

Refine Myself by Teaching Myself: Feature Refinement via Self-Knowledge Distillation

INDEX

1. 서론

- 1) CNN
- 2) 데이터 학습 - 개념

2. 본론

- 1) knowledge distillation과 self-knowledge distillation
- 2) FRSKD 개요 (self-teacher network, self-feature distillation)
- 3) 실험 과정 및 결과 분석

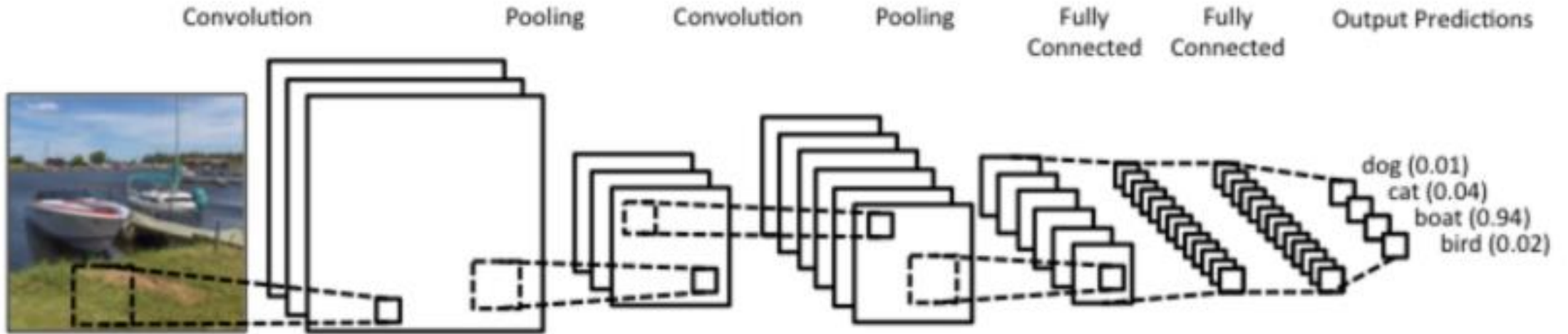
3. 결론

1. 서론

CNN(Convolution Neural Network, 합성곱 신경망)

- 시각적 영상을 분석하는데 사용되는 다층의 **feed-forward**적인 인공신경망의 한 종류
- 합성곱층 (**Convolution layer**)과 풀링층(**Pooling layer**)으로 구성
 - **Convolution layer** : 컨볼루션 연산을 통해 입력 이미지를 변환하는 역할
 - **Pooling layer** : 주위의 픽셀을 묶어서 하나의 대표 픽셀로 바꿈 (이미지 차원 축소)

CNN(Convolution Neural Network, 합성곱 신경망)



Convolution Layer

컨볼루션 계층

: 입력 이미지에서 고유한 특징을 부각시킨 이미지를 새로 만들어내는 역할 → 특징 맵(feature map)

How?

: 입력 이미지를 다른 이미지로 변환하는 필터(Filter)들이 있다. (=kernel;커널)

- 필터 행렬의 값은 신경망의 학습 과정을 통해 결정됨 (일반 신경망의 가중치 갱신과 비슷)
- 특징을 가진 값으로, 입력 데이터와 동일한 차원을 가짐
- 여러 개의 필터를 사용하여 한 번에 여러 특징을 추출할 수 있다.

