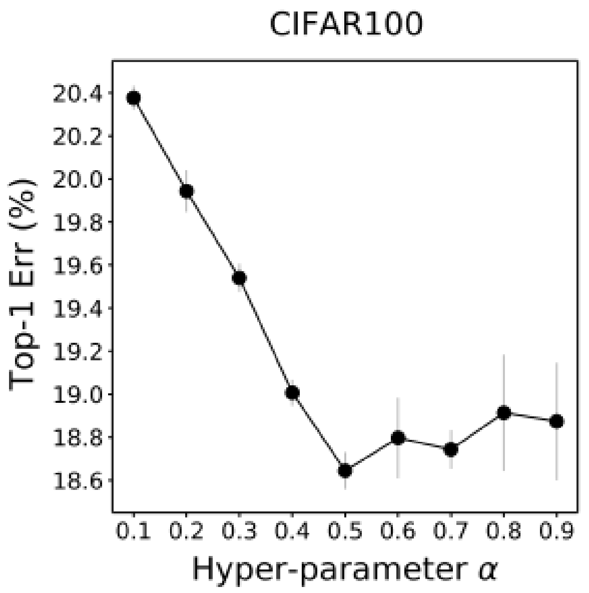



Figure 4: The effect of hyperparameter α on CIFAR-10 and CIFAR-100 datasets. Standard deviation is also plotted and another hyperparameter ω is fixed to 0.1.

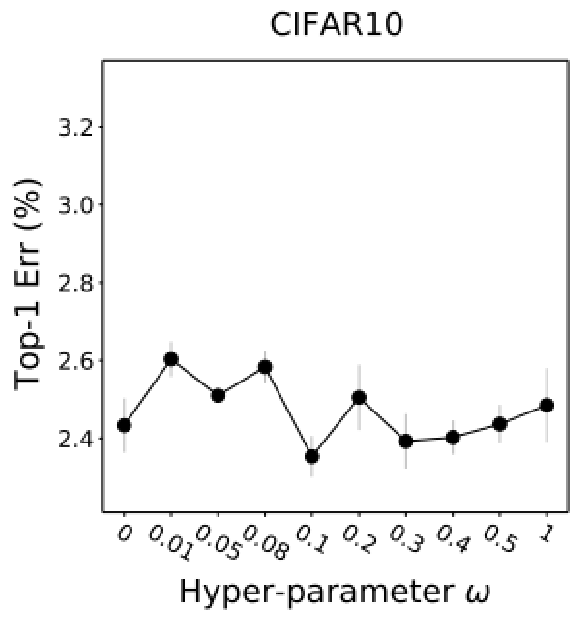
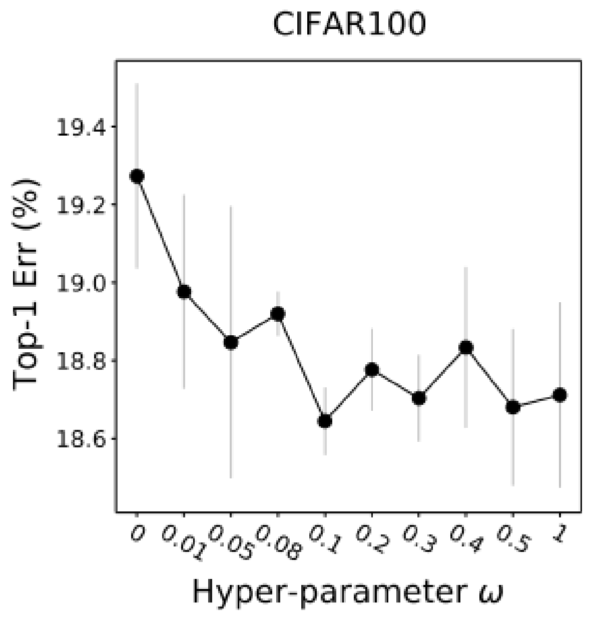
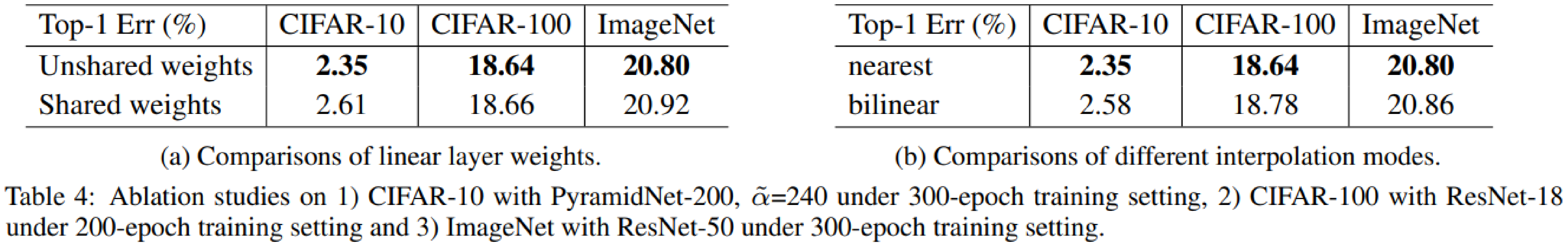



