

# Deep Convolutional Network (I)

Jin Young Choi

Seoul National University

# References

- Andrew Ng's ML class
  - <https://class.coursera.org/ml-003/lecture>
  - <http://www.holehouse.org/mlclass/> (note)
- Convolutional Neural Networks for Visual Recognition.
  - <http://cs231n.github.io/>
- TensorFlow
  - <https://www.tensorflow.org>
  - <https://github.com/aymericdamien/TensorFlow-Examples>
- TensorFlow Tutorial
  - <https://hunkim.github.io/ml/>
  - <https://github.com/Hvass-Labs/TensorFlow-Tutorials>

# Outline

- Convolution and Cross-Correlation
- Convolutional Neural Networks Basics
- Fine-Tuning for Small Datasets
  - Learning Rates in Fine-Tuning
  - Structure Design for Fine-Tuning
- Knowledge Distillation
  - Transfer of Activation Boundary
  - Design Aspects for Feature Distillation

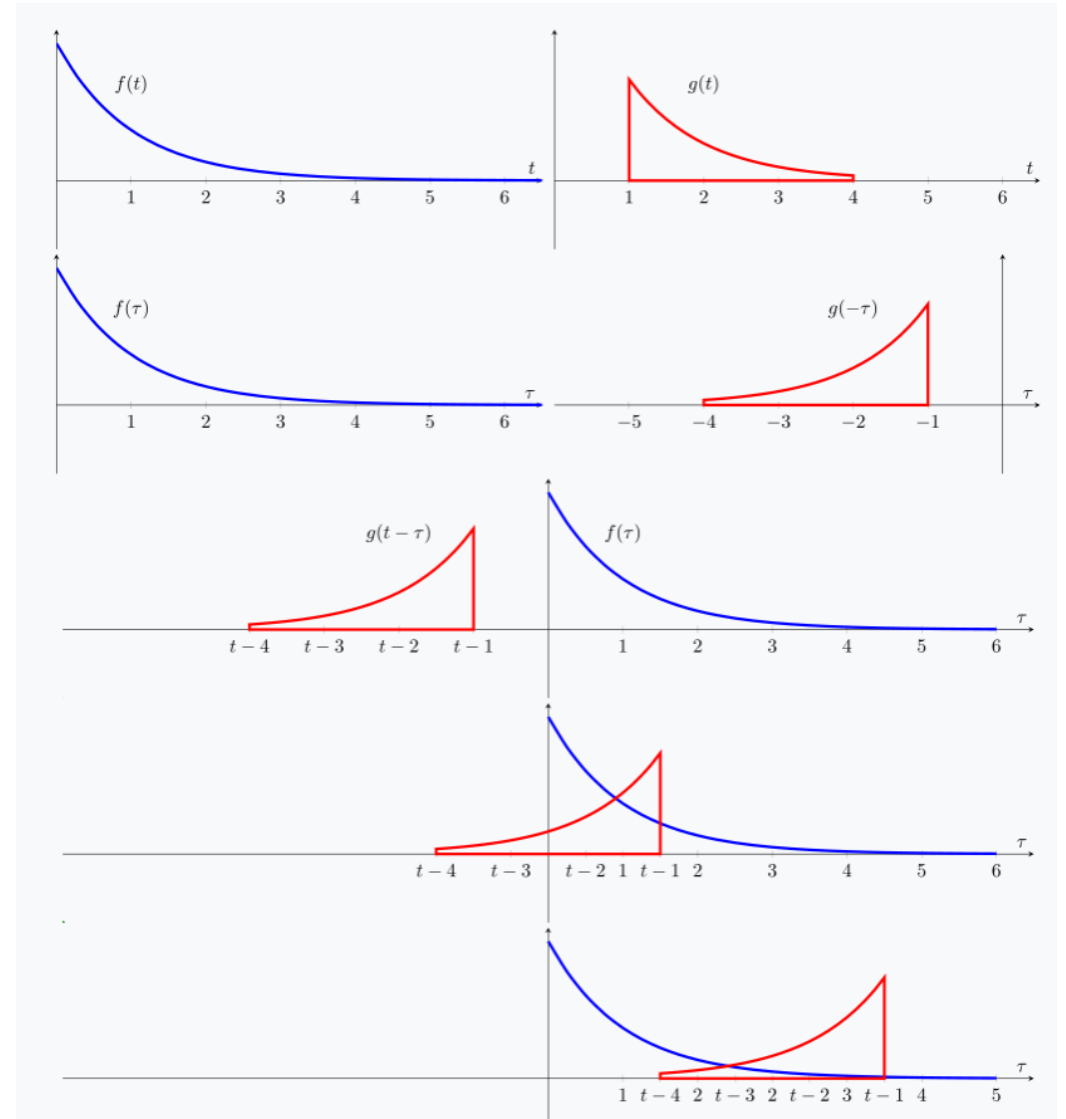
# Convolution and Cross-Correlation

- Convolution Integral (Temporal)

$$(f * g)(t) \triangleq \int_{-\infty}^{\infty} f(\tau)g(t - \tau) d\tau$$

$$(f * g)(t) \triangleq \int_{-\infty}^{\infty} f(t - \tau)g(\tau) d\tau$$

$$L \circ (f * g) = F(s)G(s)$$





# Convolution and Cross-Correlation

- **Convolution Sum (Temporal)**

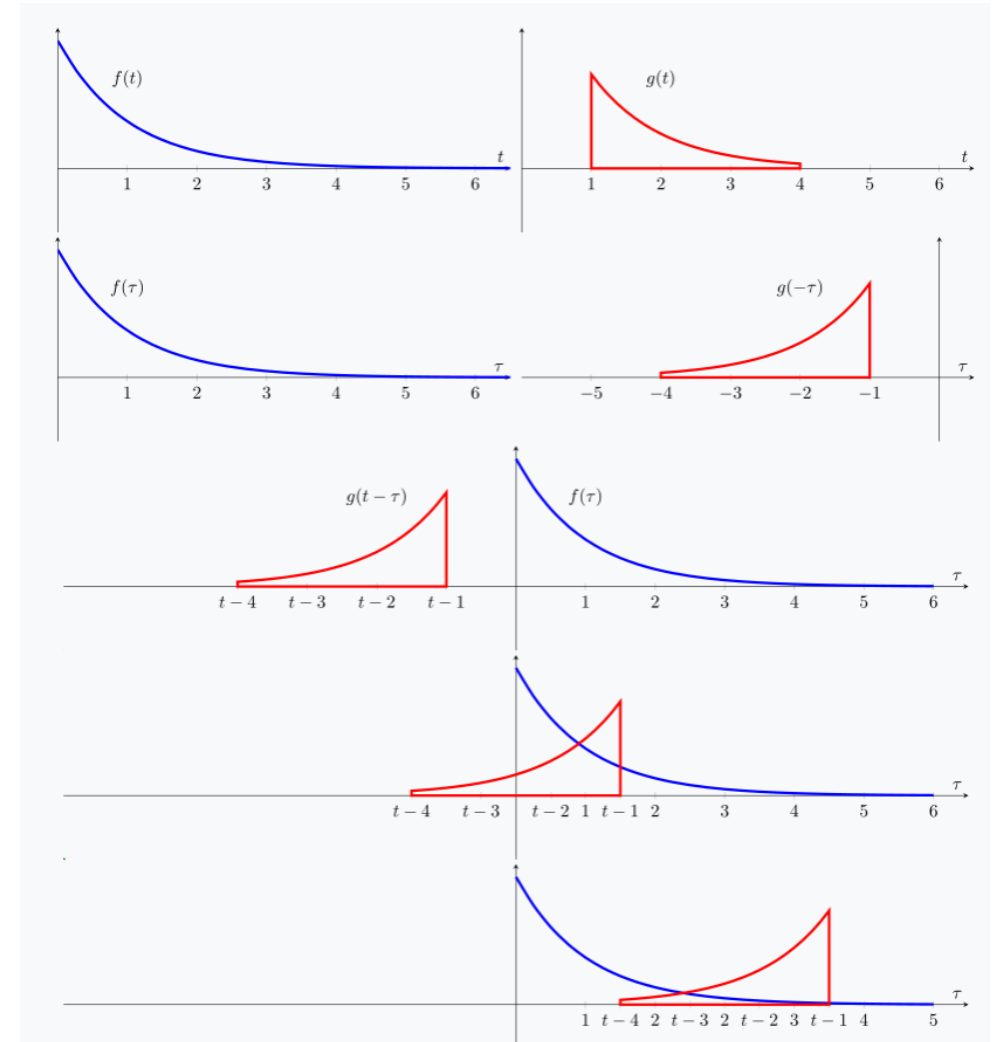
$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m]g[n - m]$$

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[n - m]g[m]$$

- **Cross correlation** (without kernel flipping)

$$\rightarrow f * g[n] = \sum_{m=-\infty}^{\infty} f[m]g[m - n]$$

- Many CNNs implement cross-correlation but call it convolution (without kernel flipping)



# Convolution and Cross-Correlation

## Circular Convolution Sum

$$h_k = f * g = \sum_{i=0}^{n-1} f_i g_{k-i}$$

$$g \triangleq (\dots, g_{n-1}, g_0, g_1, \dots, g_{n-1}, g_0, g_1, \dots, g_{n-1}, \dots)$$

$$f \triangleq (f_0, f_1, \dots, f_{n-1})$$

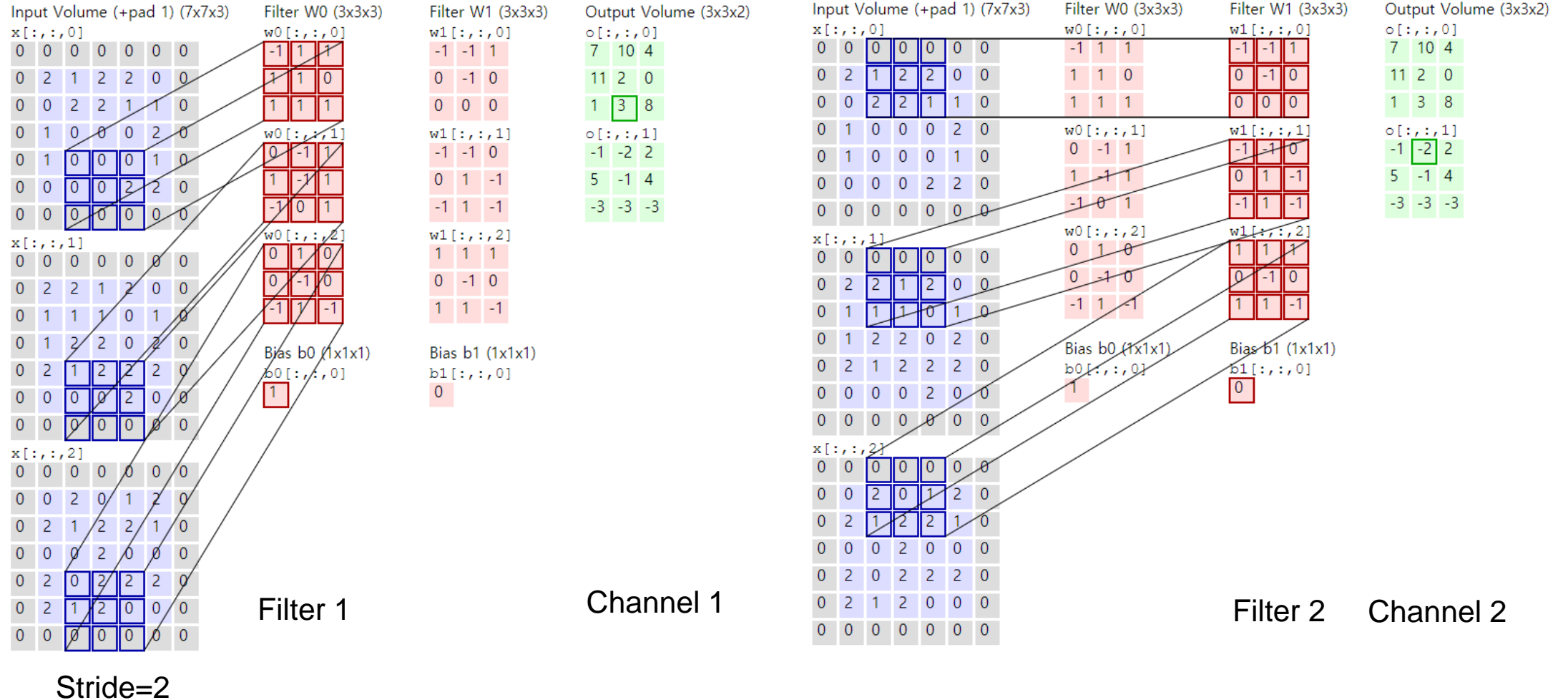
$$h \triangleq (\dots, h_{n-1}, h_0, h_1, \dots, h_{n-1}, h_0, h_1, \dots, h_{n-1}, \dots)$$

$$f * g = \begin{bmatrix} g_0 & g_{n-1} & \cdots & g_2 & g_1 \\ g_1 & g_0 & g_{n-1} & \cdots & g_2 \\ \vdots & g_1 & g_0 & \ddots & \vdots \\ g_{n-2} & \vdots & \ddots & \ddots & g_{n-1} \\ g_{n-1} & g_{n-2} & \cdots & g_1 & g_0 \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_{n-2} \\ f_{n-1} \end{bmatrix}$$

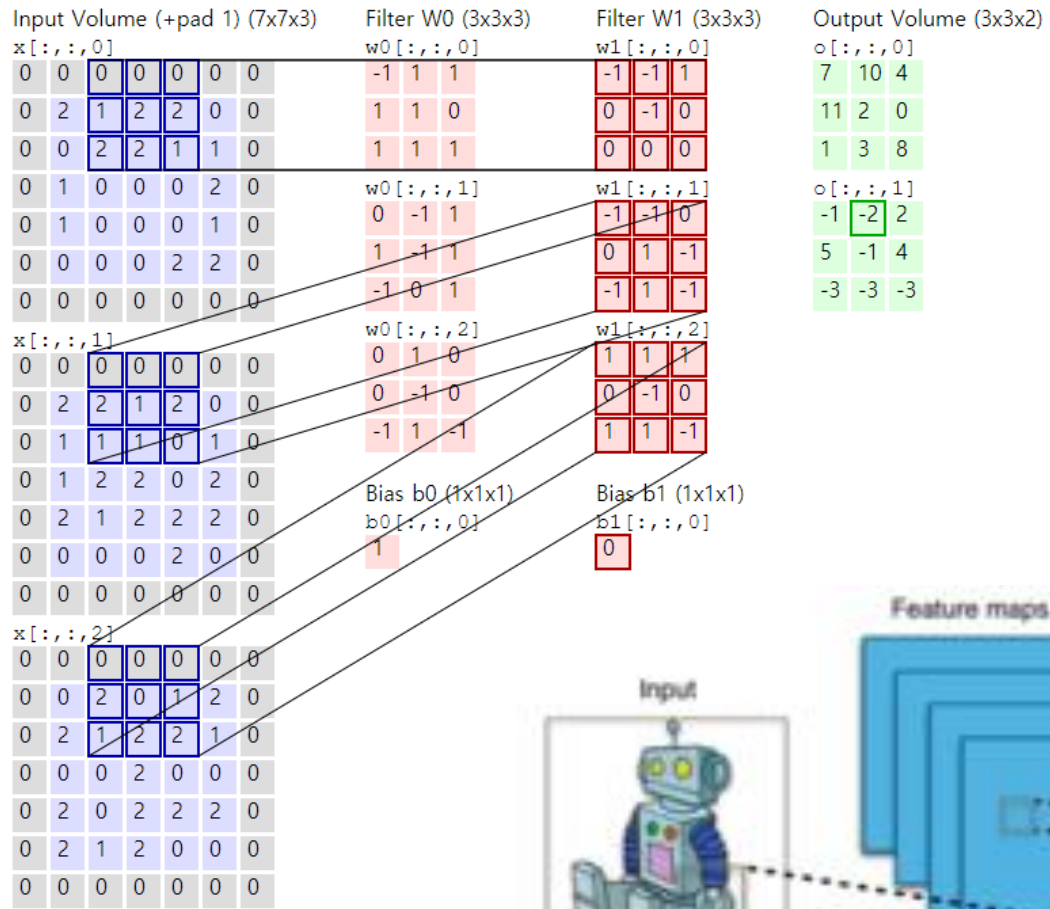
Circulant Matrix

# Convolution and Cross-Correlation

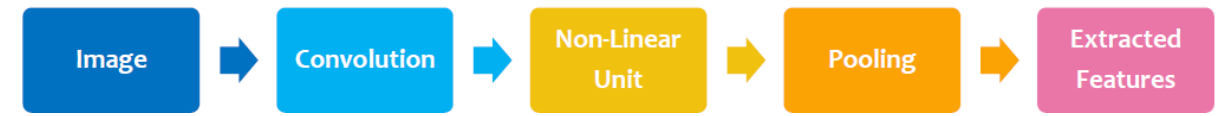
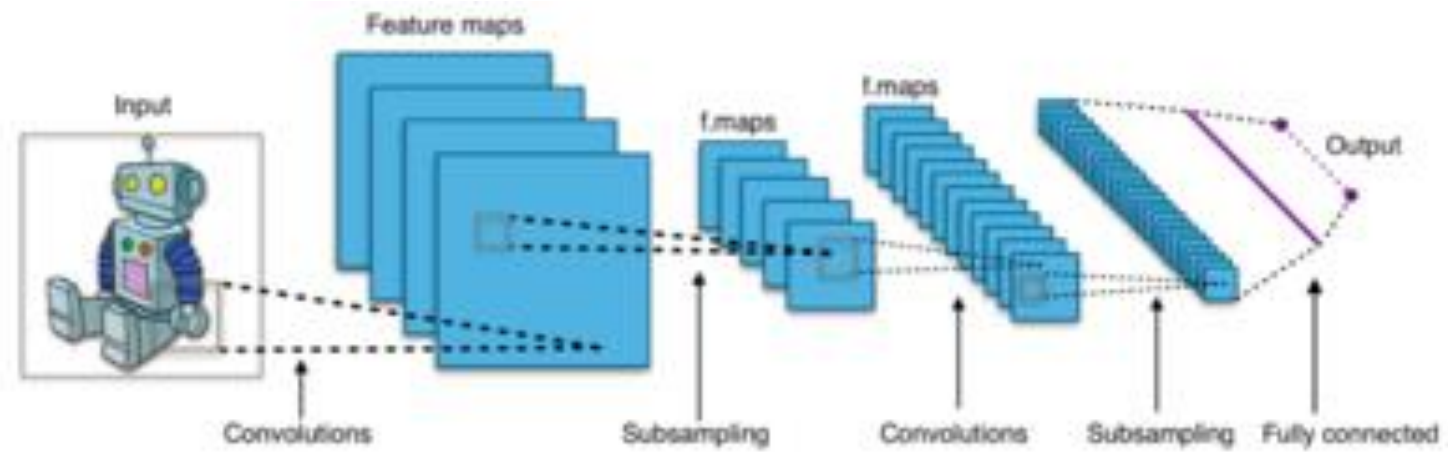
## Convolution Sum (Spatial)



# Convolutional Neural Networks (CNNs)



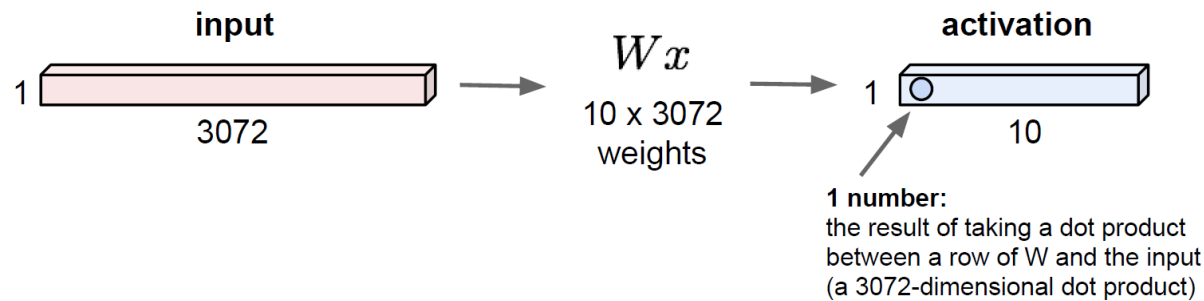
- Three key ideas behind CNN
    - Local connectivity
    - Invariance to location
    - Invariance to translation
- Convolution layer
- Pooling layer



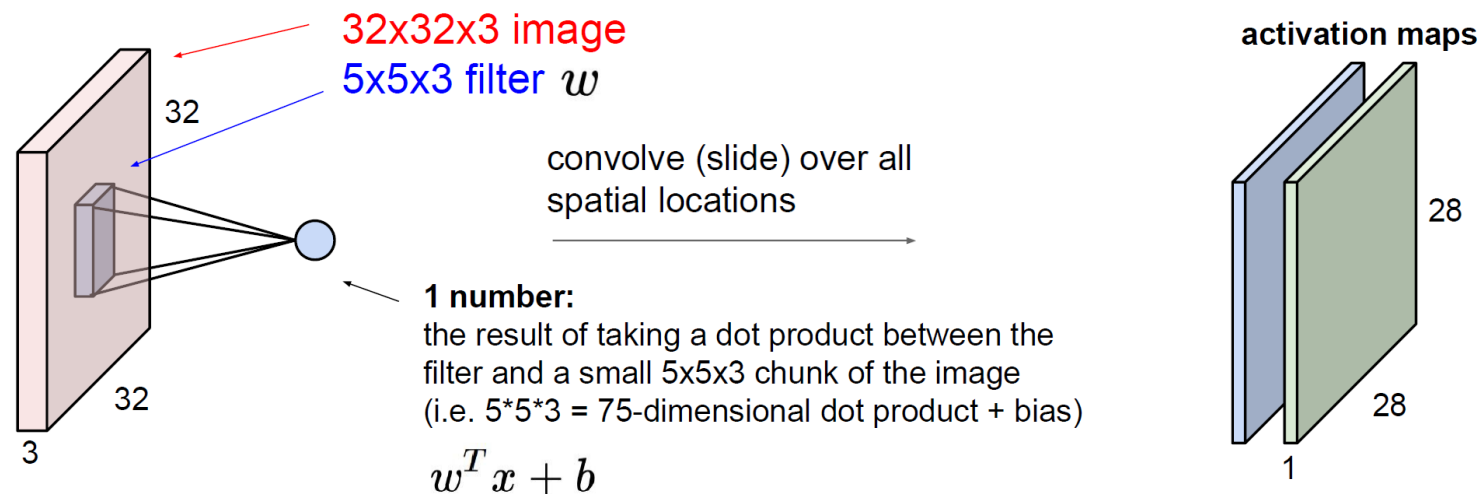
# Fully Connected Layer and Convolution Layer

- Fully connected layer

32x32x3 image -> stretch to 3072 x 1



- Convolution layer



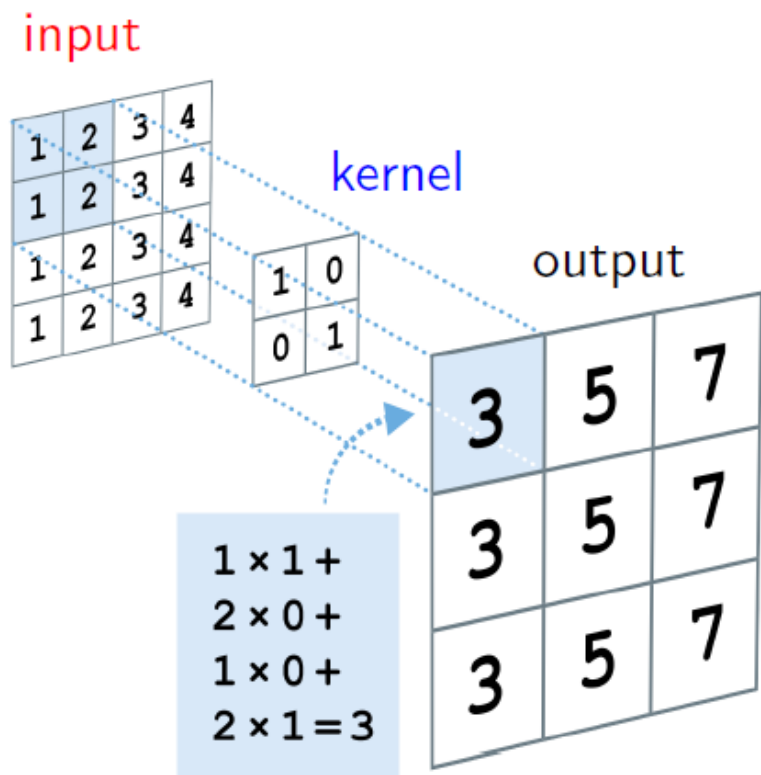
# Convolution Layer

- Input is convolved with a set of filters, giving feature maps (without kernel flipping)

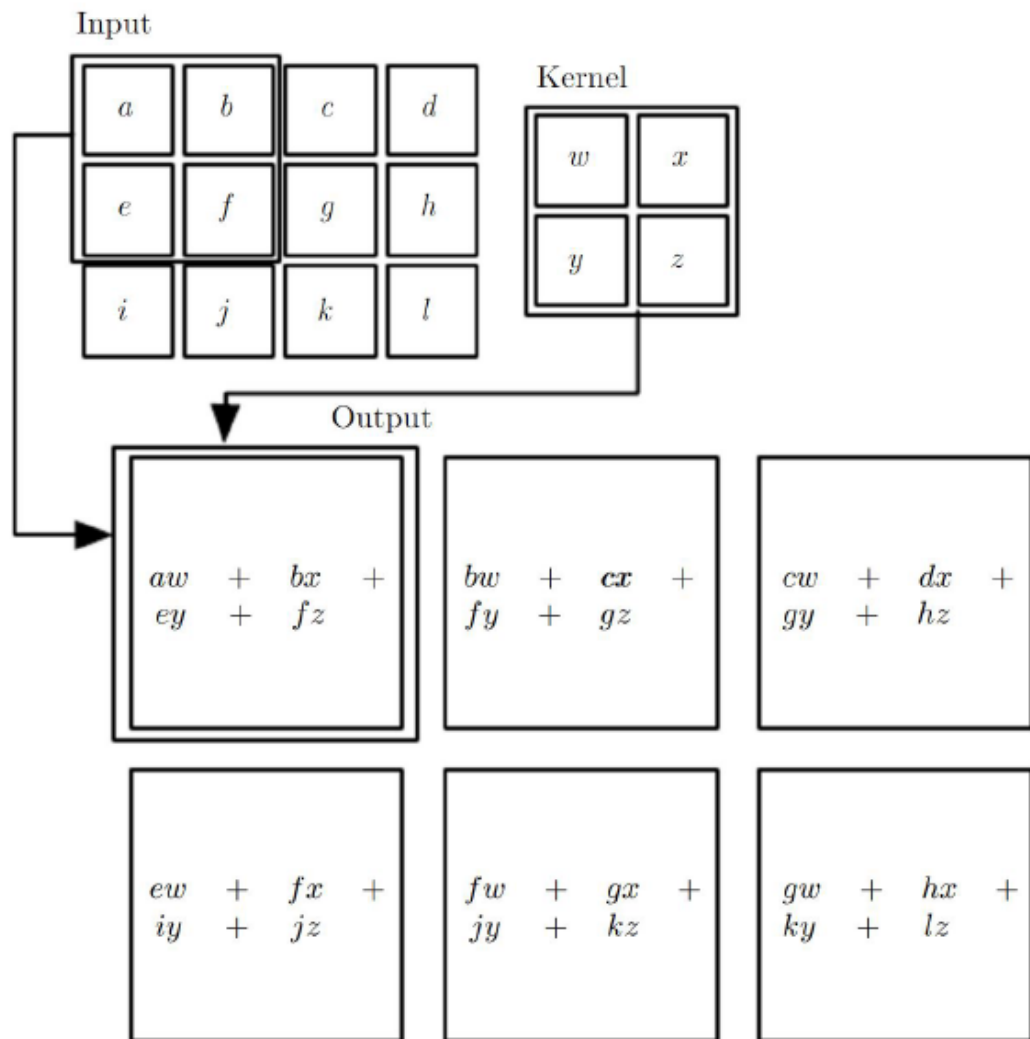


- Filter (a.k.a. kernel or convolution matrix)
  - Different filters extract different features (e.g. edges)
  - Filter weights: trained with data
  - Weight sharing & local connectivity

# Convolution Example



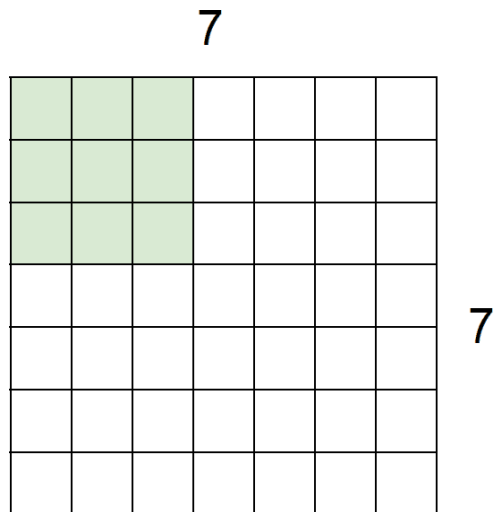
source: [Angermueller et al., 2016]



source: [Goodfellow et al., 2016]

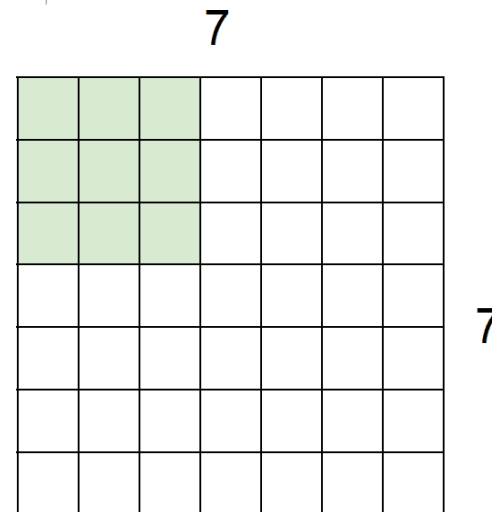
# Convolution with Stride

- The number of pixels between adjacent receptive fields = down-sampling the output of full convolution function



7x7 input  
3x3 filter with stride 1

→ 5x5 output



7x7 input  
3x3 filter with stride 2

→ 3x3 output

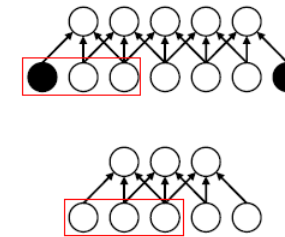


# Zero-padding

- Implicitly zero-pad input to make it wider

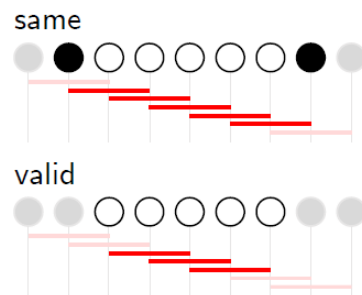
type	output	# zeros padded		
		left	right	total
same	$m$	$k$ even	$\lfloor \frac{k-1}{2} \rfloor$	$\lfloor \frac{k-1}{2} \rfloor + 1$
		$k$ odd	$\frac{k-1}{2}$	$\frac{k-1}{2}$
valid	$m - (k - 1)$	0	0	0

( $m$ : input width;  $k$ : kernel width;  $s = 1$ )

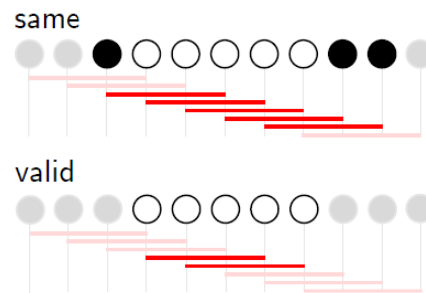


- Examples

$m = 5, k = 3$

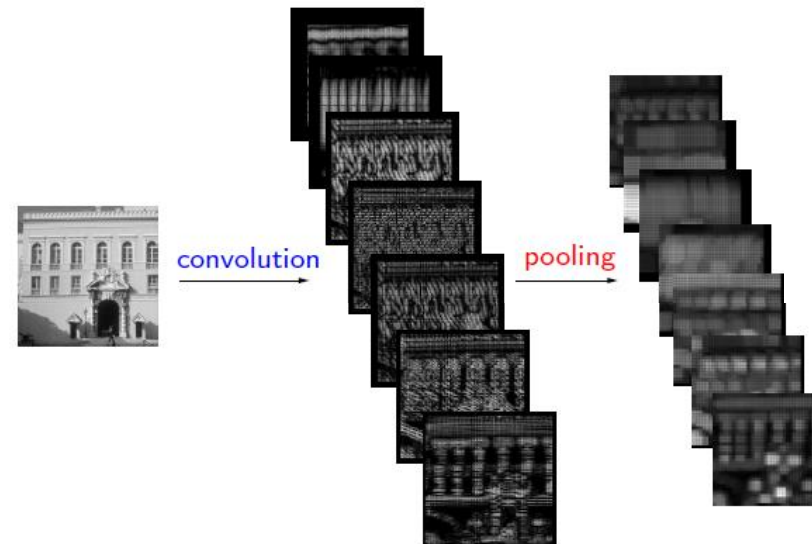
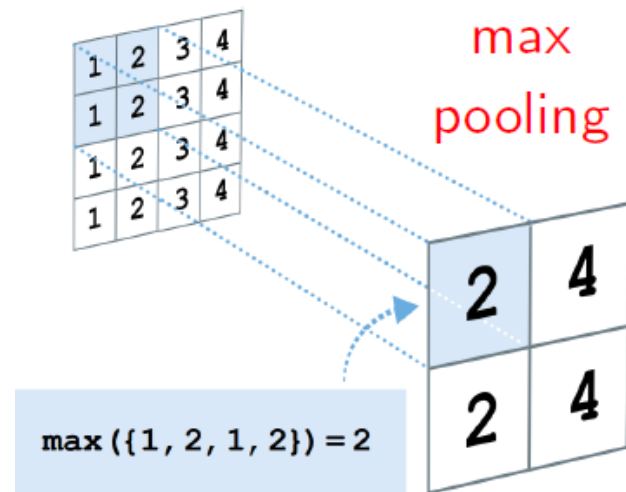


$m = 5, k = 4$



# Pooling Layer

- Nonlinear down-sampling
  - Aggregates statistics of local features (with **max** or **average** operation)
  - Reduced variance: provides invariance to local transformations



source: [Angermueller et al., 2016, Thériault et al., 2013]

# Batch Normalization

**Input:** Values of  $x$  over a mini-batch:  $\mathcal{B} = \{x_1 \dots x_m\}$ ;

Parameters to be learned:  $\gamma, \beta$

**Output:**  $\{y_i = \text{BN}_{\gamma, \beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m} \sum_{i=1}^m x_i \quad // \text{ mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2 \quad // \text{ mini-batch variance}$$

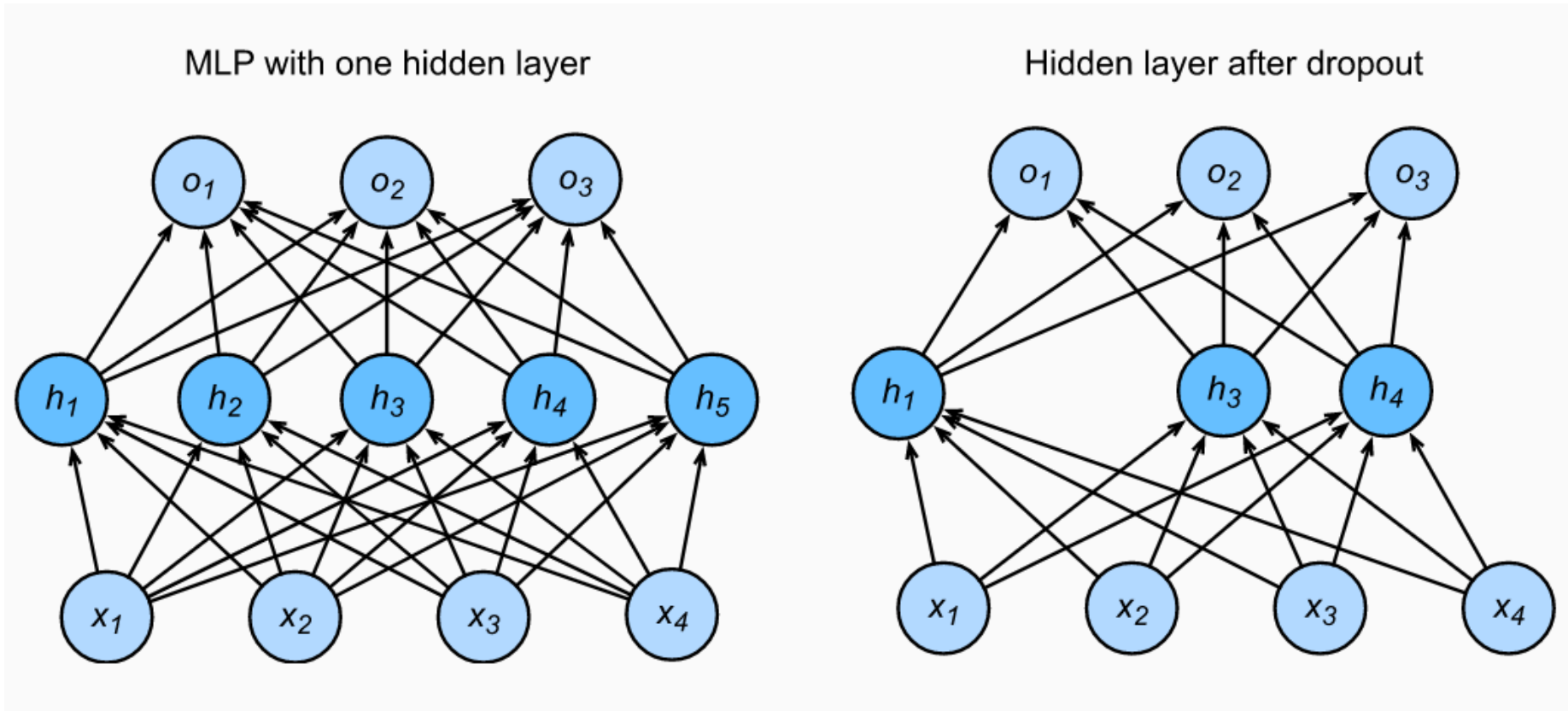
$$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \quad // \text{ normalize}$$

$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv \text{BN}_{\gamma, \beta}(x_i) \quad // \text{ scale and shift}$$

**Algorithm 1:** Batch Normalizing Transform, applied to activation  $x$  over a mini-batch.

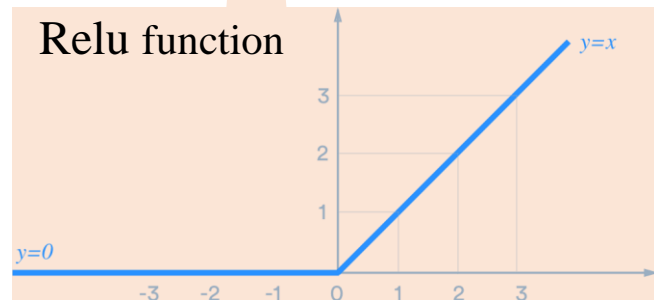
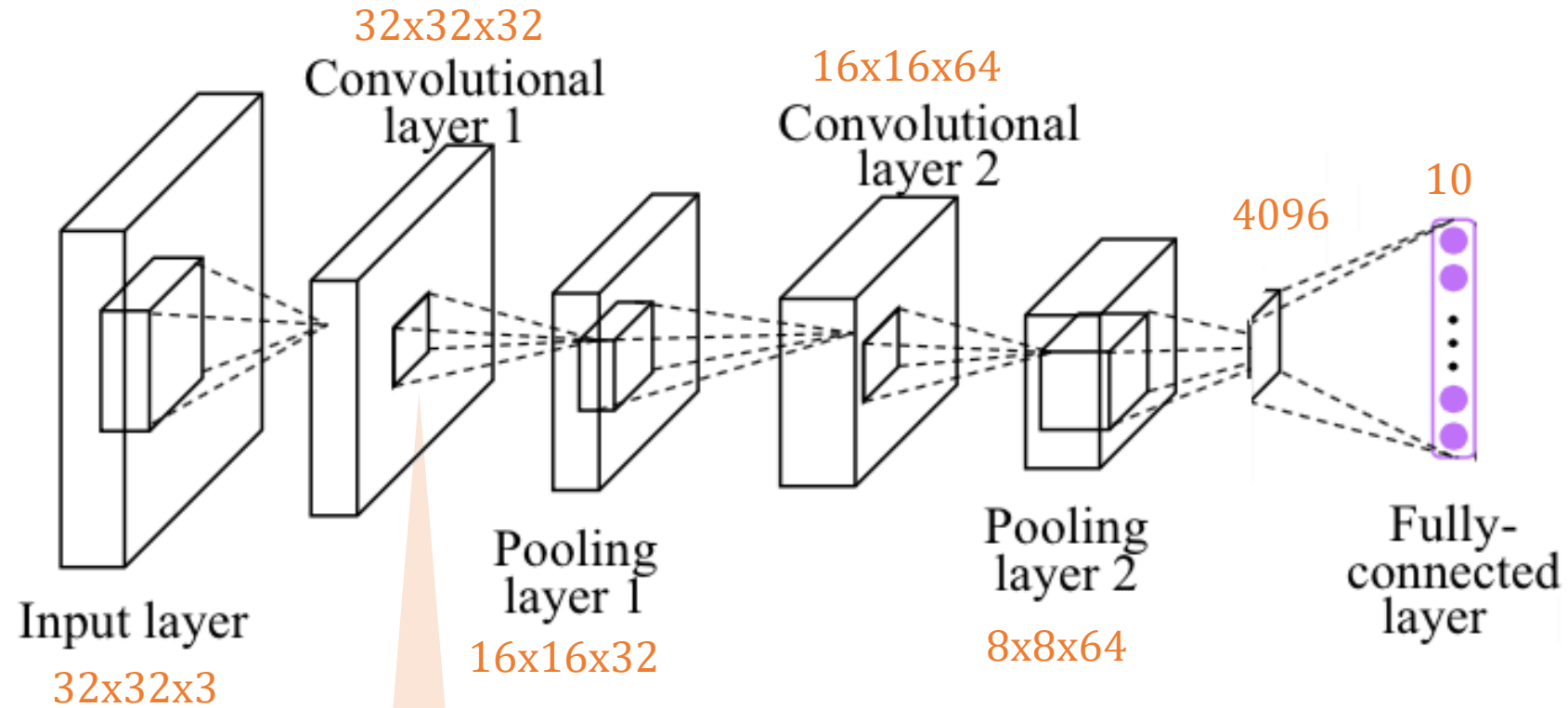
<https://sacko.tistory.com/44>