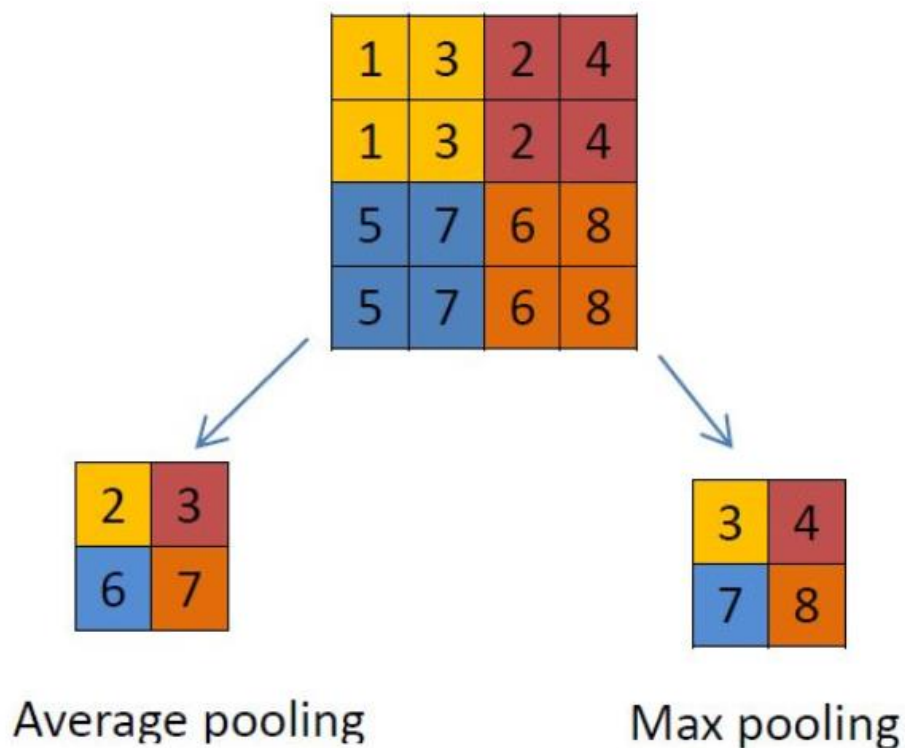
→ 입력 이미지와 컨볼루션 필터를 연산하여 특징 맵을 얻어낸다.