Figure 5: The effect of hyperparameter ω on CIFAR-10 and CIFAR-100 datasets. Standard deviation is also plotted and another hyperparameter α is fixed to 0.5.

4 Experiment



Top-1 Err (%)	CIFAR-10	CIFAR-100	ImageNet
Unshared weights	**2.35**	**18.64**	**20.80**
Shared weights	2.61	18.66	20.92

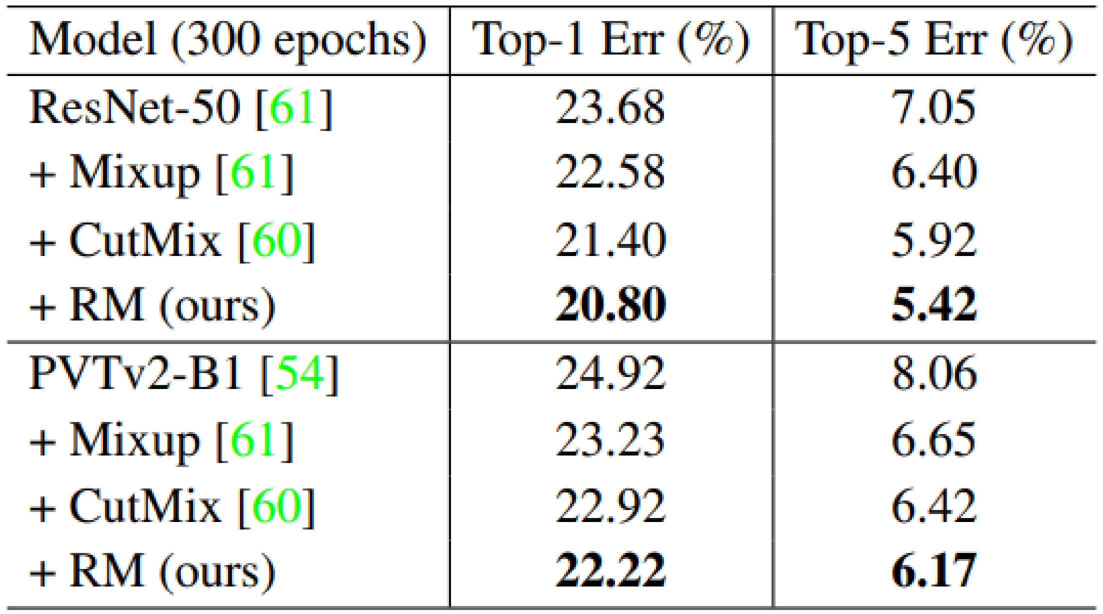
(a) Comparisons of linear layer weights.

Top-1 Err (%)	CIFAR-10	CIFAR-100	ImageNet
nearest	**2.35**	**18.64**	**20.80**
bilinear	2.58	18.78	20.86

(b) Comparisons of different interpolation modes.