# Dropout to mitigate overfitting



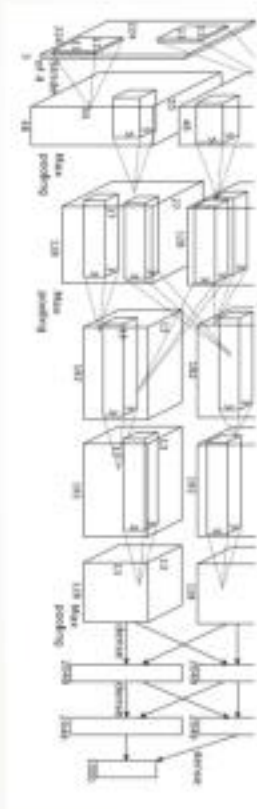
[https://ko.d2l.ai/chapter\\_deep-learning-basics/dropout.html](https://ko.d2l.ai/chapter_deep-learning-basics/dropout.html)

# Two Layer Convolutional Neural Network



# Popular Deep Networks

**“AlexNet”**



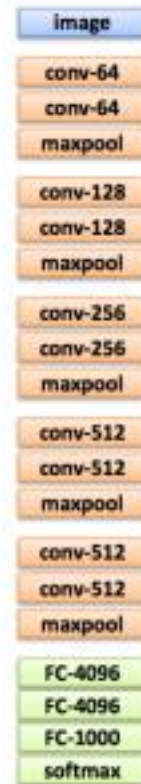
[Krizhevsky et al. NIPS 2012]

**“GoogLeNet”**



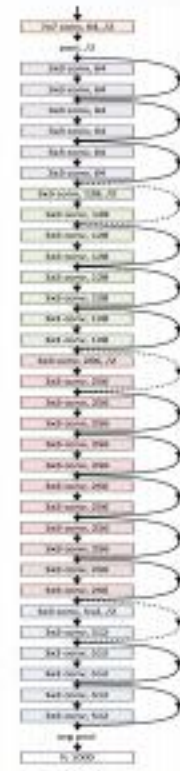
[Szegedy et al. CVPR 2015]

**“VGG Net”**



[Simonyan & Zisserman, ICLR 2015]

**“ResNet”**



[He et al. CVPR 2016]



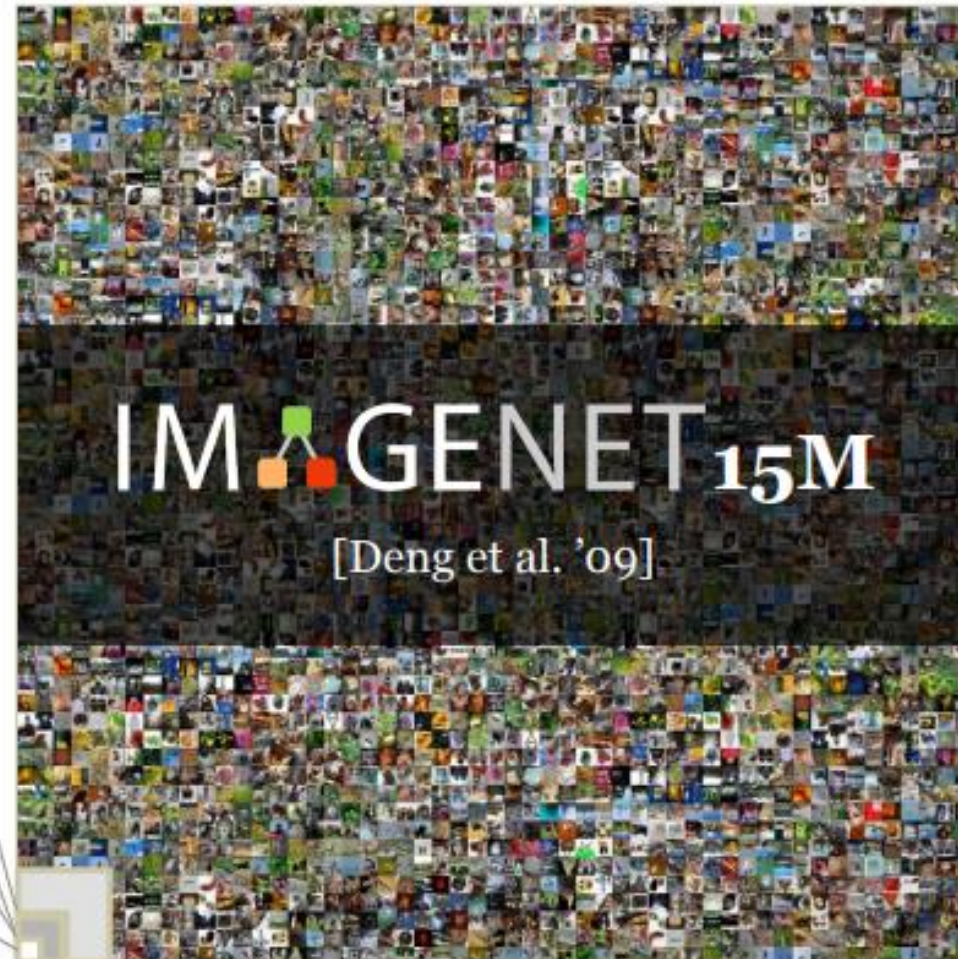
# Data Sets

**SUN, 131K**  
[Xiao et al. '10]

**LabelMe, 37K**  
[Russell et al. '07]

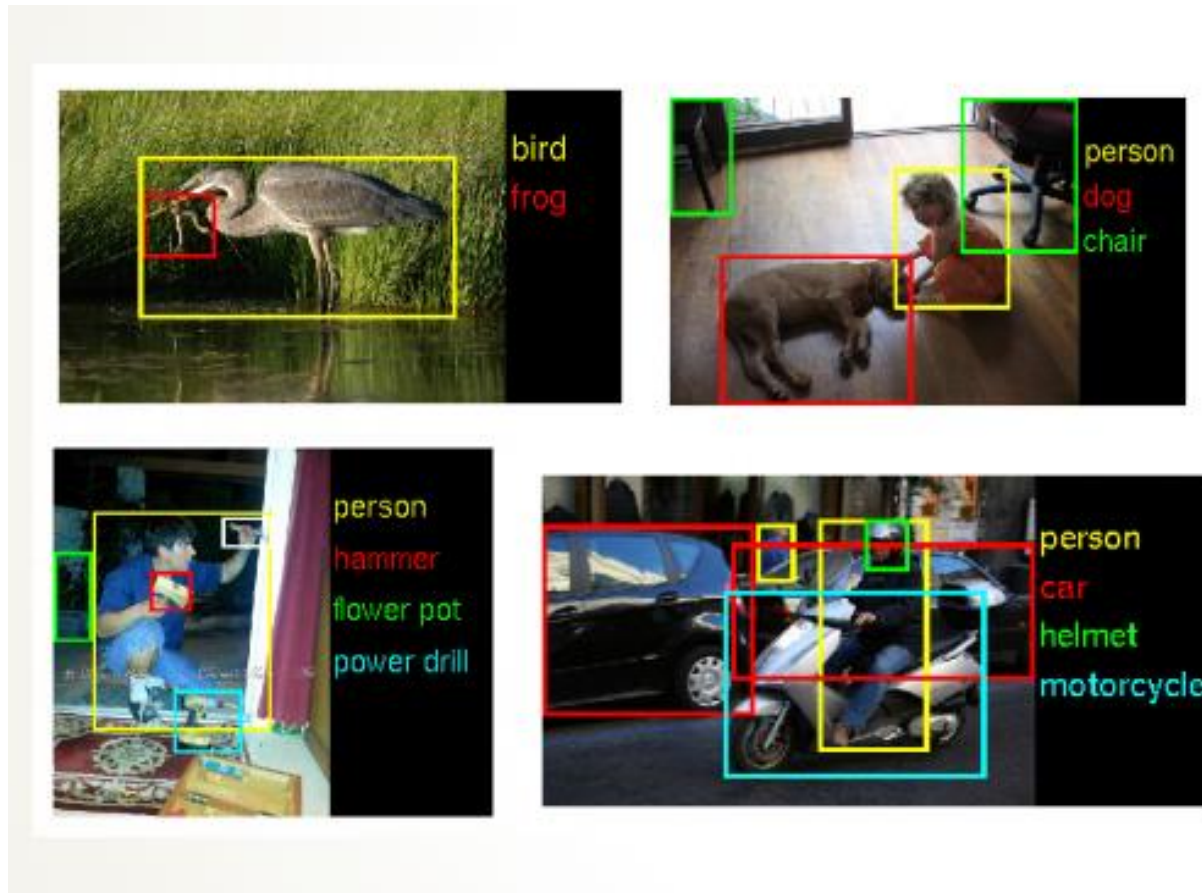
**PASCAL VOC, 30K**  
[Everingham et al. '06-'12]

**Caltech101, 9K**  
[Fei-Fei, Fergus, Perona, '03]



[http://image-net.org/challenges/talks\\_2017/imagenet\\_ilsvrc2017\\_v1.0.pdf](http://image-net.org/challenges/talks_2017/imagenet_ilsvrc2017_v1.0.pdf)

# Data Sets (detection challenge, ILSVRC)



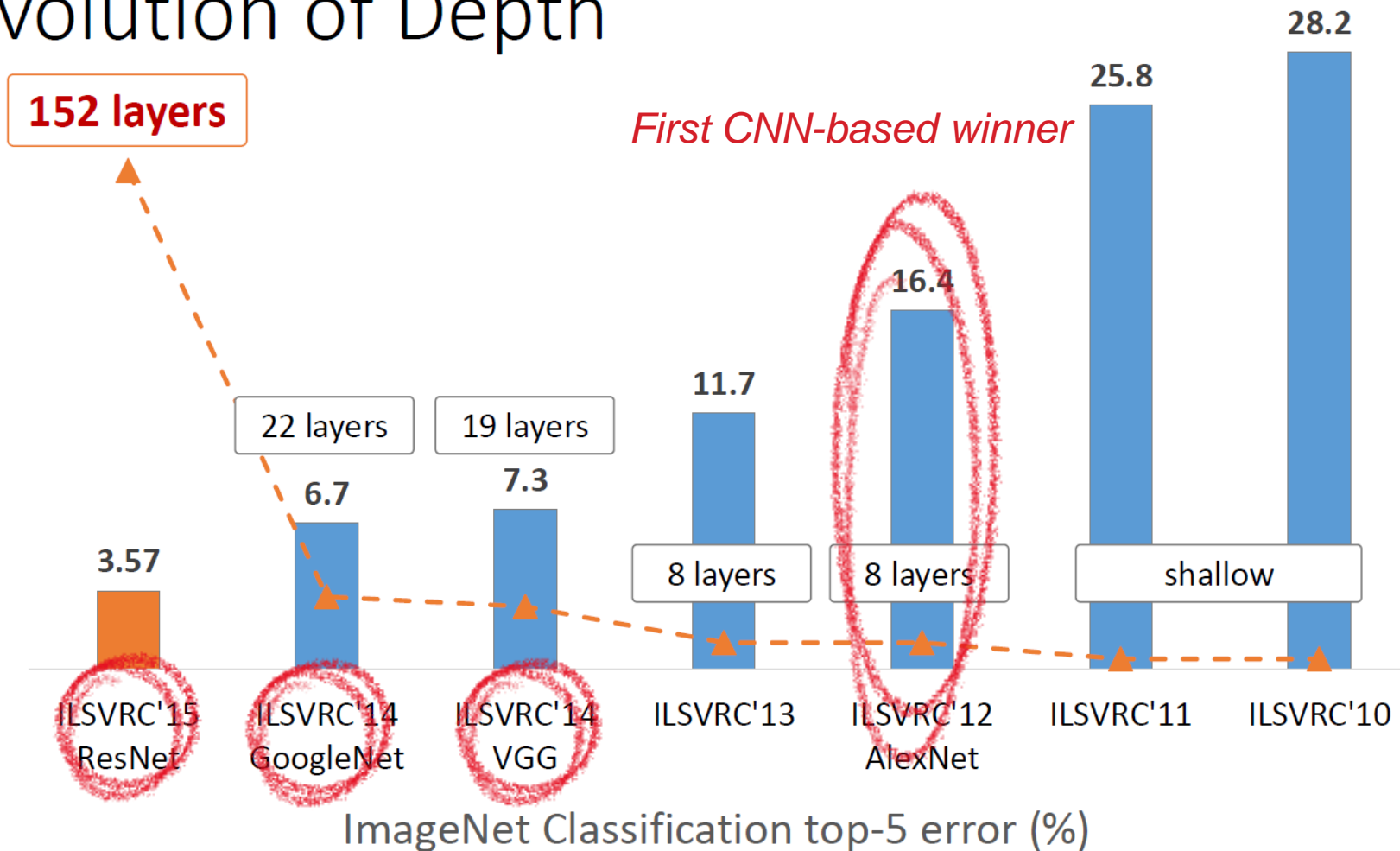
Statistics	PASCAL VOC 2012		ILSVRC 2013
Object classes	20	10x	200
Training	Images	70x	395K
	Objects	25x	345K

[http://image-net.org/challenges/talks\\_2017/imagenet\\_ilsvrc2017\\_v1.0.pdf](http://image-net.org/challenges/talks_2017/imagenet_ilsvrc2017_v1.0.pdf)



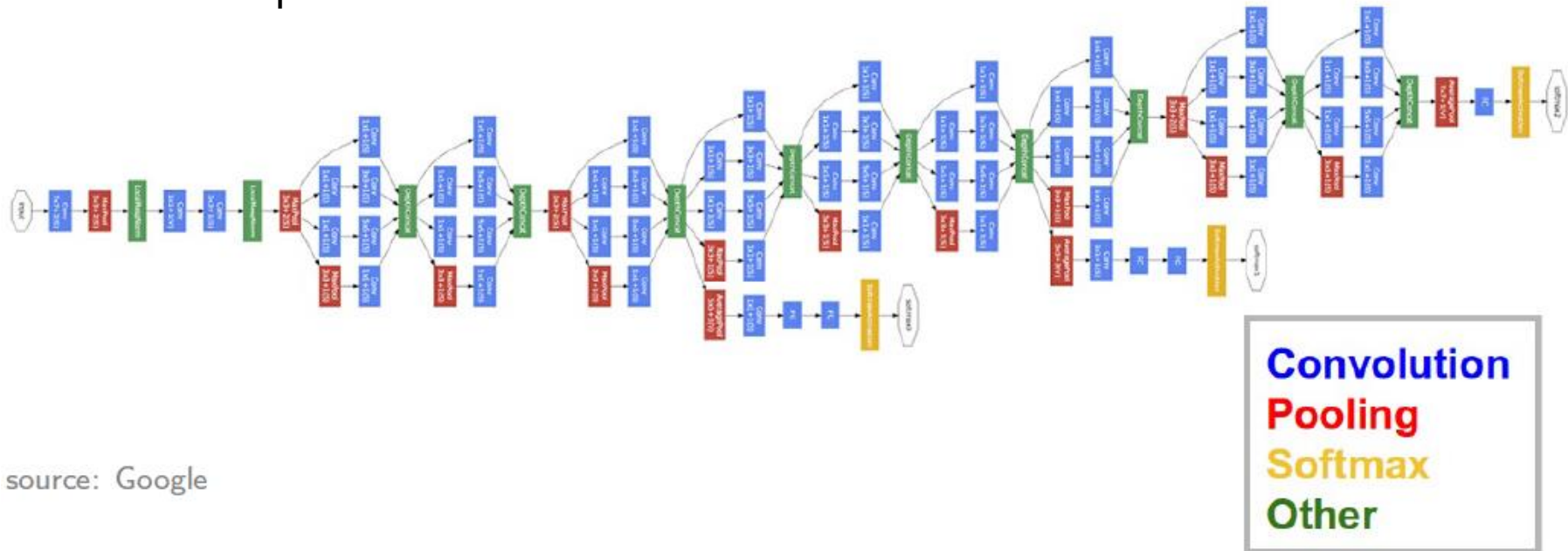
# ImageNet challenge (ILSVRC)

## Revolution of Depth



# Inception model (a.k.a GoogLeNet)

- Deeper network with computational efficiency
  - ImageNet [Large Scale](#) Visual Recognition Competition (ILSVRC) 2014 winner (6.7% top 5 error)
  - [22 layers with 5 million parameters](#) (12x less than AlexNet \*ILSVRC 2012 winner)
  - Efficient “Inception” module



# Data Sets

- The **CIFAR-10(100)** dataset consists of 32x32 color images in 10(100) classes.
- 50000 train images
- 10000 test images

airplane



automobile



bird



cat



deer



dog



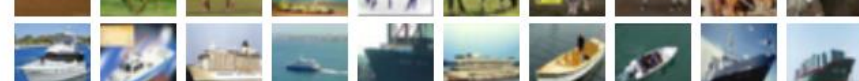
frog



horse



ship



truck



source:

<https://www.cs.toronto.edu/~kriz/cifar.html>



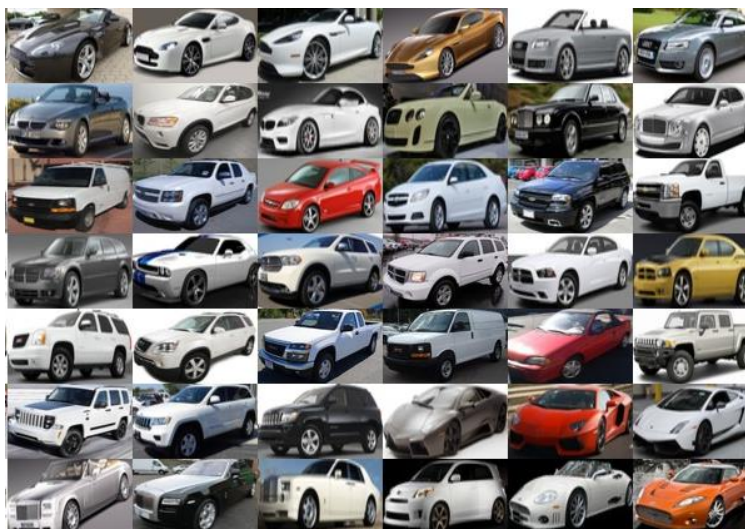
# Data Sets (Fine Grained Classification)



CUB-200(Classification)



Cars-196



FGVC-Aircraft



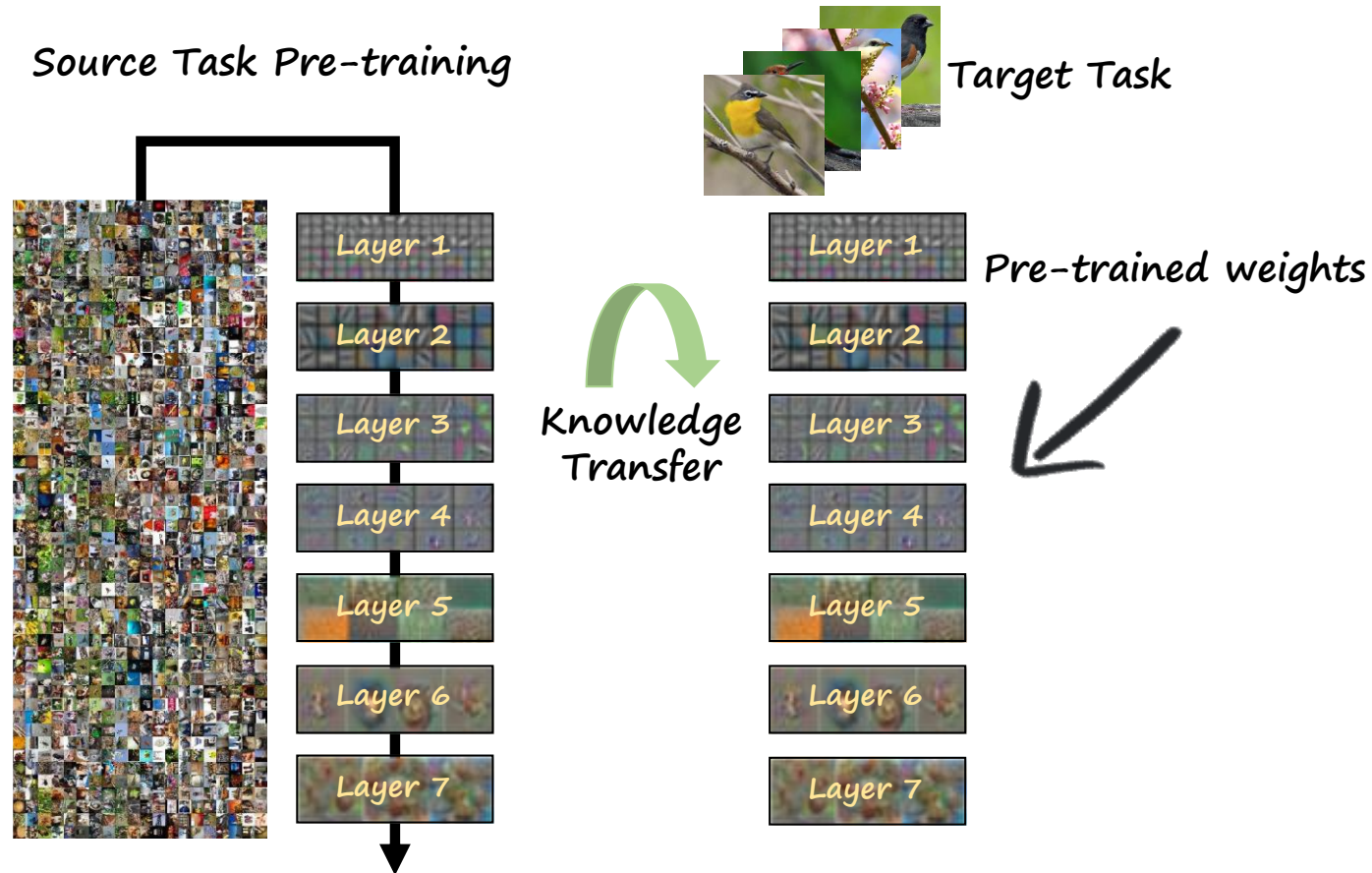
# Transfer Learning for Small Datasets: Fine-Tuning, Knowledge Distillation

Jin Young Choi

Seoul National University

# Learning Rates in Fine-Tuning

- Transfer Learning is a promising alternative in small datasets (Image Retrieval).
- **Fine-tuning** is a type of Transfer Learning. ( $\Leftrightarrow$  from scratch)





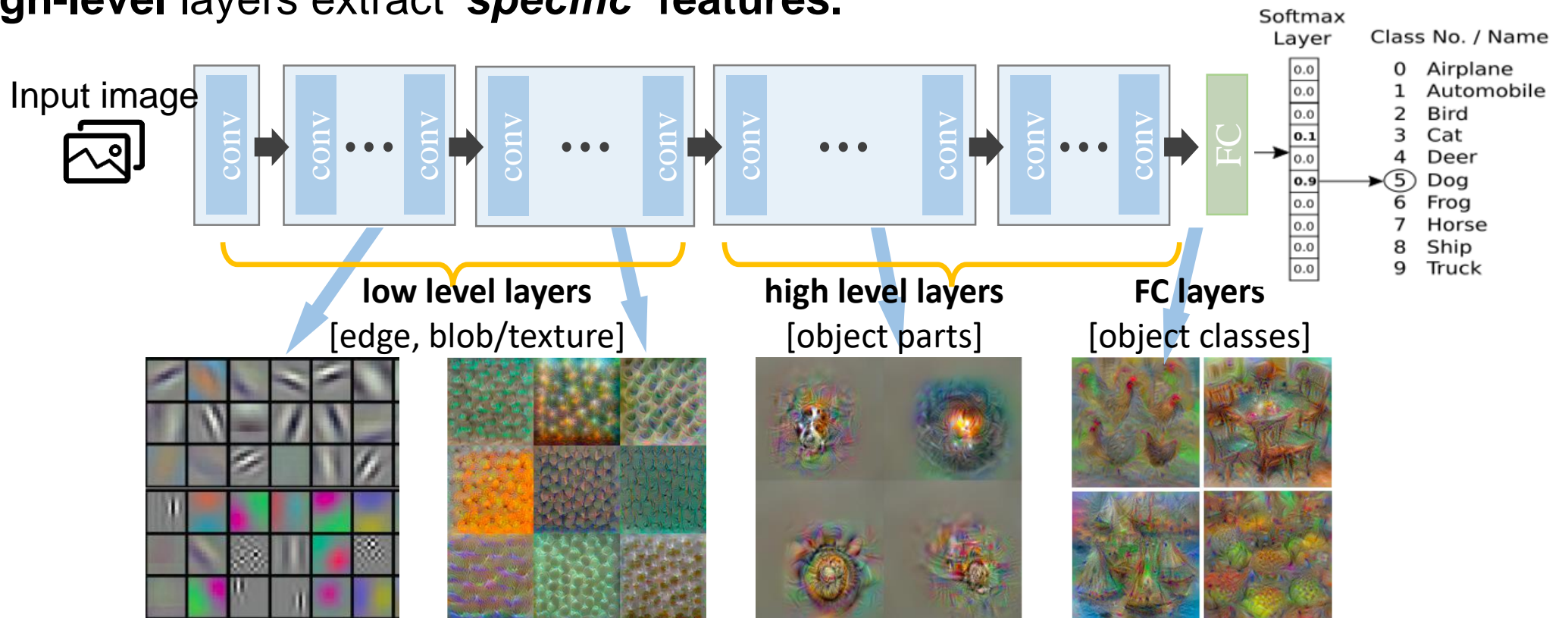
# The Characteristics of Each Layer

1) *Visualizing and understanding convolutional networks ECCV2014*

2) *How transferable are features in deep neural networks? NeurIPS2014*

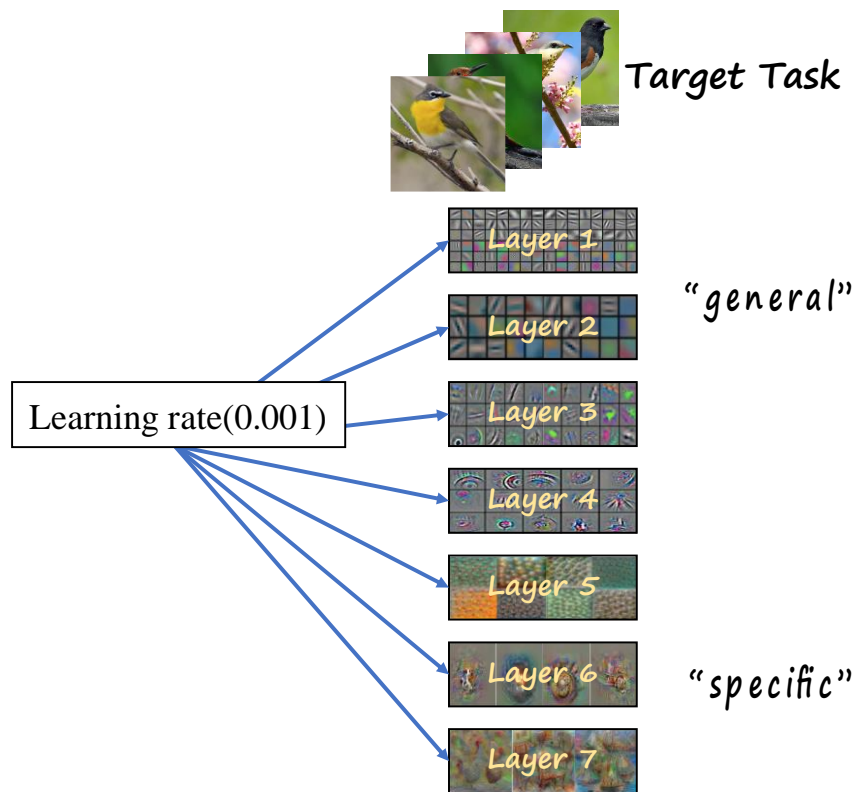
3) *Understanding deep image representations by inverting them CVPR2015*

- In previous studies,
- **Low-level layers extract ‘general’ features.**  
**High-level layers extract ‘specific’ features.**

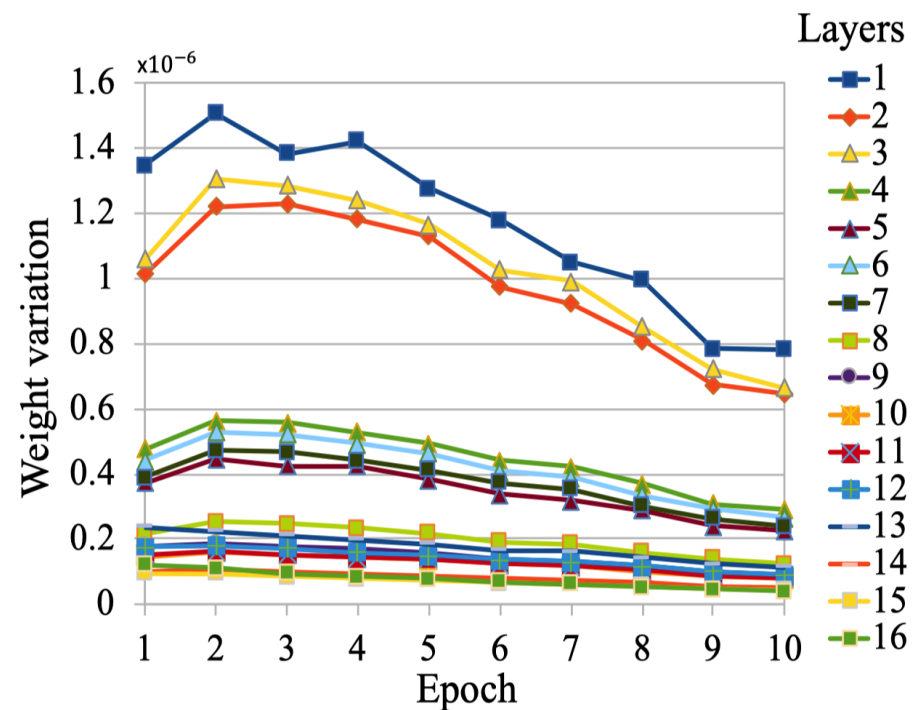


# Layer-wise Learning Rate Tuning

- Most existing fine-tuning methods adopt a single LR regardless of layers.
- *LR is a hyper-parameter that **controls how much** we are **adjusting** the weights of our network with **respect the loss gradient**.*



The ‘Layer’ is defined as 16 layers.

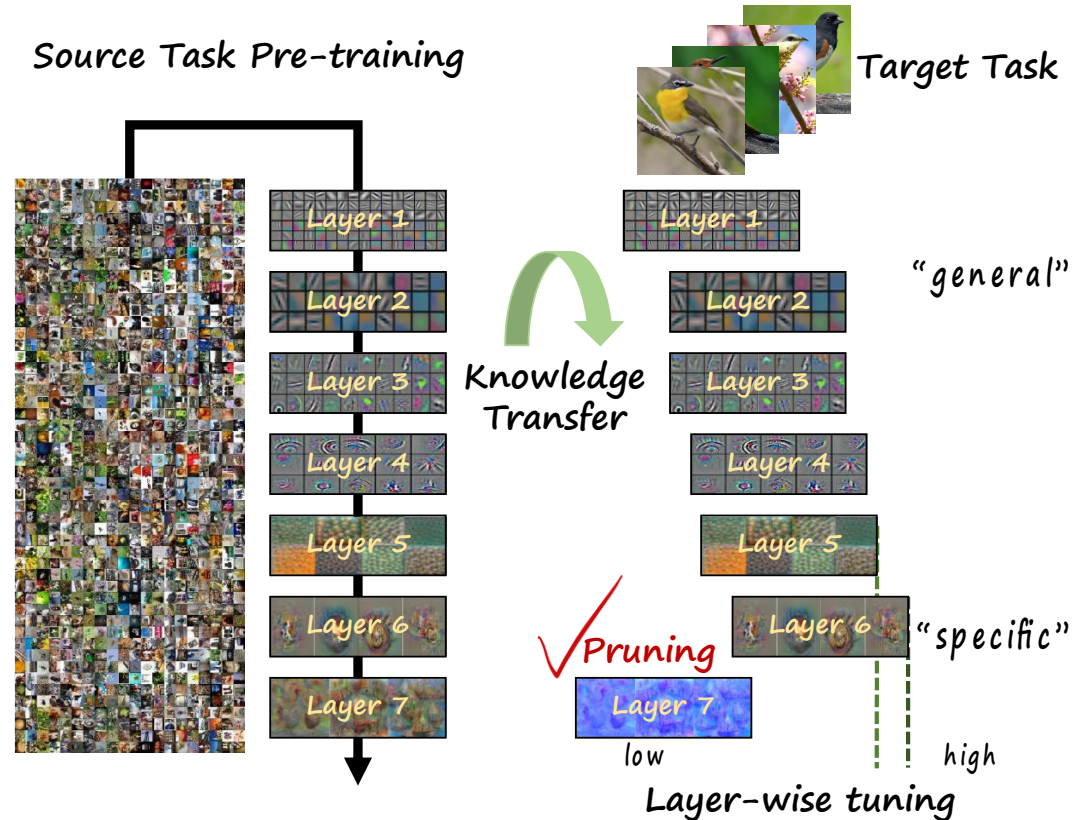




# Hypotheses

The proposed method begins on the following two hypotheses:

- Hypothesis 1: All high-level layers are **not needed for the target task**.
- Hypothesis 2: **Low-level** layers need to change **Small**.  
**High-level** layers need to change **Large**.



# Goal of Preliminary experiments

## Experiment 1

- The high-level layer with small weight variation **may not contribute to the new task learning.**
- ✓ To verify this,  
**Removed** one by one from the highest layer.

## Experiment 2

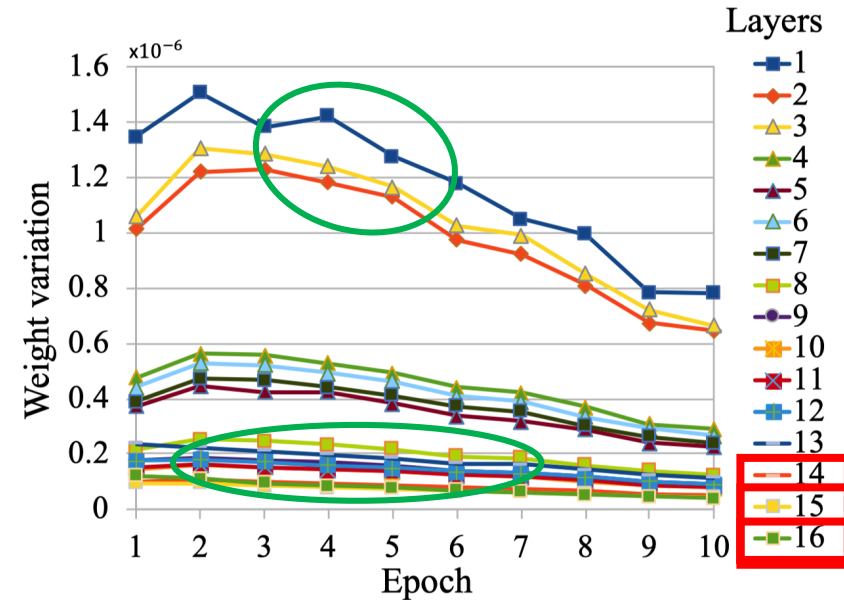
- Large variations in low-level layers may **destroys general features**
- Small variations in high-level layers may **not be adapt to target task**
- ✓ To verify this,  
**Adjusting layer-wise LRs**

# Results of Preliminary experiments

- Layer-wise pruning and Layer-wise tuning learning rates.

Recall @  $K$  score

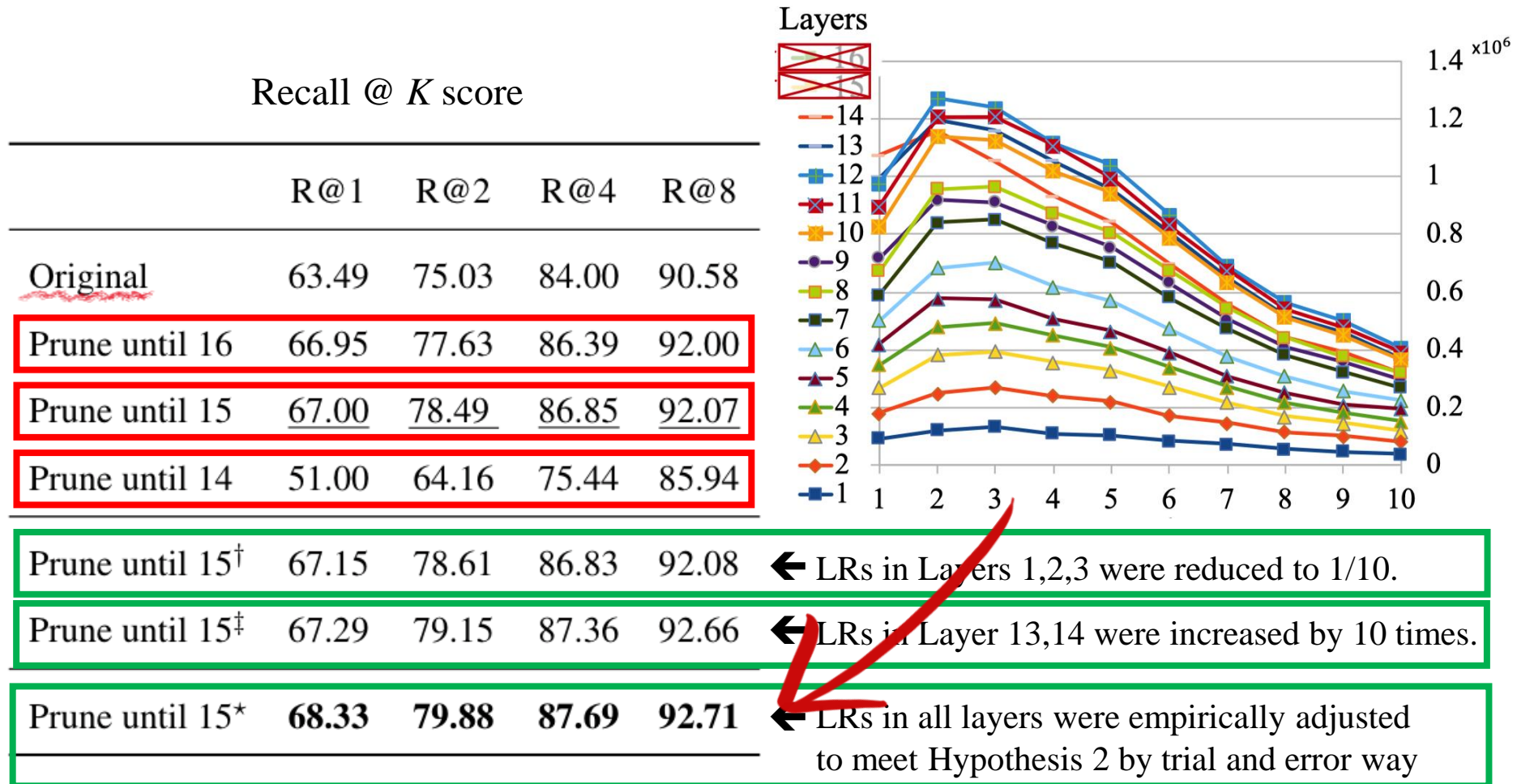
	R@1	R@2	R@4	R@8
Original	63.49	75.03	84.00	90.58
Prune until 16	66.95	77.63	86.39	92.00
Prune until 15	67.00	78.49	86.85	92.07
Prune until 14	51.00	64.16	75.44	85.94



Prune until 15 <sup>†</sup>	67.15	78.61	86.83	92.08	← LRs in Layers 1,2,3 were reduced to 1/10.
Prune until 15 <sup>‡</sup>	67.29	79.15	87.36	92.66	← LRs in Layer 13,14 were increased by 10 times.
Prune until 15 <sup>*</sup>	<b>68.33</b>	<b>79.88</b>	<b>87.69</b>	<b>92.71</b>	← LRs in all layers were empirically adjusted to meet Hypothesis 2 by trial and error way

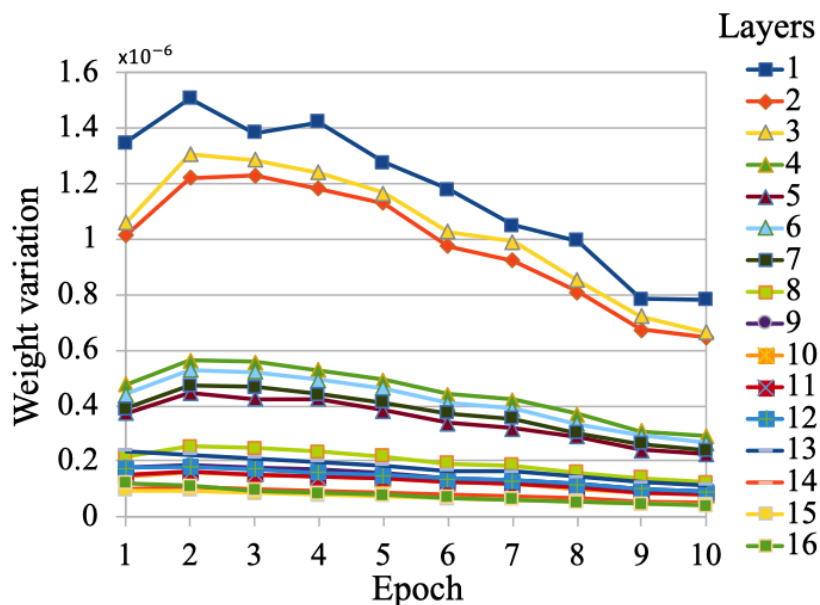
# Results of Preliminary experiments

- Layer-wise pruning and Layer-wise tuning learning rates.