Pooling Layer

풀링 계층

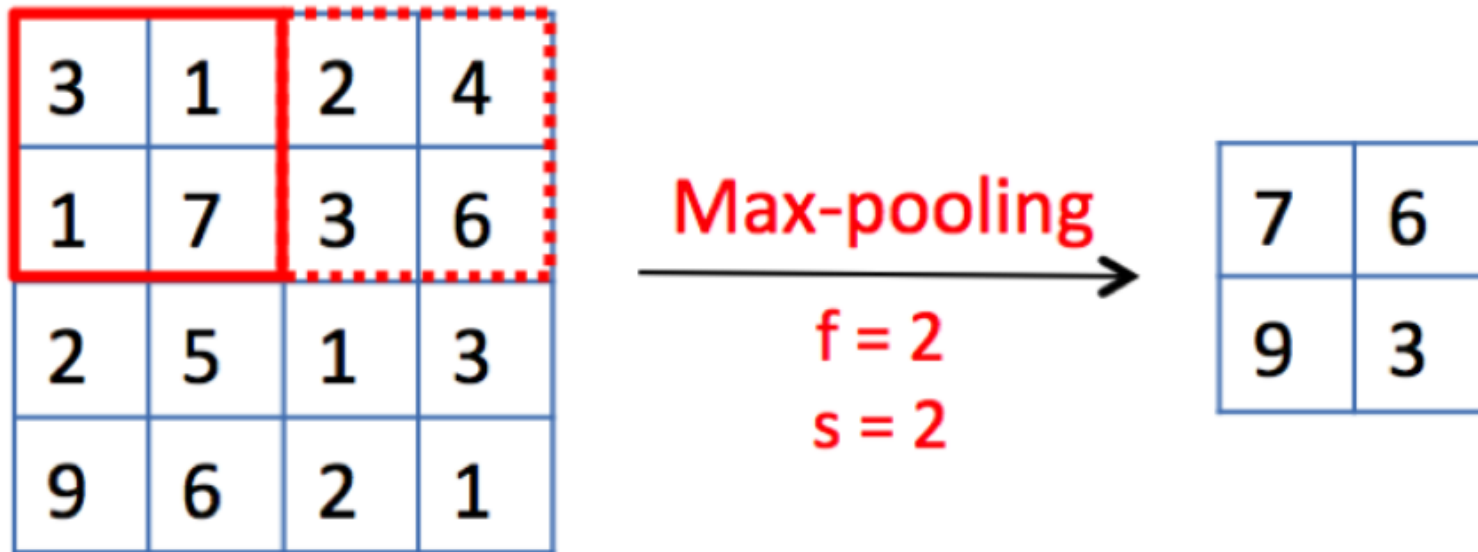
: 이미지의 크기를 줄이는 역할

- 입력 이미지에서 특정 영역에 있는 픽셀들을 묶어서 **하나의 대표 값으로 축소**
- 보통 정방 행렬(square matrix) 형태로 픽셀들을 선택하고,
선택한 픽셀들로부터 하나의 대표 값을 계산하는 연산으로는 보통 **평균값**이나 **최대값**을 많이 사용



- 수학적으로 풀링은 일종의 컨볼루션 연산!
- 컨볼루션 계층과 달리
컨볼루션 필터가 고정되어 있다.

Pooling



Convolution Operation

3	1	1	2	8	4
1	0	7	3	2	6
2	3	5	1	1	3
1	4	1	2	6	5
3	2	1	3	7	2
9	2	6	2	5	1

Original image 6x6

“Convolution”

×

1	0	-1
1	0	-1
1	0	-1

Filter 3x3

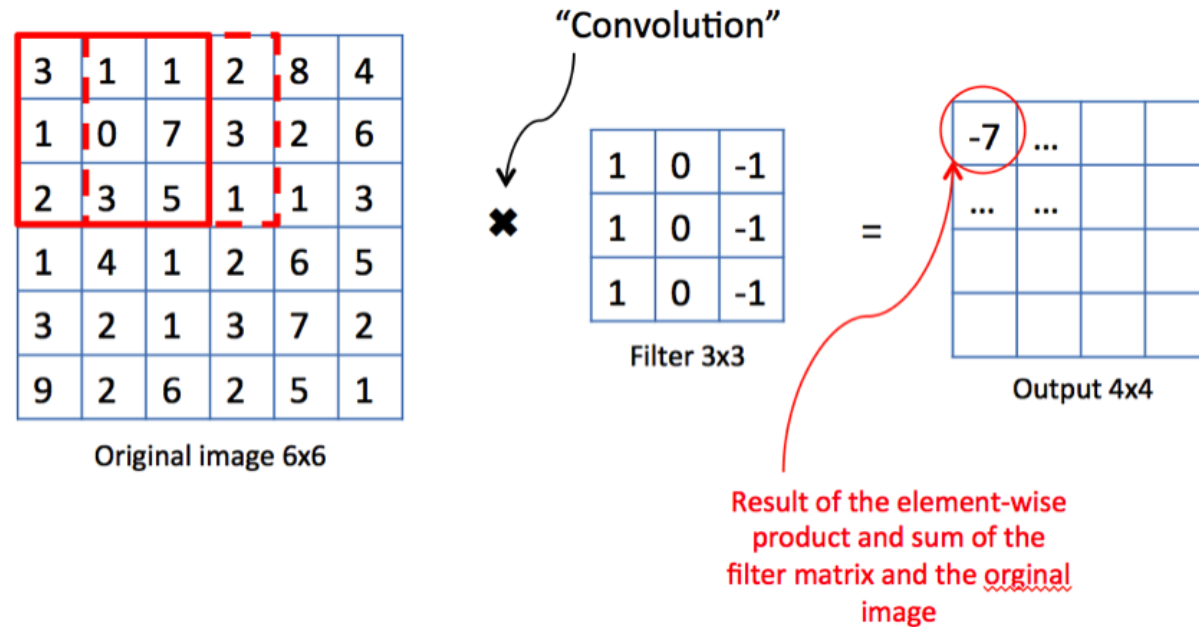
=

-7	...		
...	...		