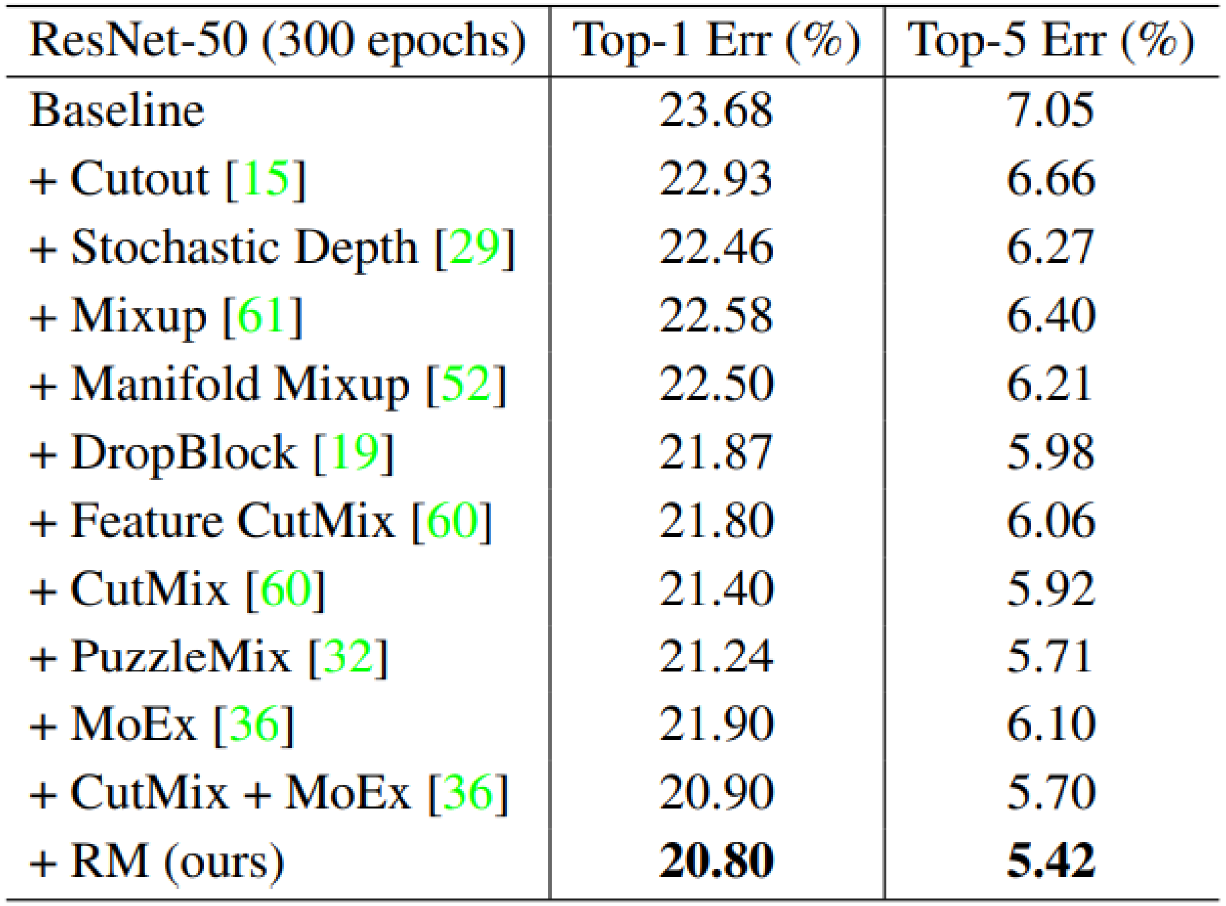
Table 4: Ablation studies on 1) CIFAR-10 with PyramidNet-200, $\tilde{\alpha}$=240 under 300-epoch training setting, 2) CIFAR-100 with ResNet-18 under 200-epoch training setting and 3) ImageNet with ResNet-50 under 300-epoch training setting.

