동국대학교 dongguk university

Modern Data Augmentation Techniques

Mixup and Cutmix 중심으로

Park, MinKyu

2020.09.04

Dongguk University

Artificial Intelligence Laboratory

mkpark73@dongguk.edu

Data Augmentation

- **Data augmentation** significantly increases the diversity of data available for training our models, without actually collecting new data samples.
- Simple image data augmentation techniques like flipping, random crop, and random rotation are commonly used to train large models.

Overview of the results of Mixup, Cutout, and CutMix.

	ResNet-50	Mixup	Cutout	CutMix
Image				
Label	Dog 1.0	Dog 0.5 Cat 0.5	Dog 1.0	Dog 0.6 Cat 0.4

mixup: Beyond Empirical Risk Minimization

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, David Lopez-Paz

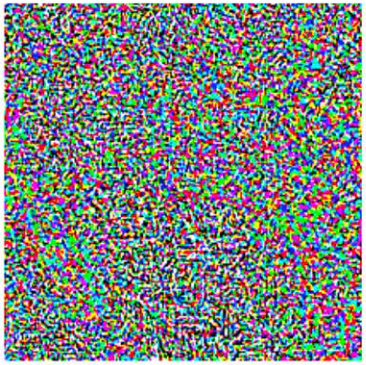
https://arxiv.org/abs/1710.09412

Abstract

- Neural networks can exhibit undesirable characteristics: memorization and sensitivity to adversarial examples
- Paper proposes a ***generic*** data augmentation technique to address this
- Claims:
  - reduces the memorization of corrupt labels
  - increases the robustness to adversarial examples
  - stabilizes the training of generative adversarial networks

참고 : adversarial example

$+ .007 \times$



$=$

"panda"

57.7% confidence

noise

"gibbon"

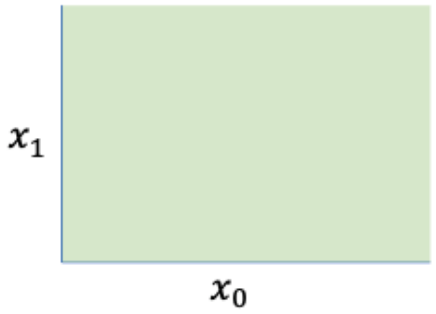
99.3% confidence

Learning Theory – Empirical Risk Minimization

- Supervised learning: find a **function** $f(x)$ that maps x to y
- We find this function by penalizing errors that our model makes

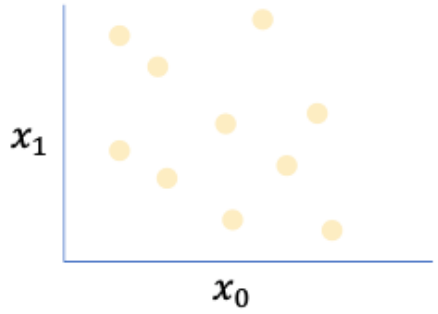
$$R(f) = \int \ell(f(x), y) \mathrm{d}P(x, y).$$

Expected risk: calculate loss over all ***possible*** data.



$$R_\delta(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i).$$

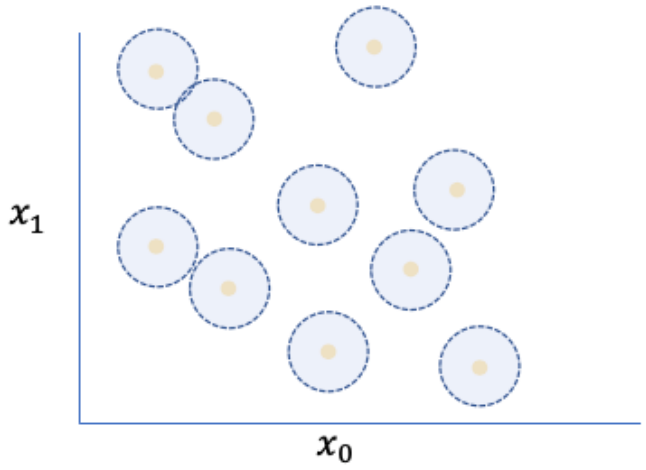
Empirical risk: calculate loss over all ***available*** data.



