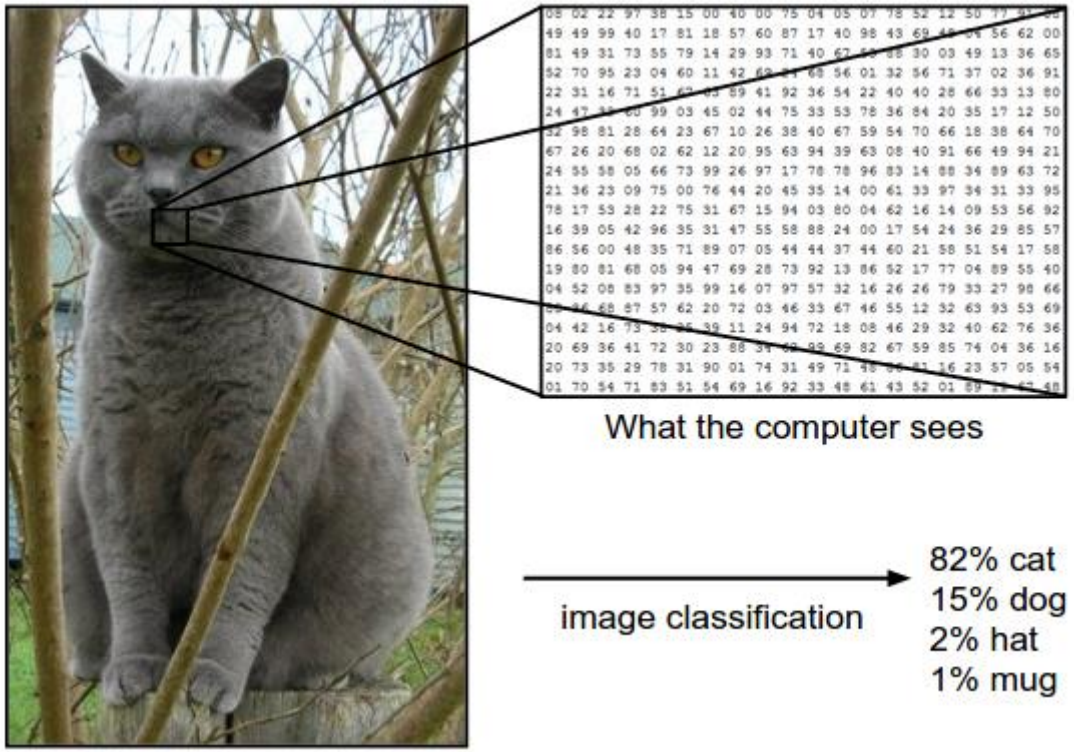
What we should focus on…

Taehyeon Kim
OsiLab

Image Classification



Supervised Learning!

Image is represented as one large 3-dimensional array of numbers.
Image has three color channels Red, Green, Blue (or RGB for short).

Cat: 248 pixels wide, 400 pixels tall

Then, it consists of 248 x 400 x 3 numbers, or a total of 297,600 numbers.

Image Classification

It is hard to make the algorithm directly, so instead of trying to specify what every one of the categories of interest look like directly in code, many provides the computer with many examples of each class and then develop learning algorithms that look at these examples and learn about the visual appearance of each class.

Data-Driven Approach

Then,

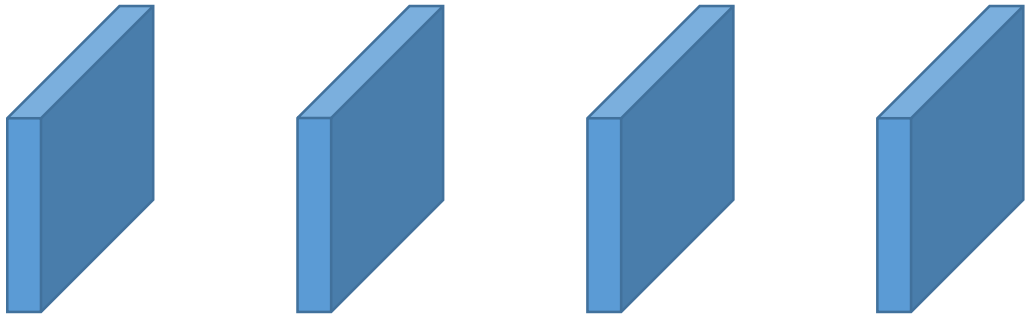
how to classify the data well with Deep Neural Network?
why we should use Deep Neural Network?
what kind of loss function we should use?
what is key idea of deep neural network?