Output 4x4

Result of the element-wise product and sum of the filter matrix and the original image

Convolution Operation



1. 필터의 윈도우(window)를 일정 간격으로 이동해 가면서, 각각의 범위 내 데이터에만 적용됩니다. (빨간 네모)
2. 단일 곱셈-누산(fused multiply-add, FMA): 입력과 필터에 대응하는 원소끼리 곱한 후 총합을 구합니다.
3. 위 과정을 모든 장소에서 수행합니다.
4. 필터를 적용한 원소에 고정값(편향)을 더합니다.

Convolution Operation

- Input feature map이 filter와 패턴이 일치할수록 → Output feature map의 값이 커짐
즉, 입력값이 filter와 얼마나 일치하는지 나타냄!
- 이미지나 음성 등에서 특정 패턴을 추출할 때 많이 사용
(계층이 얇으면 : CNN은 둥근 모양, 날카로운 모양 등 저수준의 패턴 정보를 추출,
계층이 깊으면 추상적인 정보를 추출)

Padding

0	0	0	0	0	0	0	0
0	3	3	4	4	7	0	0
0	9	7	6	5	8	2	0
0	6	5	5	6	9	2	0
0	7	1	3	2	7	8	0
0	0	3	7	1	8	3	0
0	4	0	4	3	2	2	0
0	0	0	0	0	0	0	0

$6 \times 6 \rightarrow 8 \times 8$

*

1	0	-1
1	0	-1
1	0	-1

3×3

=

-10	-13	1			
-9	3	0			

6×6

Padding이란,

- 입력 데이터 주위에 **0**을 채우는 것
- CNN의 **출력 크기**를 **조절**하기 위해 사용함
- Padding이 없으면 CNN layer를 통과할 때마다 가로/세로 크기가 감소하지만, Padding을 적용하면 Convolution을 수행해도 데이터의 크기가 동일함

개념 목차

수많은 데이터를
어떻게 해야
빠르게 학습시킬 수
있을까?



개념 : Batch, epoch

최적의 학습 결과를
찾을 수 있는 방법은?



개념 : hyper-parameter, learning rate,
cost function, backpropagation,
Gradient Descent, SGD,

학습 데이터가
너무 적다면?
Overfitting 방지는?



개념 : Data Augmentation,
Overfitting, Regularization



Batch (배치)

사전적 의미 : (일괄적으로 처리되는) 집단[무리]

딥러닝에서

batch?

: 한번에 여러 개의 데이터를 묶어서 입력하는 것

(GPU의 병렬 연산 기능을 최대한 효율적으로 사용하여 학습 속도를 줄일 수 있음)

batch size?