Learning Theory – Vicinal Risk Minimization

- Let's improve our approximation of the true distribution by sampling data from ***neighborhoods*** (i.e. vicinities) around our available data.

$$R_\nu(f) = \frac{1}{m}\sum_{i=1}^{m} \ell(f(\tilde{x}_i), \tilde{y}_i).$$



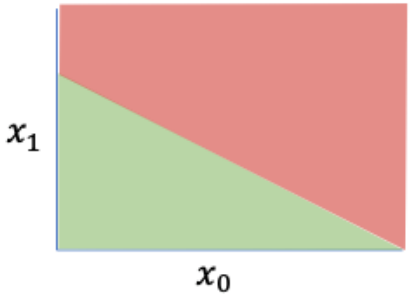
This allows to *explore more of the feature space* when learning.

You're probably already doing vicinal risk minimization – it's called **data augmentation**!

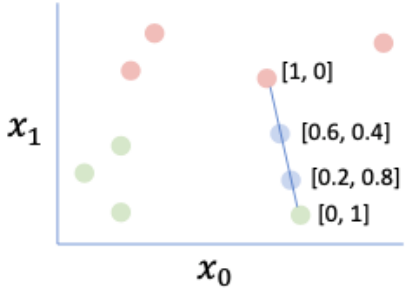
mixup

- Linearly interpolate between existing data
- This is kind of similar to SMOTE except we interpolate the targets too!

$$\tilde{x} = \lambda x_i + (1 - \lambda) x_j,$$
$$\tilde{y} = \lambda y_i + (1 - \lambda) y_j,$$