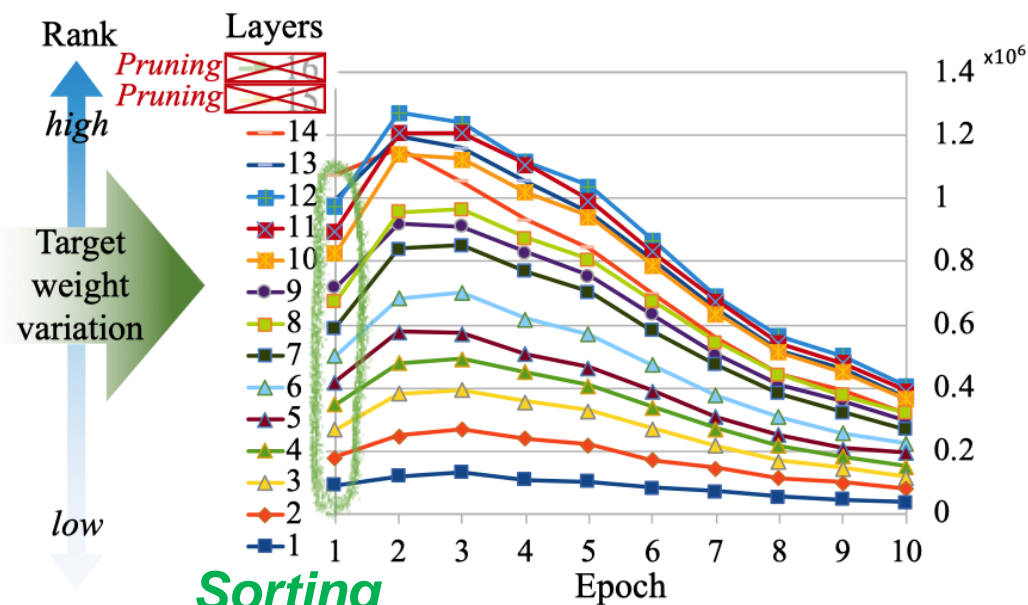


# Results of experiments

- There are a couple of high-level layers that **do not contribute**.
- Adjusts the LRs depending on the role of each layer contribute to performance improvement.



Single LR over all layers



Layer-wise pruning and adjusting LRs

Original 63.49 75.03 84.00 90.58 → Prune until 15\* 68.33 79.88 87.69 92.71



# Qualitative evaluation

- We evaluate our method qualitatively by CAM visualization.

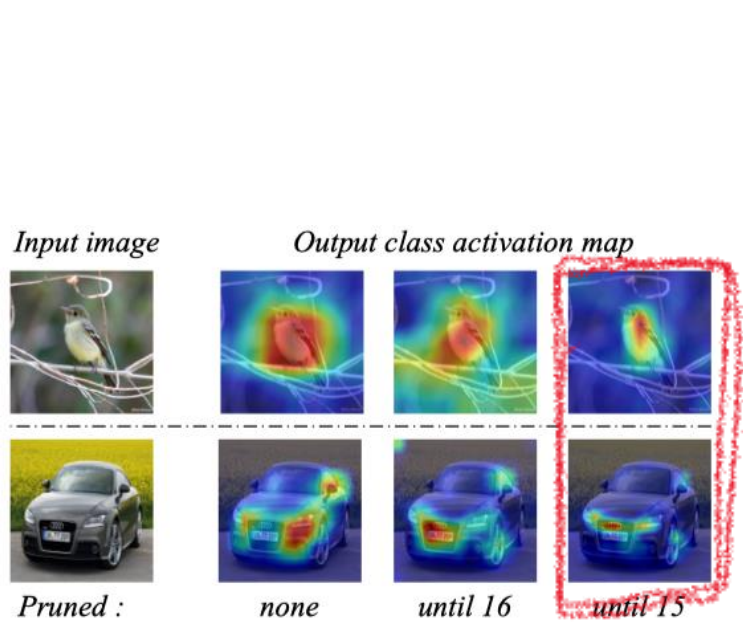


Figure 4. The class activation map (CAM) visualization of the last layers by applying different layers pruning

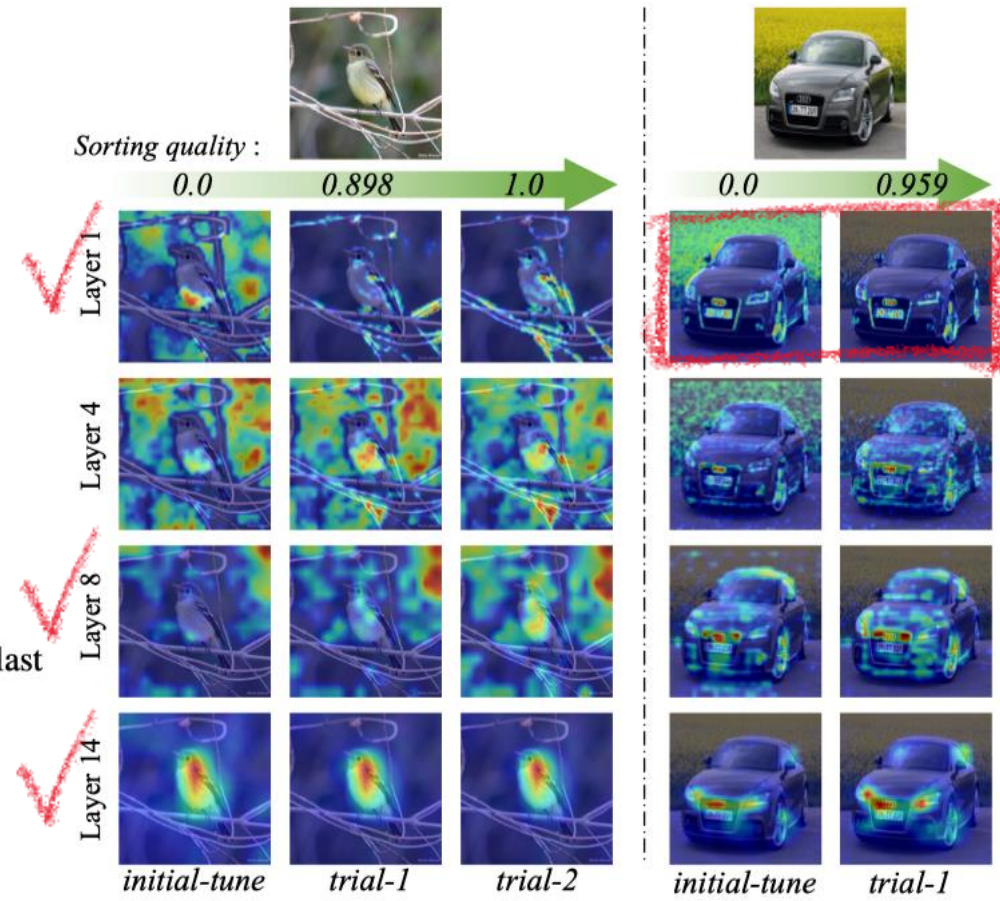


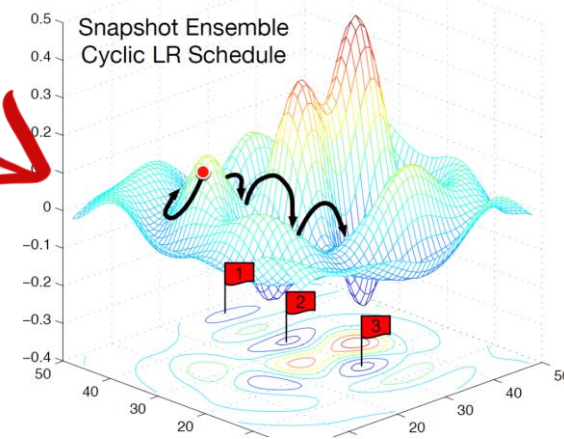
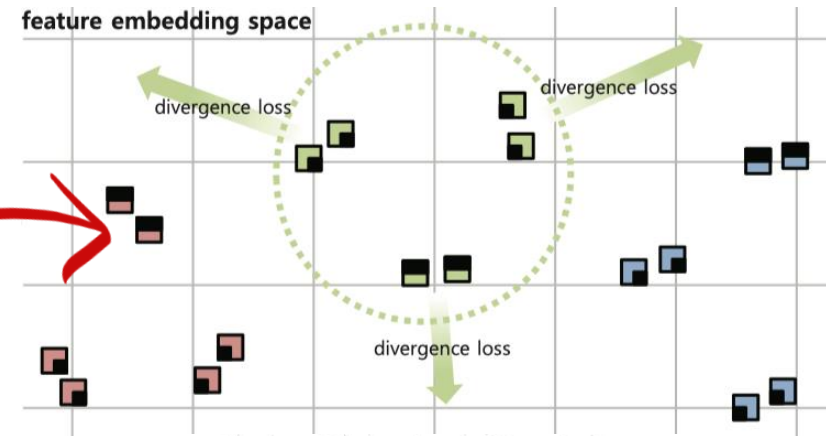
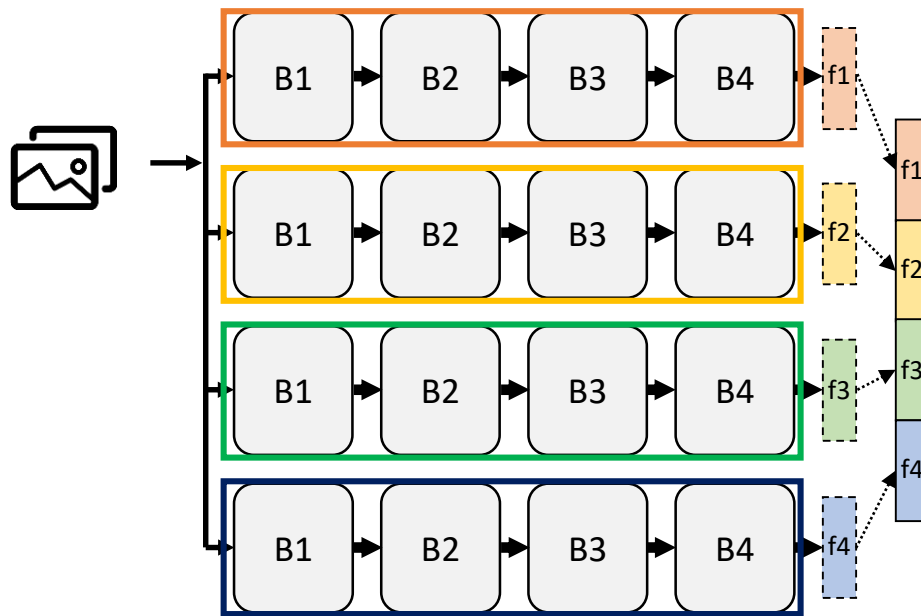
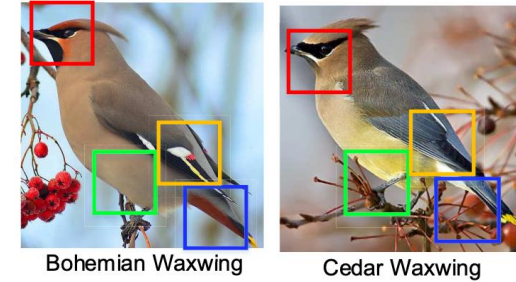
Figure 5. The class activation map (CAM) visualization of several layers (1, 4, 8, 14) according to the sorting quality. *initial-tune* is done by the conventional fine-tuning with single LR

# Structure Design for Fine-Tuning

- Feature vectors of diverse characteristics are needed

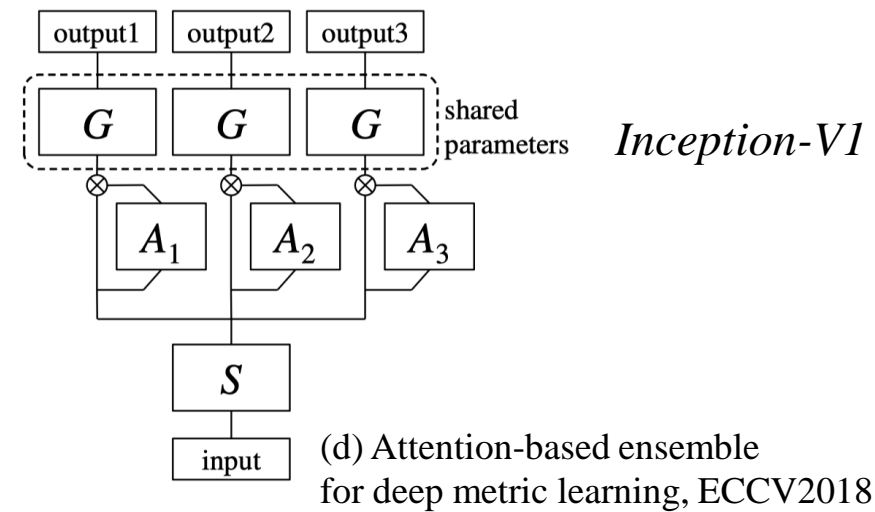
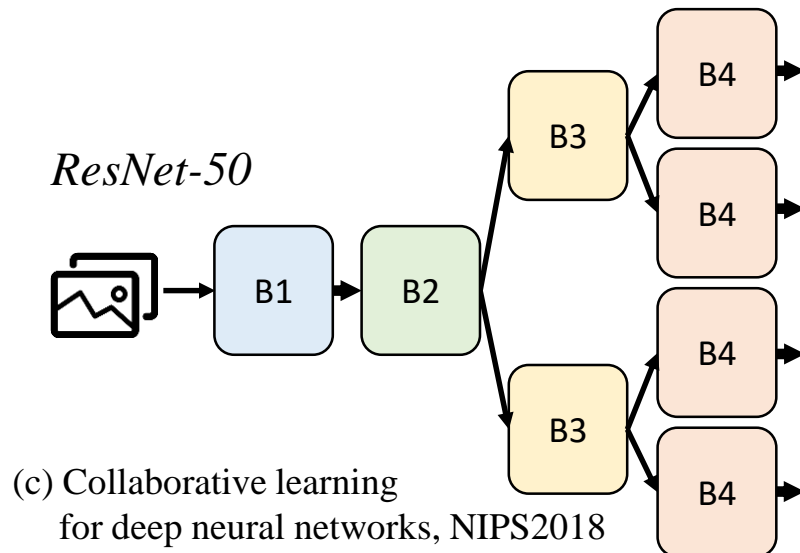
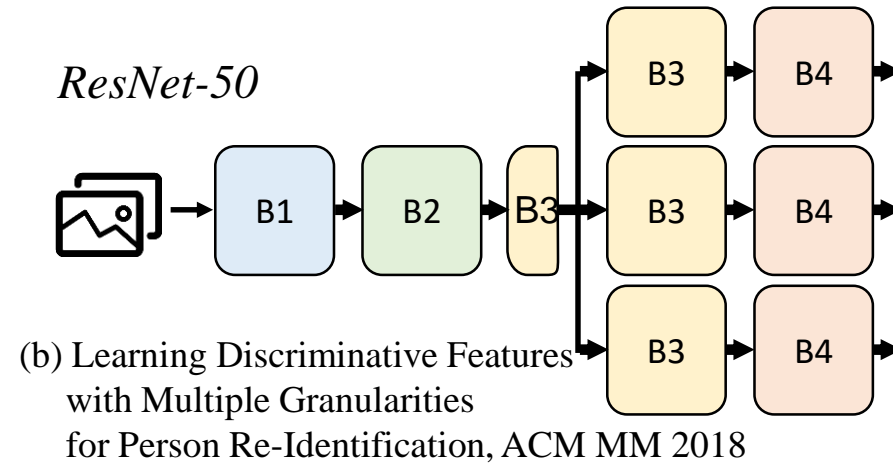
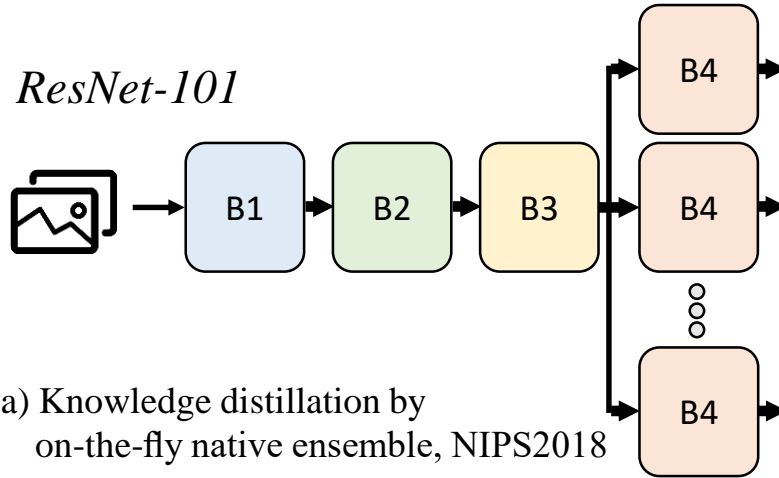
*However!!*

**Disadvantage:** Multiple models require considerable memory and heavy computation



# Multi-head structures

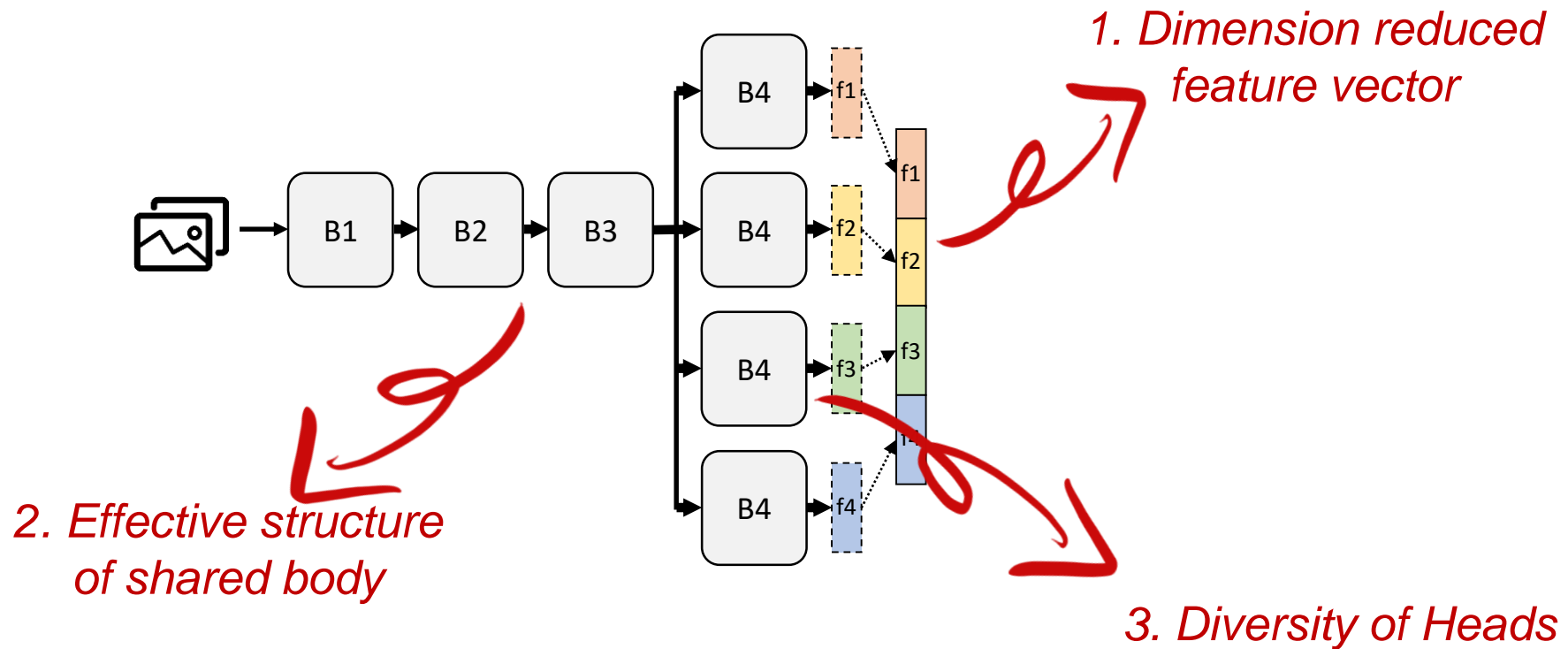
- To alleviate the memory and computation problems, Multi-head structures have been employed





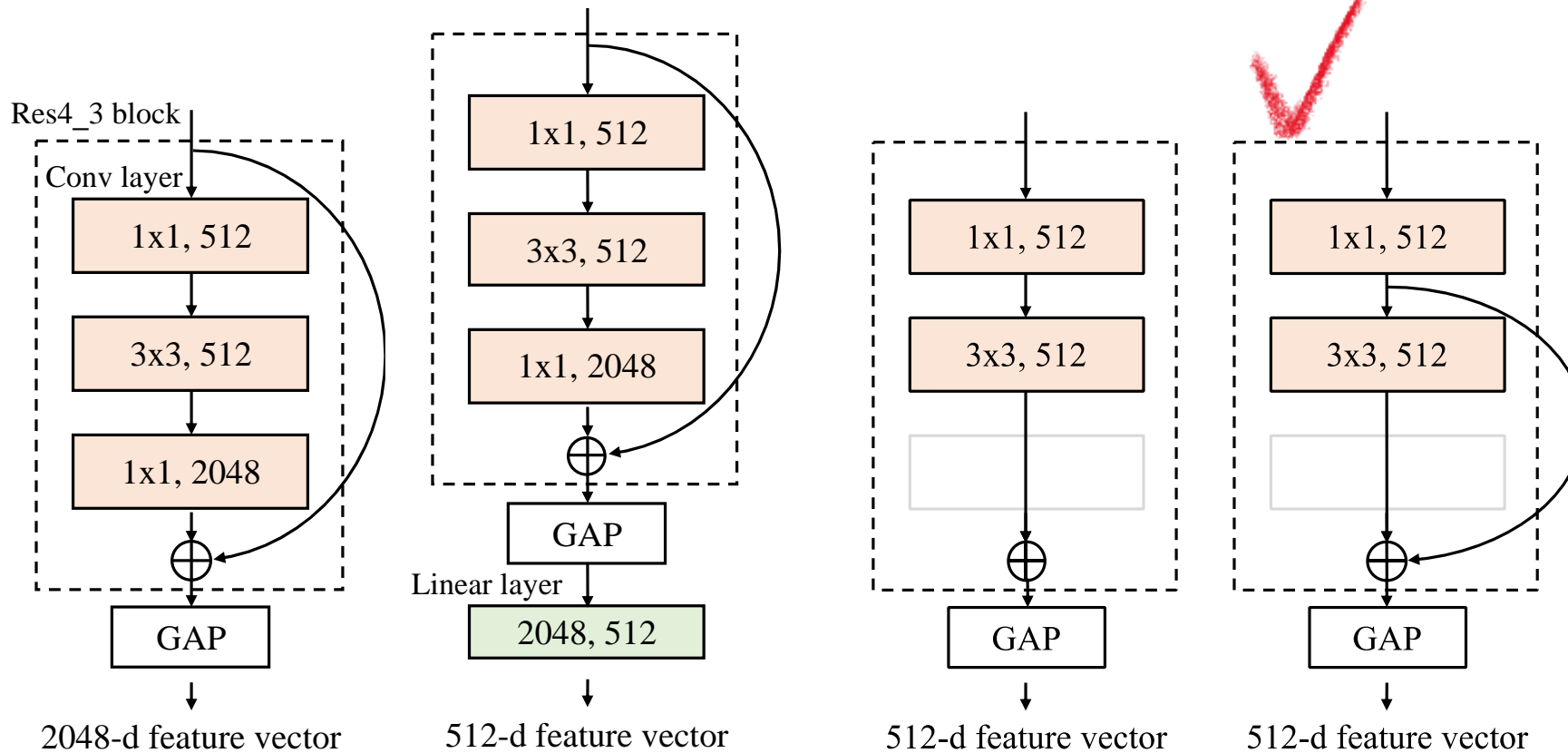
# Proposed Design Concepts

- However, previous studies did not clearly state how their multi-head structure was designed and employed.
- We provide three design concept and evaluated it.



# 1. Dimension reduced feature vector

- Goal : Extract 512-d feature from ResNet-50 network.



(a) Original

(b) Adding linear layer  
*Performance drop  
& Memory-increase*

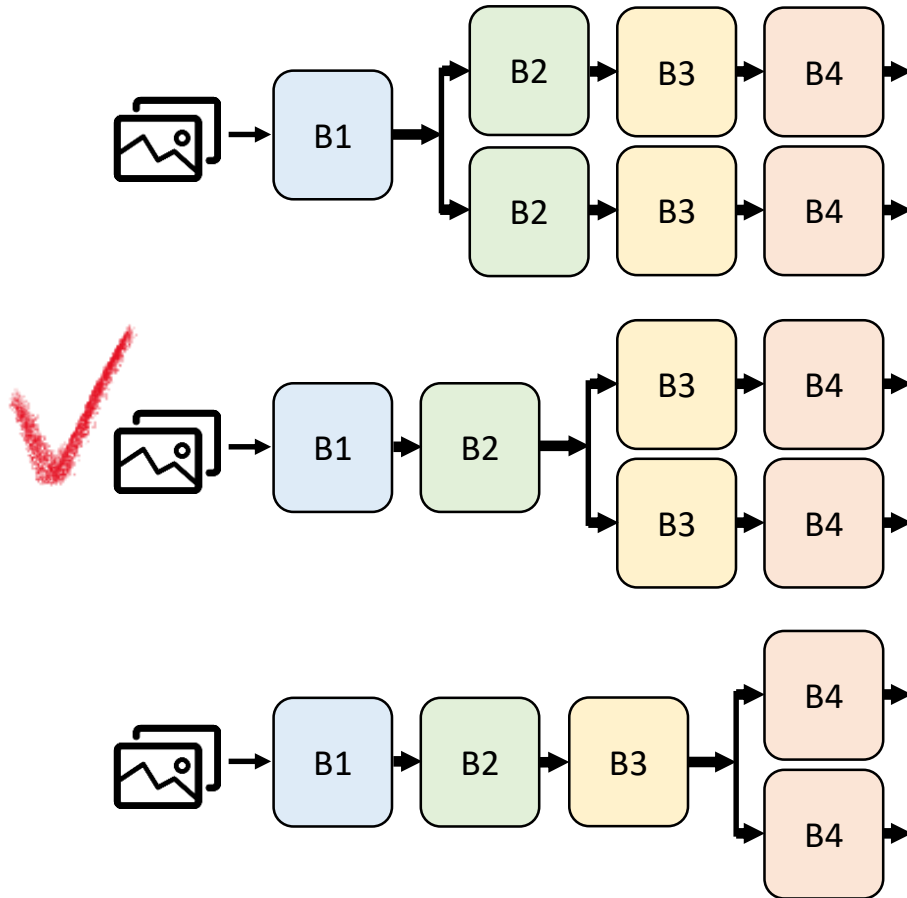
(c) Penultimate

*Performance drop slightly  
& Memory-saving*

(d) Penultimate<sup>+</sup>

## 2. Effective structure of shared low-level

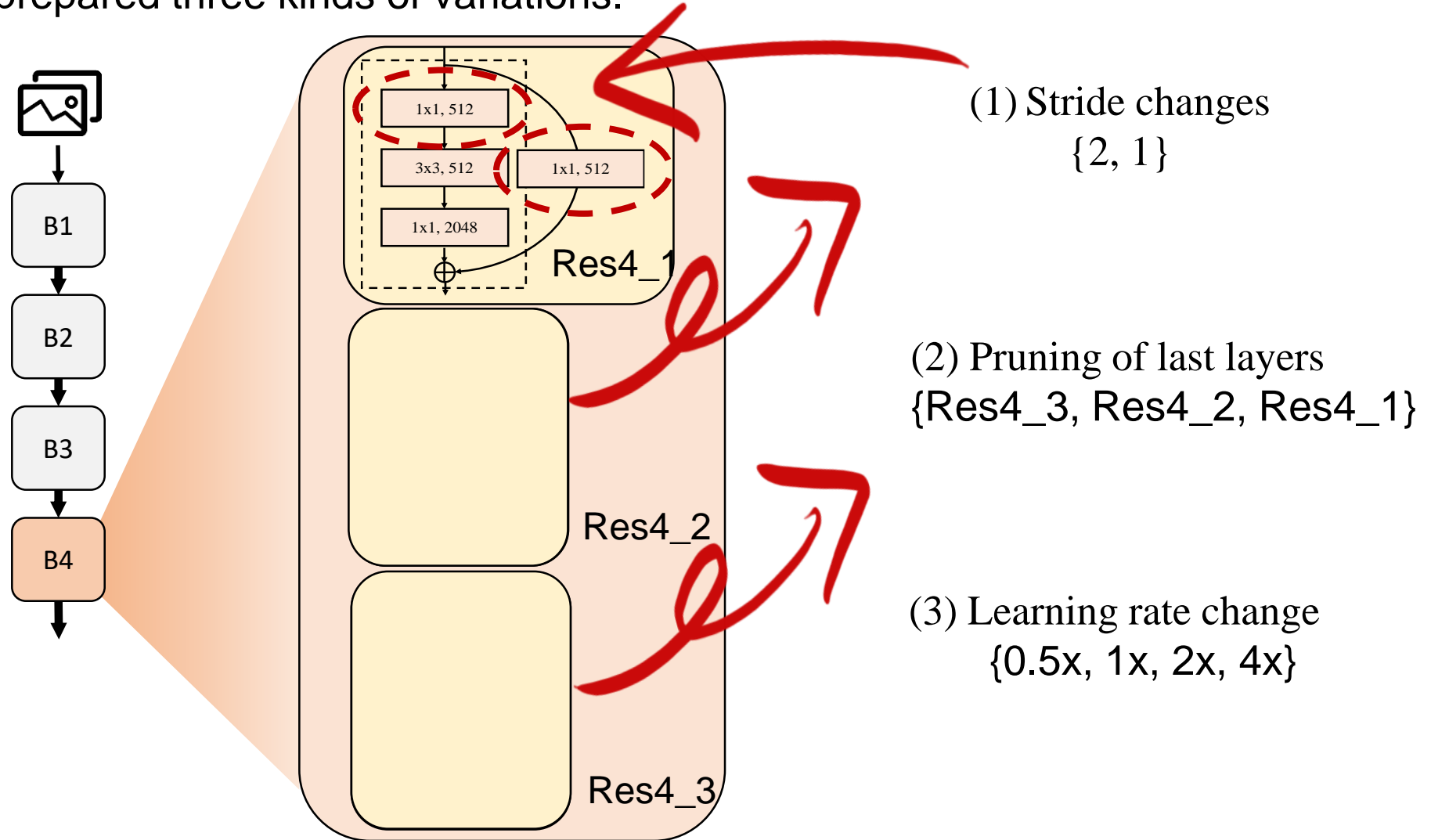
- Goal : Investigate the effective number of shared low-level layers



Memory-saving	Performance
<i>Not Good</i>	<i>Good</i>
<b><i>Good</i></b>	<b><i>Best</i></b>
<i>Best</i>	<i>Not Good</i>

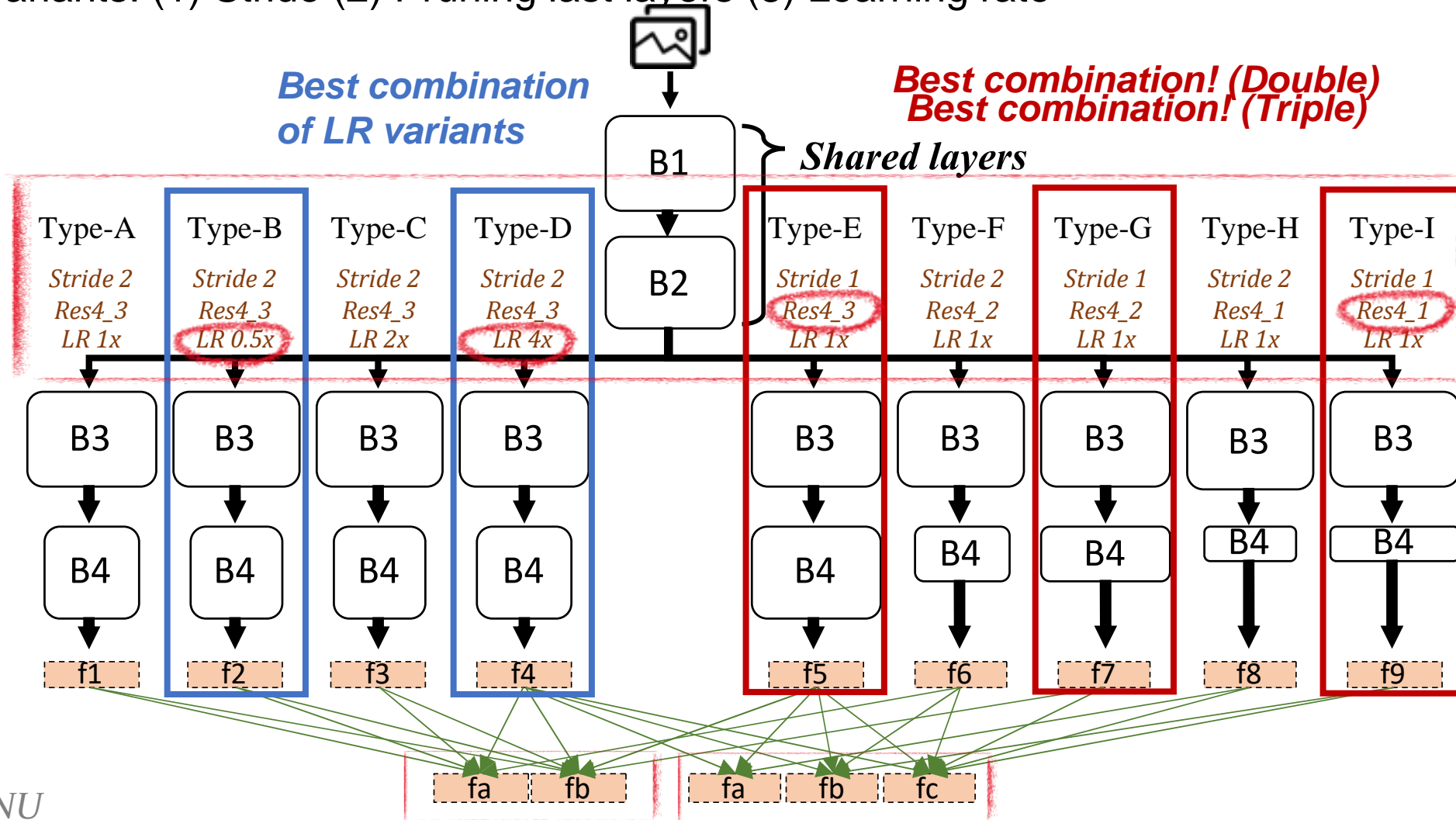
# 3. Diversity of high-level heads

- Goal : Investigate which structural combination is favorable for the ensemble
- We prepared three kinds of variations.