MicroNet Challenge

Taehyeon Kim
OsiLab

Intro

Continuity
Lipschitz of weight matrix
Orthogonality



Input
Data Augmentation

Output
One-hot vector

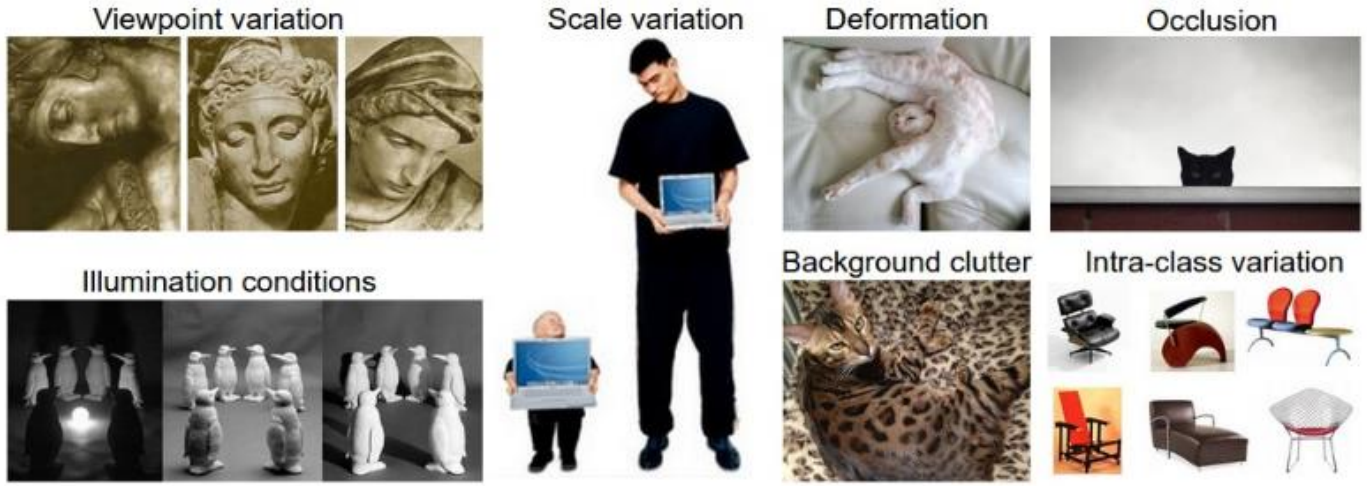
There are three main issues

Image Classification

Then, which part we should focus?

Viewpoint variation : A single instance of an object can be oriented in many ways with respect to the camera.
Scale variation : Visual classes often exhibit variation in their size (size in the real world, not only in terms of their extent in the image).
Deformation : Many objects of interest are not rigid bodies and can be deformed in extreme ways.
Occlusion : The objects of interest can be occluded. Sometimes only a small portion of an object (as little as few pixels) could be visible.
Illumination conditions : The effects of illumination are drastic on the pixel level.
Background clutter : The objects of interest may blend into their environment, making them hard to identify.
Intra-class variation : The classes of interest can often be relatively broad, such as chair. There are many different types of these objects, each with their own appearance.

1. 물체가 찍힌 구도
2. 이미지 크기
3. 물체 모양이??
4. 일부만 나온 경우
5. 조명 효과
6. 배경이랑 비슷하게 생긴…
7. 범주가 넓은 경우



CIFAR100 data augmentation

Task : required to achieve 80% top-1 accuracy on the CIFAR100 test set.

Description
CIFAR100 dataset is a bit different with others such CIFAR10 or ImageNet.

Composition
- Training dataset: 50,000 (each class has 500 images)
- Test dataset: 10,000
- Number of classes: 100
- Image size: 32 x 32 x 3 (3,072)

The number of images per each class is less than other datasets.
Namely, with data-driven approach, for especially CIFAR100, maybe one of the most important job is data augmentation.

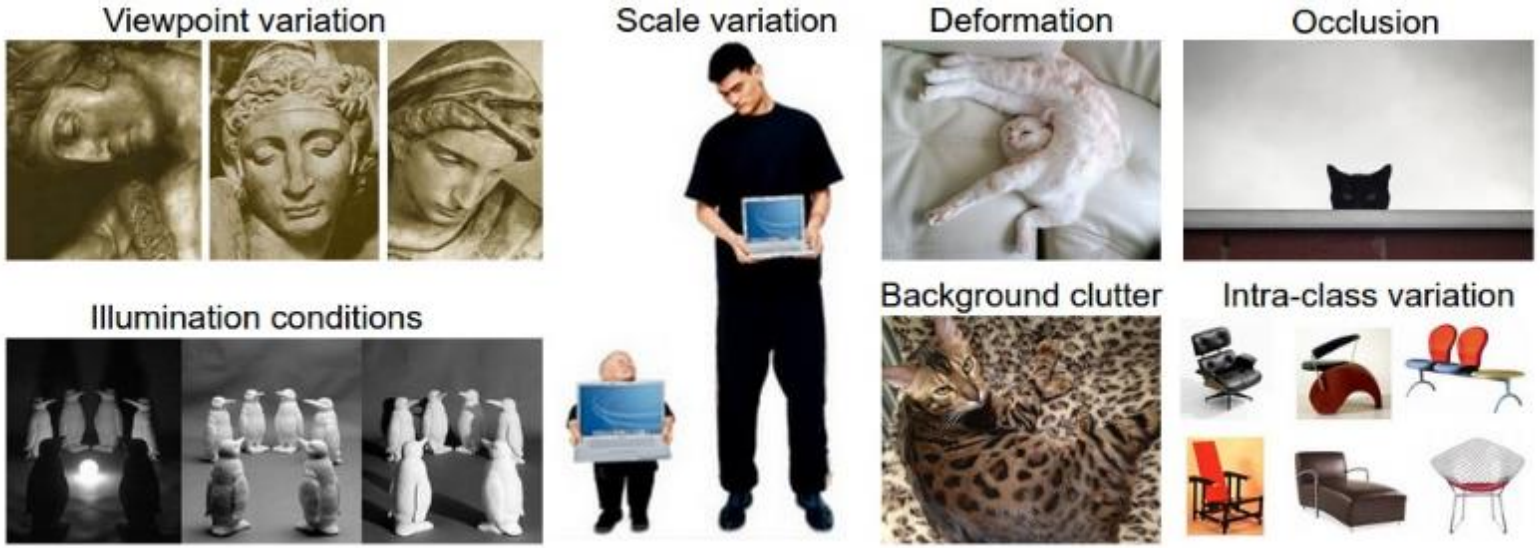
CIFAR10	IMAGENET	CIFAR100
5,000	732 – 1300 (Most: 1000)	500

CIFAR100 data augmentation

Then,

- Is it okay to use all below augmentation strategies without any consideration?
- Don’t we have to arrange the following methods any way?