True distribution

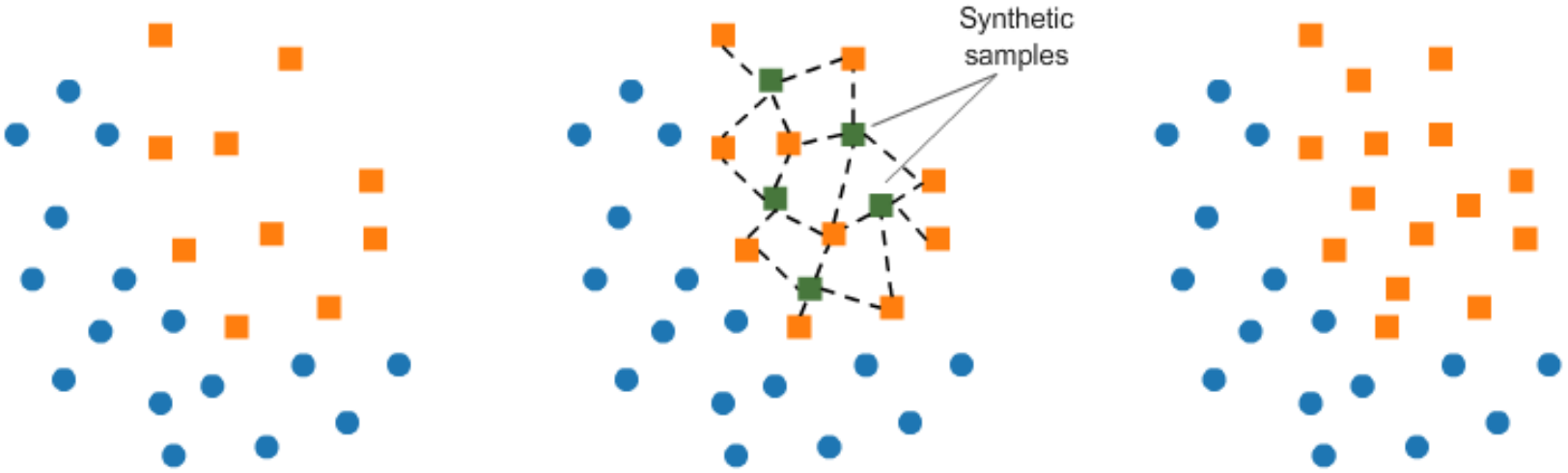


Empirical distribution

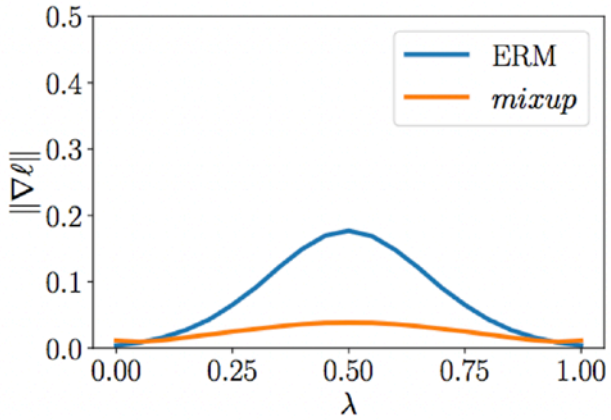
Sample points linearly between two observations

참고

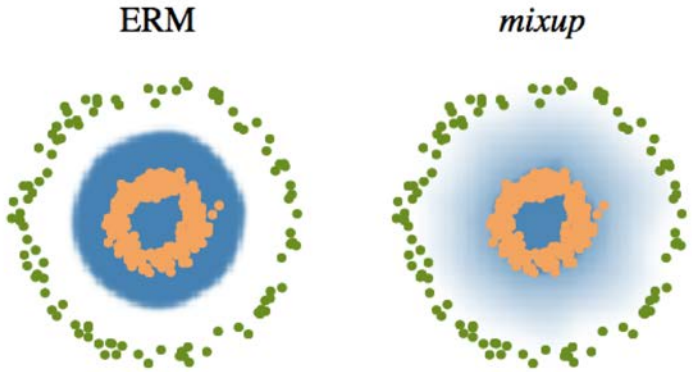
- Synthetic Minority Over-sampling Technique (SMOTE)



This leads to some nice properties



(b) Norm of the gradients of the model w.r.t. input in-between training data, evaluated at $x = \lambda x_i + (1 - \lambda)x_j$. The model trained with *mixup* has smaller gradient norms.

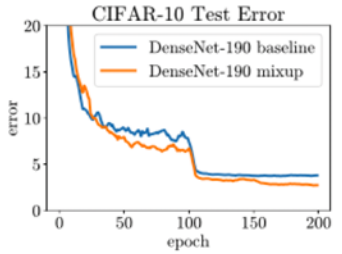


(b) Effect of *mixup* ($\alpha = 1$) on a toy problem. Green: Class 0. Orange: Class 1. Blue shading indicates $p(y = 1|x)$.

Image data

Dataset	Model	ERM	*mixup*
CIFAR-10	PreAct ResNet-18	5.6	**4.2**
	WideResNet-28-10	3.8	**2.7**
	DenseNet-BC-190	3.7	**2.7**
CIFAR-100	PreAct ResNet-18	25.6	**21.1**
	WideResNet-28-10	19.4	**17.5**
	DenseNet-BC-190	19.0	**16.8**

(a) Test errors for the CIFAR experiments.



(b) Test error evolution for the best ERM and *mixup* models.

Figure 3: Test errors for ERM and *mixup* on the CIFAR experiments.