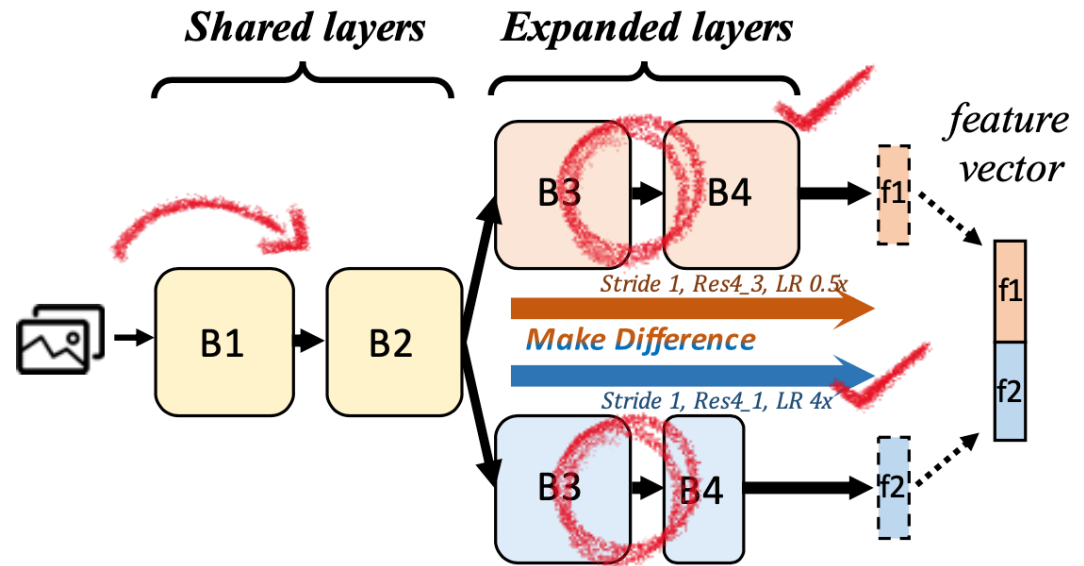
# 3. Diversity of high-level heads

- Goal : Investigate which structural combination is favorable for the ensemble
- Variants: (1) Stride (2) Pruning last layers (3) Learning rate



# Heterogeneous Double-head Ensemble(HDhE)

- *Penultimate+* (4096-d  $\rightarrow$  1024-d)
- Shared *Until B2*
- Head-1: stride 1, Res4\_3, LR 0.5x
- Head-2: stride 1, Res4\_1, LR 4x



Method	Backbone	CUB-200			Cars-196			SOP			Inshop		
		R@1	R@2	R@4	R@1	R@2	R@4	R@1	R@10	R@10 <sup>2</sup>	R@1	R@10	R@20
HDC [1]	Inception-v1†	53.6	65.7	77.0	73.7	83.2	89.5	69.5	84.4	82.8	62.1	84.9	89.0
A-BIER [2]	Inception-v1†	57.5	68.7	78.3	82.0	89.0	93.2	74.2	86.9	94.0	83.1	95.1	96.9
ABE-8 [3]	Inception-v1†	60.6	71.5	79.8	85.2	90.5	93.9	76.3	88.4	94.8	87.3	96.7	97.9
MS [4]	Inception-v1	65.7	77.0	86.4	84.1	90.4	94.0	78.2	90.5	96.0	89.7	97.9	98.5
DREML [5]	ResNet-18†	63.9	75.0	83.1	86.0	91.7	95.0	-	-	-	78.4	93.7	95.8
NormSoft [6]	ResNet-50	61.3	73.9	83.5	84.2	90.4	94.4	72.7	86.2	93.8	-	-	-
Margin [7]	ResNet-50	63.6	74.4	83.1	79.6	86.5	91.9	79.5	91.5	96.7	89.4	97.8	98.7
SCHM [8] <i>CVPR2019</i>	Inception-v1	66.2	76.3	84.1	83.6	-	-	77.6	89.1	94.7	91.9	98.0	98.7
SoftTriple [9] <i>ICCV2019</i>	Inception	65.4	76.4	84.5	84.5	90.7	94.5	78.3	90.3	95.9	-	-	-
Proxy-Anchor <i>CVPR2020</i>	Inception	68.4	79.2	86.8	86.1	91.7	95.0	79.1	90.8	96.2	91.5	98.1	98.8
Ms-HDhE (Ours)	ResNet-50†	71.3	81.4	88.4	89.8	94.3	97.1	84.0	93.5	97.1	91.2	98.0	98.6
HDhE (Ours)	ResNet-50†	72.9	82.3	89.0	90.6	95.0	97.2	84.5	93.8	97.4	92.1	98.2	98.8
<b>Tri-HDhE (Ours)</b>	ResNet-50†	<b>73.5</b>	<b>83.3</b>	<b>89.5</b>	<b>92.2</b>	<b>95.7</b>	<b>97.4</b>	<b>85.0</b>	<b>94.3</b>	<b>97.6</b>	<b>93.3</b>	<b>98.6</b>	<b>99.0</b>



# Deep Convolutional Network (II)

Jin Young Choi

Seoul National University

# Outline

- Convolution and Cross-Correlation
- Convolutional Neural Networks Basics
- Fine-Tuning for Small Datasets
  - Learning Rates in Fine-Tuning
  - Structure Design for Fine-Tuning
- Knowledge Distillation
  - Transfer of Activation Boundary
  - Design Aspects for Feature Distillation

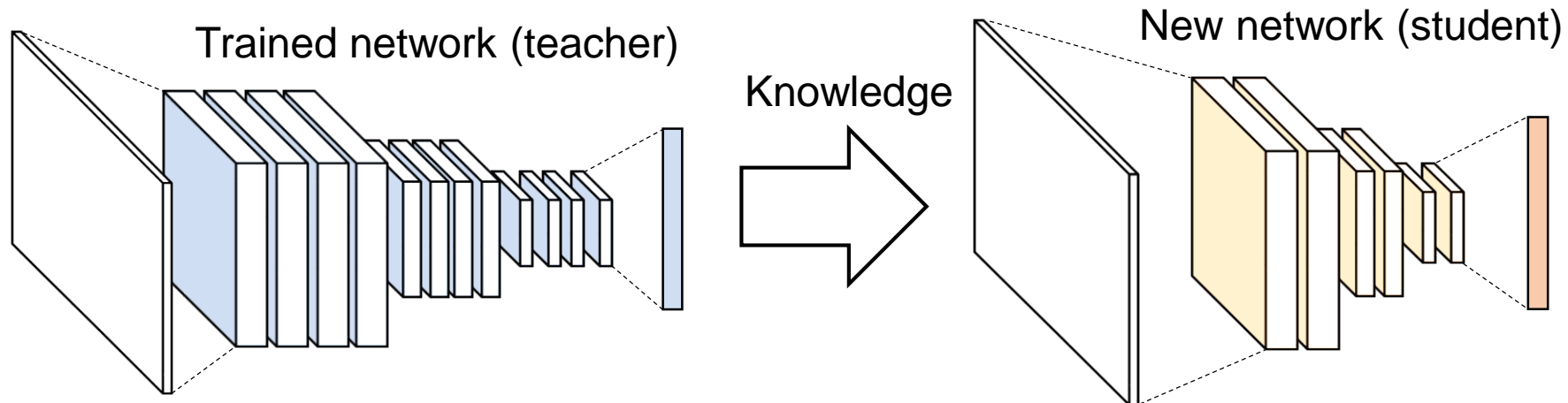
# Knowledge Distillation

Transfer knowledge from a trained network (teacher)  
to a new network (student)

To improve training of the new network (student)

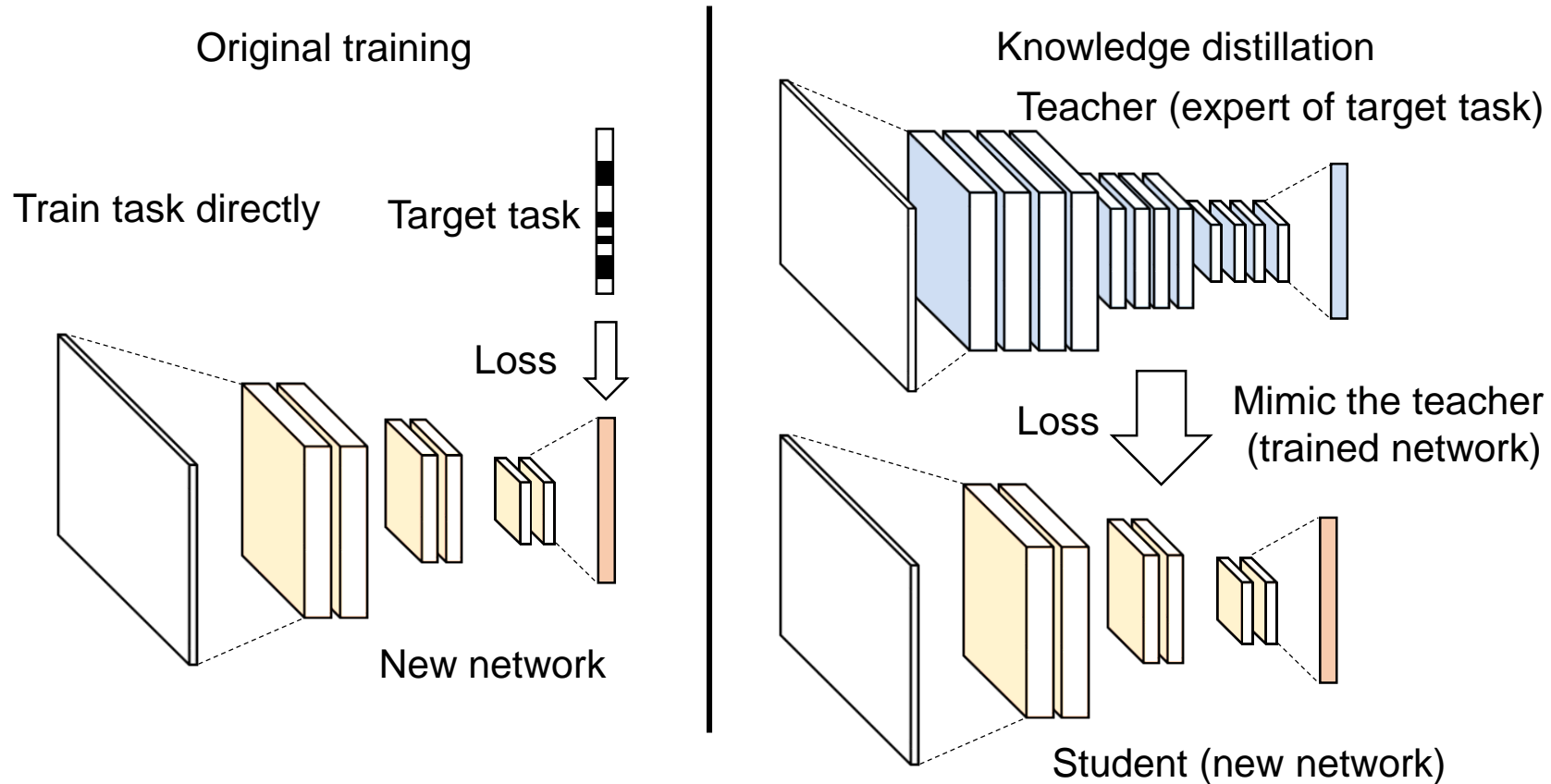
Learning method of neural network

Applications: Network compression (large network to small network)



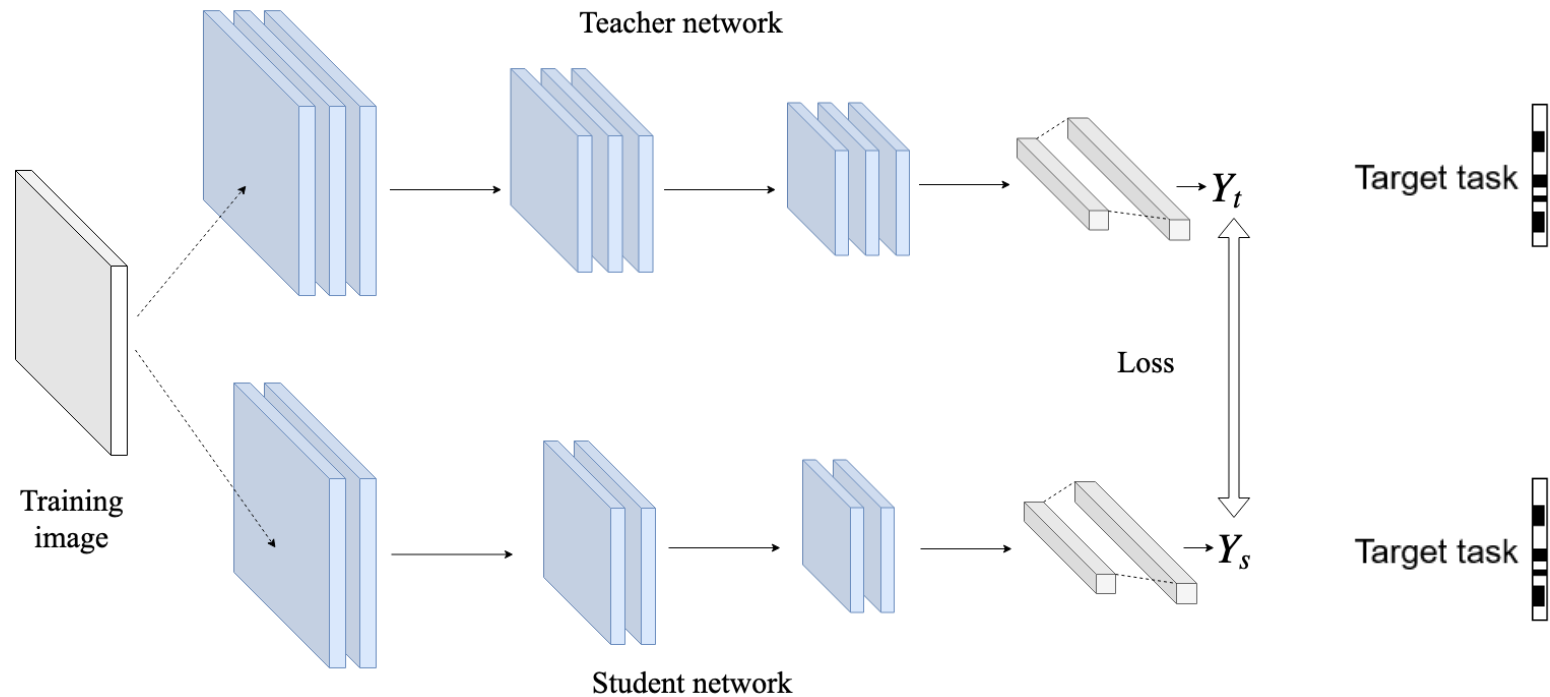
# Introduction

- Original training
  - Train network to do a target task
- Knowledge distillation
  - Train student network to mimic teacher network



# Introduction

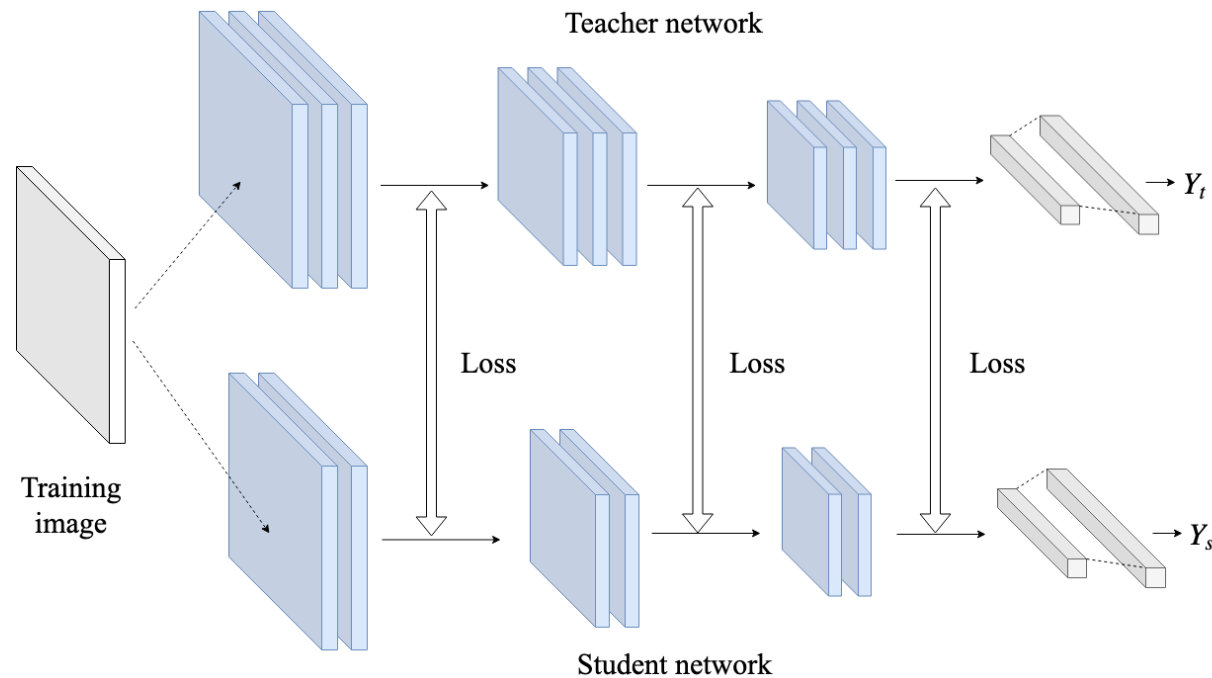
- Output distillation
  - Mimic the output of teacher network
  - Provide supervision of difficulty of training samples (hard example)
  - Smoothing effect on task loss function (easy example)



# Introduction

- **Feature** distillation

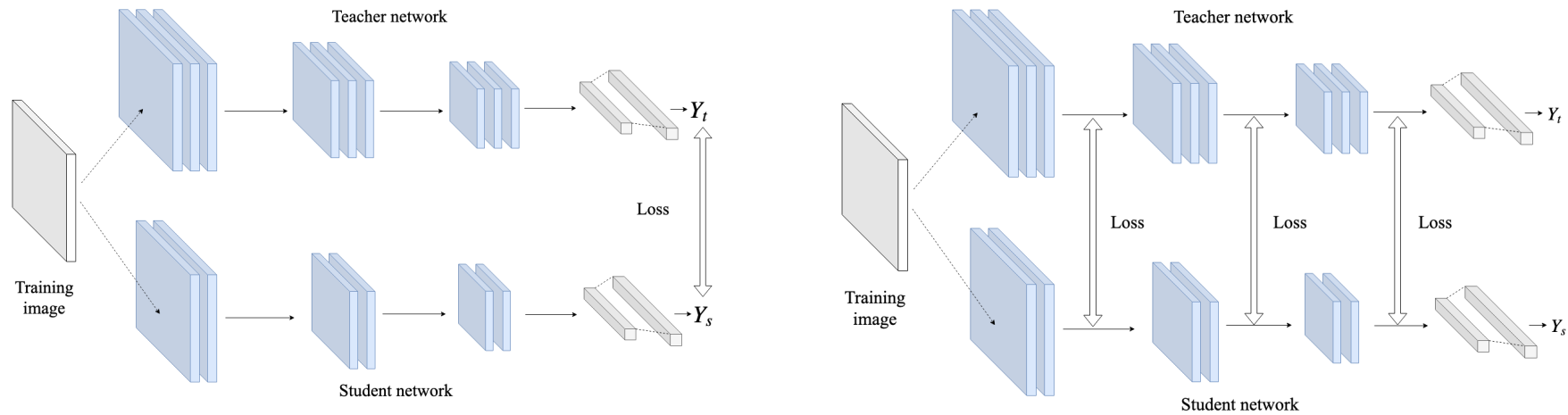
- Mimic the **hidden** layer response (feature) of teacher network
- Trains student network similar to teacher network
- Reduce redundancy on neural network (**compression**)



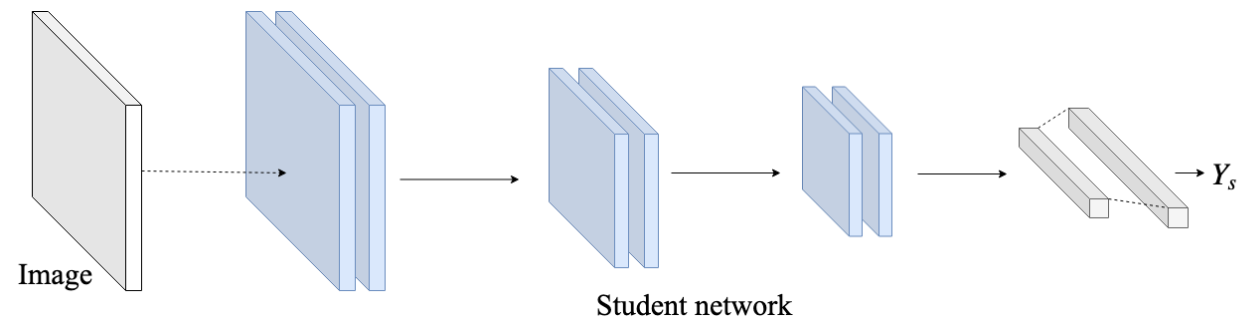


# Introduction

- Training phase (teacher & student network)

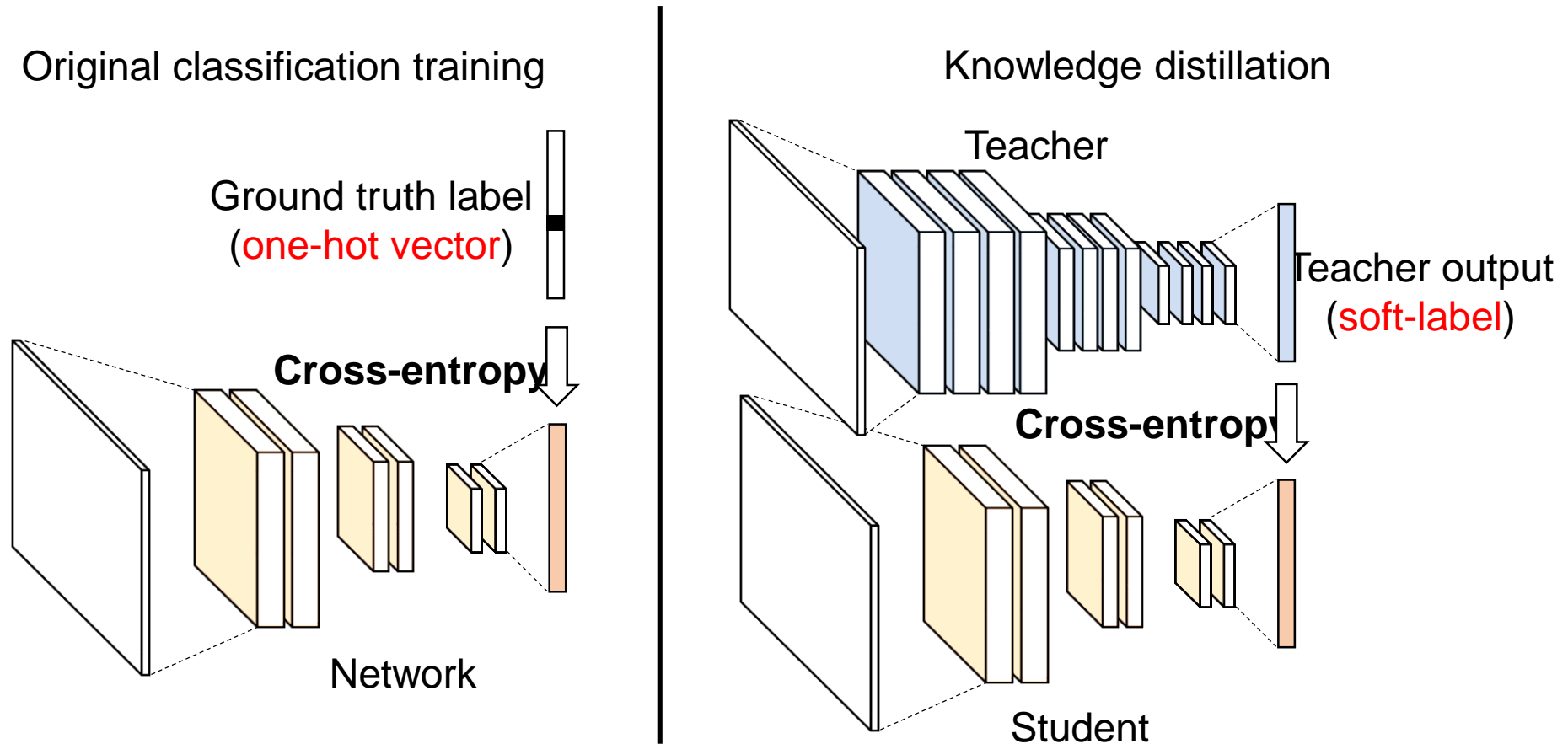


- Test phase (only student network)



# Distillation of decision boundary

- Knowledge distillation (Hinton et al., 2015)
  - Basic distillation method
  - Cross-entropy between teacher and student output



# Distillation of decision boundary

- Knowledge distillation (Hinton et al., 2015)
  - Basic distillation method
  - Cross-entropy between teacher and student output

- Formulation

- Cross-entropy :  $J(\mathbf{a}, \mathbf{b}) = -\mathbf{a}^T \log \mathbf{b}$ .

- Soft-max function :  $\sigma(\cdot)$

- Classification loss

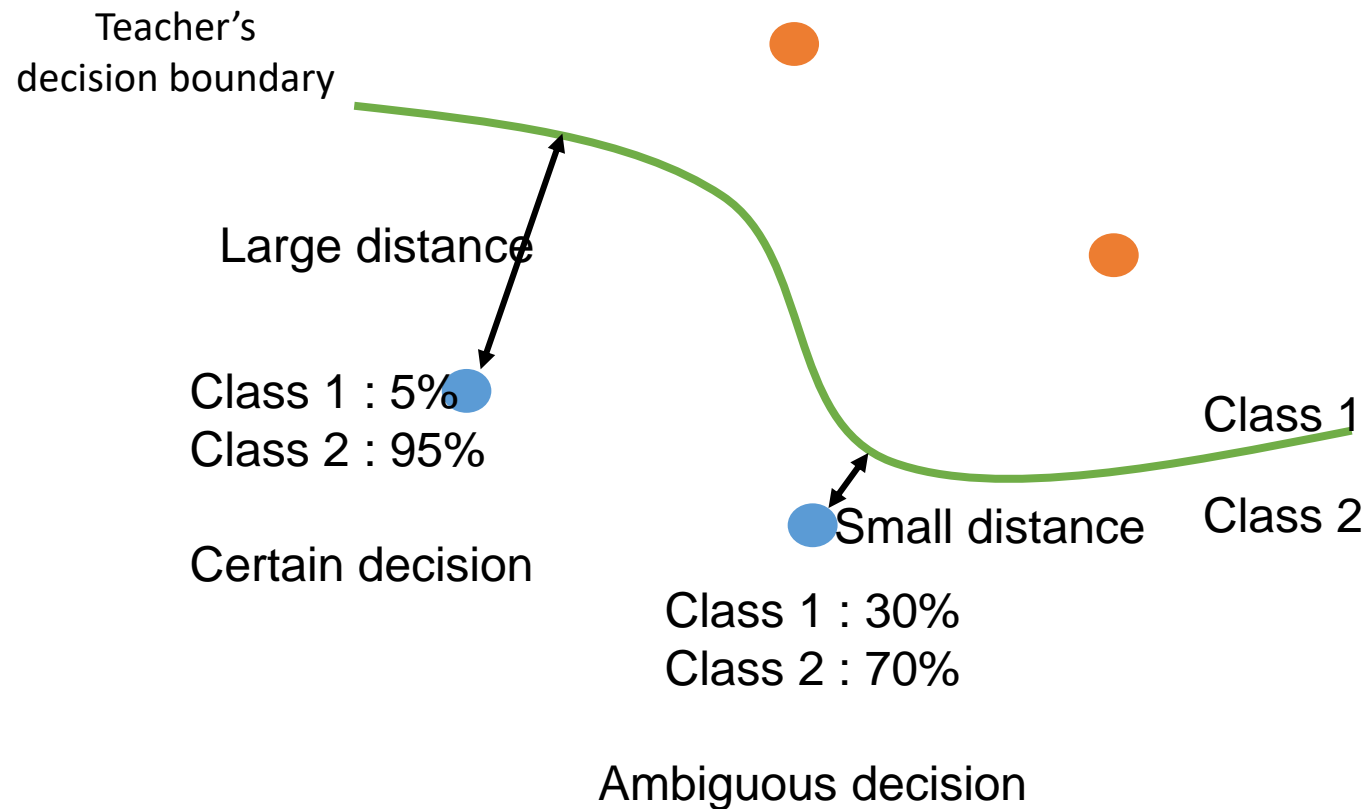
$$\mathcal{L}_{cls}(n) = J(\mathbf{y}_n^{true}, \sigma(f_s(\mathbf{x}_n)))$$

- Distillation loss

$$\mathcal{L}_{KD}(n) = J\left(\sigma\left(\frac{f_t(\mathbf{x}_n)}{T}\right), \sigma\left(\frac{f_s(\mathbf{x}_n)}{T}\right)\right)$$

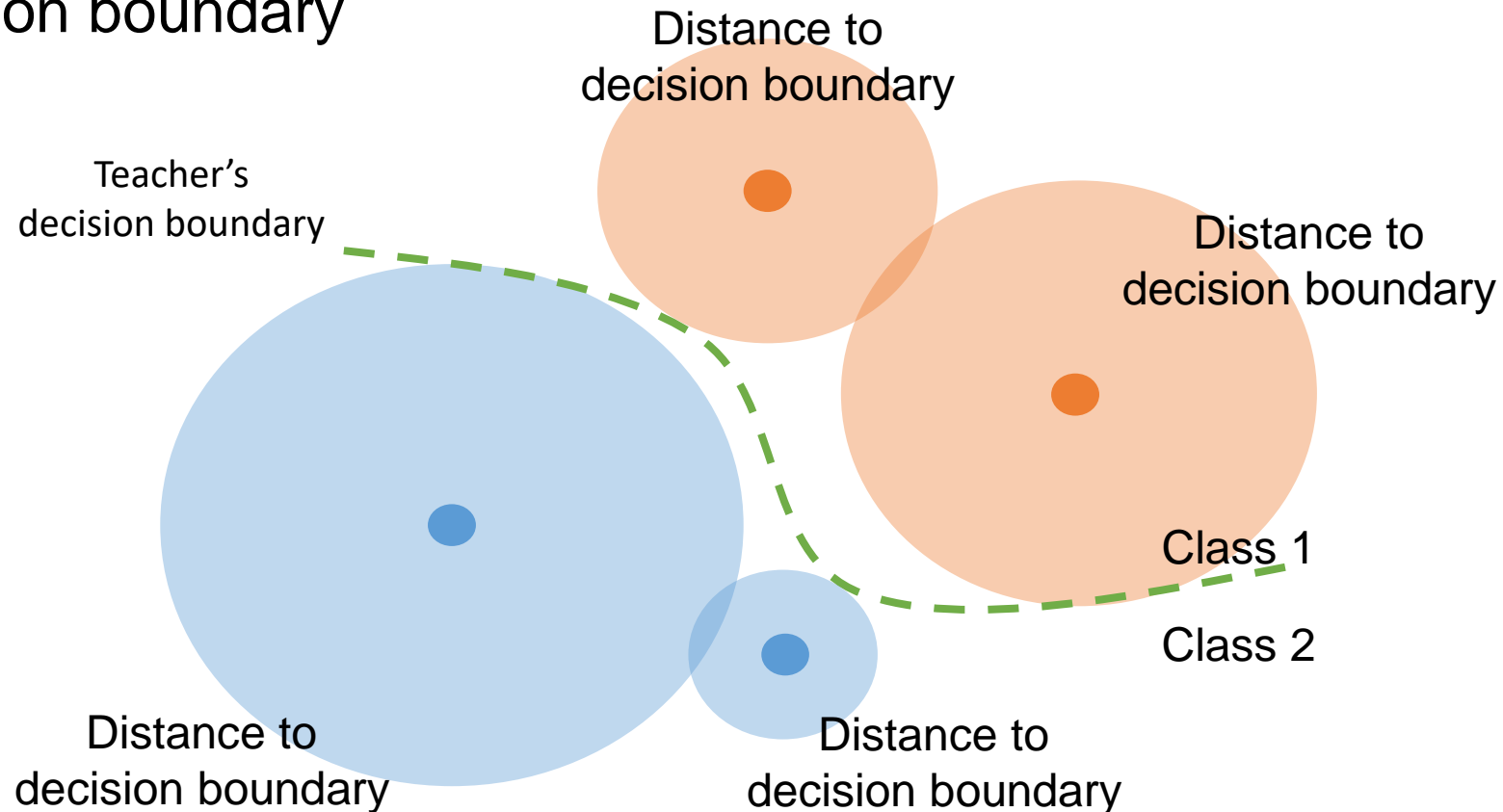
# Distillation of decision boundary

- Decision boundary
  - Network output implies distance to decision boundary
  - Certain decision: far from decision boundary
  - Ambiguous decision: close to decision boundary



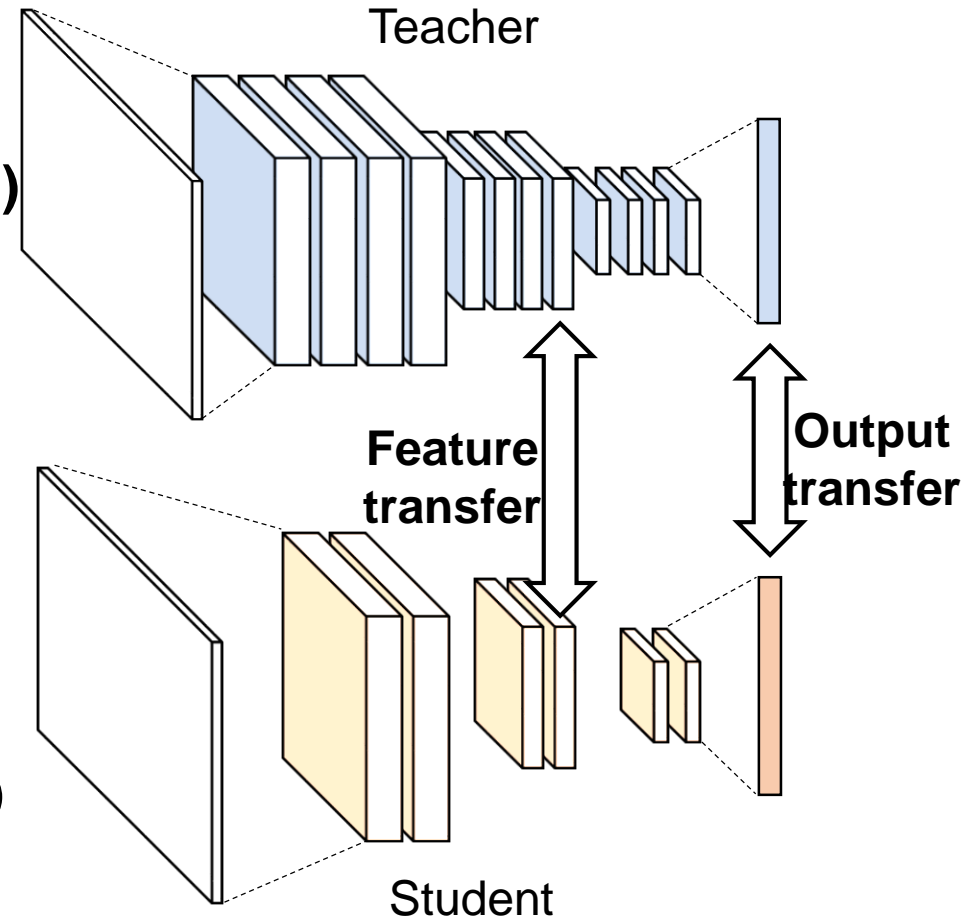
# Distillation of decision boundary

- Decision boundary
  - Soft-label implies distance to decision boundary
  - Thus, knowledge distillation is transferring distance to decision boundary



# Distillation of Activation Boundary

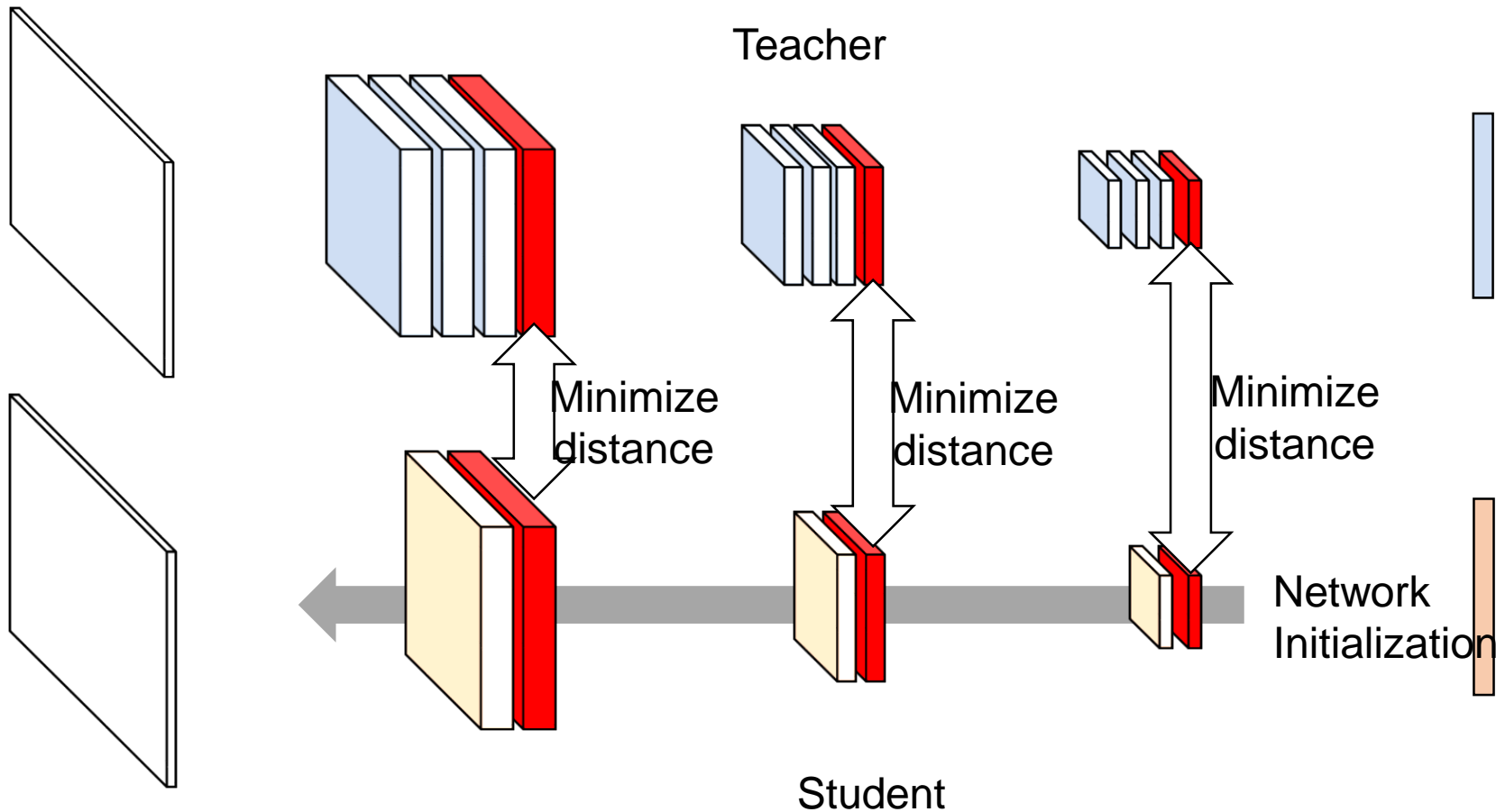
- Method of knowledge distillation
- Output distillation
  - Cross-entropy (KD, 2015)
  - **Boundary Supporting(BS, AAI 2019)**
- Feature distillation
  - (Hidden neuron response)
    - Feature response (FITNET, 2015)
    - Spatial attention (AT, 2017)
    - Gram matrix (FSP, 2017)
    - Jacobian matrix (Jacobian, 2018)
    - **Activation boundary (AB, AAI 2019)**





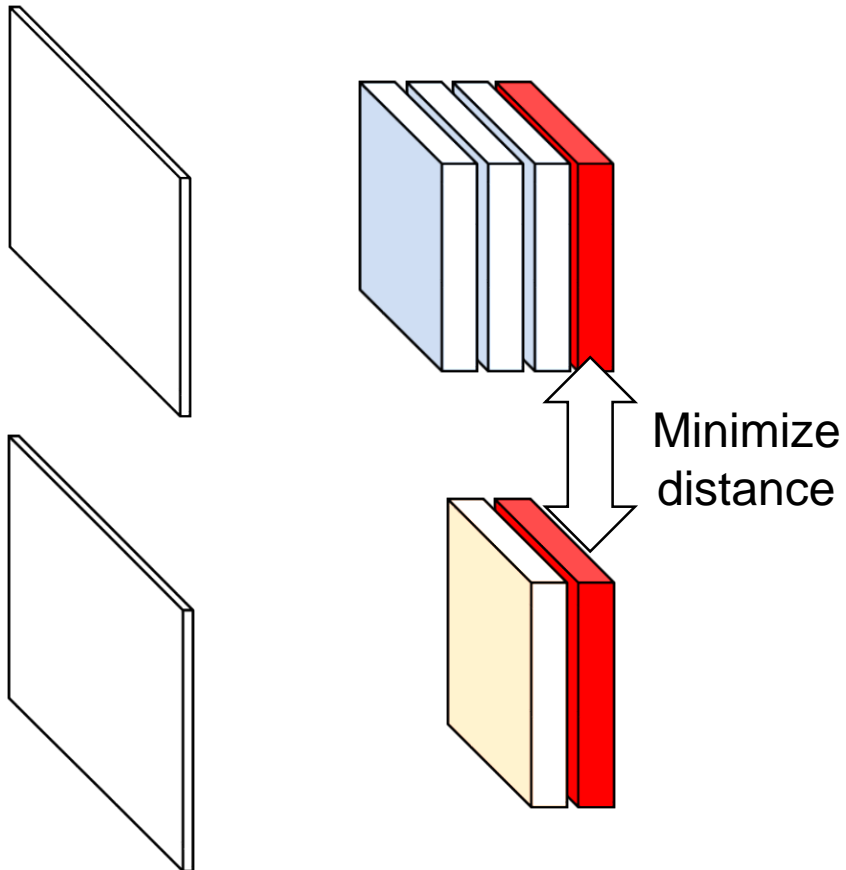
# Distillation of Activation Boundary

- Feature distillation
  - Last layer of same spatial size is selected as transfer target
  - Train student to minimize distance with teacher
  - Network initialization with knowledge transfer



# Distillation of Activation Boundary

- Feature distillation
  - Last layer of same spatial size is selected as transfer target
  - Train student to minimize distance with teacher
  - Network initialization with knowledge transfer



**Problem : how to measure distance ?**

$L_2$  distance (FITNET, 2015)

Spatial attention (AT, 2017)

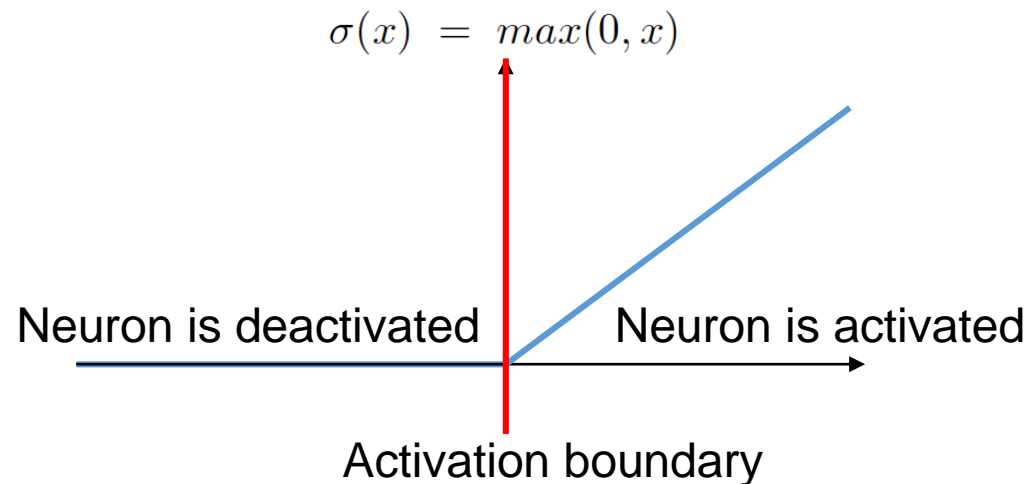
Correlation of channels (FSP, 2017)

Jacobian matrix (Jacobian, 2018)

**Activation boundary (Proposed)**

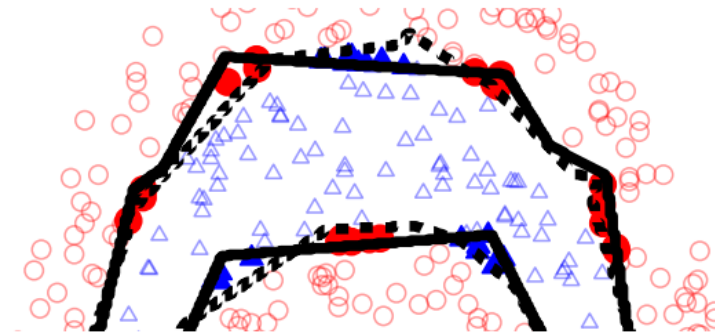
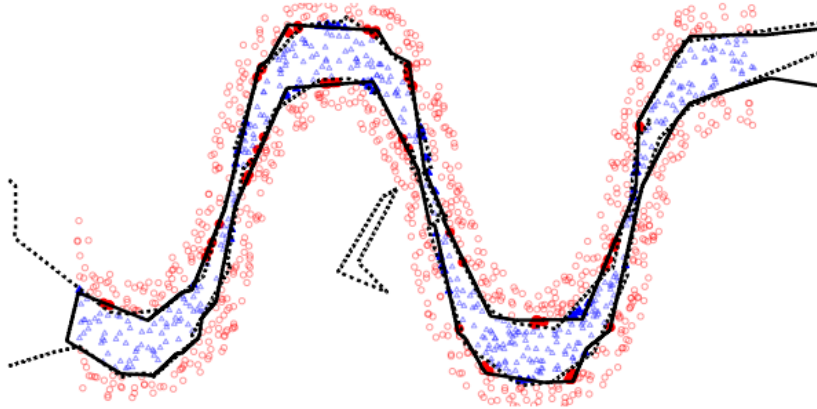
# Activation boundary

- Neuron activation is important in neural network
- Activation boundary
  - Hyperplane that determines whether a neuron is activated or deactivated
- ReLU activation
  - Widely used in image classification
  - Block negative response and pass positive response
  - Activation boundary is zero point



# Activation boundary

On the number of linear regions of deep neural networks (NIPS 2014)



- Studies about activation boundary in ReLU activation
  - On the number of linear regions of deep neural networks (NIPS 2014)
  - Expressiveness of rectifier networks (ICML 2016)
- Activation boundary is basis of decision boundary of classifier
- To create a decision boundary similar to a teacher, there must be an activation boundary similar to the teacher
- Therefore, activation boundary is important in feature distillation