4 Experiment



Model (300 epochs)	Top-1 Err (%)	Top-5 Err (%)
ResNet-50 [61]	23.68	7.05
+ Mixup [61]	22.58	6.40
+ CutMix [60]	21.40	5.92
+ RM (ours)	**20.80**	**5.42**
PVTv2-B1 [54]	24.92	8.06
+ Mixup [61]	23.23	6.65
+ CutMix [60]	22.92	6.42
+ RM (ours)	**22.22**	**6.17**

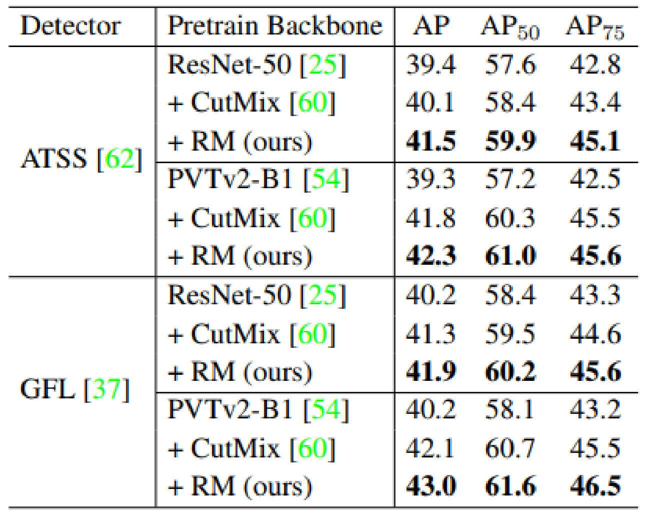
Table 5: Performance on various architectures in ImageNet under 300 epochs. The proposed RM (α=0.5, ω=0.5) shows consistent improvements over other competitive counterparts.



ResNet-50 (300 epochs)	Top-1 Err (%)	Top-5 Err (%)
Baseline	23.68	7.05
+ Cutout [15]	22.93	6.66
+ Stochastic Depth [29]	22.46	6.27
+ Mixup [61]	22.58	6.40
+ Manifold Mixup [52]	22.50	6.21
+ DropBlock [19]	21.87	5.98
+ Feature CutMix [60]	21.80	6.06
+ CutMix [60]	21.40	5.92
+ PuzzleMix [32]	21.24	5.71
+ MoEx [36]	21.90	6.10
+ CutMix + MoEx [36]	20.90	5.70
+ RM (ours)	**20.80**	**5.42**

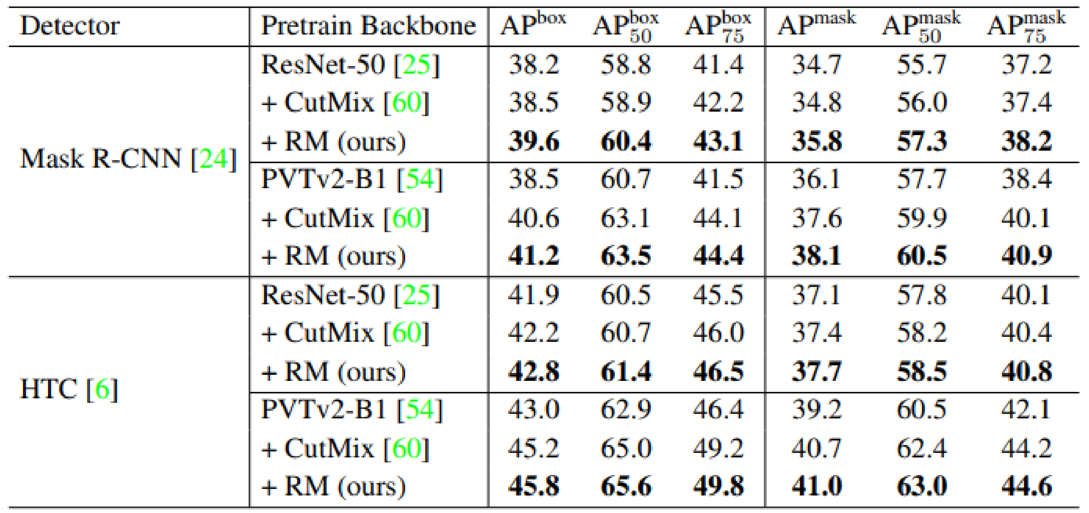
Table 6: Comparison of state-of-the-art regularization methods in ImageNet under 300 epochs.