The answer is **No.**

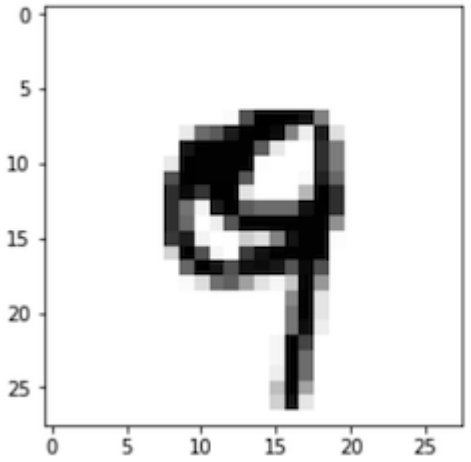


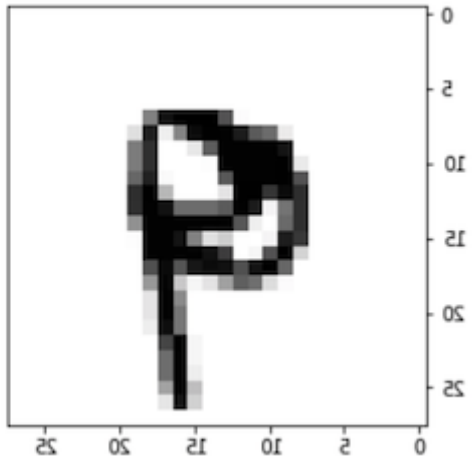
CIFAR100 data augmentation

Sometimes, this method can harm the performance
For example, let's see MNIST dataset.

If we flip the number '9', then the flipped image is useful for network? will be the network trained well?

No. It's not. Instead, the accuracy decreases.





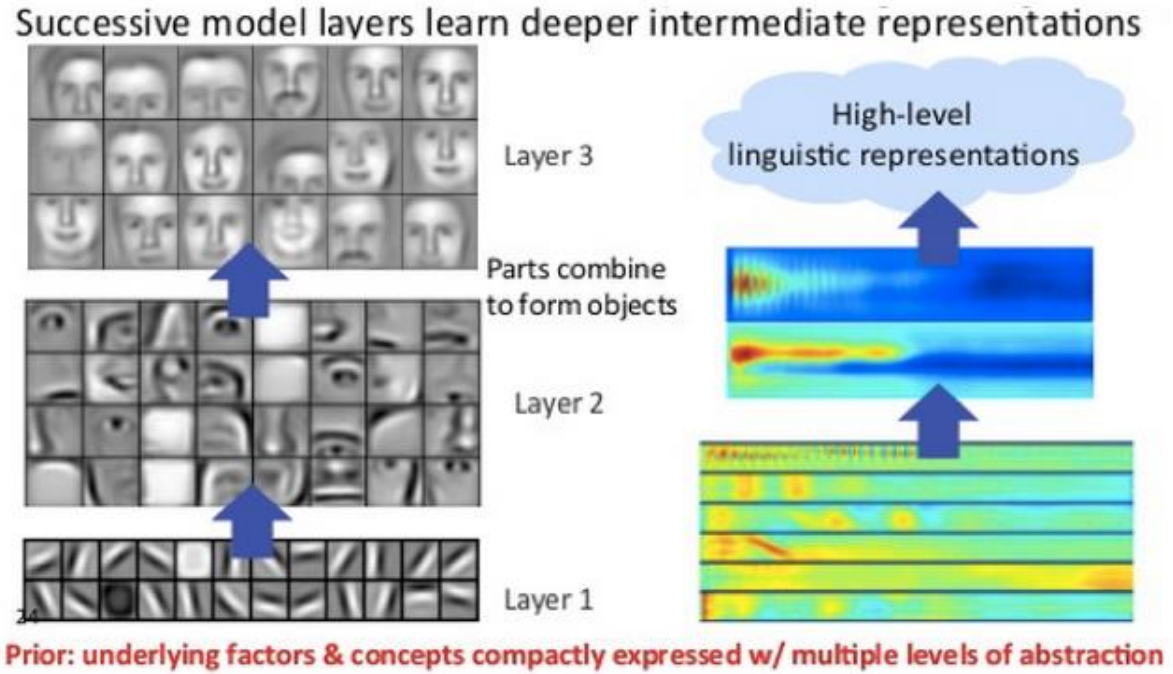
Therefore, data augmetation can improve your performance,
but it is not always true.
The more important thing is to find a proper augmentation strategy for your dataset.

CIFAR100 architecture

Main Approach of Deep learning

Each layer of nodes trains on a distinct set of features based on the previous layer's output. The further you advance into the neural net, the more complex the features your nodes can recognize, since they aggregate and recombine features from the previous layer.

Feature Hierarchy

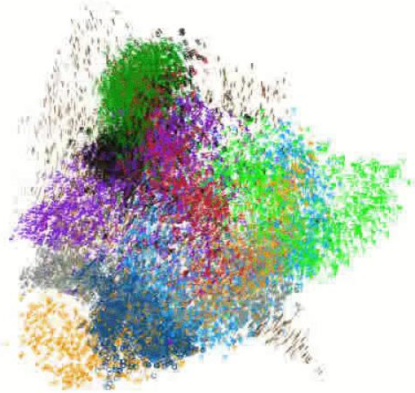


CIFAR100 architecture

My approach…

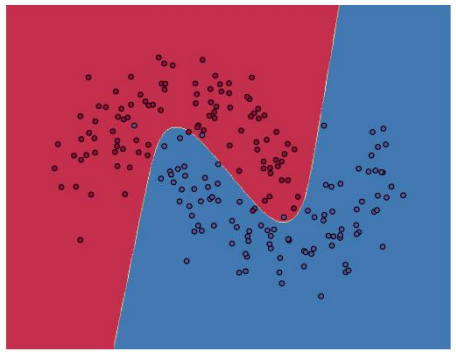
The main principle of deep learning architecture is to inject data towards high dimensional space.
Through this, they make the classification easier.