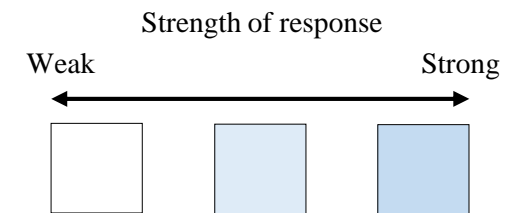
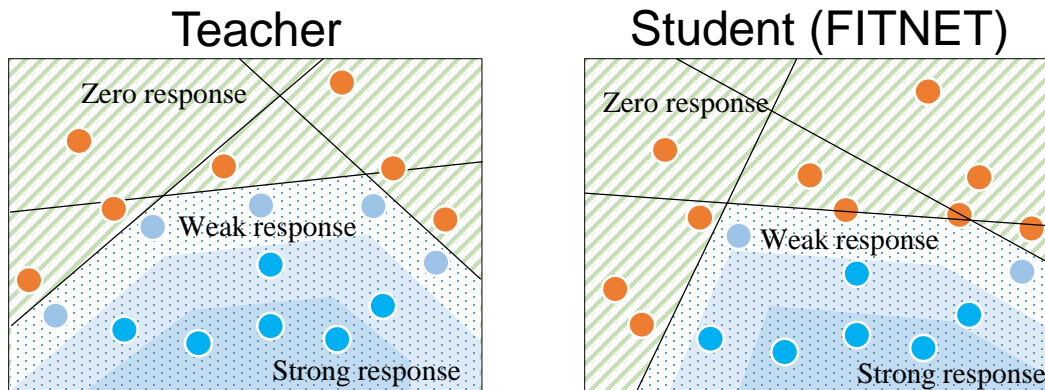
# Activation boundary

- Existing knowledge transfer does **NOT** consider activation boundary
- FITNET : Mean Squared Error (MSE) based transfer

$$\mathcal{L}(\mathbf{I}) = \|\sigma(\mathcal{T}(\mathbf{I})) - \sigma(\mathcal{S}(\mathbf{I}))\|_2^2$$

Feature of teacher  $\mathcal{T}(\mathbf{I})$   
Feature of student  $\mathcal{S}(\mathbf{I})$

- Focused on large response
- Difference between zero and small response is ignored
- Activation boundary is NOT transferred properly



# Activation boundary

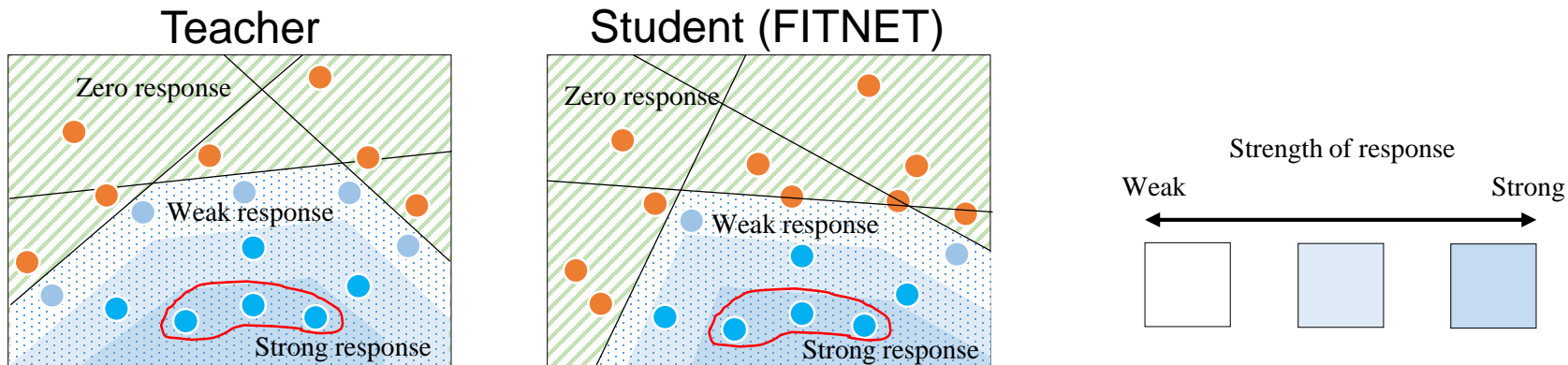
- Existing knowledge transfer does **NOT** consider activation boundary
- FITNET : Mean Squared Error (MSE) based transfer

$$\mathcal{L}(\mathbf{I}) = \|\sigma(\mathcal{T}(\mathbf{I})) - \sigma(\mathcal{S}(\mathbf{I}))\|_2^2$$

Feature of teacher  $\mathcal{T}(\mathbf{I})$

Feature of student  $\mathcal{S}(\mathbf{I})$

- Focused on large response
- Difference between zero and small response is ignored
- Activation boundary is NOT transferred properly





# Activation boundary

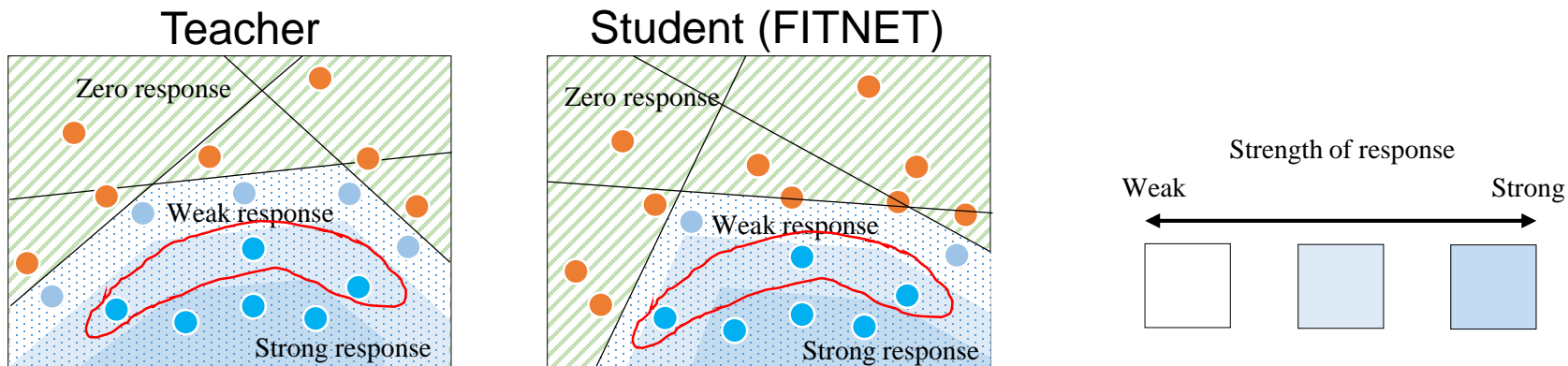
- Existing knowledge transfer does **NOT** consider activation boundary
- FITNET : Mean Squared Error (MSE) based transfer

$$\mathcal{L}(\mathbf{I}) = \|\sigma(\mathcal{T}(\mathbf{I})) - \sigma(\mathcal{S}(\mathbf{I}))\|_2^2$$

Feature of teacher  $\mathcal{T}(\mathbf{I})$

Feature of student  $\mathcal{S}(\mathbf{I})$

- Focused on large response
- Difference between zero and small response is ignored
- Activation boundary is NOT transferred properly



# Activation boundary

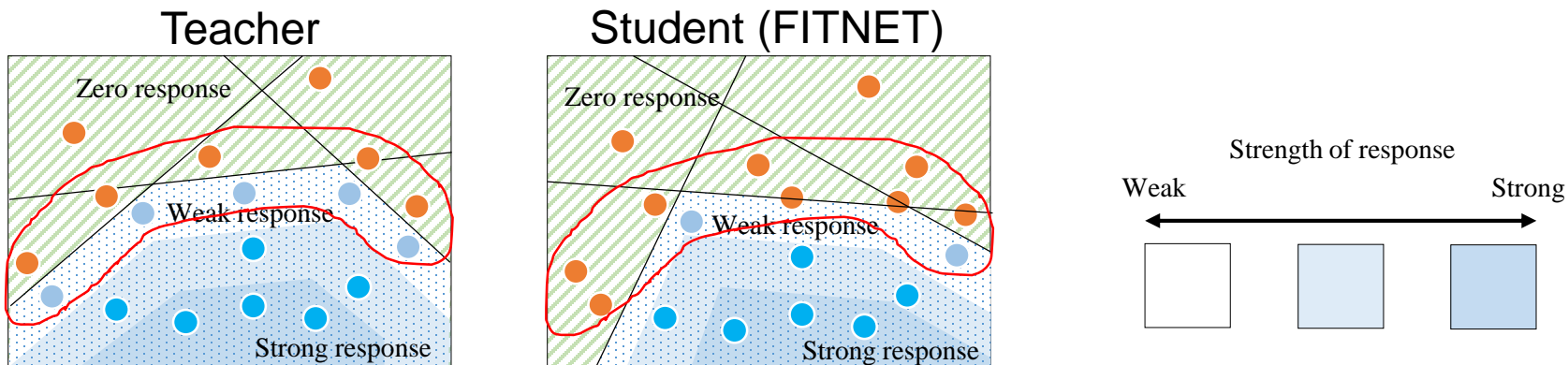
- Existing knowledge transfer does **NOT** consider activation boundary
- FITNET : Mean Squared Error (MSE) based transfer

$$\mathcal{L}(\mathbf{I}) = \|\sigma(\mathcal{T}(\mathbf{I})) - \sigma(\mathcal{S}(\mathbf{I}))\|_2^2$$

Feature of teacher  $\mathcal{T}(\mathbf{I})$

Feature of student  $\mathcal{S}(\mathbf{I})$

- Focused on large response
- Difference between zero and small response is ignored
- Activation boundary is NOT transferred properly



# Activation boundary

- Existing knowledge transfer does **NOT** consider activation boundary
- FITNET : Mean Squared Error (MSE) based transfer

$$\mathcal{L}(\mathbf{I}) = \|\sigma(\mathcal{T}(\mathbf{I})) - \sigma(\mathcal{S}(\mathbf{I}))\|_2^2$$

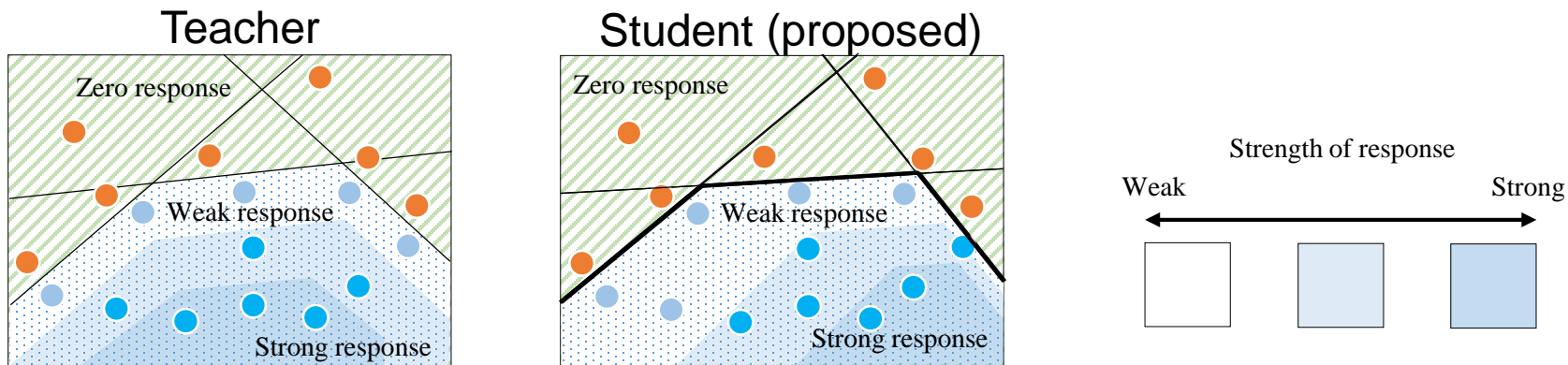
- Simple experiment
  - Student network is trained to minimize  $l_p$  distance with teacher
  - Measure percentage of same activation on teacher and student
  - Method based on  $l_p$  distance can not transfer activation boundary

Percentage of same activation on teacher and student

	$l_2$ loss	$l_1$ loss	$l_{0.5}$ loss
Layer 1	56.1%	66.9%	68.8%
Layer 2	67.3%	72.5%	73.0%
Layer 3	56.0%	56.1%	56.4%

# Activation boundary

- Motivation
  - Activation boundaries are important for knowledge transfer
  - We propose a knowledge transfer focused on activation boundaries
  - Proposed method transfers activation boundaries
  - Although strength of response could be different



# Proposed method

- Activation indicator function

$$\rho(x) = \begin{cases} 1, & \text{if } x > 0 \\ 0, & \text{otherwise.} \end{cases}$$

Neuron is activated  
Neuron is deactivated

- Activation transfer loss

$$loss = \sum_{i=1}^N |\rho(T_i) - \rho(S_i)|$$

Feature of teacher  $T_i$   
Feature of student  $S_i$   $i \in [1, \dots, N]$

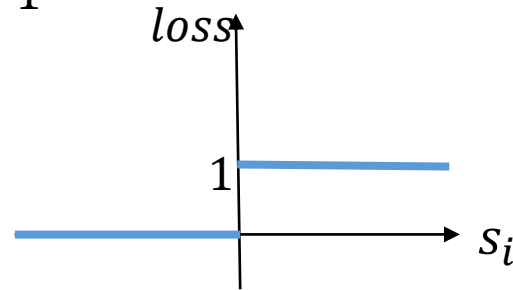
- Indicates number of different activations
- Magnitude of response is ignored
- Only [activated or deactivated] is considered
- Represents similarity of activation boundaries

# Proposed method

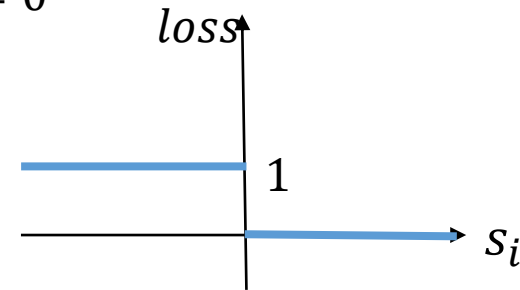
- Activation transfer loss

$$loss = \sum_{i=1}^N |\rho(T_i) - \rho(S_i)|$$

if  $\rho(T_i) = 1$



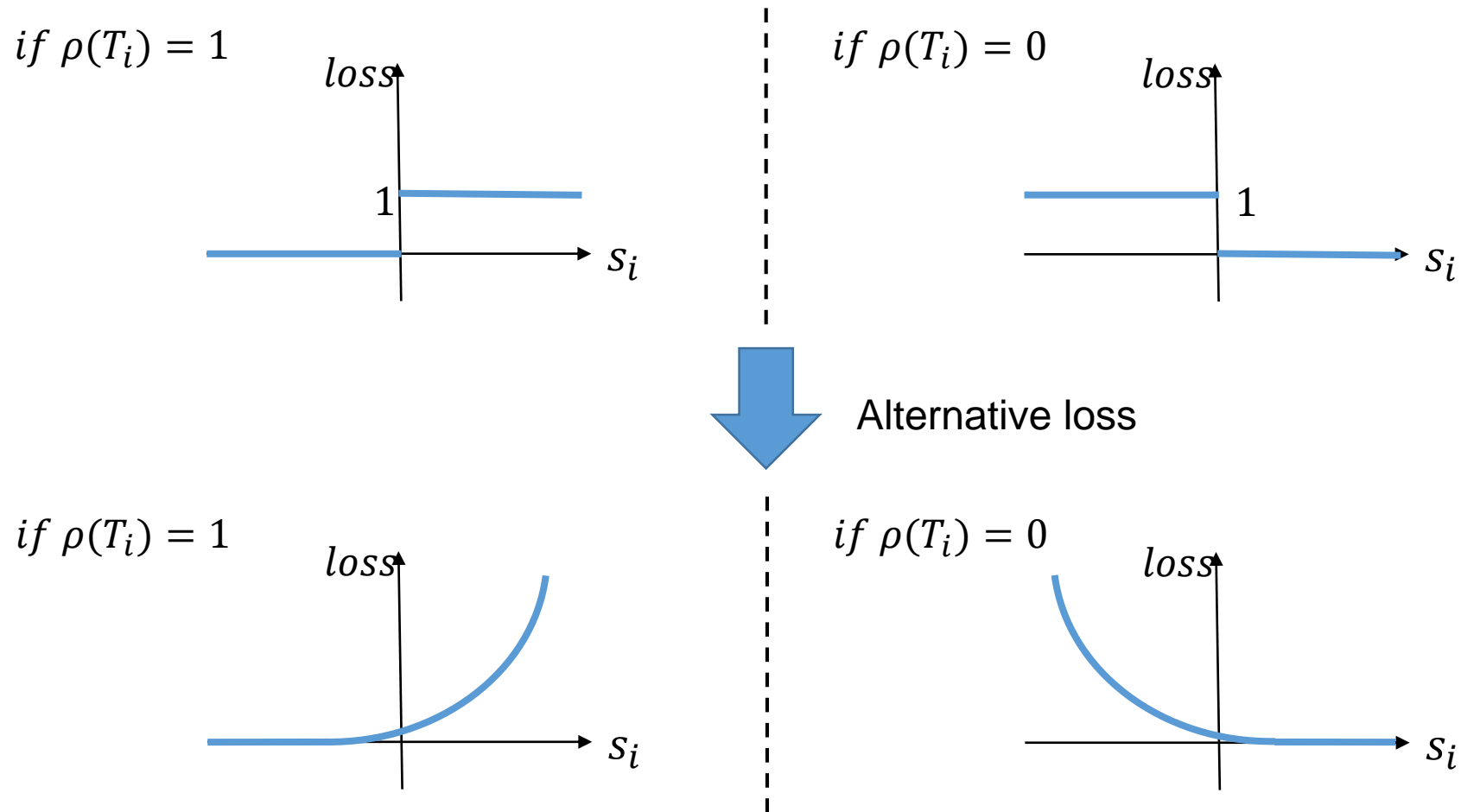
if  $\rho(T_i) = 0$



- Non differentiable function
- Needs alternative function for gradient descent method

# Proposed method

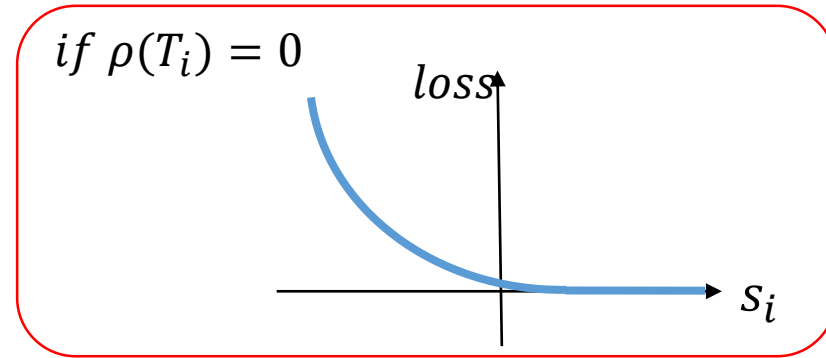
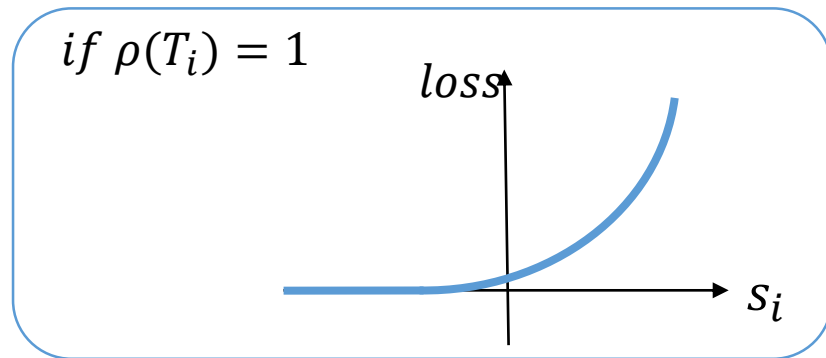
- Activation transfer loss





# Proposed method

- Alternative loss

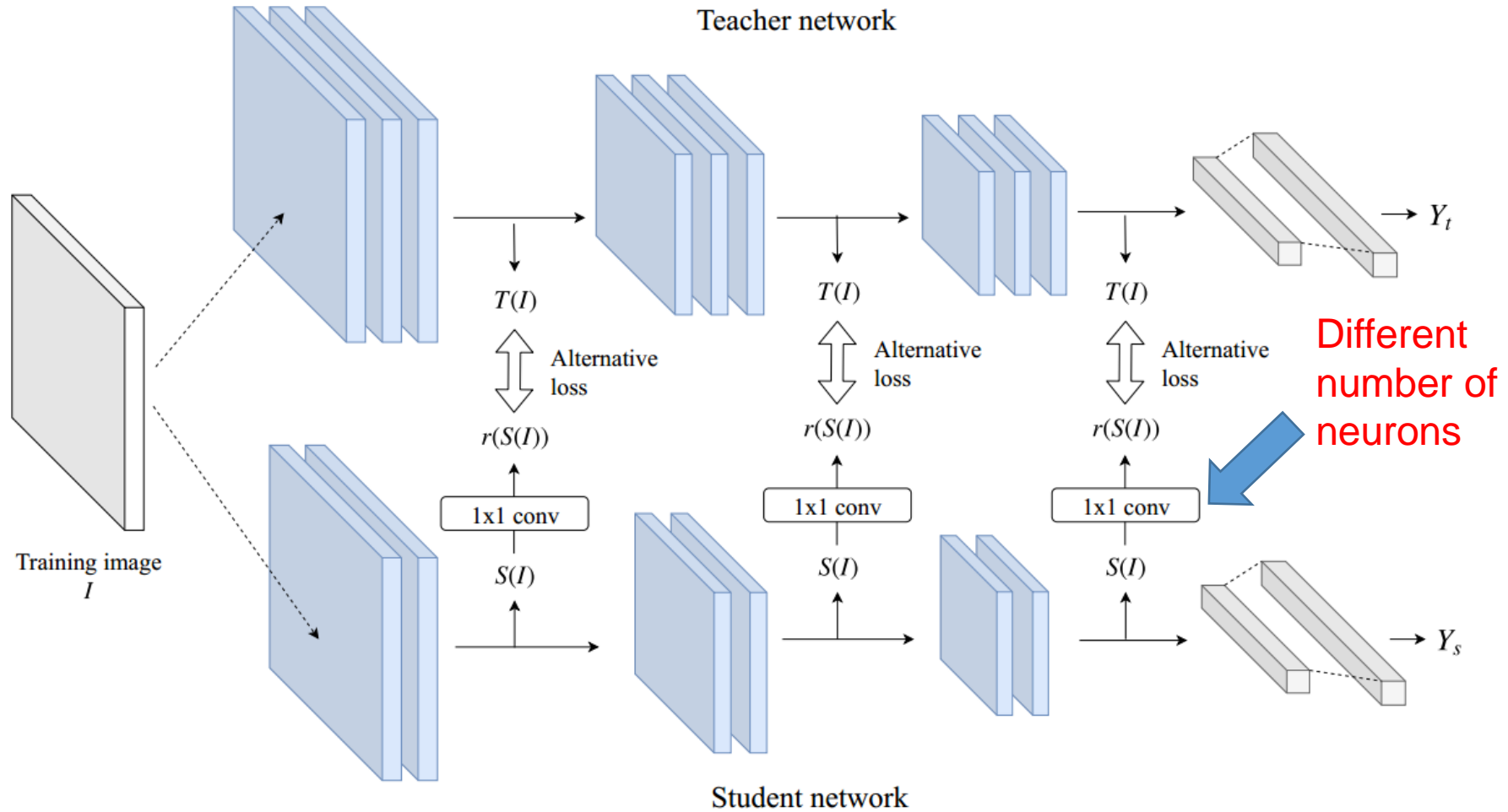


$$loss = \sum_{i=1}^N \rho(T_i) (\sigma(\mu - S_i))^2 + (1 - \rho(T_i)) (\sigma(\mu + S_i))^2 \quad \sigma(x) = \max(0, x)$$

- Inspired by hinge loss function
- Differentiable function
- $\mu$  is margin (for using non-activative samples)

# Proposed method

- Overall framework



# Experiments

- Neuron activation
  - Student network is trained to minimize alternative loss
  - Much better than  $l_p$  distance
  - Activations of teacher and student are almost same
  - Alternative loss well transfers activation boundaries
  - CIFAR-10 / Wide Residual Network 16-4

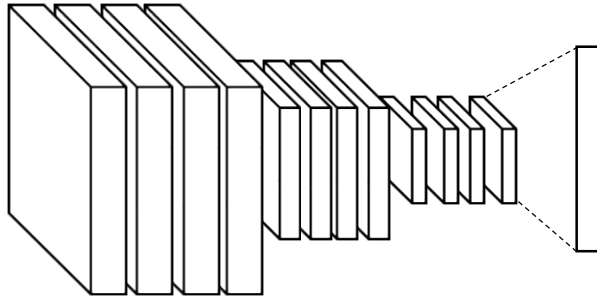
Percentage of same activation on teacher and student

	$l_2$ loss	$l_1$ loss	$l_{0.5}$ loss	Proposed
Layer 1	56.1%	66.9%	68.8%	<b>96.3%</b>
Layer 2	67.3%	72.5%	73.0%	<b>96.4%</b>
Layer 3	56.0%	56.1%	56.4%	<b>92.8%</b>

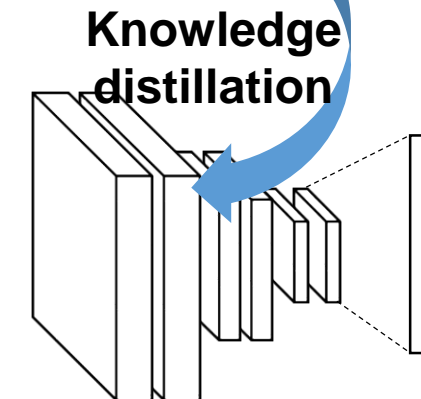
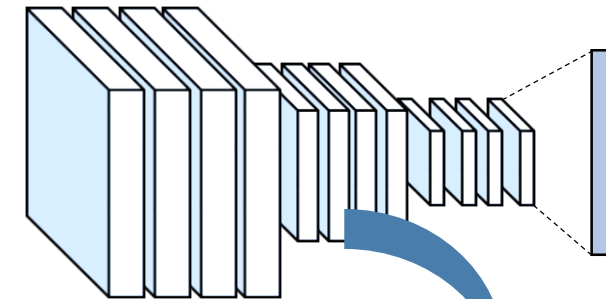
# Experiments

- Knowledge transfer for the same task
  - When there is trained large network
  - The goal is to train a small network that does the same task

1. Teacher network (large) is trained on CIFAR-10



2. Student network (small) is trained on CIFAR-10 with distillation from teacher



**Knowledge distillation**

# Experiments

- **Fast training**
  - Knowledge transfer can accelerate training student
  - **Training epochs** (distillation + classification)
  - Proposed method is **much better in small epochs**
  - CIFAR-10, WRN 22-4 (teacher), WRN 16-2 (student)

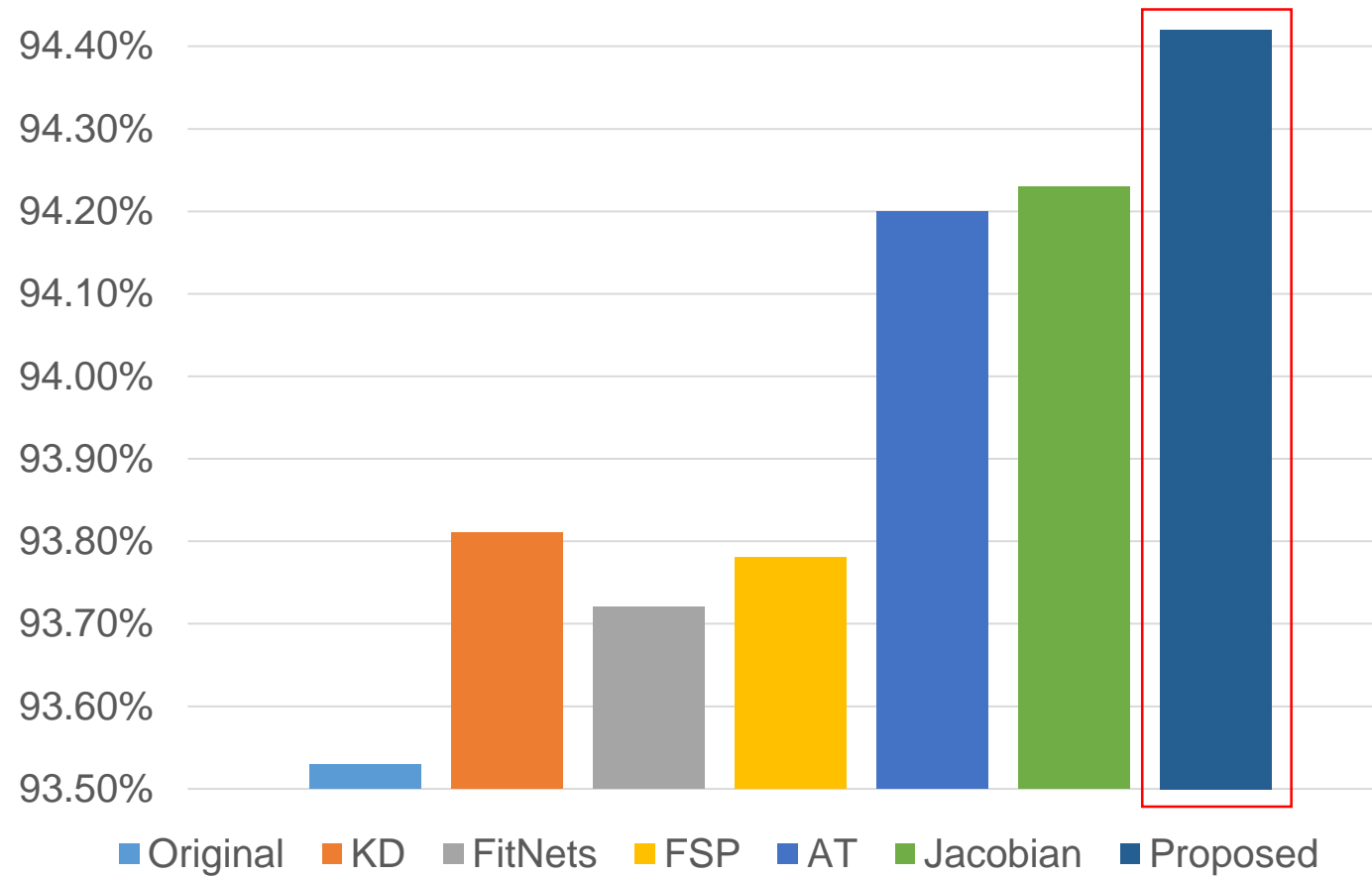
← Less training epochs

Training epochs	1+1	1+5	3+12	5+25	10+50	20+100
Without distillation	43.37%	17.72%	11.76%	8.63%	7.41%	6.47%
Output distillation (KD)	48.42%	19.80%	12.09%	8.66%	6.80%	6.19%
FITNET + KD	48.16%	19.82%	11.10%	8.38%	7.02%	6.28%
FSP + KD	43.51%	19.29%	11.15%	8.48%	6.87%	6.22%
AT + KD	37.66%	14.14%	8.35%	6.68%	5.94%	5.80%
Jacobian + KD	38.46%	14.29%	8.37%	6.98%	5.98%	5.77%
Proposed + KD	<b>16.39%</b>	<b>11.16%</b>	<b>6.95%</b>	<b>6.08%</b>	<b>5.72%</b>	<b>5.58%</b>

Error rate (%)

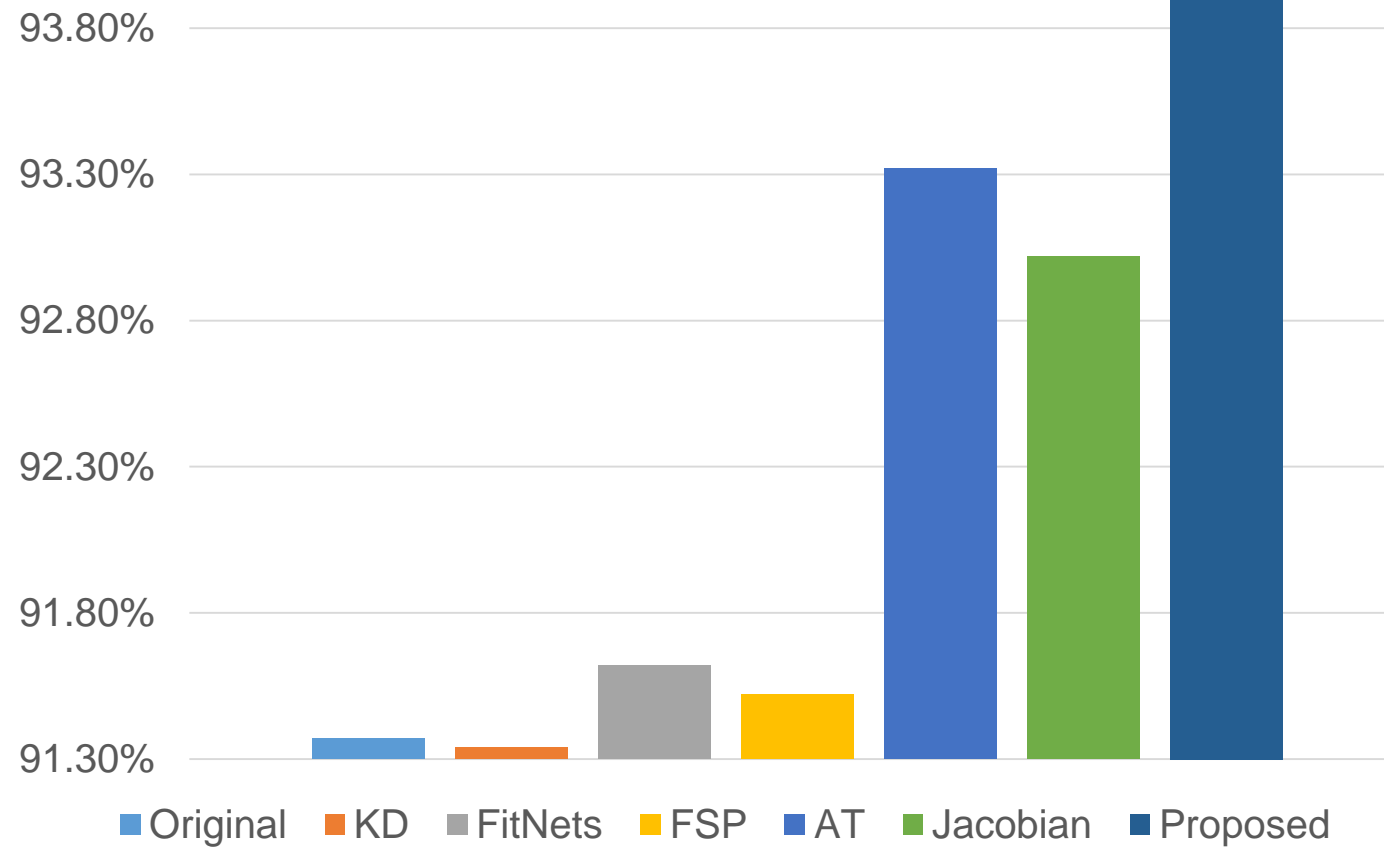
# Experiments

- Fast training (120 epoch)



# Experiments

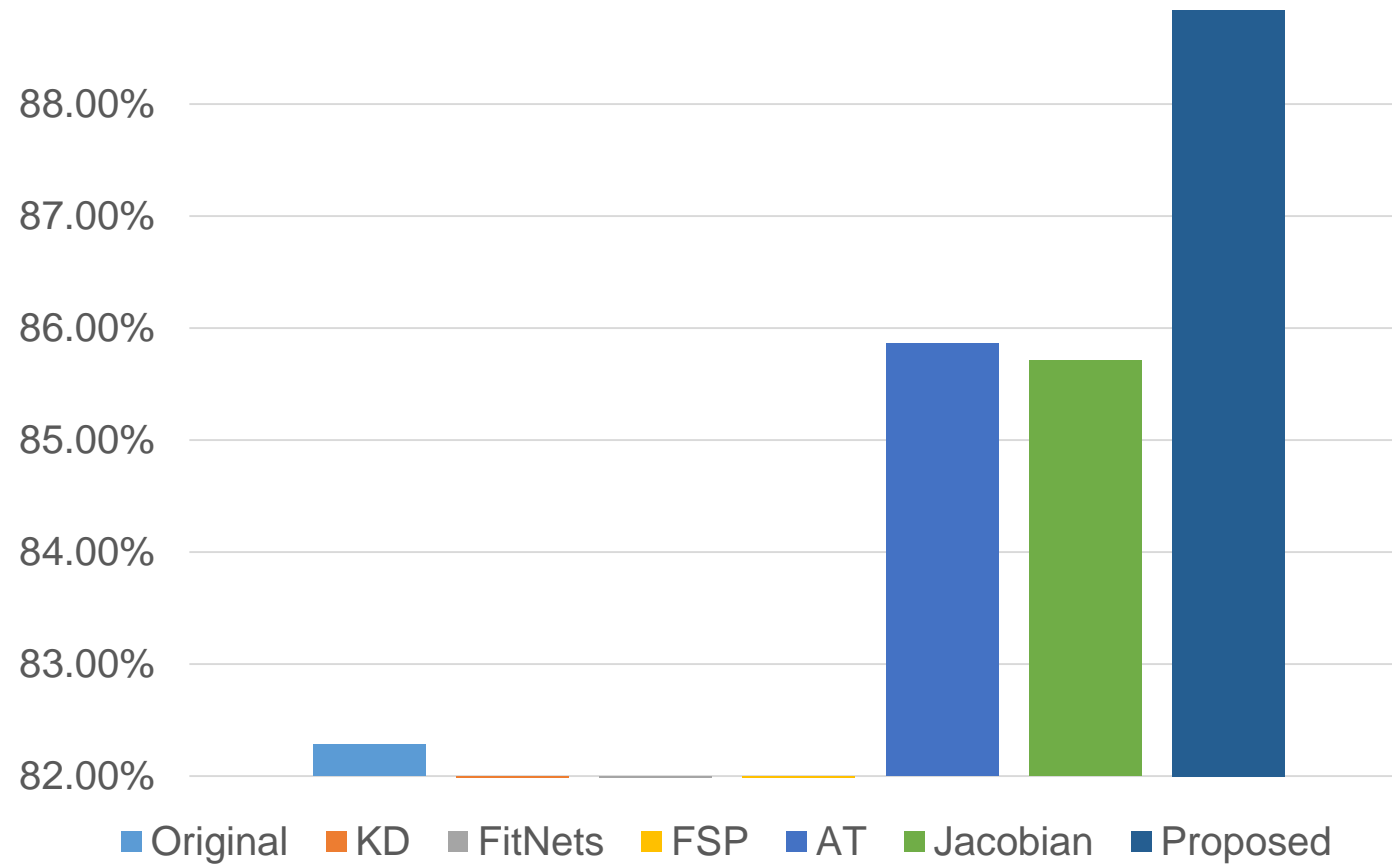
- Fast training (30 epoch)





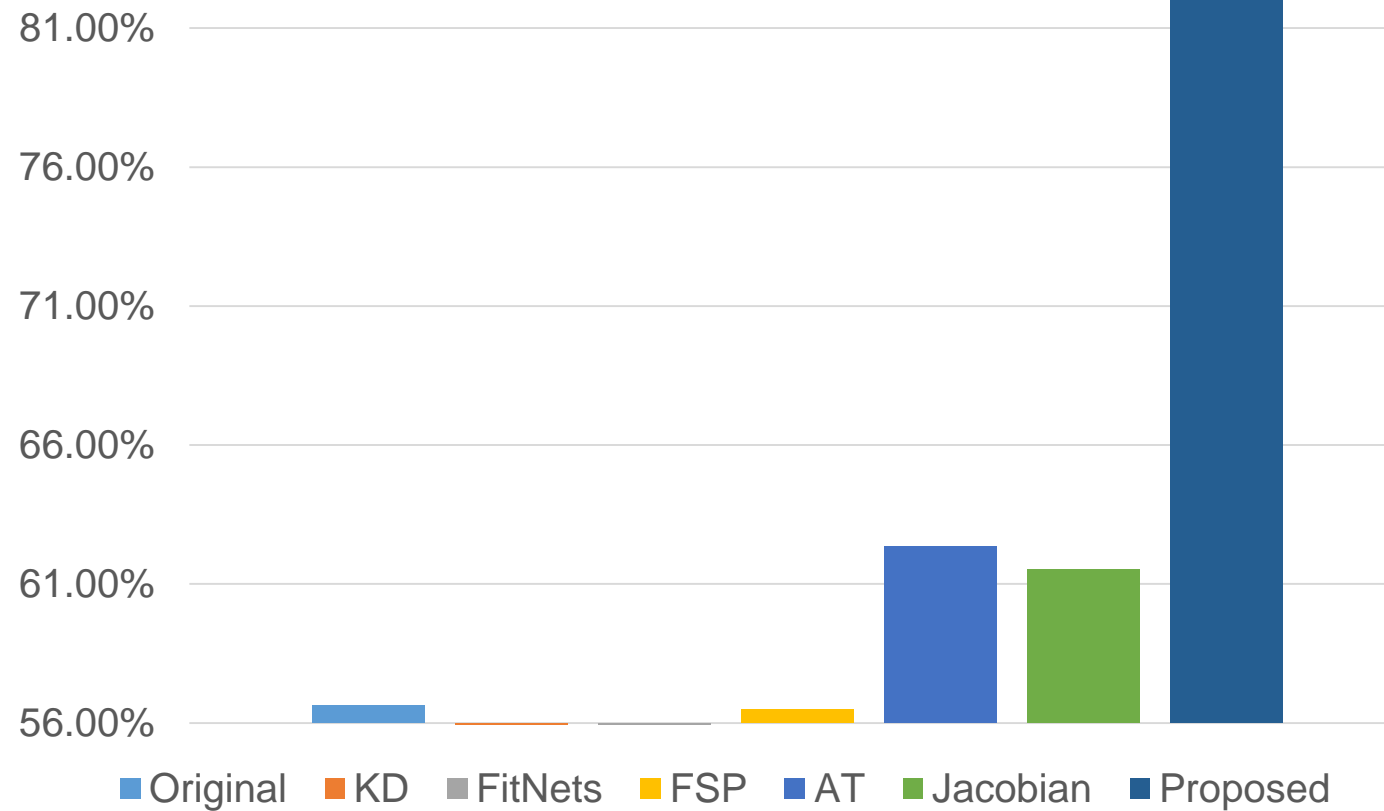
# Experiments

- Fast training (6 epoch)