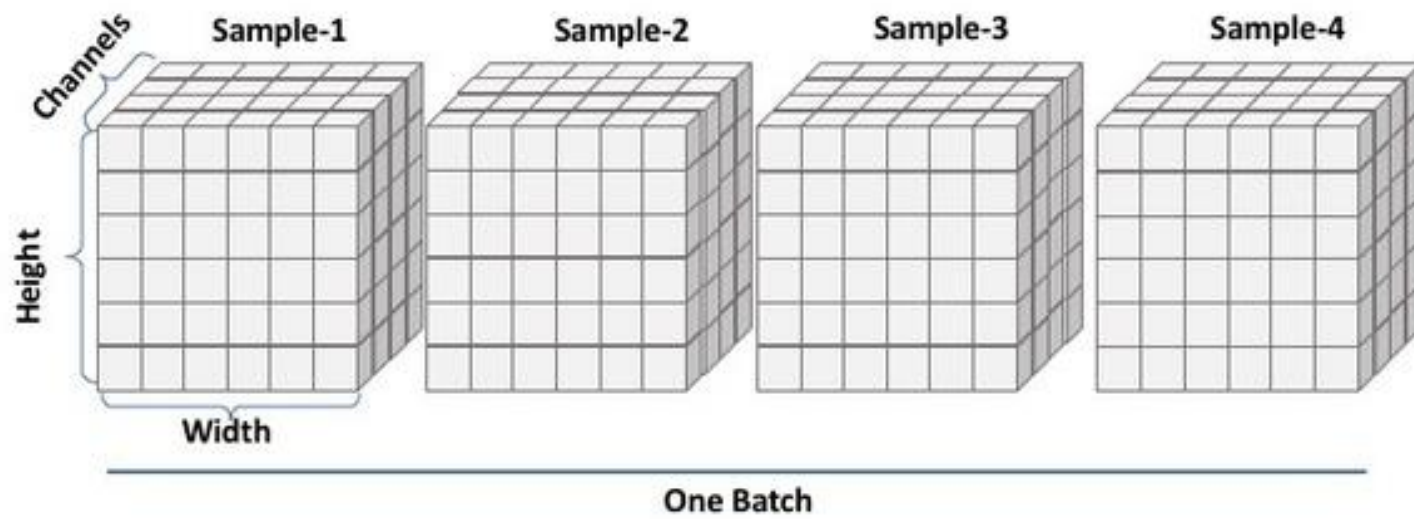
: 한 번의 batch마다 주는 데이터 샘플의 크기

ex) 1000개의 샘플 데이터, batch size=20개

→ 20개의 샘플 단위마다 모델의 가중치를 한번씩 업데이트, 총 50번($1000/20$) 가중치 업데이트

→ 총 500번 ($1000/20$)의 iteration(반복)으로 가중치 업데이트가 이뤄짐

Batch (배치)



Batch (배치)



1 Epoch : **모든** 데이터 셋을 한 번 학습

1 iteration : 1회 학습

minibatch : 데이터 셋을 **batch size** 크기로 쪼개서 학습

ex) 총 데이터가 100개, batch size가 10이면,
1 iteration = 10개 데이터에 대해서 학습
1 Epoch = $100 / \text{batch size} = 10$ iteration

epoch (에포크)

N epochs?

: 전체 데이터 셋에 대해 인경신경망을 통해 **N번 학습**(forward/backward)하는 것


- epoch가 너무 작으면 → underfitting
 - epoch가 너무 크면 → overfitting
- 발생할 수 있으므로 **적절한 값**을 설정하는 것이 중요

Q. 10000개의 데이터 셋을 학습시킨다.
한 턴에 1000개씩 10번, 총 5턴 학습
→ batch size 는 1000
→ iteration = 10
→ epoch = 5

hyper-parameter(하이퍼파라미터)

hyper-parameter?

- 머신러닝에서 분류기 등 모형의 학습 양상을 결정하는 중요한 수치
- 분류기를 학습하는 과정에서 업데이트되는 파라미터(W, b)와 달리 하이퍼파라미터는 **분류기를 학습하기 전에 설정**하며, 학습 과정에서는 일반적으로 업데이트되지 않는다.
- ex. 학습률(learning rate; η), 정규화 강도(regularization strength; λ), CNN에서 filter 크기 및 stride, 경사하강법 외 다른 최적화 방법에 필요한 하이퍼파라미터 등


$$W_{new} = W_{old} - \eta \frac{\partial L}{\partial W}$$
$$b_{new} = b_{old} - \eta \frac{\partial L}{\partial b}$$

learning rate (학습률)

경사하강법 알고리즘은 기울기에 학습률(learning rate) 또는 보폭(step size)이라 불리는 **스칼라**를 곱해 다음 지점을 결정

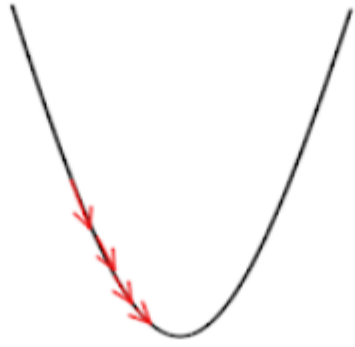
- 학습률이 **큰** 경우 : 데이터가 무질서하게 이탈하며, 최저점에 수렴하지 못함
- 학습률이 **작은** 경우 : 학습시간이 오래 걸리며, 최저점에 도달하지 못함

따라서 Local minimum에 효율적으로 도달할 수 있도록 너무 크지도 작지도 않은 **적절한 학습률**을 세팅해야 한다.