Increasing dimension ↑



high dimensional space

Decreasing dimension ↓

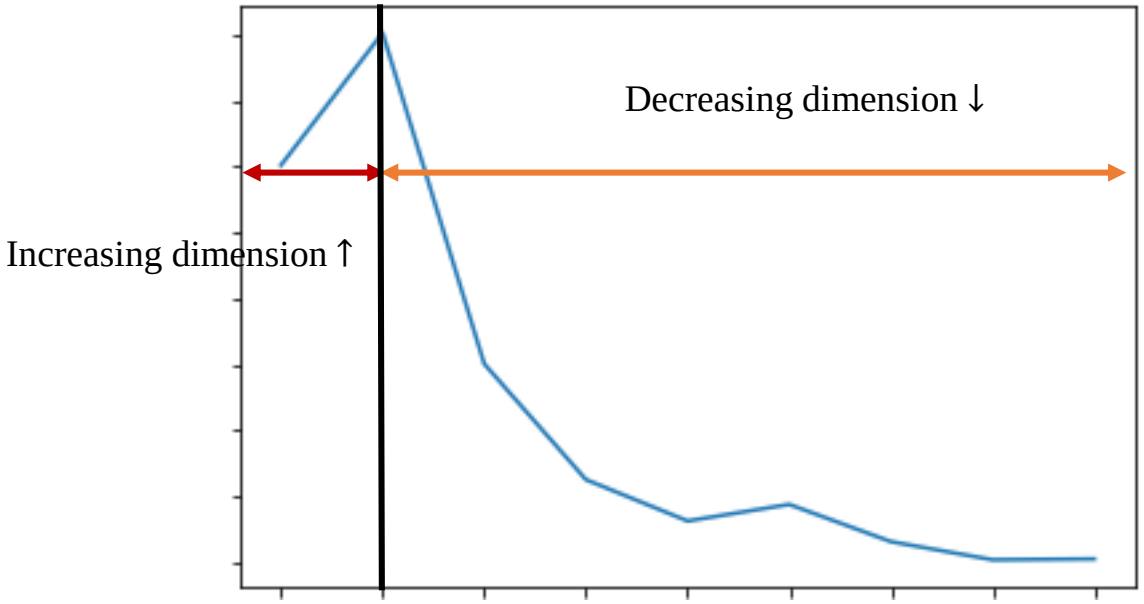


Decision Boundary

CIFAR100 architecture design

In many architecture, they pay more attention on decreasing the dimension of input data.

Namely, maybe, the key of CNN is finding the way how to reducing the dimension effectively.



I want to find the optimal curve of above graph…

Input	Operator	exp size	#out	SE	NL	s
$224^2 \times 3$	conv2d	-	16	-	HS	2
$112^2 \times 16$	bneck, 3x3	16	16	-	RE	1
$112^2 \times 16$	bneck, 3x3	64	24	-	RE	2
$56^2 \times 24$	bneck, 3x3	72	24	-	RE	1
$56^2 \times 24$	bneck, 5x5	72	40	✓	RE	2
$28^2 \times 40$	bneck, 5x5	120	40	✓	RE	1
$28^2 \times 40$	bneck, 5x5	120	40	✓	RE	1
$28^2 \times 40$	bneck, 3x3	240	80	-	HS	2
$14^2 \times 80$	bneck, 3x3	200	80	-	HS	1
$14^2 \times 80$	bneck, 3x3	184	80	-	HS	1
$14^2 \times 80$	bneck, 3x3	184	80	-	HS	1
$14^2 \times 80$	bneck, 3x3	480	112	✓	HS	1
$14^2 \times 112$	bneck, 3x3	672	112	✓	HS	1
$14^2 \times 112$	bneck, 5x5	672	160	✓	HS	2
$7^2 \times 160$	bneck, 5x5	960	160	✓	HS	1
$7^2 \times 160$	bneck, 5x5	960	160	✓	HS	1
$7^2 \times 160$	conv2d, 1x1	-	960	-	HS	1
$7^2 \times 960$	pool, 7x7	-	-	-	-	1
$1^2 \times 960$	conv2d 1x1, NBN	-	1280	-	HS	1
$1^2 \times 1280$	conv2d 1x1, NBN	-	k	-	-	1