# Experiments

- Fast training (2 epoch)



# Experiments

- **Small training data**
  - Teacher networks helps generalization of student network
  - Most of algorithms can improves performance with small size data
  - Proposed method **much better in small training data**
  - CIFAR-10, WRN 22-4 (teacher), WRN 16-2 (student)

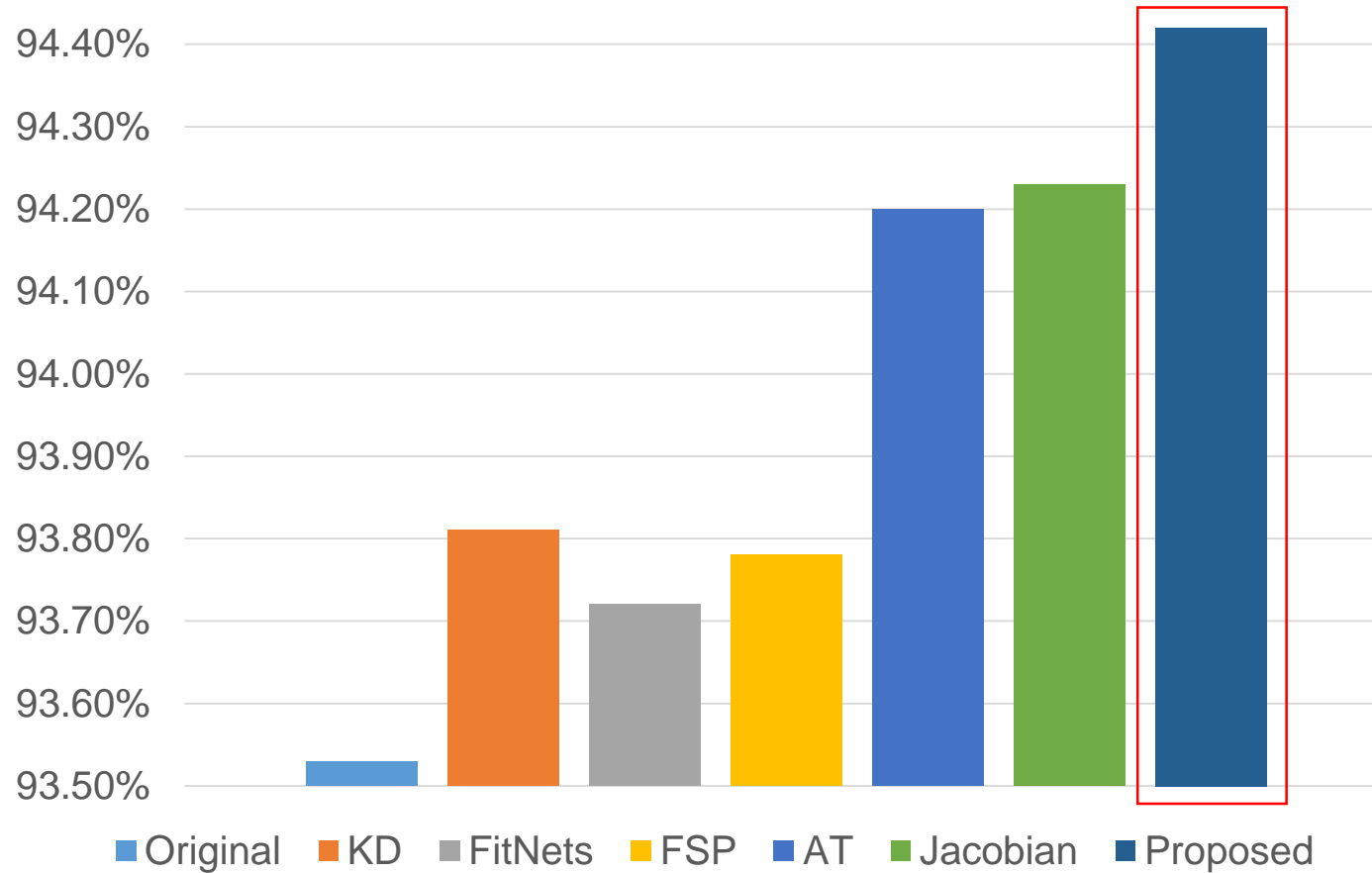
Less training data ←

Percentage of data	0.1%	1%	5%	10%	20%	100%
Without distillation	73.83%	48.41%	28.12%	21.76%	15.26%	6.47%
Output distillation (KD)	74.62%	48.34%	28.13%	21.21%	14.62%	6.19%
FITNET + KD	74.81%	49.91%	27.28%	21.42%	14.52%	6.28%
FSP + KD	73.96%	47.98%	26.90%	20.80%	13.85%	6.22%
AT + KD	67.54%	37.11%	18.86%	15.61%	9.94%	5.80%
Jacobian + KD	68.65%	36.99%	18.34%	15.03%	9.83%	5.77%
Proposed + KD	<b>50.32%</b>	<b>21.54%</b>	<b>14.99%</b>	<b>13.09%</b>	<b>9.16%</b>	<b>5.58%</b>

Error rate (%)

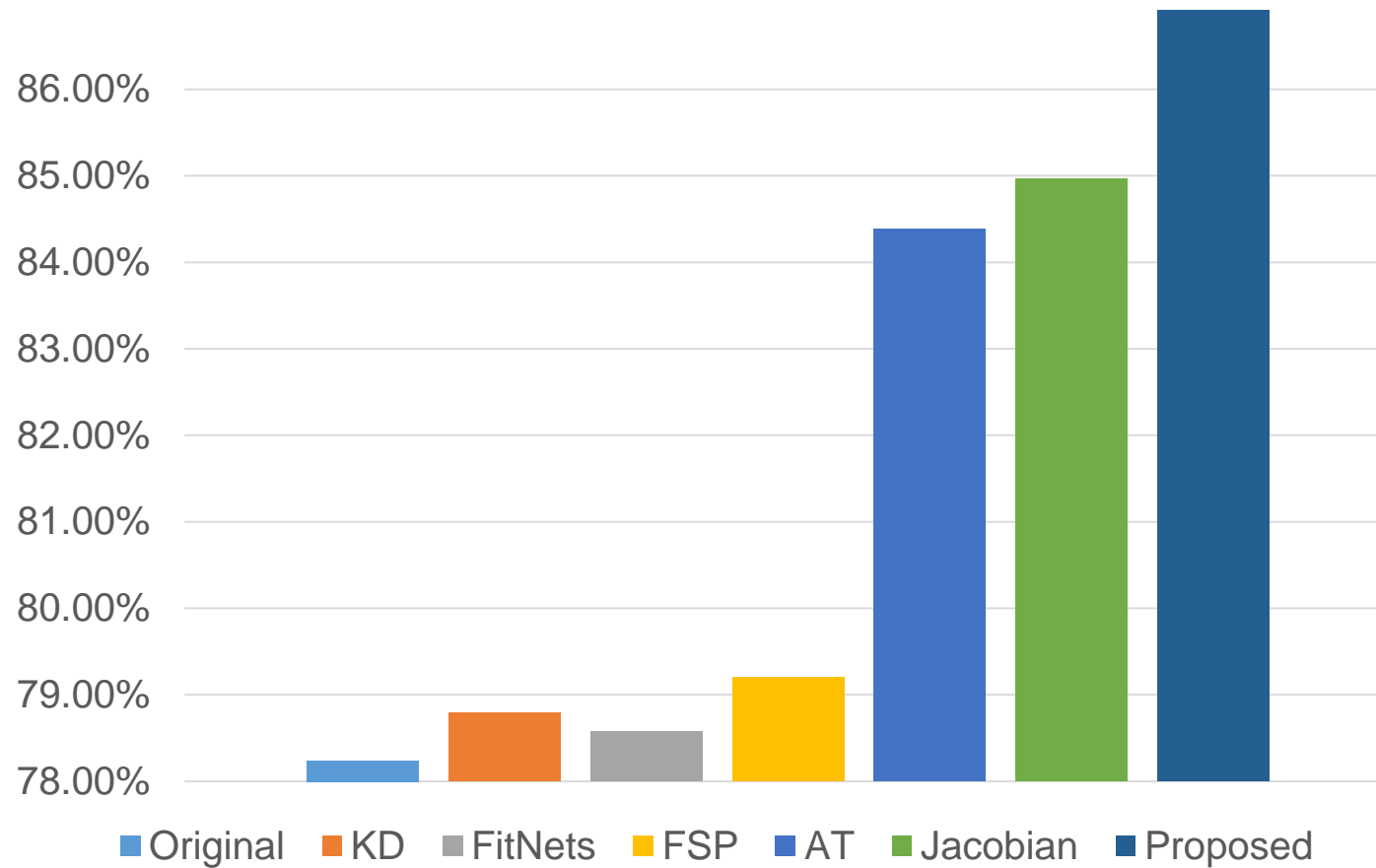
# Experiments

- Small training data (100%)



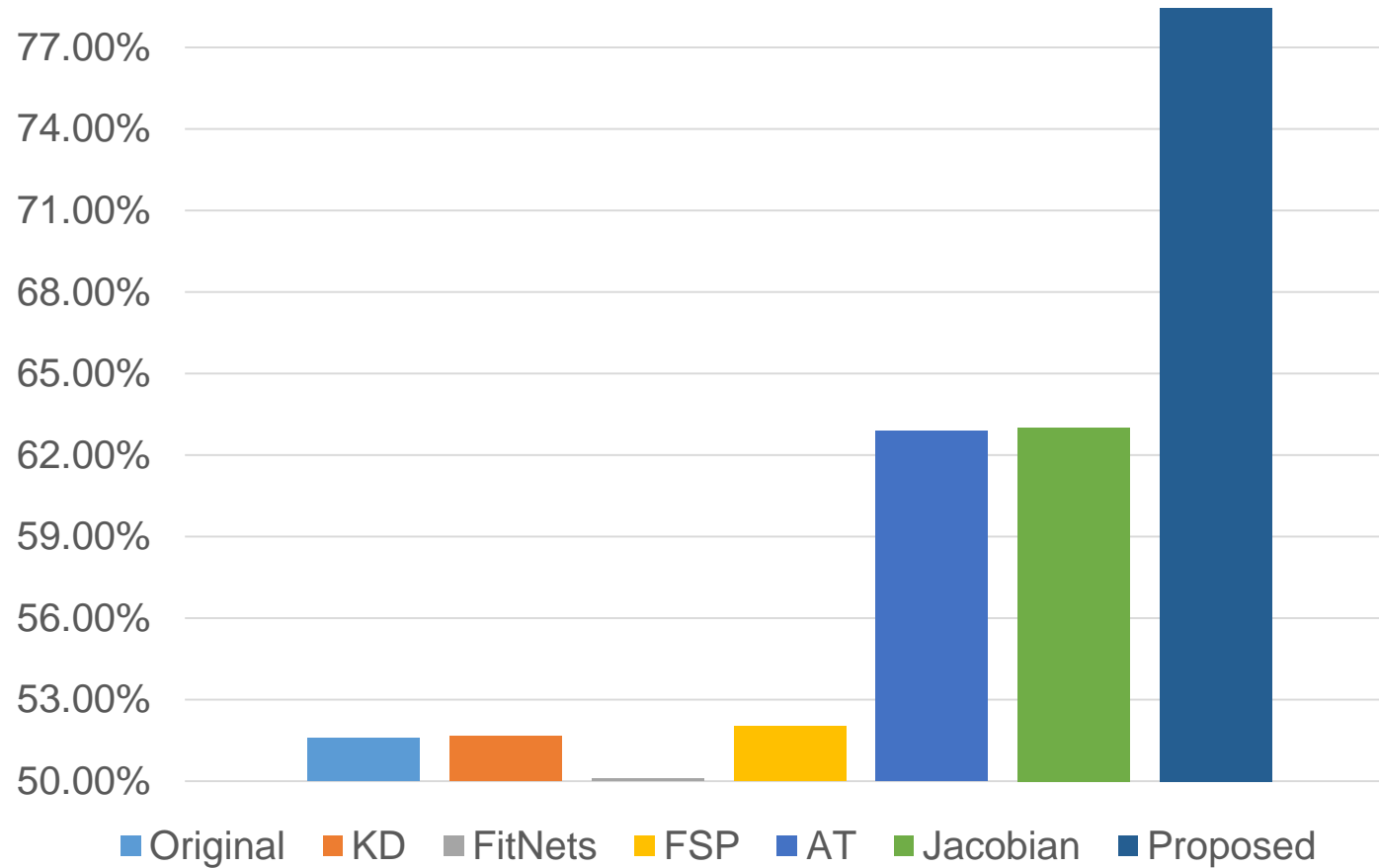
# Experiments

- Small training data (10%)



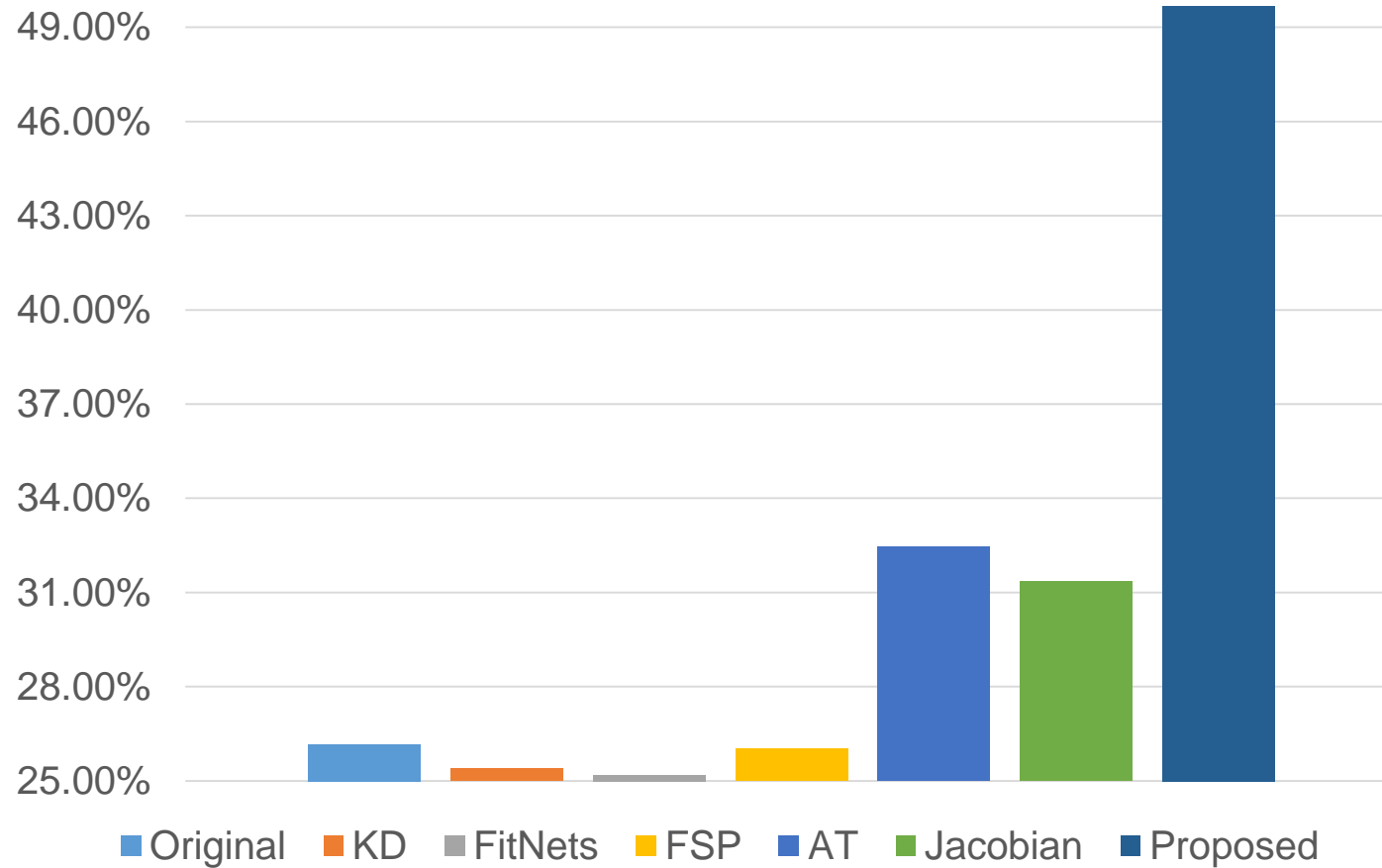
# Experiments

- Small training data (1%)



# Experiments

- Small training data (0.1%)





# Experiments

- Various **type of compression**

Compression type	Teacher	Student
Depth	WRN 22-4	WRN 10-4
Channel	WRN 16-4	WRN 16-2
Depth & Channel	WRN 22-4	WRN 16-2
Tiny network	WRN 22-4	WRN 10-1
Same network	WRN 16-4	WRN 16-4

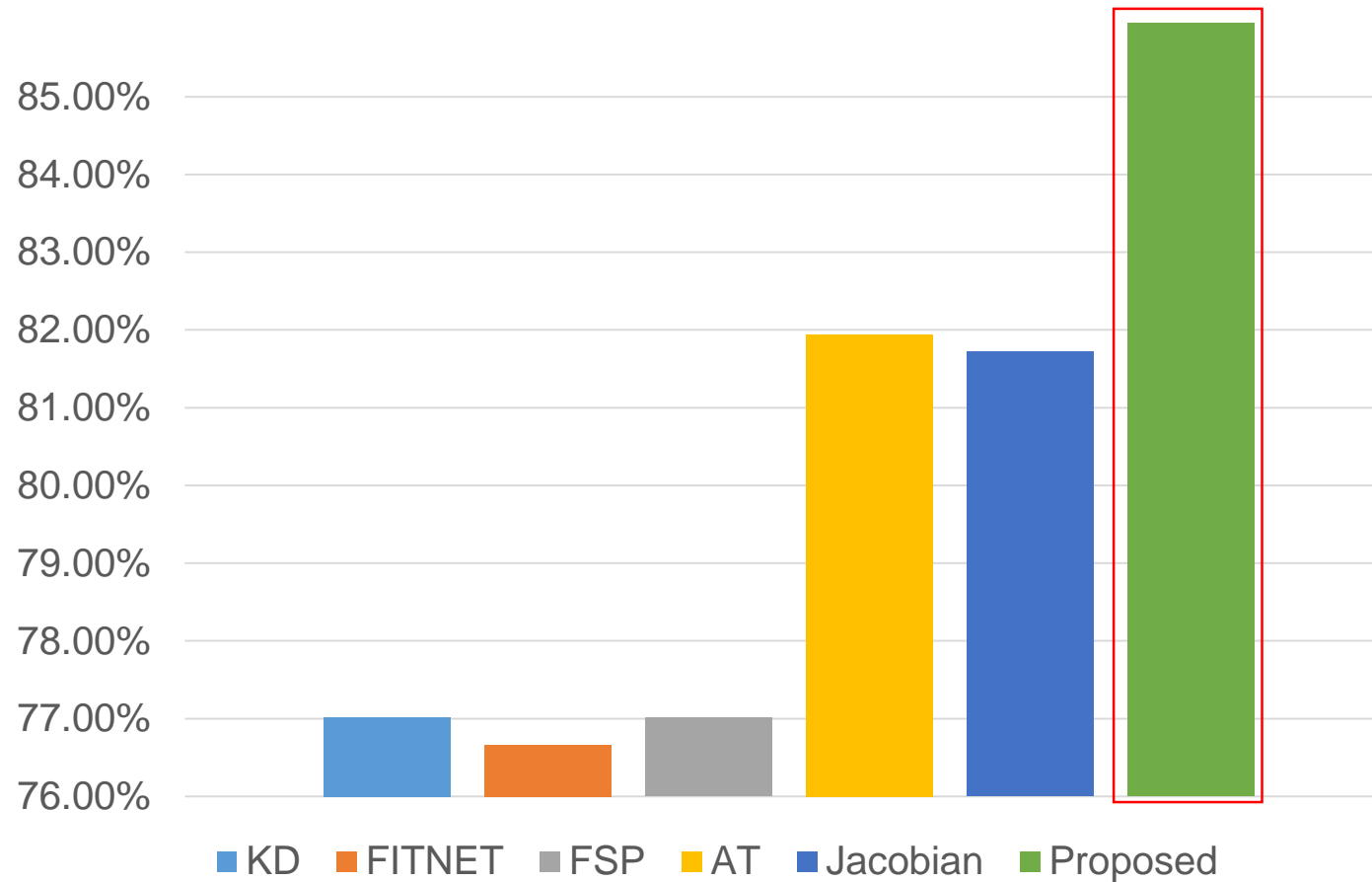
- Knowledge transfer can be applied with various kinds of network
- We measured performance on various type of compression
- Proposed method has **the best performance**
- 10% of training data, 120 epochs

Compression type	Size ratio	KD	FITNET	FSP	AT	Jacobian	Proposed
Depth	27.9%	22.98%	23.34%	22.99%	18.06%	18.28%	<b>14.05%</b>
Channel	25.2%	20.48%	19.98%	19.78%	14.81%	14.41%	<b>11.62%</b>
Depth & Channel	16.1%	21.21%	21.42%	20.80%	15.61%	15.03%	<b>13.09%</b>
Tiny network	1.8%	29.57%	29.18%	28.70%	29.44%	28.70%	<b>23.27%</b>
Same network	100%	18.29%	17.91%	17.81%	12.03%	11.28%	<b>6.63%</b>

Error rate (%)

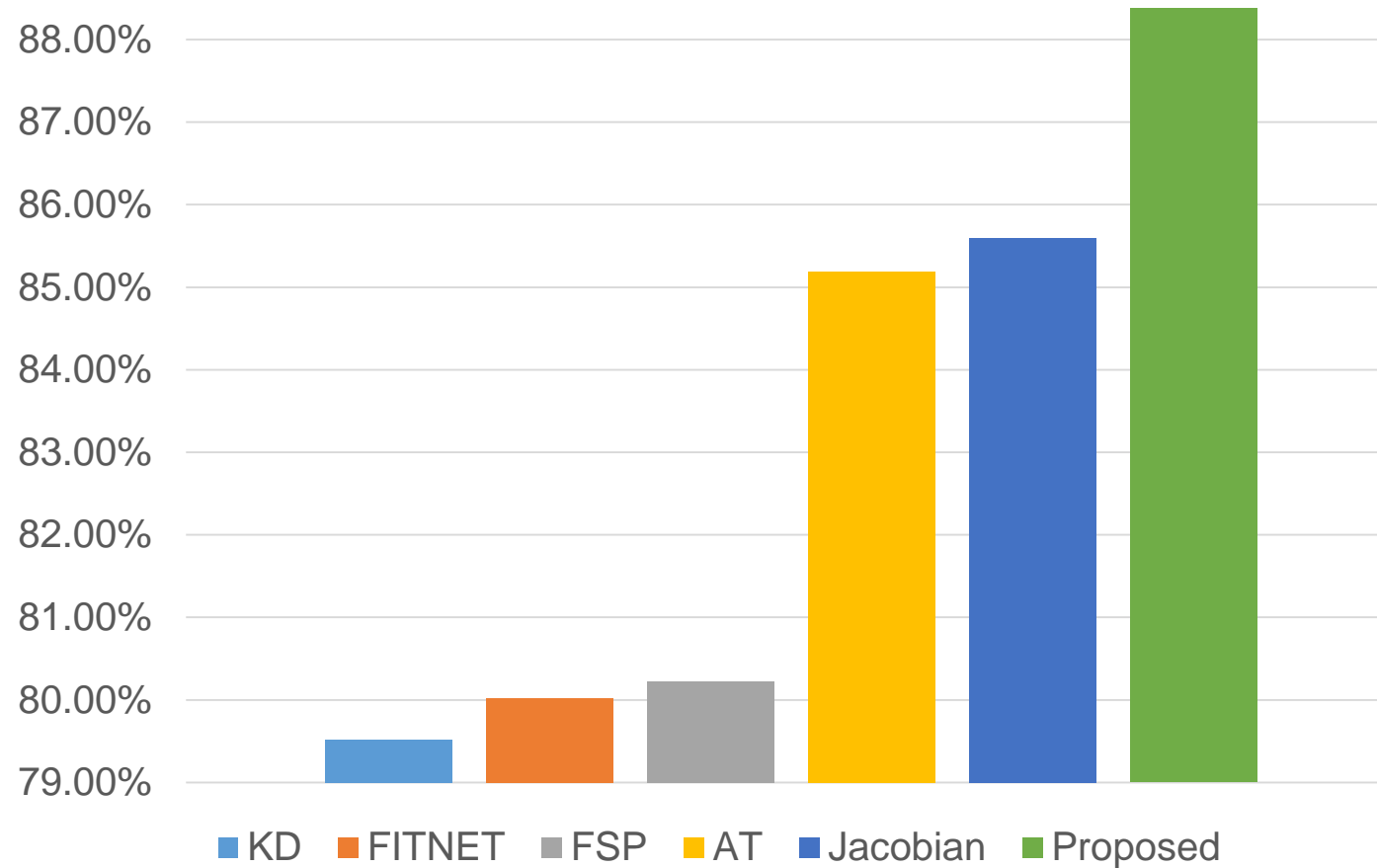
# Experiments

- Depth compression (depth 22 => 10)
  - Compression ratio 27.9%



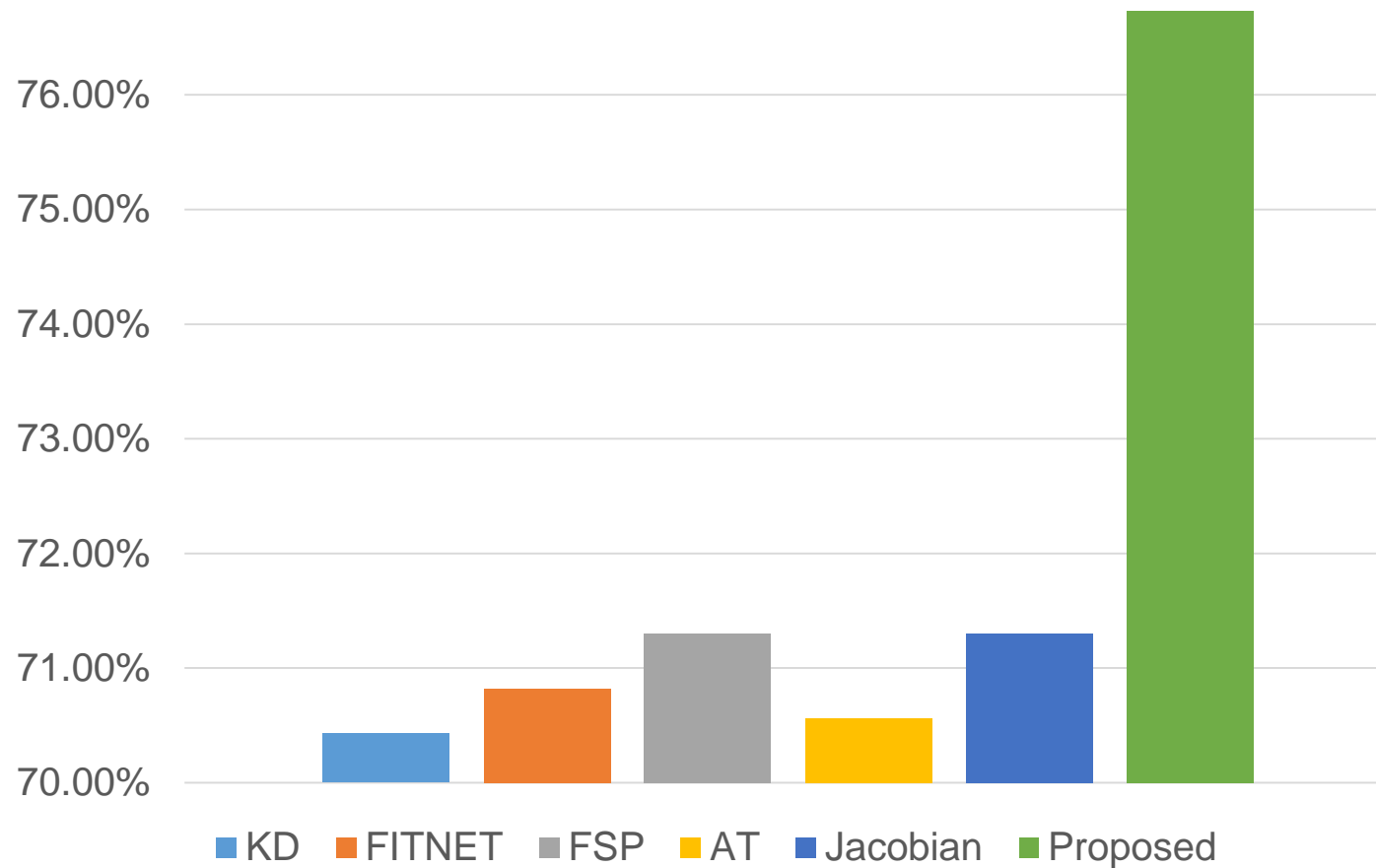
# Experiments

- Channel compression (channel 50%)
  - Compression ratio 25.2%



# Experiments

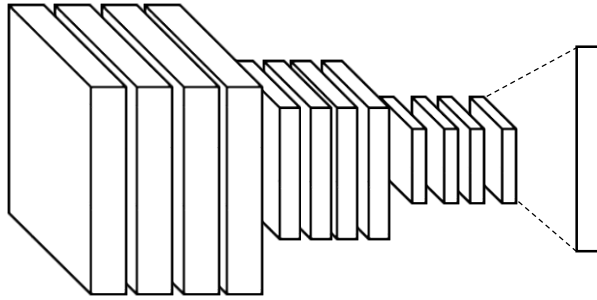
- Tiny network (depth 22 => 10 / channel 25%)
  - Compression ratio 1.8%



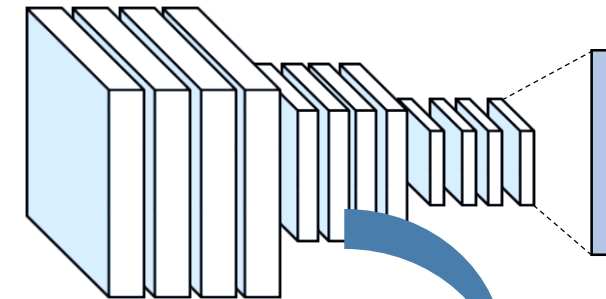
# Experiments

- **Transfer learning (Different Datasets)**
  - Training a new network without pre-training
  - Knowledge transfer can make similar effect of pre-training
  - Teacher (ResNet 50), Student (MobileNet)

1. Teacher network is trained on **ImageNet**

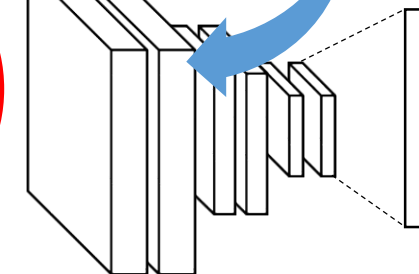


2. Student network is trained on **target dataset** with distillation from teacher



Target dataset

**Knowledge Transfer**



# Experiments

- MIT scenes dataset (Indoor scenes classification)





# Experiments

- MIT scenes dataset (Indoor scenes classification)
  - ImageNet pre-trained student(same as teacher) has good performance
  - Randomly initialized student has poor performance
  - Knowledge transfer can improve performance without pre-training
  - Proposed method has best performance in knowledge transfer
  - Proposed method is even better than pre-trained student

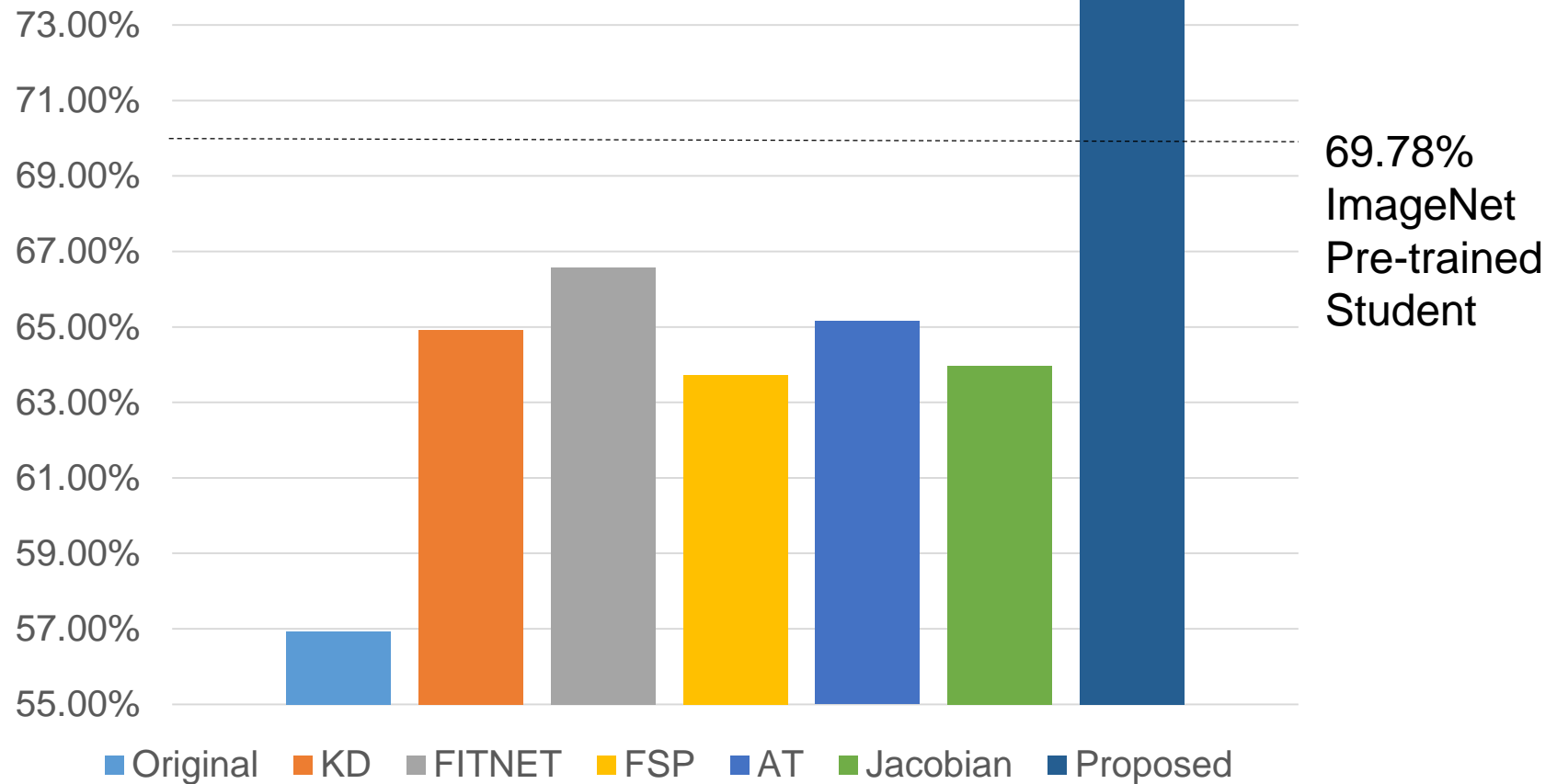
Number of training data	5	10	20	40	80
ImageNet pre-trained network	32.39%	42.46%	52.99%	62.54%	69.78%
Randomly initialized network	11.12%	17.99%	30.37%	43.88%	56.94%
Output distillation (DTL)	20.52%	34.55%	49.78%	59.48%	64.93%
FITNET + DTL	29.55%	43.81%	53.51%	63.36%	66.57%
FSP + DTL	25.60%	39.63%	51.42%	60.08%	63.73%
AT + DTL	22.61%	35.37%	49.10%	60.15%	65.15%
Jacobian + DTL	26.64%	38.28%	51.34%	61.19%	63.96%
Proposed + DTL	<b>43.36%</b>	<b>56.72%</b>	<b>66.34%</b>	<b>70.75%</b>	<b>74.10%</b>

Accuracy (%)



# Experiments

- MIT scenes dataset (Indoor scenes classification)



# Experiments

- CUB 2011 dataset (Bird classification)



# Experiments

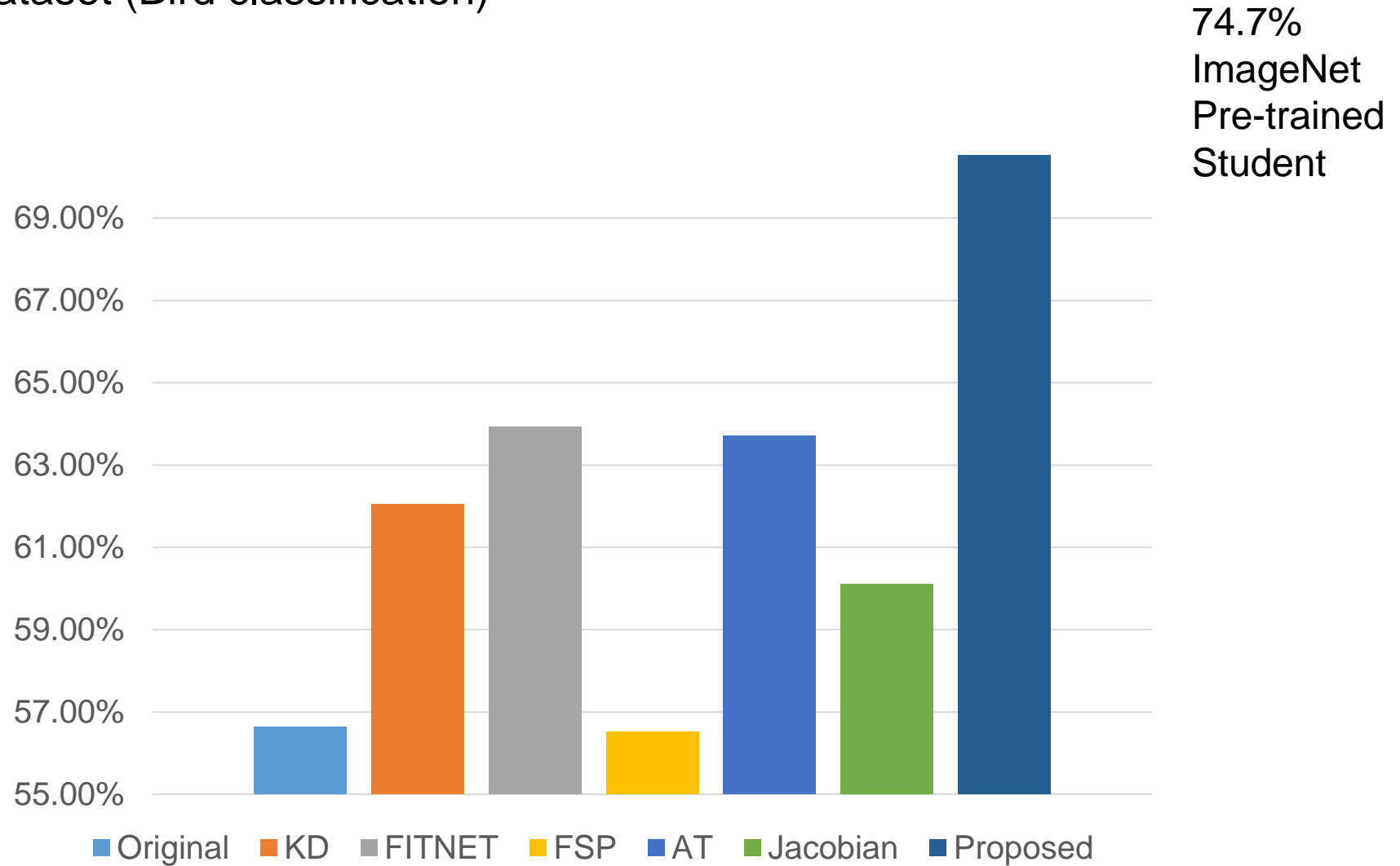
- CUB 2011 dataset (Bird classification)
  - Proposed method has the best performance in knowledge transfer
  - Not always better than pre-trained network
  - **Better than pre-trained network** in case of small training data

Number of training data	1	5	10	20	30
ImageNet pre-trained network	8.01%	29.84%	48.83%	69.02%	74.70%
Randomly initialized network	2.40%	10.68%	25.54%	44.43%	56.63%
Output distillation (DTL)	5.35%	26.29%	42.15%	48.24%	62.05%
FITNET + DTL	10.70%	36.75%	50.74%	52.85%	63.93%
FSP + DTL	7.06%	29.43%	42.44%	43.96%	56.52%
AT + DTL	6.65%	28.56%	44.43%	52.80%	63.70%
Jacobian + DTL	7.85%	31.02%	45.39%	49.41%	60.11%
Proposed + DTL	<b>13.38%</b>	<b>45.39%</b>	<b>57.11%</b>	<b>62.93%</b>	<b>70.54%</b>