4 Experi



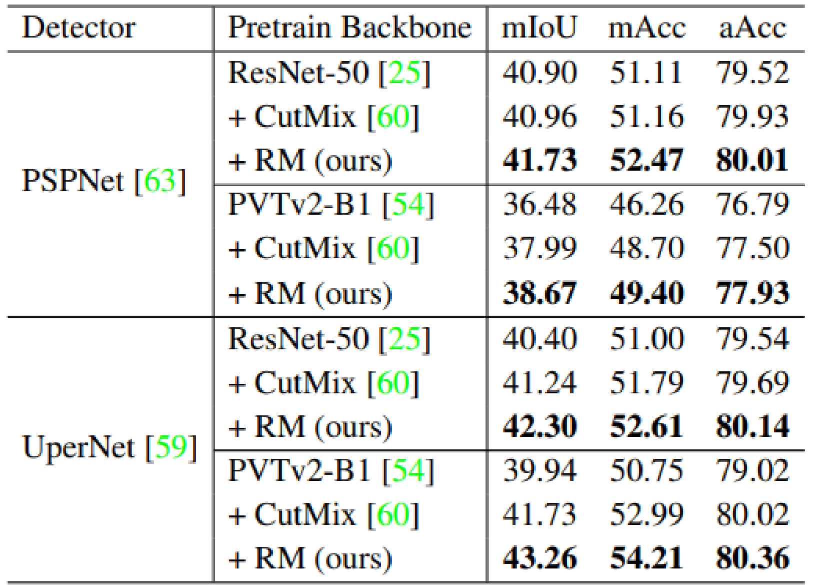
Detector	Pretrain Backbone	AP	AP_{50}	AP_{75}
ATSS [62]	ResNet-50 [25]	39.4	57.6	42.8
	+ CutMix [60]	40.1	58.4	43.4
	+ RM (ours)	**41.5**	**59.9**	**45.1**
	PVTv2-B1 [54]	39.3	57.2	42.5
	+ CutMix [60]	41.8	60.3	45.5
	+ RM (ours)	**42.3**	**61.0**	**45.6**
GFL [37]	ResNet-50 [25]	40.2	58.4	43.3
	+ CutMix [60]	41.3	59.5	44.6
	+ RM (ours)	**41.9**	**60.2**	**45.6**
	PVTv2-B1 [54]	40.2	58.1	43.2
	+ CutMix [60]	42.1	60.7	45.5
	+ RM (ours)	**43.0**	**61.6**	**46.5**

Table 7: Object detection fine-tuned on COCO with the 1x schedule by ATSS [62] and GFL [37].



Detector	Pretrain Backbone	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}
Mask R-CNN [24]	ResNet-50 [25]	38.2	58.8	41.4	34.7	55.7	37.2
	+ CutMix [60]	38.5	58.9	42.2	34.8	56.0	37.4
	+ RM (ours)	**39.6**	**60.4**	**43.1**	**35.8**	**57.3**	**38.2**
	PVTv2-B1 [54]	38.5	60.7	41.5	36.1	57.7	38.4
	+ CutMix [60]	40.6	63.1	44.1	37.6	59.9	40.1
	+ RM (ours)	**41.2**	**63.5**	**44.4**	**38.1**	**60.5**	**40.9**
HTC [6]	ResNet-50 [25]	41.9	60.5	45.5	37.1	57.8	40.1
	+ CutMix [60]	42.2	60.7	46.0	37.4	58.2	40.4
	+ RM (ours)	**42.8**	**61.4**	**46.5**	**37.7**	**58.5**	**40.8**
	PVTv2-B1 [54]	43.0	62.9	46.4	39.2	60.5	42.1
	+ CutMix [60]	45.2	65.0	49.2	40.7	62.4	44.2
	+ RM (ours)	**45.8**	**65.6**	**49.8**	**41.0**	**63.0**	**44.6**

Table 8: Object detection and instance segmentation fine-tuned on COCO with the 1x schedule by Mask R-CNN [24] and HTC [6].