Figure. MobileNet V3

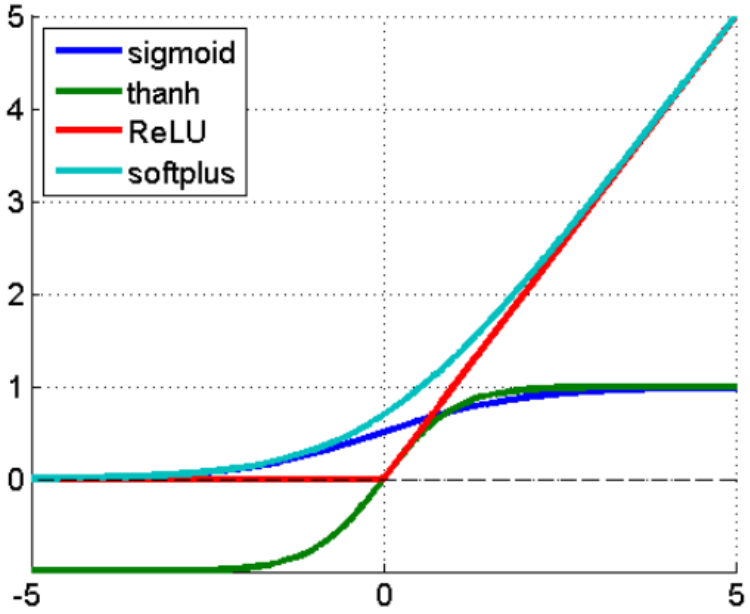
To get this,

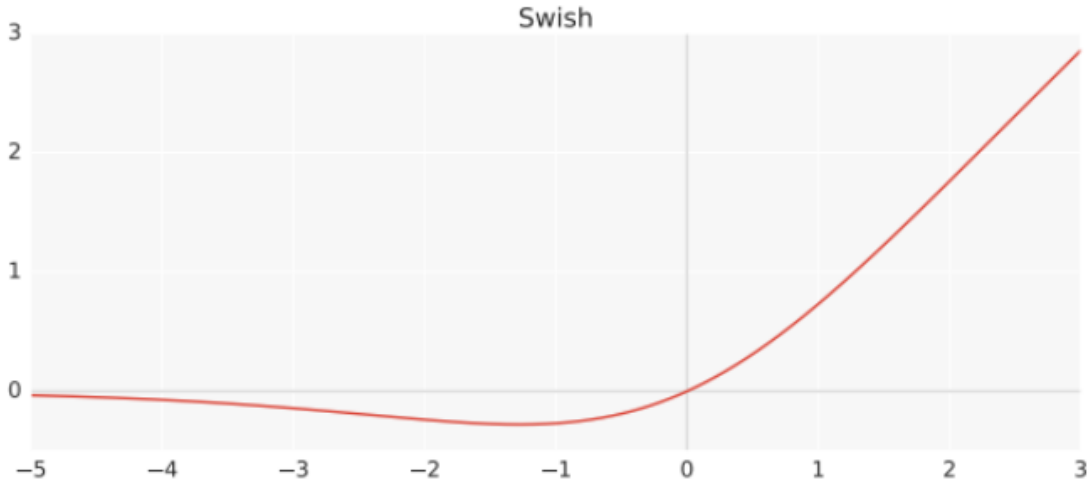
1. Analyze the information gain and loss with the layerwise fashion.
2. Make visualization of each input tensor.

CIFAR100 activation function

To reduce the information loss when passing the next layer, nowadays, many use ‘swish’ activation function.

swish
f(x) = x · sigmoid(x)





Recently, the ‘hard swish’ function which approximates the swish is also proposed to reduce the computational costs.

Then, what should we do

Taehyeon Kim
OsiLab

CIFAR100 data augmentation

Now, I focus on the data augmentation policy.
Also, study about the architecture design strategy, but still there's no remarkable analysis, so today i'll introduce some recent data augmentation policies.

I can improve the accuracy with 'AutoAugment method' from 74.25% to 79.08%.
4.83% Improvement!!!

Parameters : 1M
Flops: 0.98B

HighpathNet (Mine)	No	With AutoAugment
Accurcay	74.25%	79.08%

But we cannot use autoaugment policy… so we have to make new augment rule.!!

CIFAR100 data augmentation

These are the state-of-the-art augment polices or popular augment polices.

- **Cutout [arXiv'17]**
- **Mixup [ICLR'18]**
- **DropBlock [NeurIPS'18]**
- Manifold Mixup [ICML'19]
- **AutoAugment [CVPR'19]**
- Cutmix [ICCV'19]
- label smoothing [Under review]

CIFAR100 data augmentation

Cutout [arXiv'17]

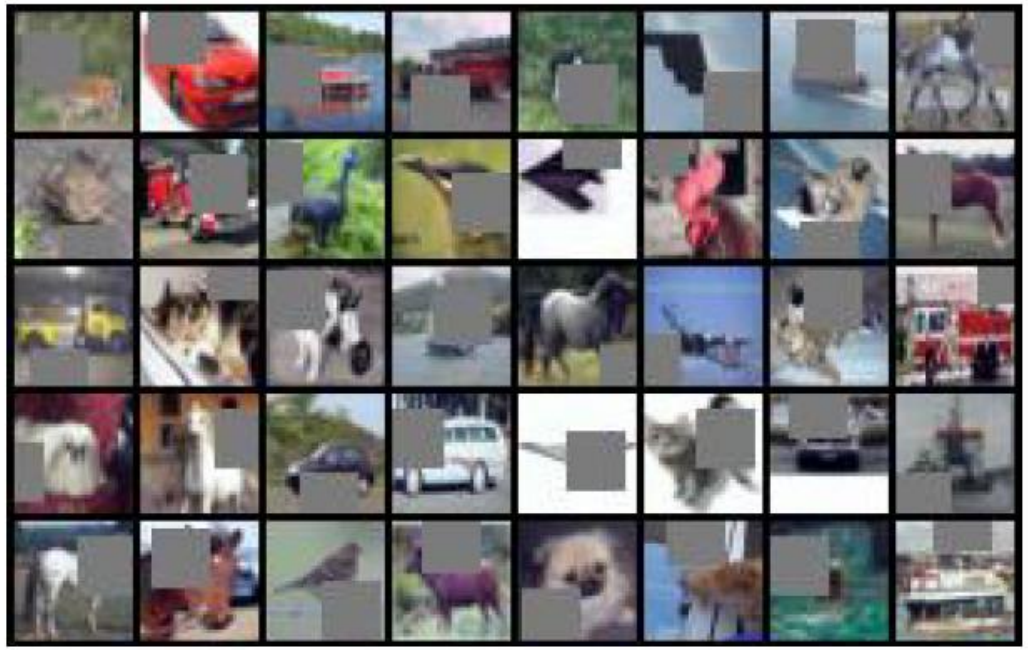
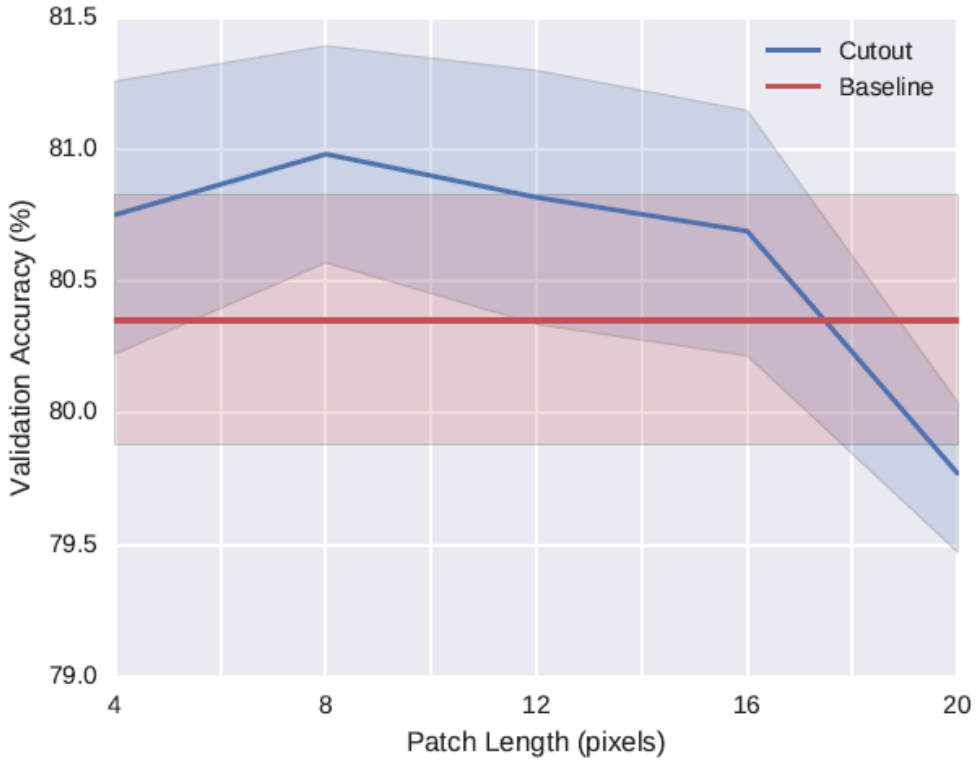