Accuracy (%)

# Experiments

- CUB 2011 dataset (Bird classification)



# Interim Summary

- Activation boundaries are important in neural networks
- But, existing methods does not transfer activation boundary
- We proposed knowledge **transfer for activation boundaries**
  
- Proposed method accurately transfers activation boundary
- Higher performance than the state-of-the-art
  
- Our method **significantly outperforms state-of-the-art feature distillation**

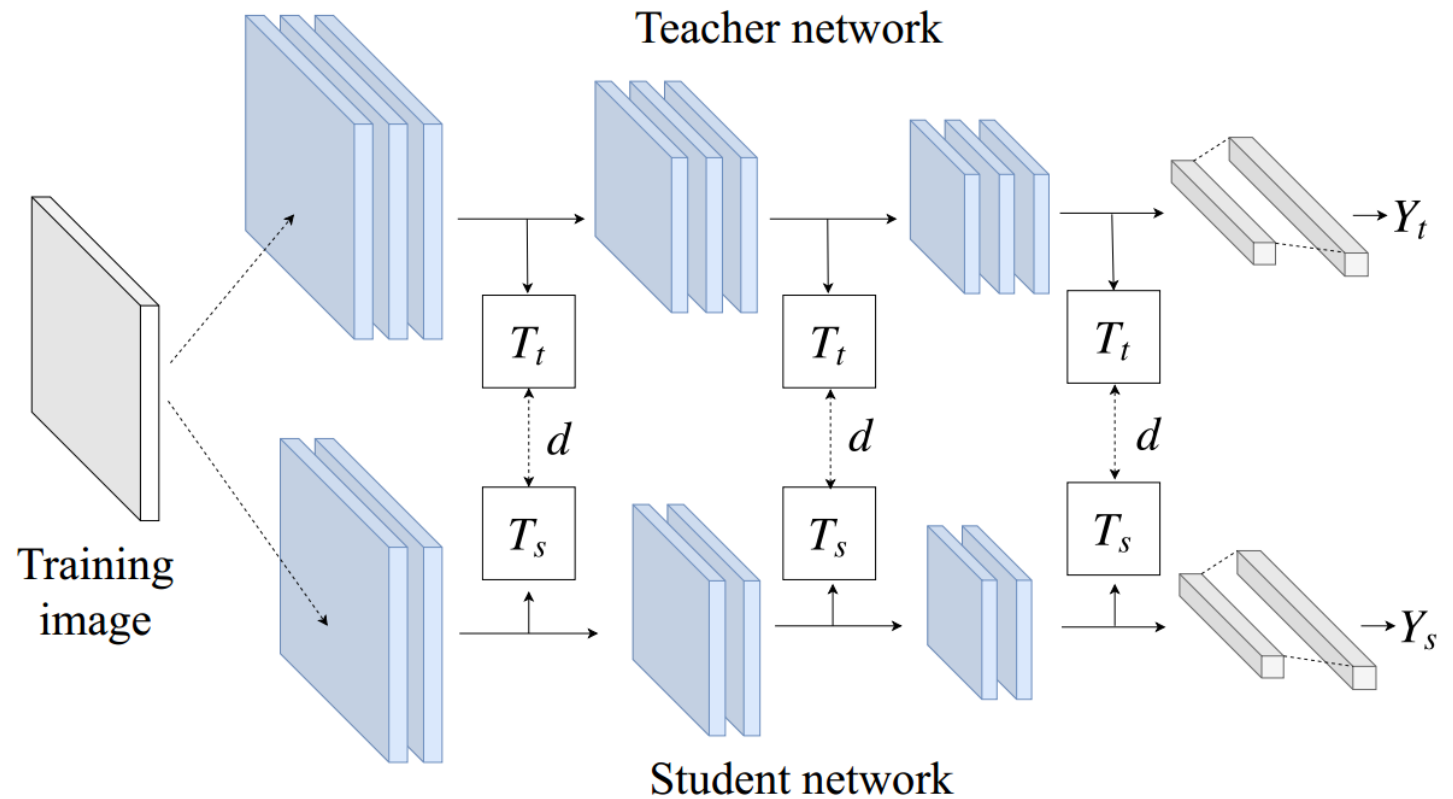
# Design of feature distillation

- Design aspects of feature distillation
  - Teacher transform
  - Student transform
  - Distance function
  - Feature position

[ICCV 2019] A Comprehensive Overhaul of Feature Distillation

Generalized loss function of feature distillation

$$L_{distill} = d(T_t(\mathbf{F}_t), T_s(\mathbf{F}_s))$$



# Design of feature distillation

- **Design aspects** of feature distillation
  - We analyzed six design aspects of feature distillation methods
  - And design a new feature distillation

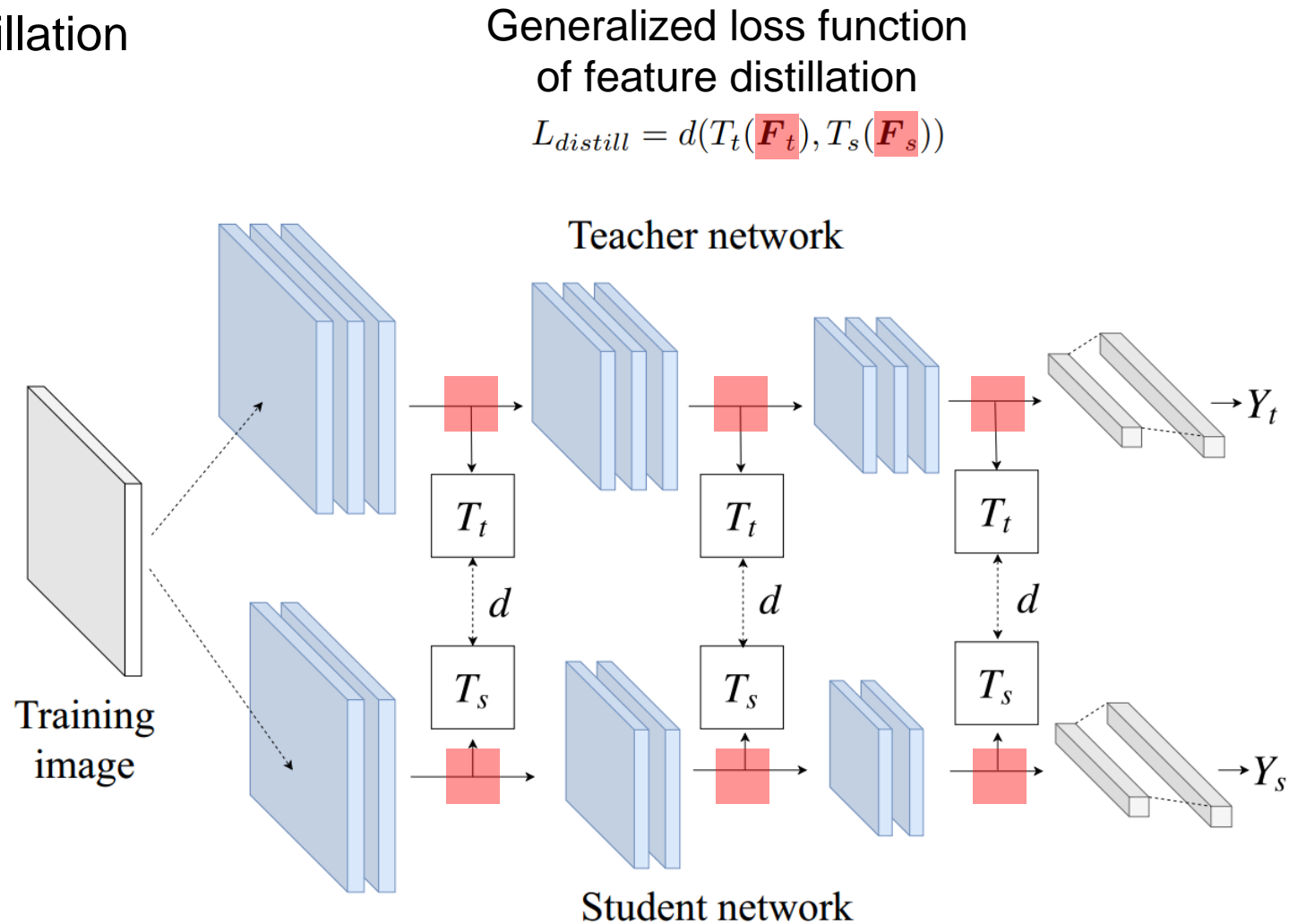
Method	Teacher transform	Student transform	Distillation feature position	Distance	Missing information
FitNets [22]	None	$1 \times 1$ conv	Mid layer	$L_2$	None
AT [30]	Attention	Attention	End of group	$L_2$	Channel dims
FSP [28]	Correlation	Correlation	End of group	$L_2$	Spatial dims
Jacobian [26]	Gradient	Gradient	End of group	$L_2$	Channel dims
FT [14]	Auto-encoder	Auto-encoder	End of group	$L_1$	Auto-encoded
AB [7]	Binarization	$1 \times 1$ conv	Pre-ReLU	Marginal $L_2$	Feature values
Proposed	Margin ReLU	$1 \times 1$ conv	Pre-ReLU	Partial $L_2$	Negative features

[ICCV 2019] A Comprehensive Overhaul of Feature Distillation

# Proposed method: Feature position

- Design aspects of feature distillation
  - Teacher transform
  - Student transform
  - Distance function
  - **Feature position**

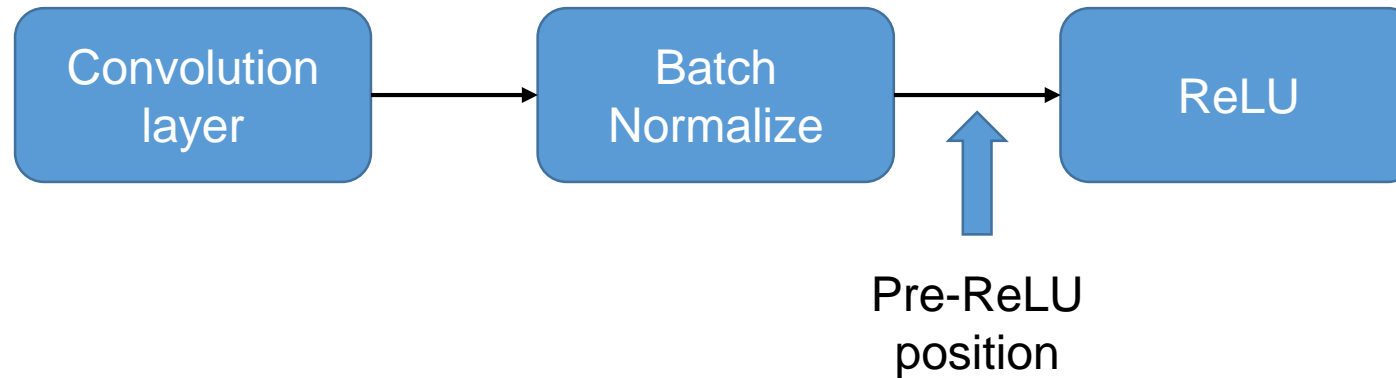
[ICCV 2019] A Comprehensive Overhaul of Feature Distillation



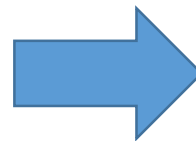
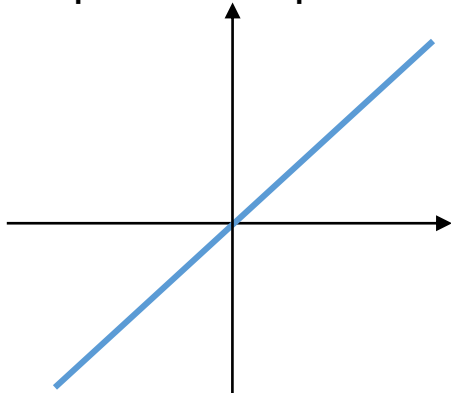


# Proposed method

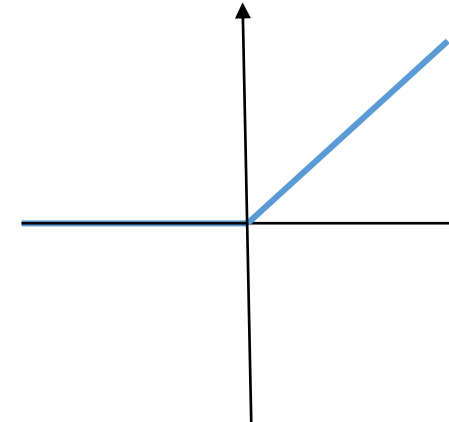
- Distillated Feature Position



Feature values  
at pre-ReLU position

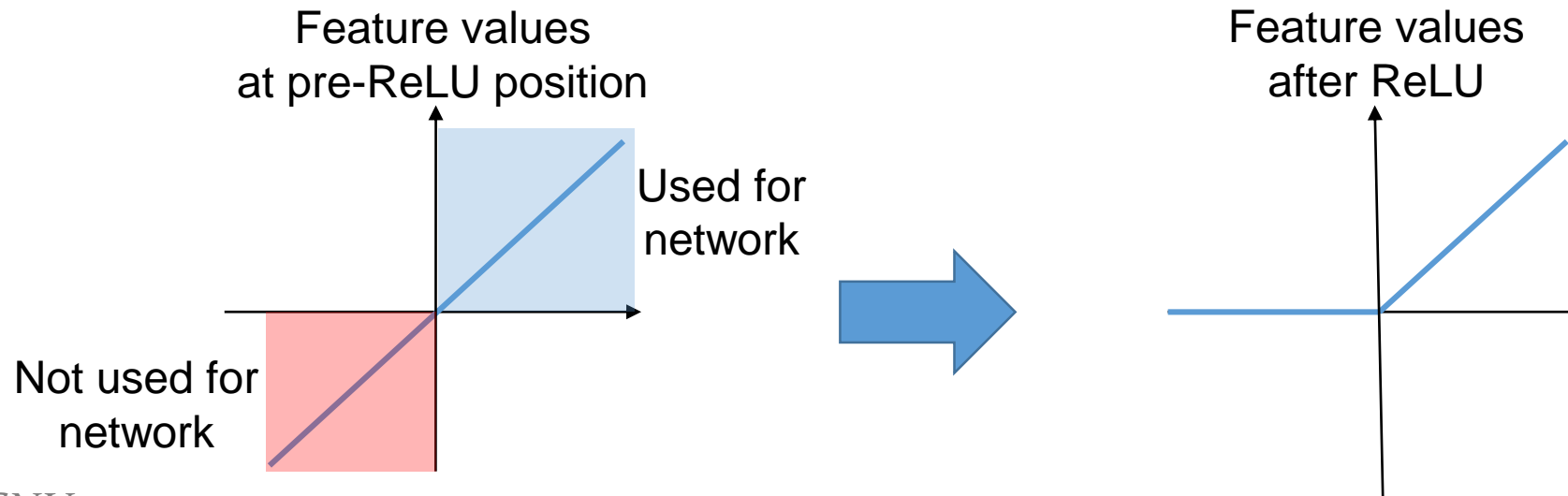
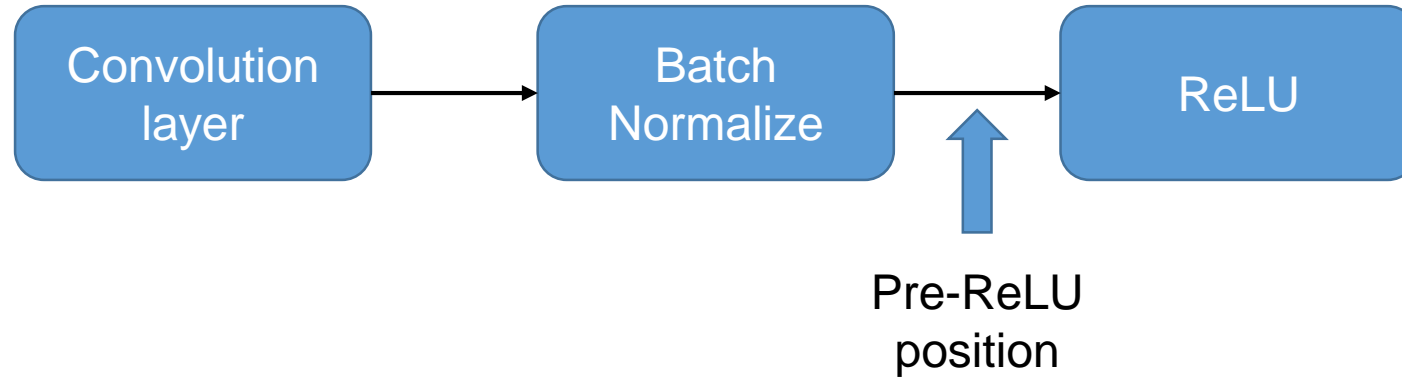


Feature values  
after ReLU



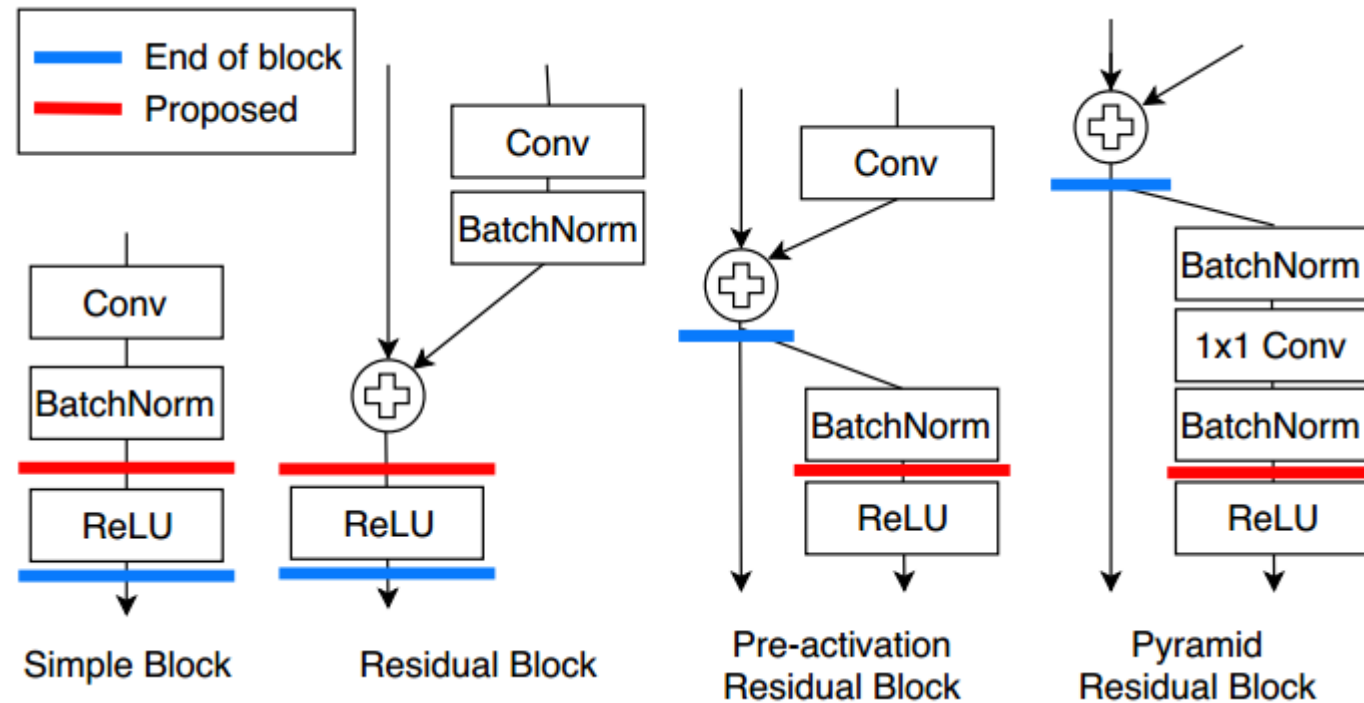
# Proposed method

- Distillated Feature Position



# Proposed method

- Distillated Feature Position
  - Most of distillation uses end of block position
  - We change distillation position to pre-ReLU position



# Proposed method: Teacher Transform

- Design aspects of feature distillation

## 1. Teacher transform

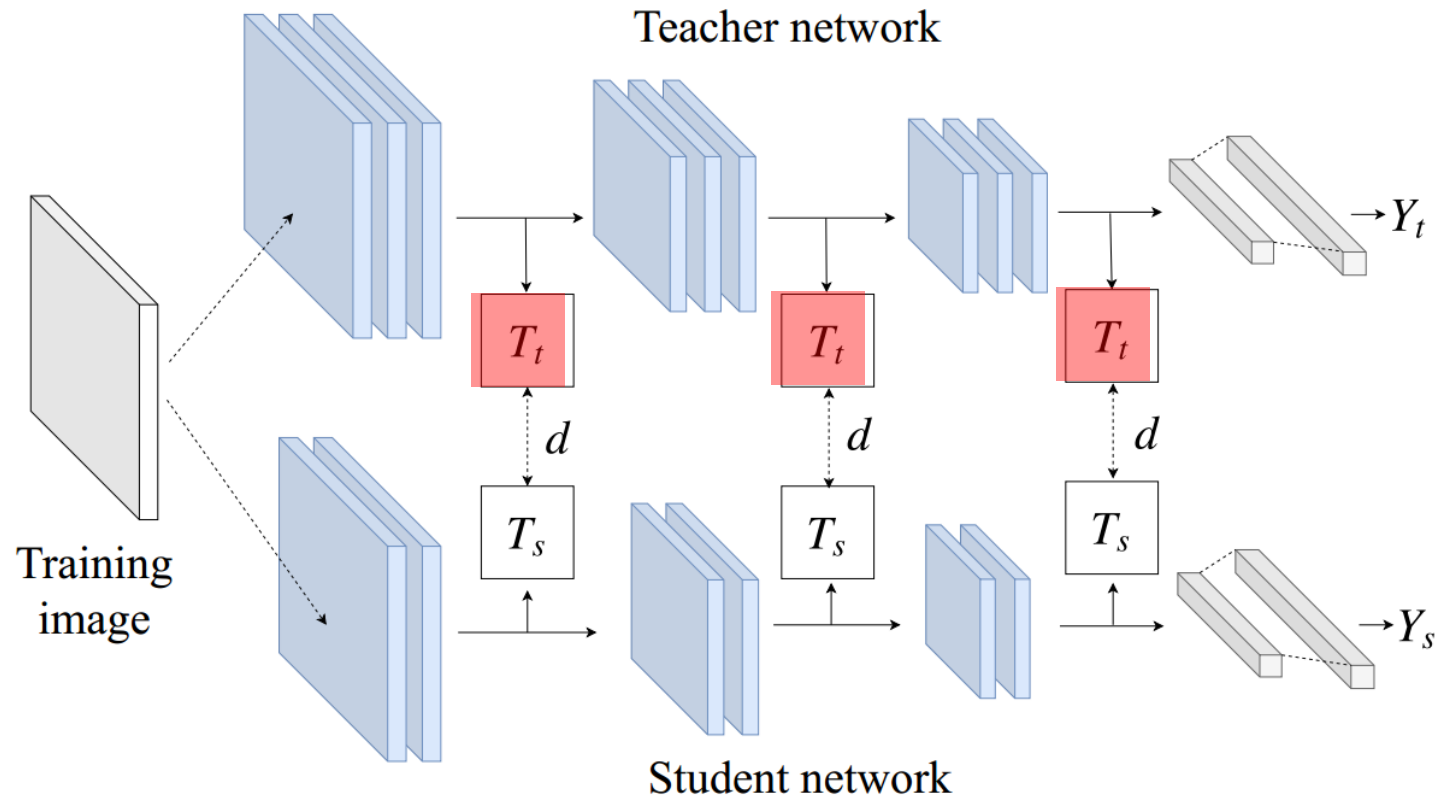
## 2. Student transform

## 3. Distance function

## 4. Feature position

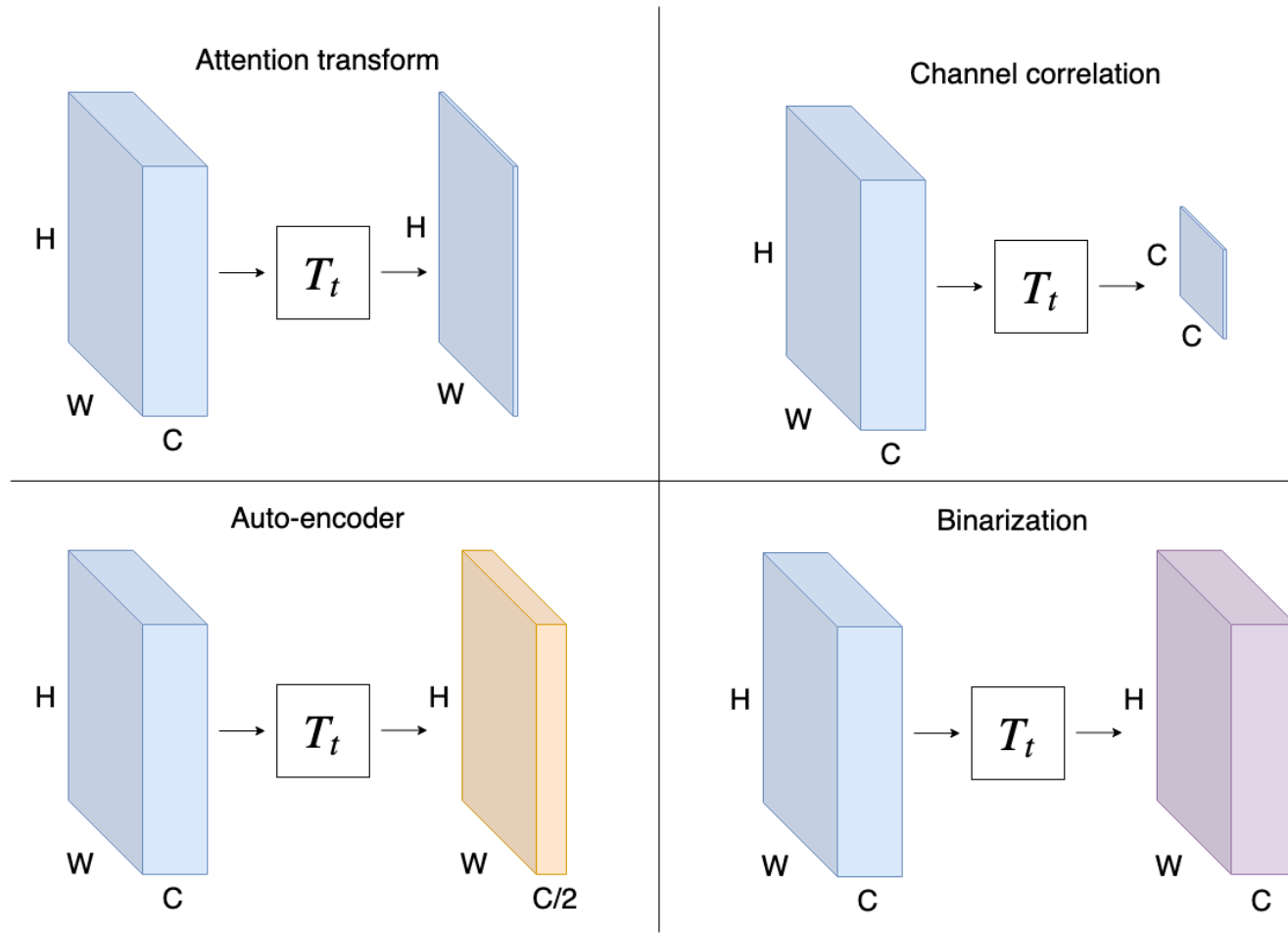
Generalized loss function  
of feature distillation

$$L_{distill} = d(T_t(F_t), T_s(F_s))$$



# Proposed method

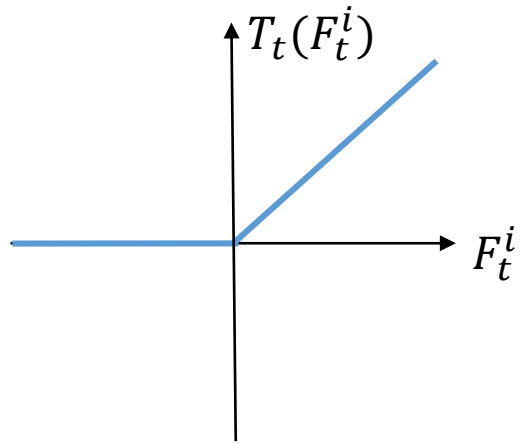
- Teacher transform
  - Feature representation is changed
  - Some information may be missed
  - Goal is to use the teacher transform with the least information missing



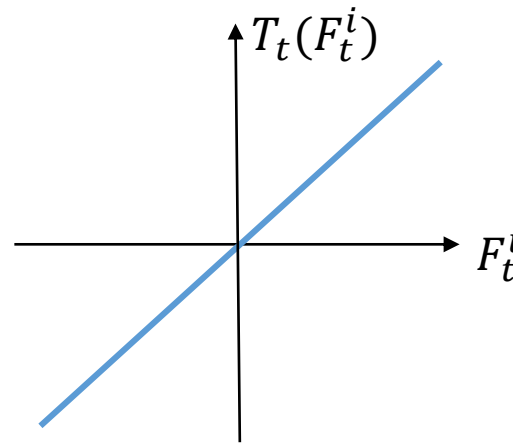
# Proposed method

- Margin ReLU
  - Teacher transform of our method
  - Preserving feature values of teacher
  - But, suppress large negative values
  - Margin value : a small value of negative responses

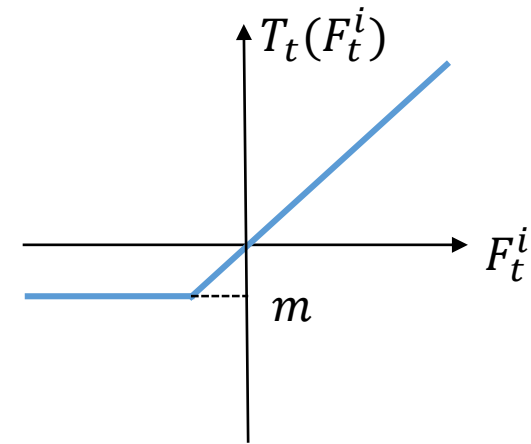
$$m_C = E[F_t^i | F_t^i < 0, i \in C].$$



ReLU



Identity

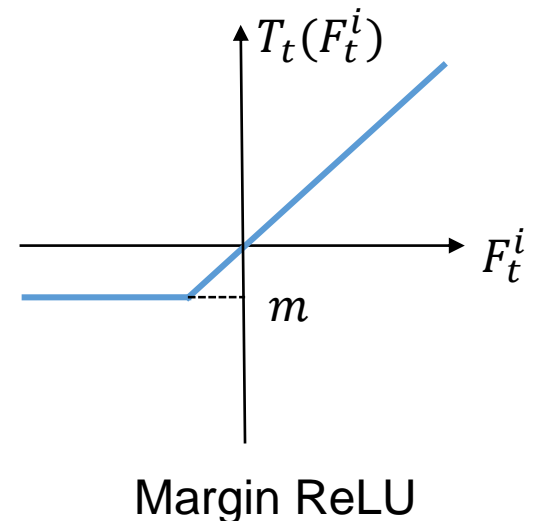
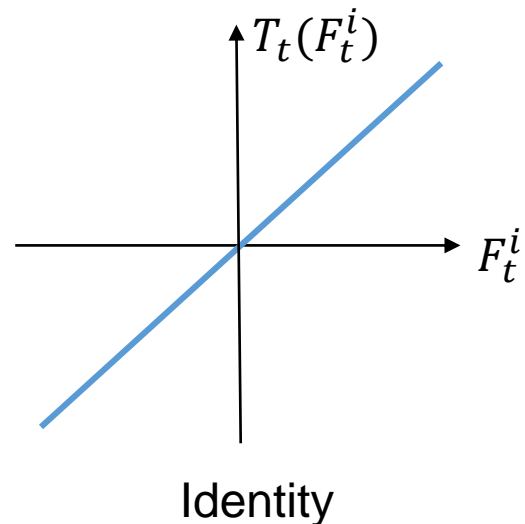
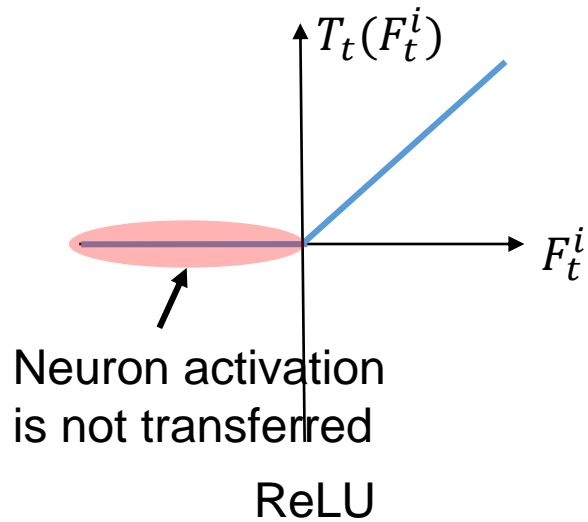


Margin ReLU

# Proposed method

- Margin ReLU
  - Teacher transform of our method
  - Preserving feature values of teacher
  - But, suppress large negative values
  - Margin value : expectation value of negative responses

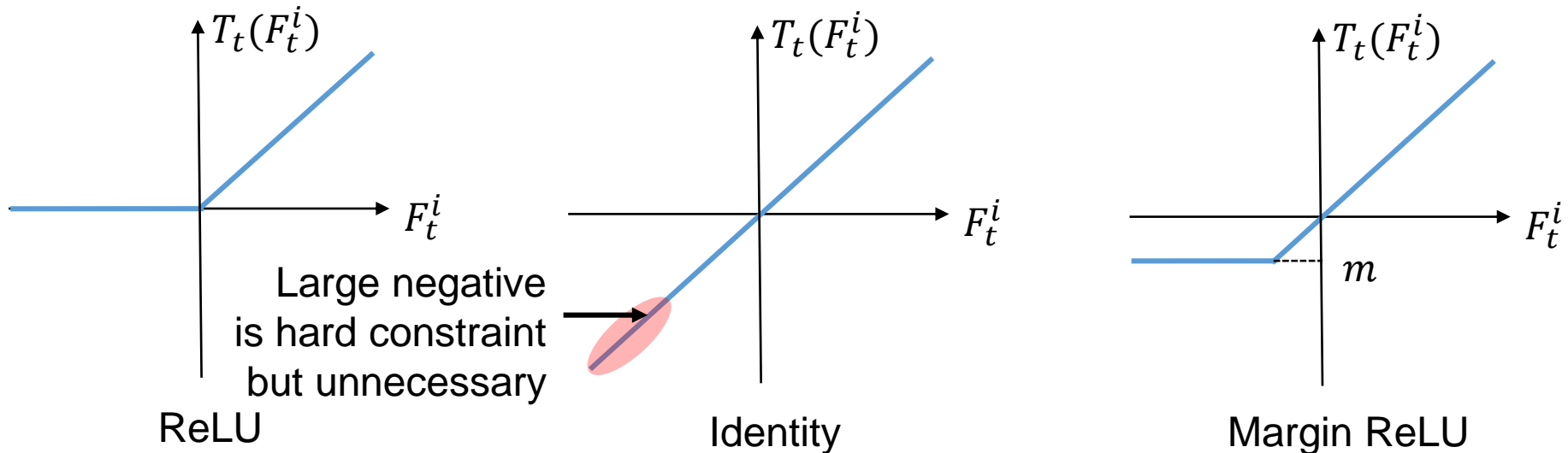
$$m_C = E[F_t^i | F_t^i < 0, i \in C].$$



# Proposed method

- Margin ReLU
  - Teacher transform of our method
  - Preserving feature values of teacher
  - But, suppress large negative values
  - Margin value : expectation value of negative responses

$$m_C = E[F_t^i | F_t^i < 0, i \in C].$$

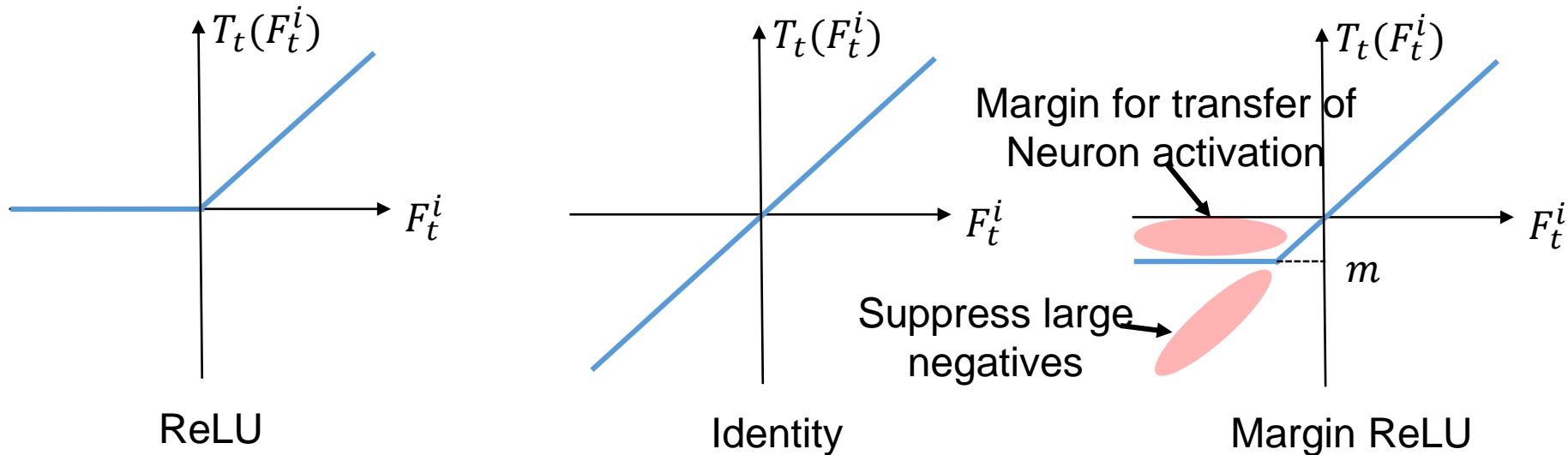




# Proposed method

- Margin ReLU
  - Teacher transform of our method
  - Preserving feature values of teacher
  - But, suppress large negative values
  - Margin value : expectation value of negative responses

$$m_C = E[F_t^i | F_t^i < 0, i \in C].$$

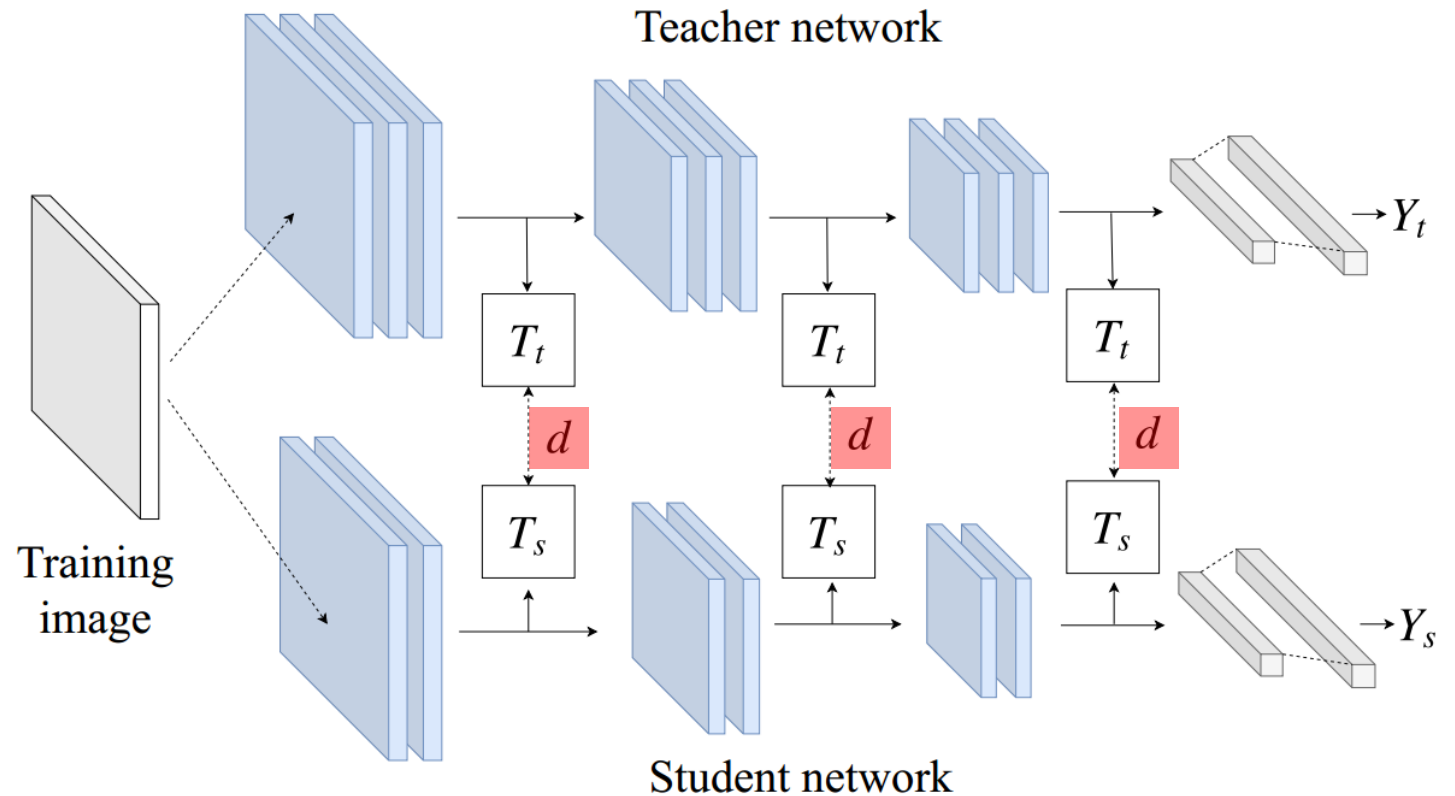


# Proposed method: Distance Function

- Design aspects of feature distillation
  - Teacher transform
  - Student transform
  - **Distance function**
  - Feature position