Detector	Pretrain Backbone	mIoU	mAcc	aAcc
PSPNet [63]	ResNet-50 [25]	40.90	51.11	79.52
	+ CutMix [60]	40.96	51.16	79.93
	+ RM (ours)	**41.73**	**52.47**	**80.01**
	PVTv2-B1 [54]	36.48	46.26	76.79
	+ CutMix [60]	37.99	48.70	77.50
	+ RM (ours)	**38.67**	**49.40**	**77.93**
UperNet [59]	ResNet-50 [25]	40.40	51.00	79.54
	+ CutMix [60]	41.24	51.79	79.69
	+ RM (ours)	**42.30**	**52.61**	**80.14**
	PVTv2-B1 [54]	39.94	50.75	79.02
	+ CutMix [60]	41.73	52.99	80.02
	+ RM (ours)	**43.26**	**54.21**	**80.36**

Table 9: Semantic segmentation fine-tuned on ADE20K [66] for 80k iterations by PSPNet [63] and UperNet [59].

4 Experiment

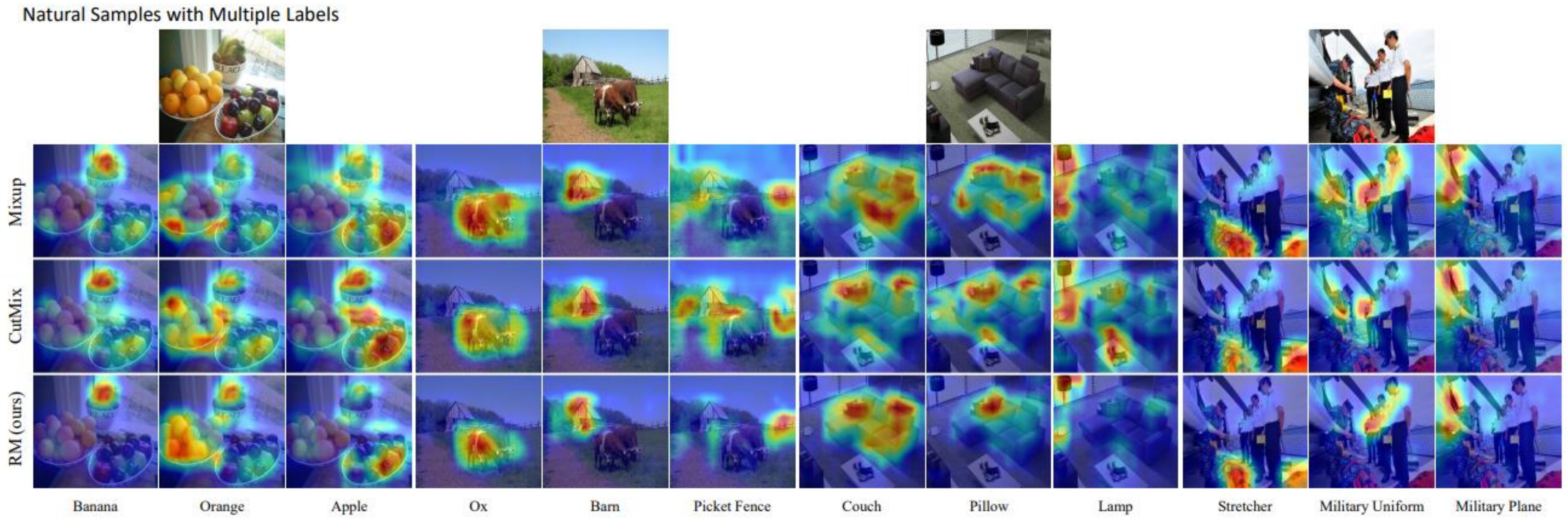


Figure 6: CAM [65] visualization comparing RM with Mixup [61] and CutMix [60] on natural samples with multiple labels.