Corrupted labels

Label corruption	Method	Test error		Training error	
		Best	Last	Real	Corrupted
20%	ERM	12.7	16.6	0.05	0.28
	ERM + dropout ($p = 0.7$)	8.8	10.4	5.26	83.55
	mixup ($\alpha = 8$)	**5.9**	6.4	2.27	86.32
	mixup + dropout ($\alpha = 4, p = 0.1$)	6.2	**6.2**	1.92	85.02
50%	ERM	18.8	44.6	0.26	0.64
	ERM + dropout ($p = 0.8$)	14.1	15.5	12.71	86.98
	mixup ($\alpha = 32$)	11.3	12.7	5.84	85.71
	mixup + dropout ($\alpha = 8, p = 0.3$)	**10.9**	**10.9**	7.56	87.90
80%	ERM	36.5	73.9	0.62	0.83
	ERM + dropout ($p = 0.8$)	30.9	35.1	29.84	86.37
	mixup ($\alpha = 32$)	25.3	30.9	18.92	85.44
	mixup + dropout ($\alpha = 8, p = 0.3$)	**24.0**	**24.8**	19.70	87.67

Table 2: Results on the corrupted label experiments for the best models.

Speech data

Model	Method	Validation set	Test set
LeNet	ERM	**9.8**	**10.3**
	mixup ($\alpha = 0.1$)	10.1	10.8
	mixup ($\alpha = 0.2$)	10.2	11.3
VGG-11	ERM	5.0	4.6
	mixup ($\alpha = 0.1$)	4.0	3.8
	mixup ($\alpha = 0.2$)	**3.9**	**3.4**

Figure 4: Classification errors of ERM and *mixup* on the Google commands dataset.

Tabular data

Dataset	ERM	*mixup*
Abalone	74.0	73.6
Arcene	57.6	**48.0**
Arrhythmia	56.6	**46.3**

Dataset	ERM	*mixup*
Htru2	2.0	2.0
Iris	21.3	**17.3**
Phishing	16.3	15.2

Table 4: ERM and *mixup* classification errors on the UCI datasets.

GAN stabilization

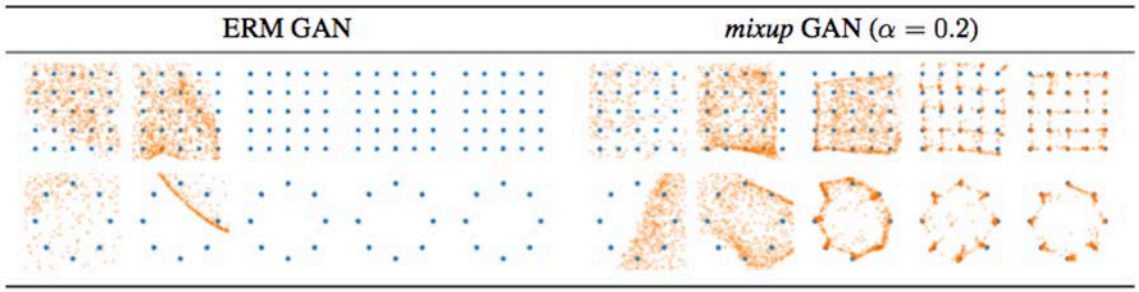


Figure 5: Effect of *mixup* on stabilizing GAN training at iterations 10, 100, 1000, 10000, and 20000.

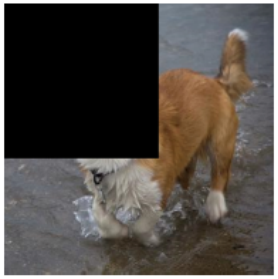
CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features

Sangdoo Yun, Dongyoon Han, Seong Joon Oh,
Sanghyuk Chun, Junsuk Choe, Youngjoon Yoo
Clova AI Research, NAVER Corp.

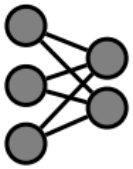
https://arxiv.org/abs/1905.04899

Image Classification

- Regional dropout strategy for “occlusion-robust” classifier[a, b]