Generalized loss function  
of feature distillation

$$L_{distill} = d(T_t(F_t), T_s(F_s))$$

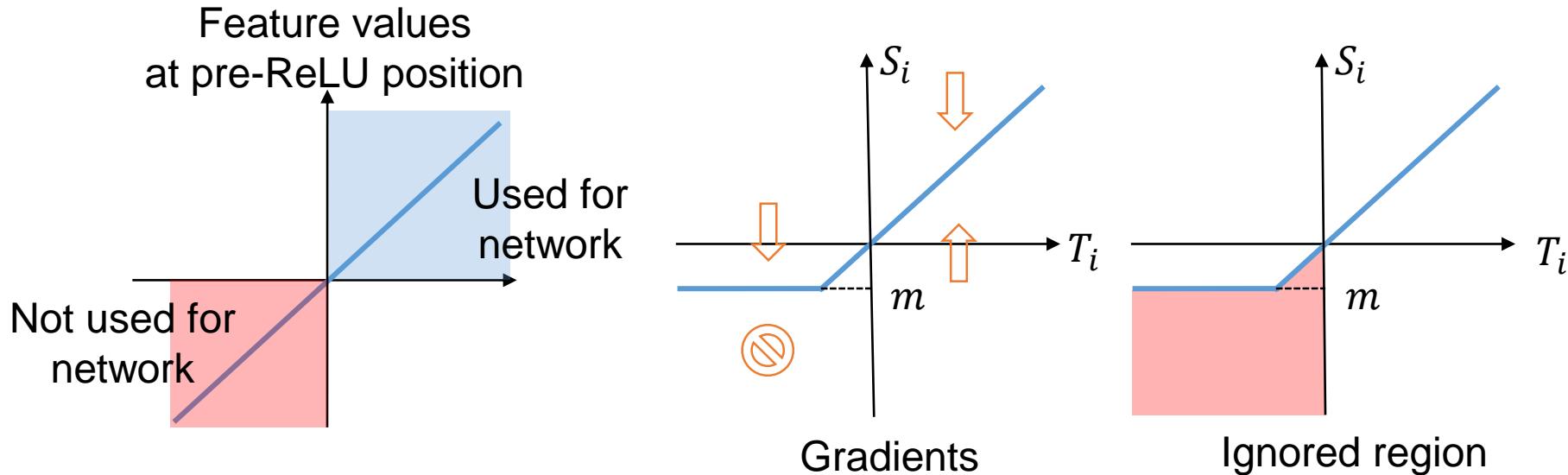


# Proposed method

- Partial  $L_2$  distance
  - Distance function of our method

$$d_p(\mathbf{T}, \mathbf{S}) = \sum_i^{WHC} \begin{cases} 0 & \text{if } S_i \leq T_i \leq 0 \\ (T_i - S_i)^2 & \text{otherwise.} \end{cases}$$

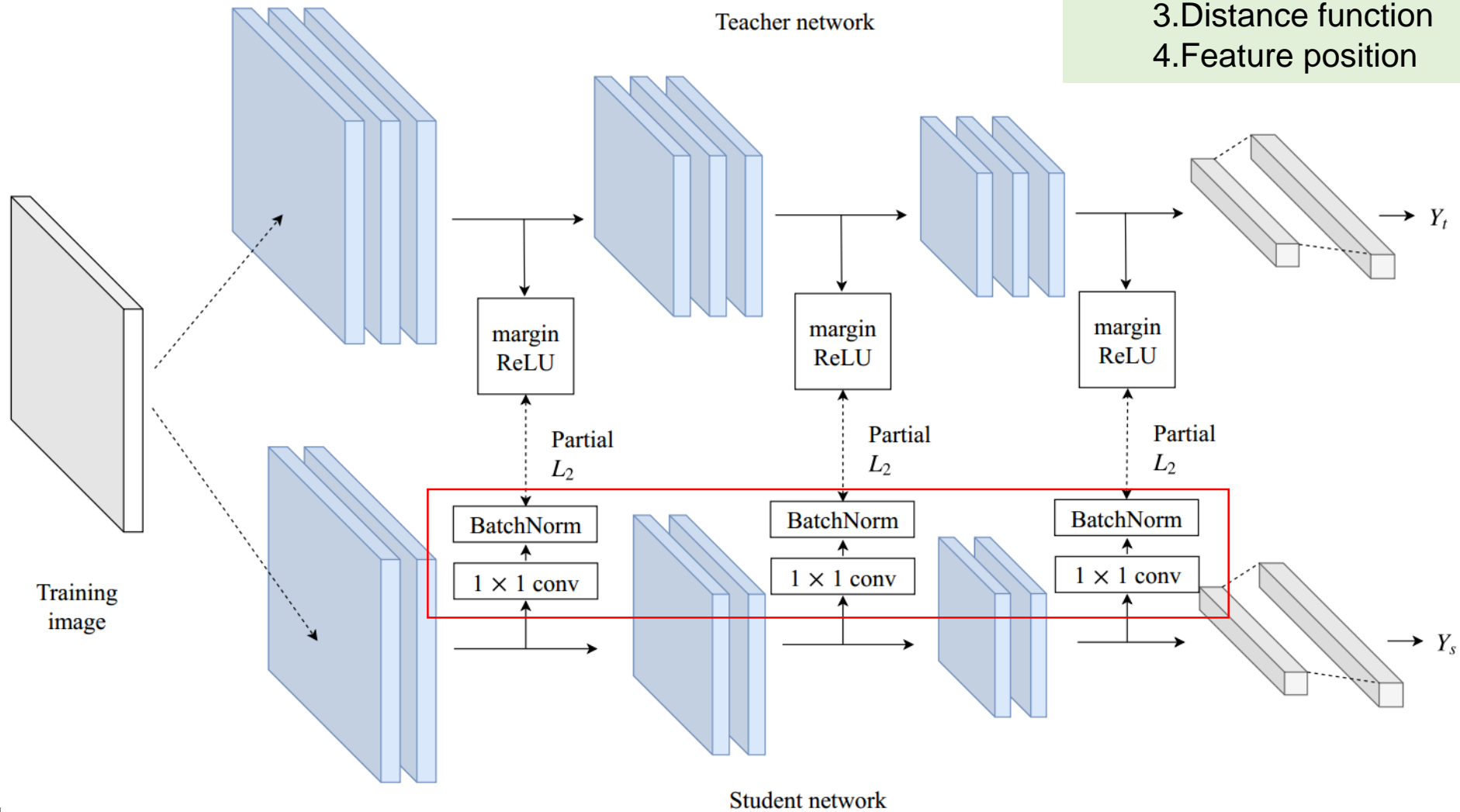
- Positive feature should be same as teacher
- Negative feature should be smaller than margin ReLU
- Margin ReLU + Partial  $L_2$  distance



# Proposed method

- Student transform

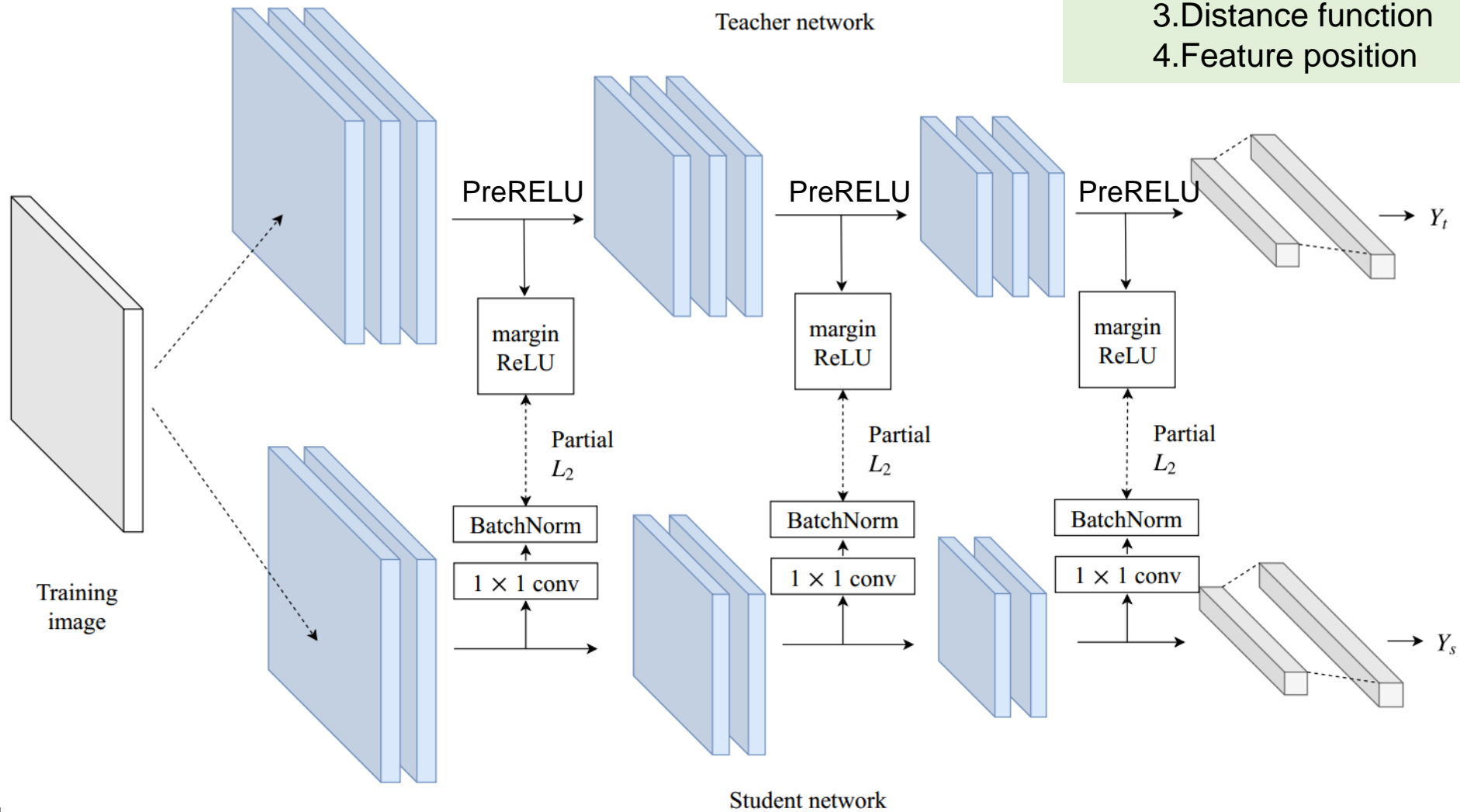
1. Teacher transform
2. Student transform
3. Distance function
4. Feature position



# Proposed method

- Overall Scheme

1. Teacher transform
2. Student transform
3. Distance function
4. Feature position



# Experiments

- **Various Compressions** on CIFAR-100
  - Our method was evaluated on various setup
  - Significantly outperforms state-of-the-art feature distillation

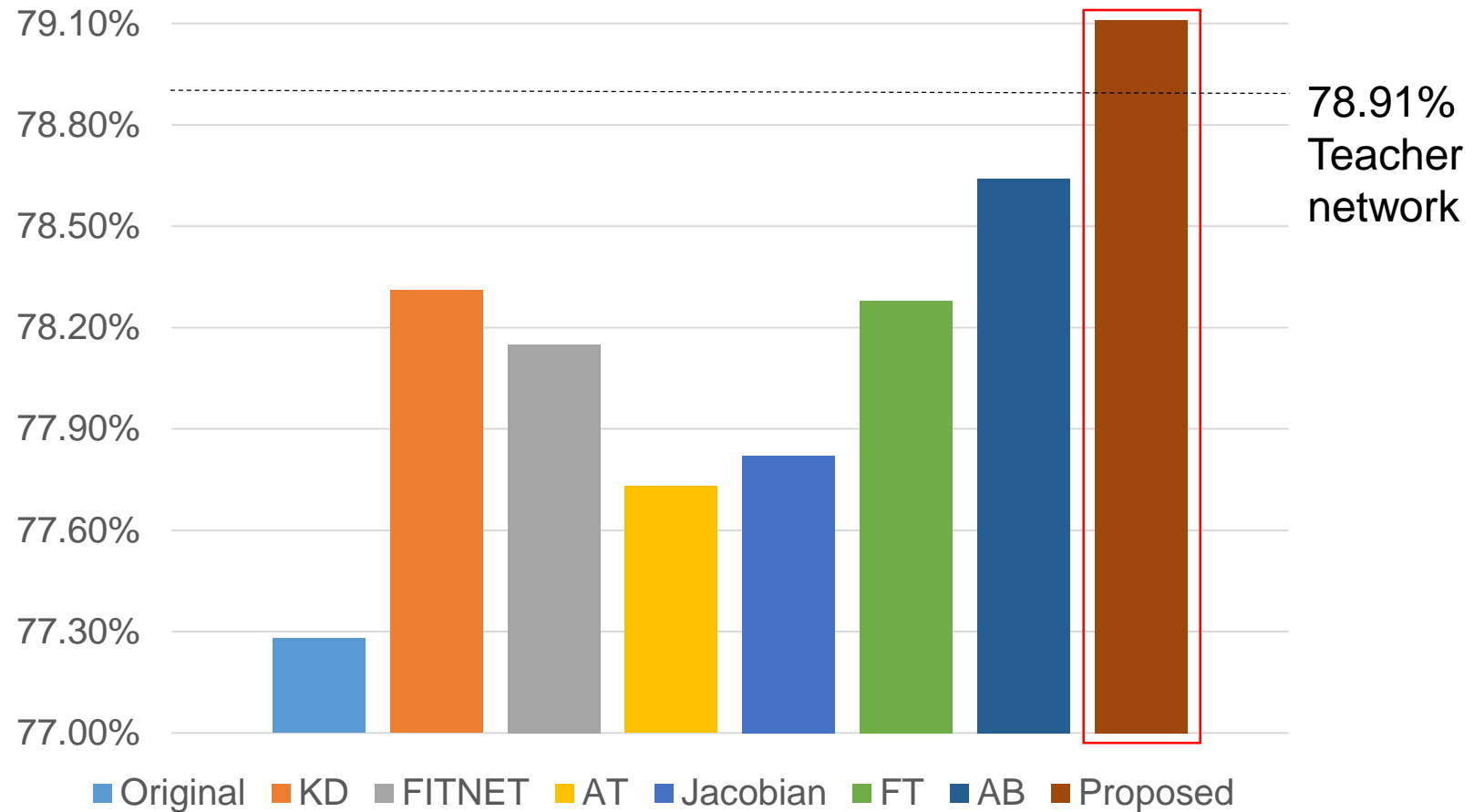
Setup	Compression type	Teacher network	Student network	# of params teacher	# of params student	Compress ratio
(a)	Depth	WideResNet 28-4	WideResNet 16-4	5.87M	2.77M	47.2%
(b)	Channel	WideResNet 28-4	WideResNet 28-2	5.87M	1.47M	25.0%
(c)	Depth & channel	WideResNet 28-4	WideResNet 16-2	5.87M	0.70M	11.9%
(d)	Different architecture	WideResNet 28-4	ResNet 56	5.87M	0.86M	14.7%
(e)	Different architecture	PyramidNet-200 (240)	WideResNet 28-4	26.84M	5.87M	21.9%
(f)	Different architecture	PyramidNet-200 (240)	PyramidNet-110 (84)	26.84M	3.91M	14.6%

Setup	Teacher	Baseline	KD [8]	FitNets [22]	AT [30]	Jacobian [26]	FT [14]	AB [7]	Proposed
(a)	21.09	22.72	21.69	21.85	22.07	22.18	21.72	21.36	<b>20.89</b>
(b)	21.09	24.88	23.43	23.94	23.80	23.70	23.41	23.19	<b>21.98</b>
(c)	21.09	27.32	26.47	26.30	26.56	26.71	25.91	26.02	<b>24.08</b>
(d)	21.09	27.68	26.76	26.35	26.66	26.60	26.20	26.04	<b>24.44</b>
(e)	15.57	21.09	20.97	22.16	19.28	20.59	19.04	20.46	<b>17.80</b>
(f)	15.57	22.58	21.68	23.79	19.93	23.49	19.53	20.89	<b>18.89</b>

ICLR 2017 ICML 2018 NIPS 2018 AAAI 2019

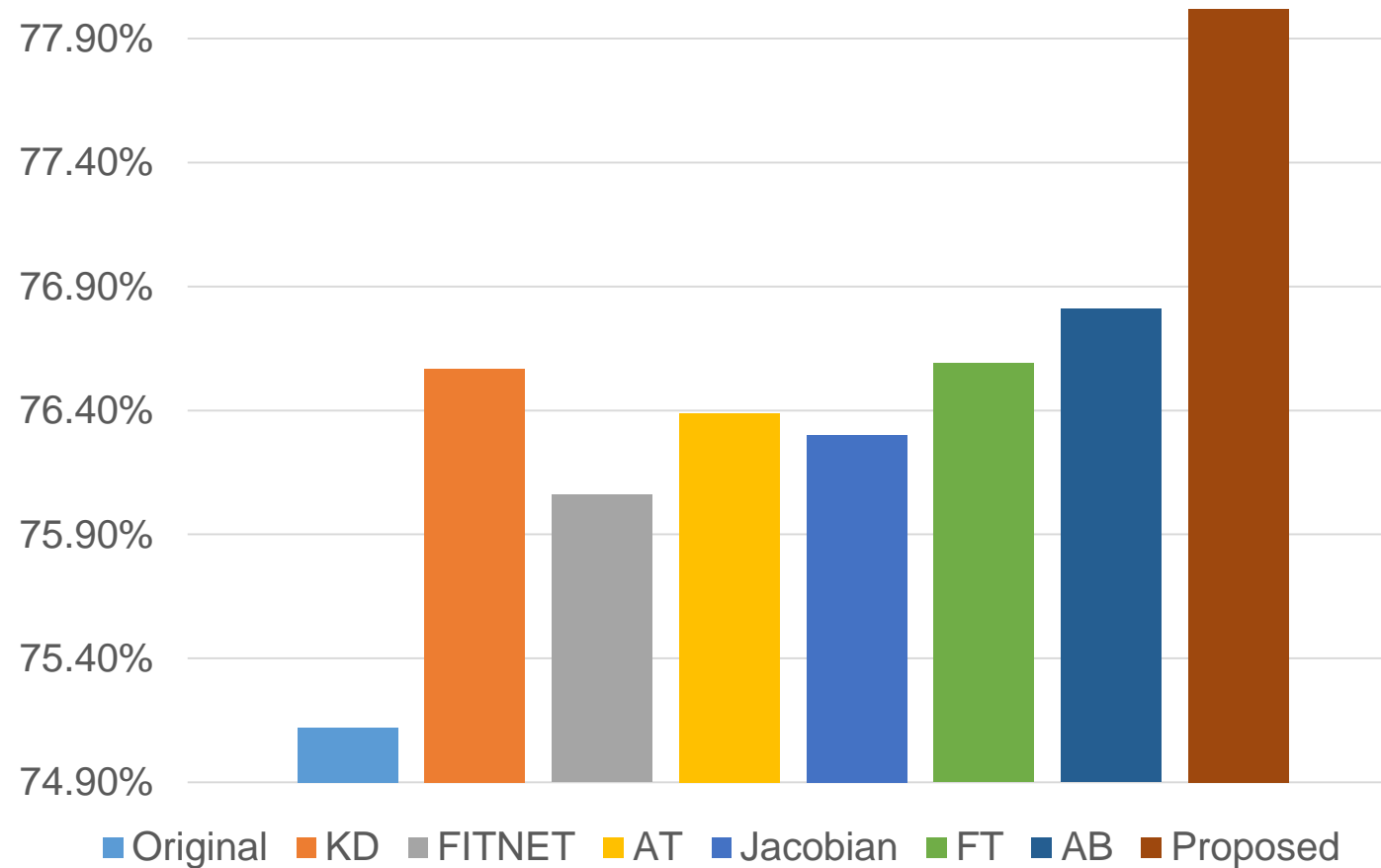
# Experiments

- Depth compression
  - Depth : 28 -> 16
  - Compression rate : 47.2%



# Experiments

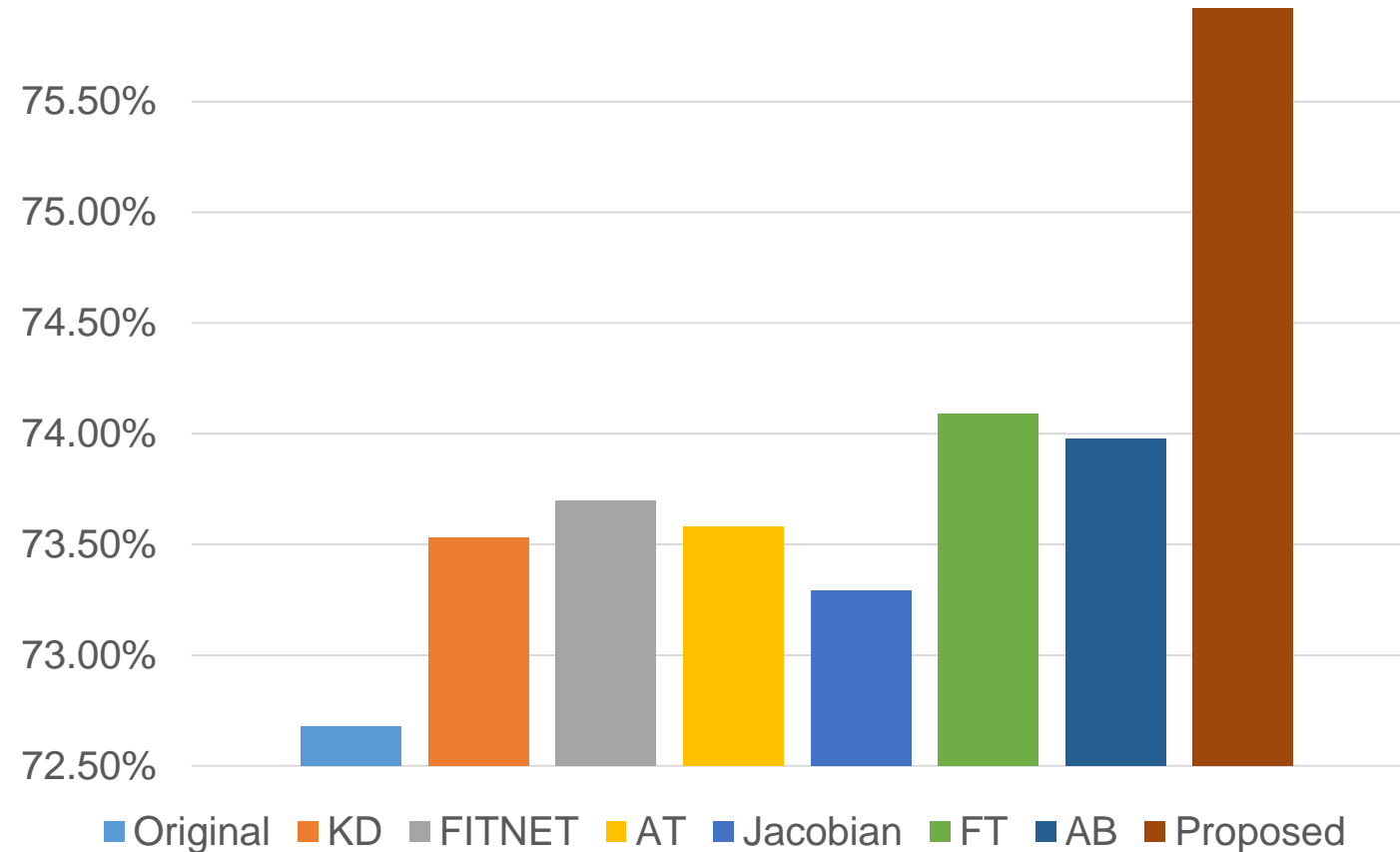
- Channel compression
  - Channel 50%
  - Compression rate : 25%





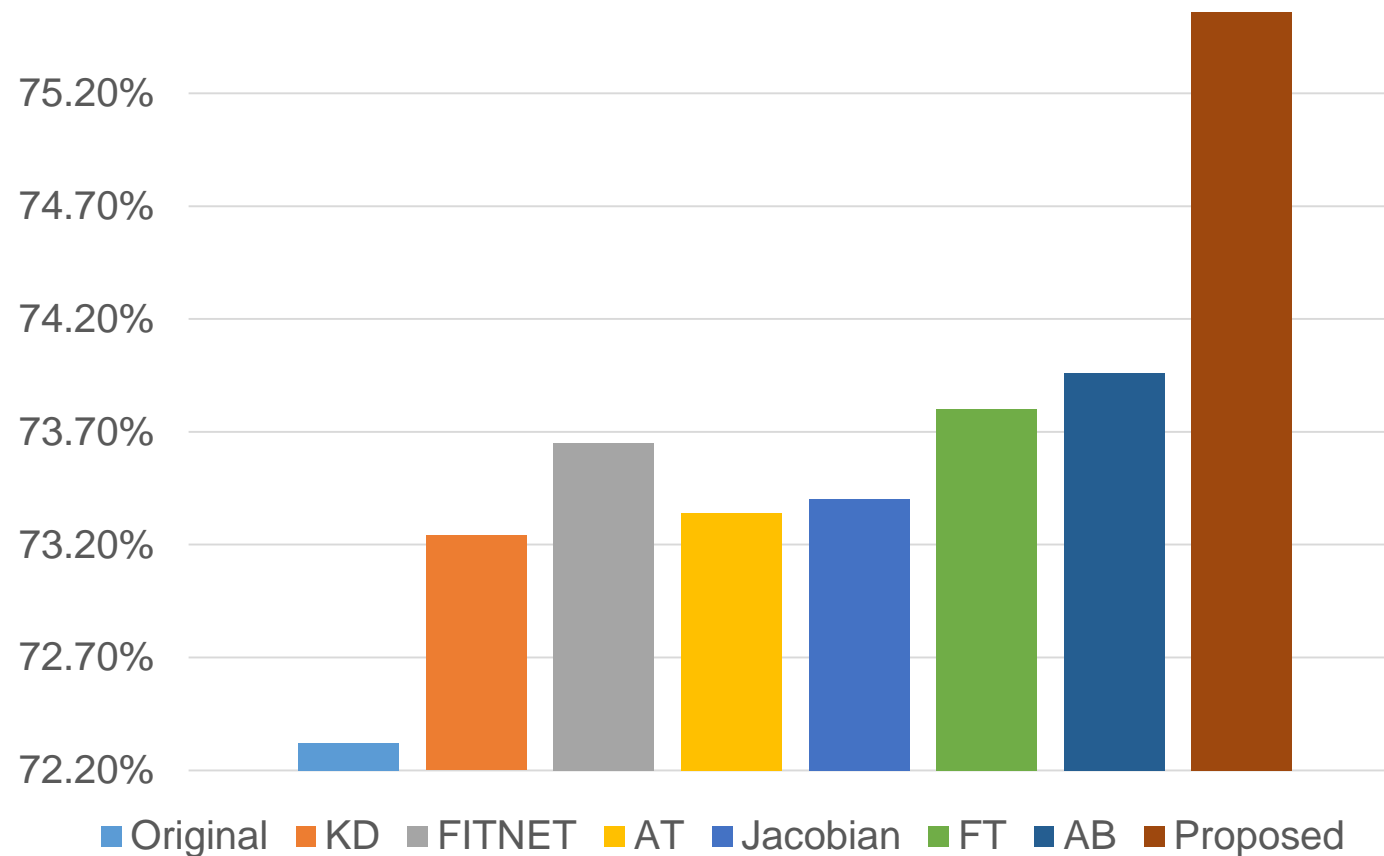
# Experiments

- Depth & channel compression
  - Depth : 28 -> 16 / Channel : 50%
  - Compression rate : 11.9%



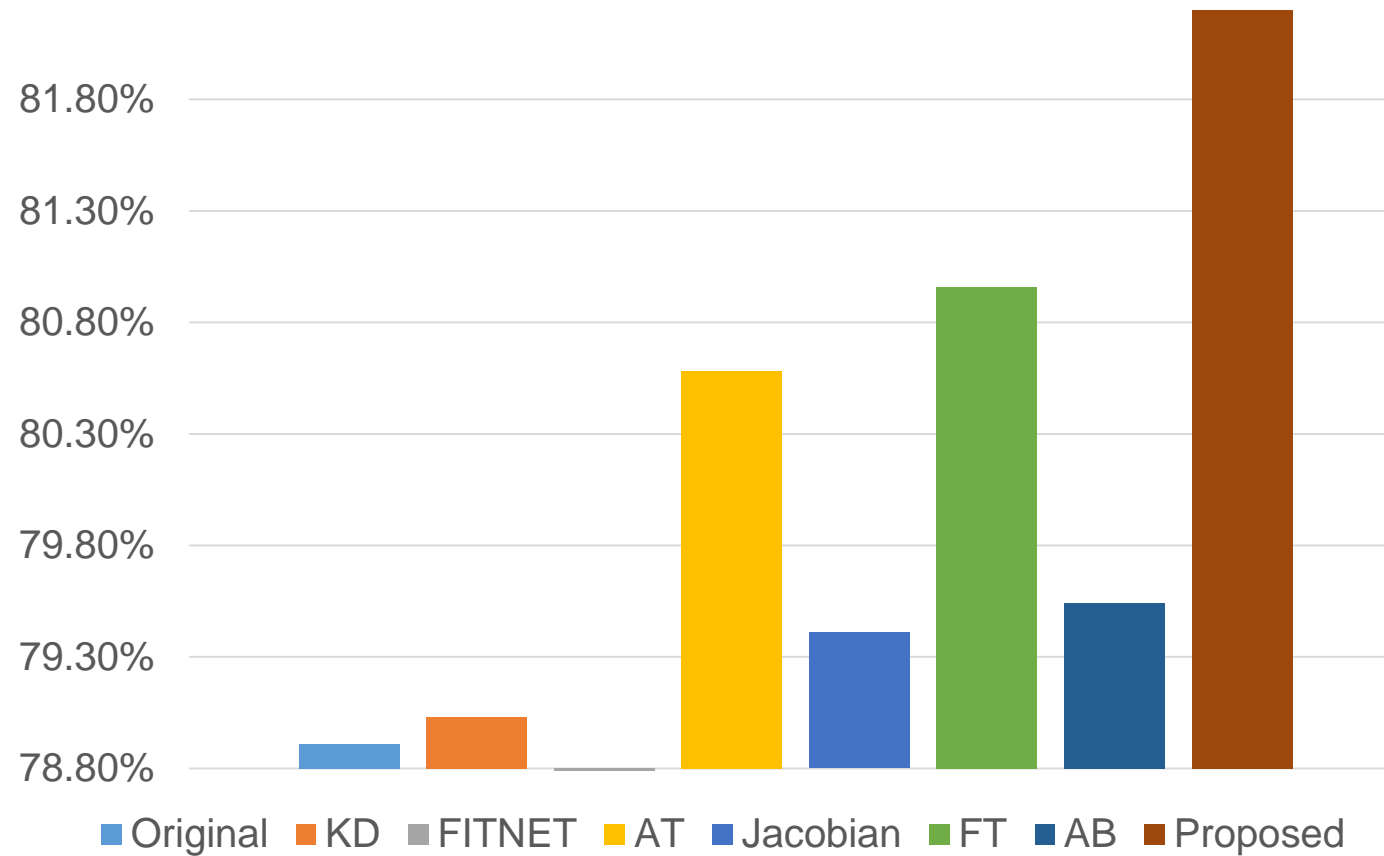
# Experiments

- WideResNet 28-4 => ResNet 56
  - Depth : 28 -> 56 / Channel : 25%
  - Compression rate : 14.7%



# Experiments

- PyramidNet => WideResNet
  - Compression rate : 21.9%



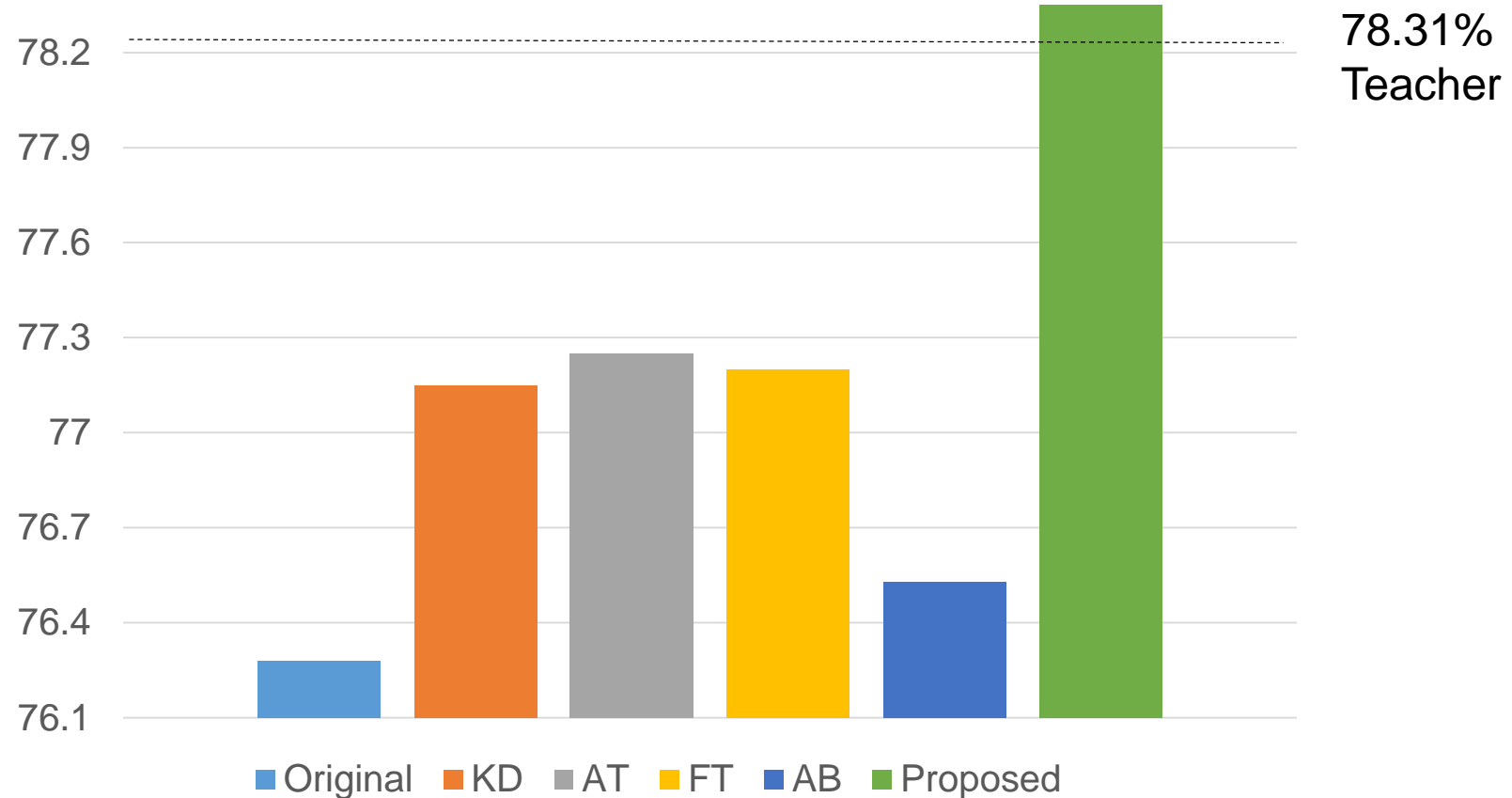
# Experiments

- ImageNet
  - 1000 classes / 1.2 million training images
  - 224 x 224 image size

Network	# of param (ratio)	Method	Top-1 error(%)	Top-5 error(%)
ResNet152	60.19M	Teacher	21.69	5.95
ResNet50	25.56M (42.5%)	Baseline	23.72	6.97
		AT [30]	22.75	6.35
		FT [14]	22.80	6.49
		AB [7]	23.47	6.94
		Proposed	<b>21.65</b>	<b>5.83</b>
ResNet50	25.56M	Teacher	23.84	7.14
MobileNet	4.23M (16.5%)	Baseline	31.13	11.24
		AT [30]	30.44	10.67
		FT [14]	30.12	10.50
		AB [7]	31.11	11.29
		Proposed	<b>28.75</b>	<b>9.66</b>

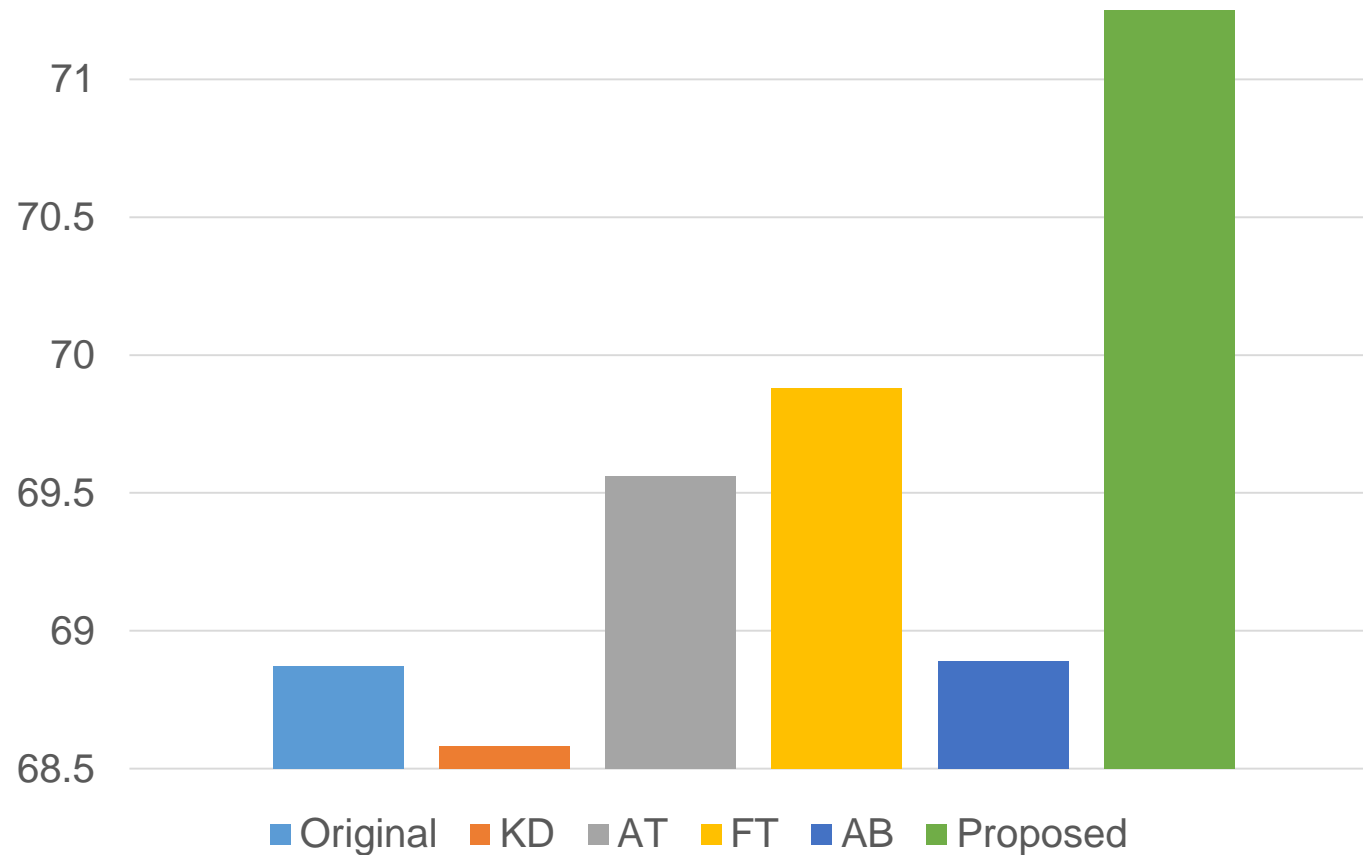
# Experiments

- ImageNet
  - ResNet 152 => ResNet 50
  - Compression rate : 42.5%



# Experiments

- ImageNet
  - ResNet 50 => MobileNet
  - Compression rate : 16.5%



# Lesson from the works

- The feature distillation consists of several design aspects, and should be designed based on an understanding of each design aspects.
  - 1. Teacher transform** is a function that transforms teacher's features to determine what information is transferred in the distillation.
  - 2. Student transform** is a function that transforms student's features to be similar to the transformed feature of the teacher.
  - 3. Distance function** measures a distance between two transformed features to used as a loss function.
  - 4. Distillation positions** are distillation locations of two networks.

# Lesson from the works

- In order to transfer the activation boundary, four design aspects need to be considered carefully, as a result, our feature distillation is designed comprehensively as
  - **Pre-ReLU**, our distillation position before the ReLU activation to **prevent missing information**.
  - **Margin ReLU**, our teacher transform, can **transfer the activation boundary** and response simultaneously by truncating a large negative response in the feature.
  - **Partial L2**, our distance function for the margin ReLU, **eliminates the adverse effect of Margin ReLU in a negative area** smaller than a margin.
  - **1×1 conv layer**, our student transform, requires the least computation to **match the channel size** of student with that of teacher.
- The proposed comprehensive design based on the four design aspects significantly improves the performance of knowledge distillation by accurate transfer of activation boundaries and responses of hidden neurons simultaneously.