4 Experiment

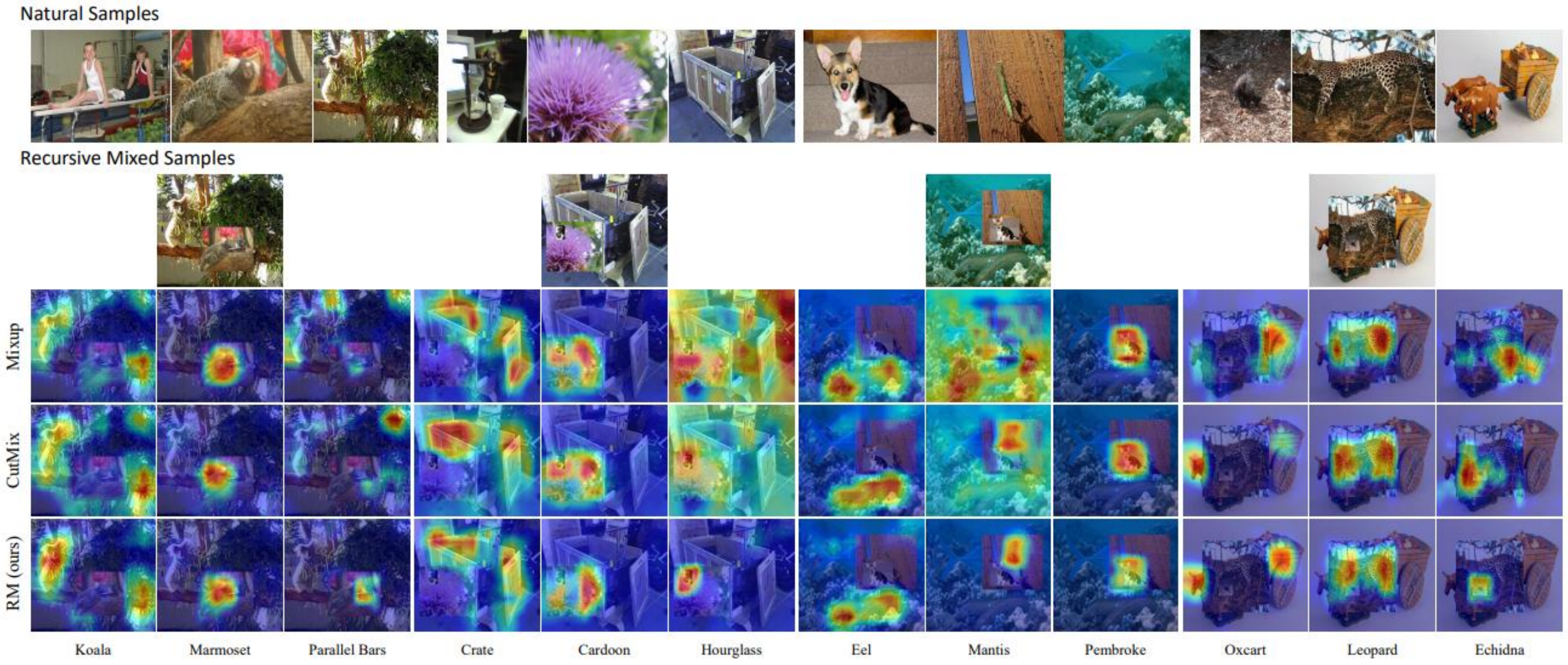


Figure 7: CAM [65] visualization comparing RM with Mixup [61] and CutMix [60] on samples created using RM.

Effective Class Number

Training Efficiency. As shown in Table 10, we improve the performance with negligible additional memory cost. Notably, the training hour and computation complexity are competitive to existing methods.

Effective Class Number. To illustrate the diversity in the supervision signals of RM, we denote the number of classes in the regularized target whose supervision values are higher than a given threshold as effective classes. We then record the average effective number over 3 runs under 100 iterations on RM and other methods in ImageNet training set. Note that the label smoothing threshold is roughly set to "10^{-4}". We suspect that a signal value above the label smoothing threshold can be used as effective supervision, then RM can achieve a learning signal containing an average of about 4∼5 objects in a single image, which confirms the diversity of its supervision (Fig. 8).