Input Image



Model

Dog = 1.0

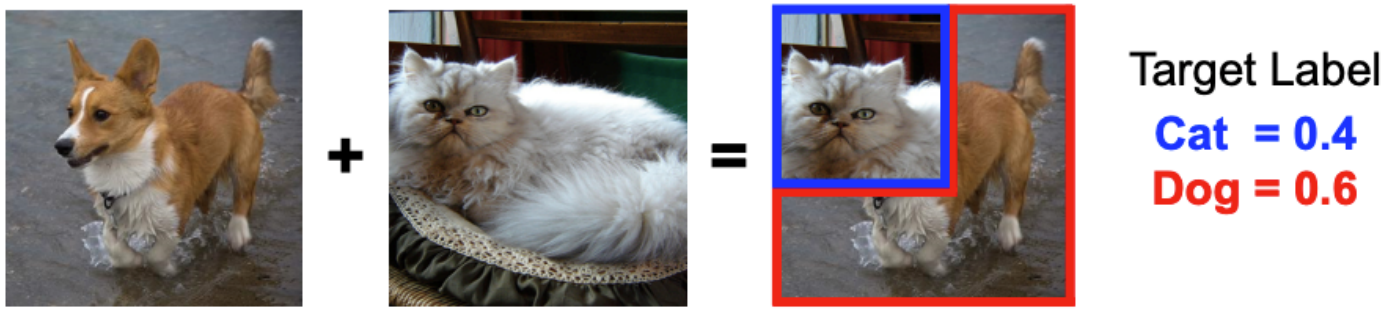
Target

✓ Good generalization ability

✘ Cannot utilize full image regions

[a] Devries et al., “Improved regularization of convolutional neural networks with cutout”, arXiv 2017.
[b] Zhong et al., “Random erasing data augmentation”, arXiv 2017.

CutMix in a Nutshell



- Cut and paste two images and labels.
- In this way, the classifier learns “what” and “where” objects are in the image.

CutMix in a Nutshell

Overview of the results of Mixup, Cutout, and CutMix.

	ResNet-50	Mixup	Cutout	CutMix
Image				
Label	Dog 1.0	Dog 0.5 Cat 0.5	Dog 1.0	Dog 0.6 Cat 0.4

- Unlike Cutout, CutMix uses all input pixels for training.
- Unlike Mixup, CutMix presents realistic local image patches.
- CutMix is simple: only 20 lines of pytorch code.

Algorithm

Let $x \in \mathbb{R}^{W \times H \times C}$ and y denote a training image and its label, respectively. The goal of CutMix is to generate a new training sample $(\tilde{x}, \tilde{y})$ by combining two training samples (x_A, y_A) and (x_B, y_B). The generated training sample $(\tilde{x}, \tilde{y})$ is used to train the model with its original loss function. We define the combining operation as

$$\begin{aligned} \tilde{x} &= \mathbf{M} \odot x_A + (\mathbf{1} - \mathbf{M}) \odot x_B \\ \tilde{y} &= \lambda y_A + (1 - \lambda) y_B, \end{aligned} \tag{1}$$

where $\mathbf{M} \in \{0,1\}^{W \times H}$ denotes a binary mask indicating where to drop out and fill in from two images, $\mathbf{1}$ is a binary mask filled with ones, and $\odot$ is element-wise multiplication. Like Mixup [48], the combination ratio λ between two data points is sampled from the beta distribution Beta(α, α). In our all experiments, we set α to 1, that is λ is sampled from the uniform distribution $(0, 1)$. Note that the major difference is that CutMix replaces an image region with a patch from another training image and generates more locally natural image than Mixup does.

To sample the binary mask $\mathbf{M}$, we first sample the bounding box coordinates $\mathbf{B} = (r_x, r_y, r_w, r_h)$ indicating the cropping regions on x_A and x_B. The region $\mathbf{B}$ in x_A is removed and filled in with the patch cropped from $\mathbf{B}$ of x_B.

In our experiments, we sample rectangular masks $\mathbf{M}$ whose aspect ratio is proportional to the original image. The box coordinates are uniformly sampled according to:

$$\begin{aligned} r_x &\sim \text{Unif}\ (0, W), \quad r_w = W\sqrt{1-\lambda}, \\ r_y &\sim \text{Unif}\ (0, H), \quad r_h = H\sqrt{1-\lambda} \end{aligned} \tag{2}$$

making the cropped area ratio $\frac{r_w r_h}{WH} = 1 - \lambda$. With the cropping region, the binary mask $\mathbf{M} \in \{0,1\}^{W \times H}$ is decided by filling with 0 within the bounding box $\mathbf{B}$, otherwise 1.