Figure: cutout applied to images from the CIFAR-10 dataset



This can improve up to 1.5%.

Pros: Very simple
Cons: Not work on ImageNet data

CIFAR100 data augmentation

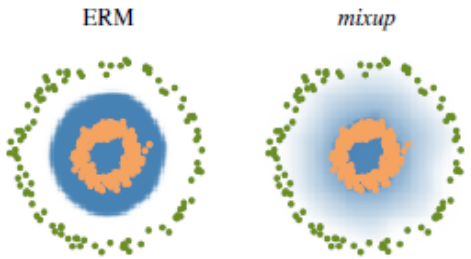
Mixup [ICLR'18]

$$\tilde{x} = \lambda x_i + (1-\lambda)x_j, \quad \text{where } x_i, x_j \text{ are raw input vectors}$$
$$\tilde{y} = \lambda y_i + (1-\lambda)y_j, \quad \text{where } y_i, y_j \text{ are one-hot label encodings}$$

$\lambda \sim \text{Beta}(\alpha, \alpha)$, for $\alpha \in (0, \infty)$.

```
# y1, y2 should be one-hot vectors
for (x1, y1), (x2, y2) in zip(loader1, loader2):
    lam = numpy.random.beta(alpha, alpha)
    x = Variable(lam * x1 + (1. - lam) * x2)
    y = Variable(lam * y1 + (1. - lam) * y2)
    optimizer.zero_grad()
    loss(net(x), y).backward()
    optimizer.step()
```

(a) One epoch of *mixup* training in PyTorch.



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

Figure 1: Illustration of *mixup*, which converges to ERM as $\alpha \to 0$.

From paper, this is called vicinity distribution that measures the probability of finding the virtual feature-target pair $(\tilde{x}, \tilde{y})$.
This encourages the model f to behave linearly in-between training examples.

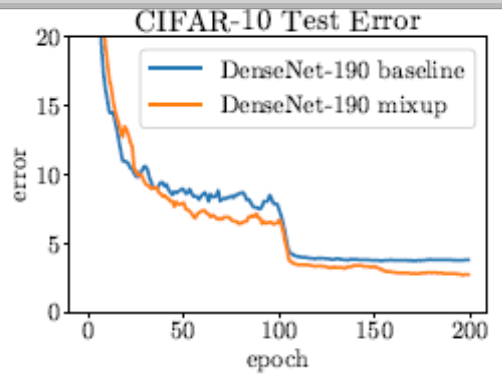
Pros: Very simple
Cons: Not work on ImageNet data

CIFAR100 data augmentation

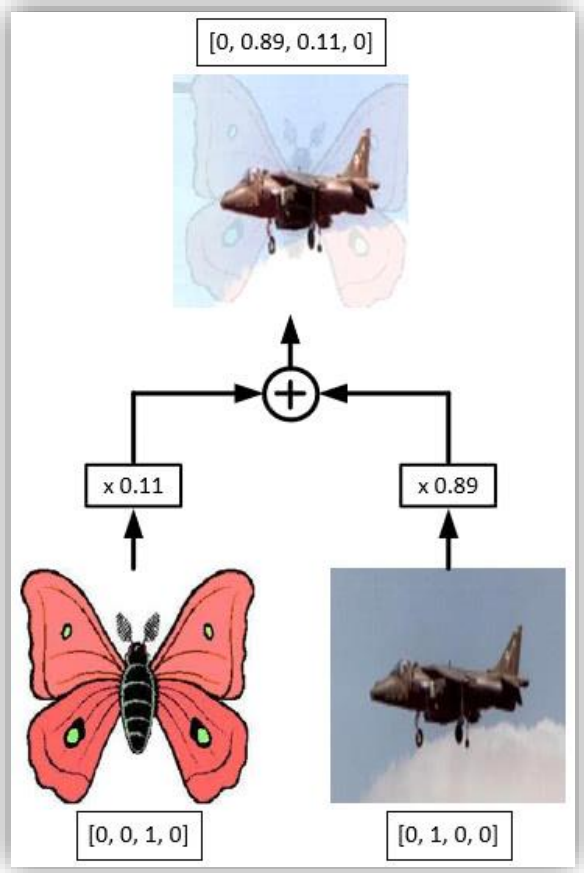
Mixup [ICLR'18]

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.