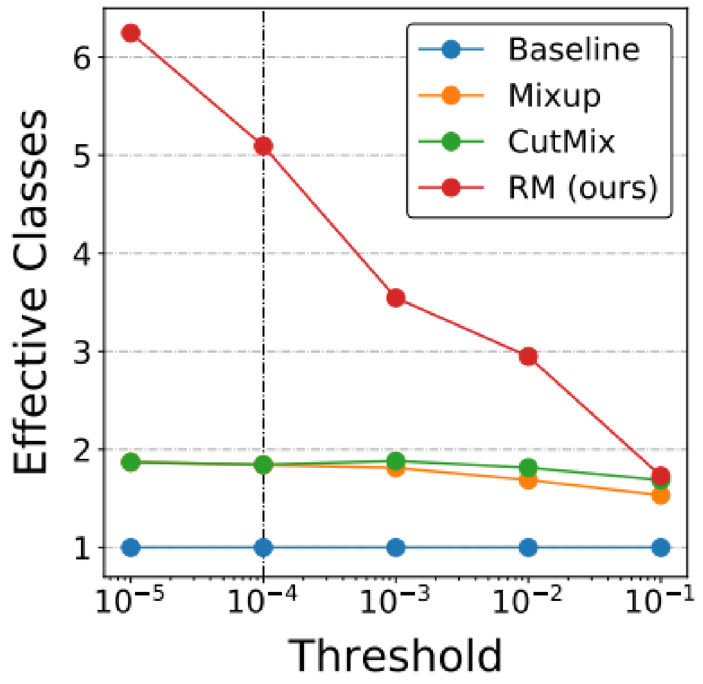
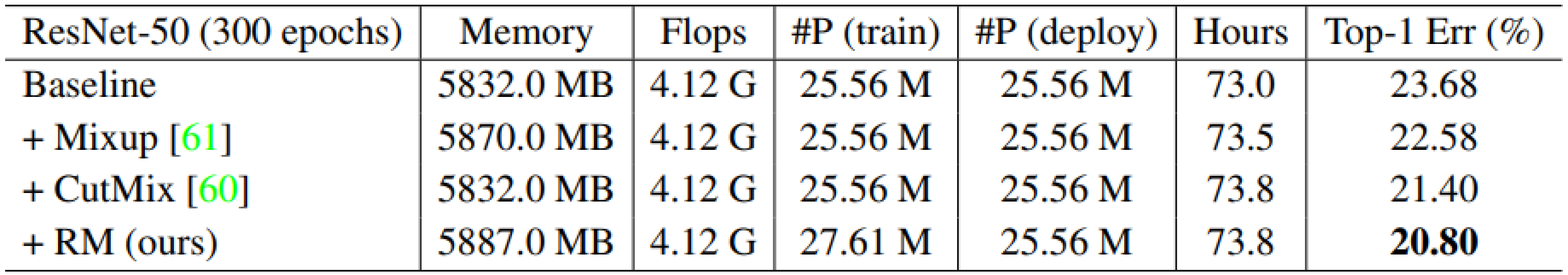


Figure 8: Comparisons of the effective classes during training under different thresholds.

Training Efficiency



ResNet-50 (300 epochs)	Memory	Flops	#P (train)	#P (deploy)	Hours	Top-1 Err (%)
Baseline	5832.0 MB	4.12 G	25.56 M	25.56 M	73.0	23.68
+ Mixup [61]	5870.0 MB	4.12 G	25.56 M	25.56 M	73.5	22.58
+ CutMix [60]	5832.0 MB	4.12 G	25.56 M	25.56 M	73.8	21.40
+ RM (ours)	5887.0 MB	4.12 G	27.61 M	25.56 M	73.8	**20.80**

Table 10: Comparisons of the training efficiency by hours, evaluated on 8 TITAN Xp GPUs. “#P” denotes the number of parameters.

吉林大学

THANKS