Overview of the results

	ResNet-50	Mixup [48]	Cutout [3]	CutMix
Image				
Label	Dog 1.0	Dog 0.5 Cat 0.5	Dog 1.0	Dog 0.6 Cat 0.4
ImageNet Cls (%)	76.3 (+0.0)	77.4 (+1.1)	77.1 (+0.8)	**78.6** **(+2.3)**
ImageNet Loc (%)	46.3 (+0.0)	45.8 (-0.5)	46.7 (+0.4)	**47.3** **(+1.0)**
Pascal VOC Det (mAP)	75.6 (+0.0)	73.9 (-1.7)	75.1 (-0.5)	**76.7** **(+1.1)**

Table 1: Overview of the results of Mixup, Cutout, and our CutMix on ImageNet classification, ImageNet localization, and Pascal VOC 07 detection (transfer learning with SSD [24] finetuning) tasks. Note that CutMix significantly improves the performance on various tasks.

Experiments

ImageNet Classification

Model	# Params	Top-1 Err (%)	Top-5 Err (%)
ResNet-152*	60.3 M	21.69	5.94
ResNet-101 + SE Layer* [15]	49.4 M	20.94	5.50
ResNet-101 + GE Layer* [14]	58.4 M	20.74	5.29
ResNet-50 + SE Layer* [15]	28.1 M	22.12	5.99
ResNet-50 + GE Layer* [14]	33.7 M	21.88	5.80
ResNet-50 (Baseline)	25.6 M	23.68	7.05
ResNet-50 + Cutout [3]	25.6 M	22.93	6.66
ResNet-50 + StochDepth [17]	25.6 M	22.46	6.27
ResNet-50 + Mixup [48]	25.6 M	22.58	6.40
ResNet-50 + Manifold Mixup [42]	25.6 M	22.50	6.21
ResNet-50 + DropBlock* [8]	25.6 M	21.87	5.98
ResNet-50 + Feature CutMix	25.6 M	21.80	6.06
ResNet-50 + CutMix	25.6 M	**21.40**	**5.92**

Table 3: ImageNet classification results based on ResNet-50 model. '*' denotes results reported in the original papers.

Experiments

Weakly-supervised object localization (WSOL) on CUB and ImageNet.

Method	CUB200-2011 Loc Acc (%)	ImageNet Loc Acc (%)
VGG-GAP + CAM [52]	37.12	42.73
VGG-GAP + ACoL* [49]	45.92	45.83
VGG-GAP + ADL* [2]	52.36	44.92
GoogLeNet + HaS* [33]	-	45.21
InceptionV3 + SPG* [50]	46.64	48.60
VGG-GAP + Mixup [48]	41.73	42.54
VGG-GAP + Cutout [3]	44.83	43.13
VGG-GAP + CutMix	**52.53**	**43.45**
ResNet-50 + CAM [52]	49.41	46.30
ResNet-50 + Mixup [48]	49.30	45.84
ResNet-50 + Cutout [3]	52.78	46.69
ResNet-50 + CutMix	**54.81**	**47.25**

Table 9: Weakly supervised object localization results on CUB200-2011 and ImageNet. * denotes results reported in the original papers.