From paper, this is called vicinity distribution that measures the probability of finding the virtual feature-target pair $(\tilde{x}, \tilde{y})$.
This encourages the model f to behave linearly in-between training examples.

CIFAR100 data augmentation

Mixup [ICLR'18]

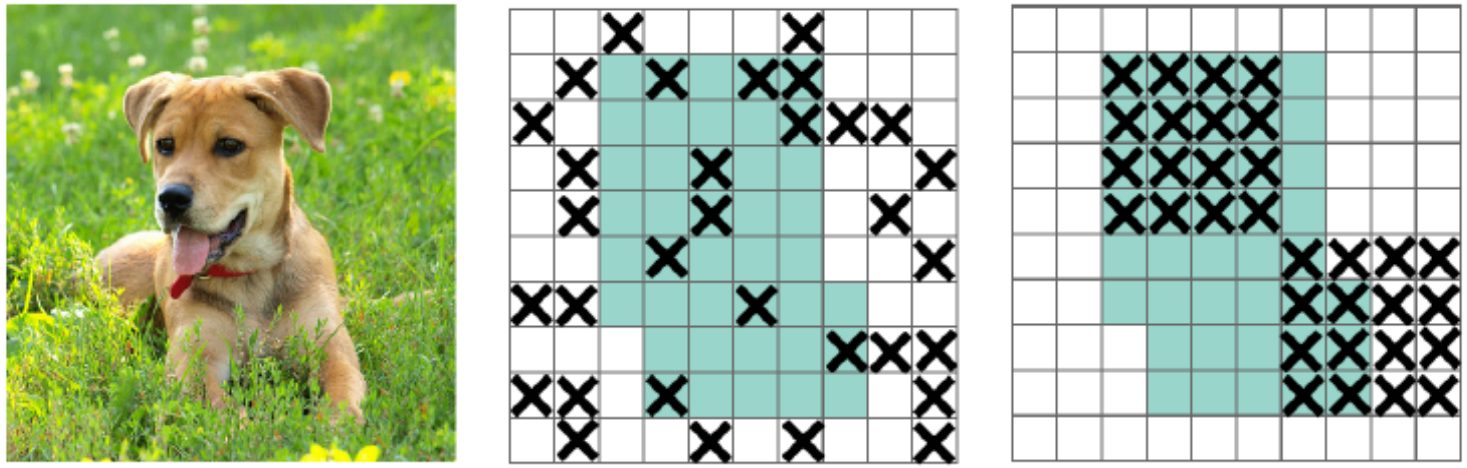
1. They also tried to use combinations of three or more examples with weights sampled from a Dirichlet distribution does not provide further gain, but increases the computation cost of mixup.

2. Their current implementation uses a single data loader to obtain one minibatch, and then mixup is applied to the same minibatch after random shuffling.

3. Interpolating only between inputs with equal label did not lead to the performance gains of mixup discussed in the sequel.

CIFAR100 data augmentation

DropBlock [NeurIPS'18]



(a) (b) (c)

Figure 1: (a) input image to a convolutional neural network. The green regions in (b) and (c) include the activation units which contain semantic information in the input image. Dropping out activations at random is not effective in removing semantic information because nearby activations contain closely related information. Instead, dropping continuous regions can remove certain semantic information (e.g., head or feet) and consequently enforcing remaining units to learn features for classifying input image.

CIFAR100 Feature regularization

DropBlock [NeurIPS'18]

Algorithm 1 DropBlock

1: **Input:**output activations of a layer (A), $block\_size$, γ, $mode$
2: **if** $mode$ == $Inference$ **then**
3: return A
4: **end if**
5: Randomly sample mask M: $M_{i,j} \sim Bernoulli(\gamma)$
6: For each zero position $M_{i,j}$, create a spatial square mask with the center being $M_{i,j}$, the width, height being $block\_size$ and set all the values of M in the square to be zero (see Figure 2).
7: Apply the mask: $A = A \times M$
8: Normalize the features: $A = A \times \textbf{count}(M)/\textbf{count\_ones}(M)$

Gradually, increasing number of dropped units during training leads to better accuracy and more robust to hyperparameter choices.

Model	top-1(%)	top-5(%)
ResNet-50	76.51 ± 0.07	93.20 ± 0.05
ResNet-50 + dropout (kp=0.7) [1]	76.80 ± 0.04	93.41 ± 0.04
ResNet-50 + DropPath (kp=0.9) [17]	77.10 ± 0.08	93.50 ± 0.05
ResNet-50 + SpatialDropout (kp=0.9) [20]	77.41 ± 0.04	93.74 ± 0.02
ResNet-50 + Cutout [23]	76.52 ± 0.07	93.21 ± 0.04
ResNet-50 + AutoAugment [27]	77.63	93.82
ResNet-50 + label smoothing (0.1) [28]	77.17 ±0.05	93.45 ±0.03
ResNet-50 + DropBlock, (kp=0.9)	78.13 ± 0.05	94.02 ± 0.02
ResNet-50 + DropBlock (kp=0.9) + label smoothing (0.1)	78.35 ± 0.05	94.15 ± 0.03