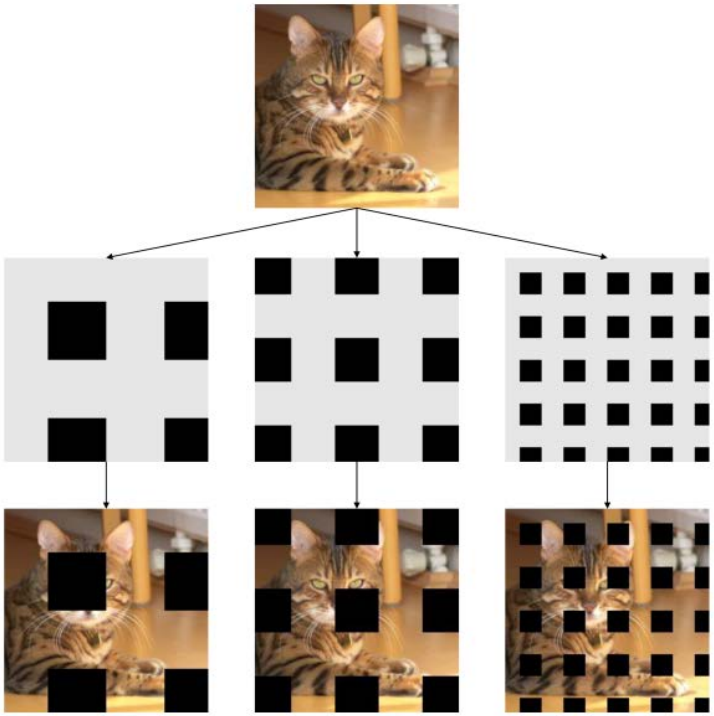
Experiments

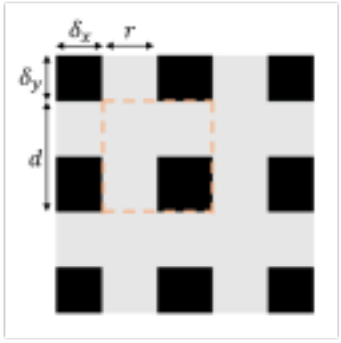
Transfer Learning

Backbone Network	ImageNet Cls Top-1 Error (%)	Detection		Image Captioning	
		SSD [24] (mAP)	Faster-RCNN [30] (mAP)	NIC [43] (BLEU-1)	NIC [43] (BLEU-4)
ResNet-50 (Baseline)	23.68	76.7 (+0.0)	75.6 (+0.0)	61.4 (+0.0)	22.9 (+0.0)
Mixup-trained	22.58	76.6 (-0.1)	73.9 (-1.7)	61.6 (+0.2)	23.2 (+0.3)
Cutout-trained	22.93	76.8 (+0.1)	75.0 (-0.6)	63.0 (+1.6)	24.0 (+1.1)
CutMix-trained	21.40	**77.6 (+0.9)**	**76.7 (+1.1)**	**64.2 (+2.8)**	**24.9 (+2.0)**

Table 10: Impact of CutMix on transfer learning of pretrained model to other tasks, object detection and image captioning.

기타 : GridMask





Model	Accuracy(%)
ResNet50 [6]	76.5
ResNet50 + Cutout [23]	77.1
ResNet50 + HaS [17]	77.2
ResNet50 + AutoAugment [3]	77.6
ResNet50 + GridMask (Our Impl.)	**77.9**
ResNet101 [26]	78.0
ResNet101 + GridMask (Our Impl.)	**79.1**
ResNet152 [10]	78.3
ResNet152 + GridMask (Our Impl.)	**79.7**

Table 2. This table summarizes the result of ImageNet. We can see our model improves the baseline of various models.

References

[1] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz, “mixup: Beyond empirical risk minimization,” in ICLR, 2018.

[2] 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.

[3] https://sangdooyun.github.io/papers/yun2019iccv_talk.pdf

[4] https://app.wandb.ai/authors/tfaugmentation/reports/Modern-Data-Augmentation-Techniques-for-Computer-Vision--VmlldzoxODA3NTQ

[5] https://towardsdatascience.com/cutmix-a-new-strategy-for-data-augmentation-bbc1c3d29aab

[6] https://www.kaggle.com/saife245/cutmix-vs-mixup-vs-gridmask-vs-cutout

[7] GridMask Data Augmentation https://arxiv.org/abs/2001.04086v1