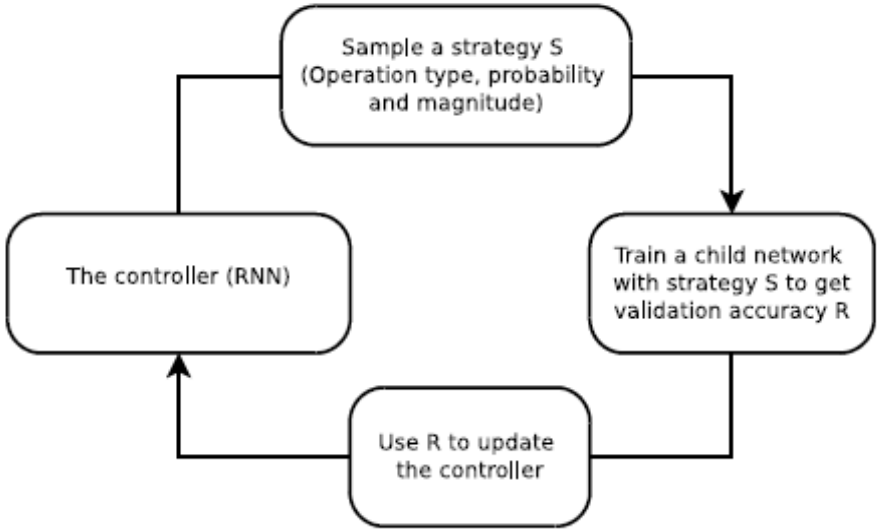
CIFAR100 data augmentation

AutoAugment [CVPR'19]

They use Reinforcement Learning as the search algorithm.

Key: The policy S will be used to train a neural network with a fixed architecture, whose validation accuracy R will be sent back to update the controller.

Dataset	GPU hours	Best published results	Our results
CIFAR-10	5000	2.1	1.5
CIFAR-100	0	12.2	10.7
SVHN	1000	1.3	1.0
Stanford Cars	0	5.9	5.2
ImageNet	15000	3.9	3.5

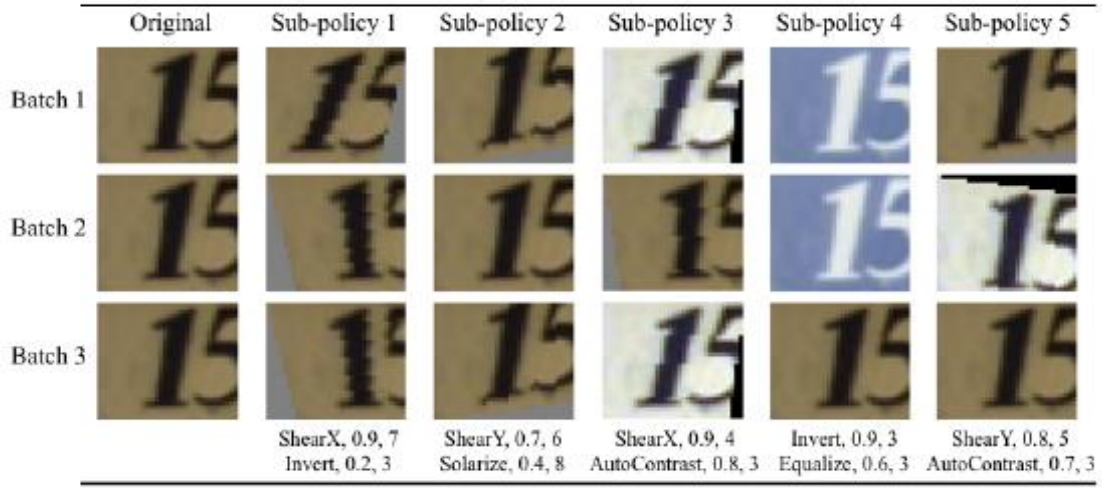


CIFAR100 data augmentation

AutoAugment [CVPR'19]

They use Reinforcement Learning as the search algorithm.

Key: The policy S will be used to train a neural network with a fixed architecture, whose validation accuracy R will be sent back to update the controller.



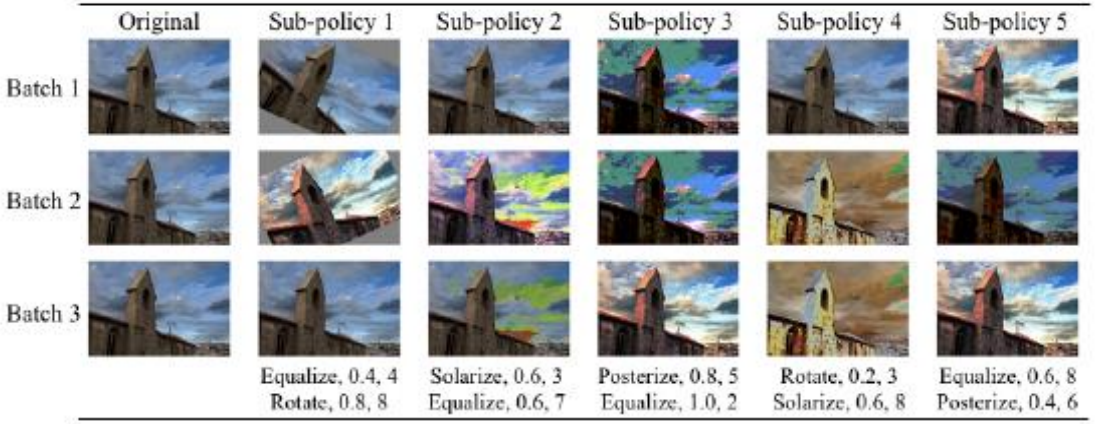


Figure 3. One of the successful policies on ImageNet. As described in the text, most of the policies found on ImageNet used color-based transformations.

Thank You

Taehyeon Kim
OsiLab

Reference

1. http://cs231n.github.io/classification/
2.