Big Learning Rate



Just right



Too small



learning rate (학습률)

경사하강법 알고리즘은 기울기에 학습률(learning rate) 또는 보폭(step size)이라 불리는 **스칼라**를 곱해 다음 지점을 결정

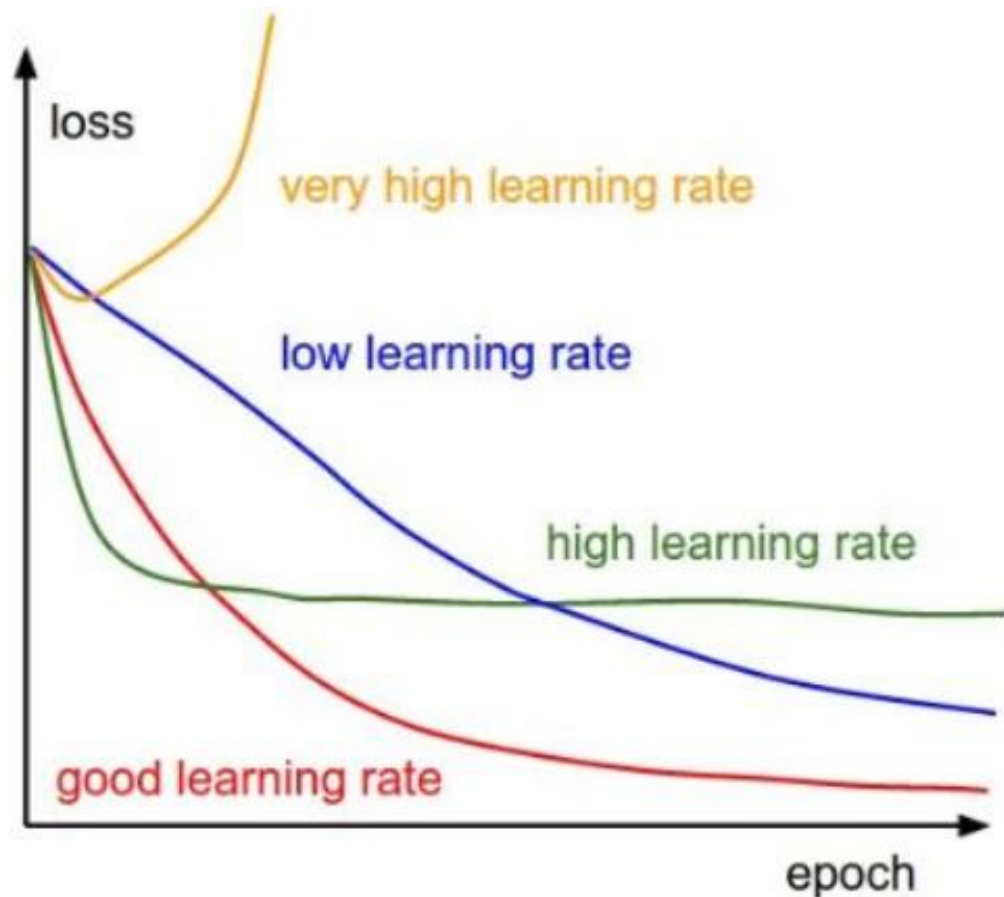
- 학습률이 **큰** 경우 : 데이터가 무질서하게 이탈하며, 최저점에 수렴하지 못함
- 학습률이 **작은** 경우 : 학습시간이 오래 걸리며, 최저점에 도달하지 못함

따라서 Local minimum에 효율적으로 도달할 수 있도록 너무 크지도 작지도 않은 **적절한 학습률**을 세팅해야 한다.

[파라미터 업데이트 알고리즘]

- **Step decay** ~번의 반복(epochs)마다 ~만큼의 learning rate를 줄이는 방법.
- **Exponential decay** $\alpha = (\alpha_0)e^{-kt}$ (k : 하이퍼파라미터, t : 반복수)
- **1/t decay** $\alpha = (\alpha_0)/(1+kt)$

learning rate (학습률)



low learning rate:

손실(loss) 감소가 선형의 형태를 보이면서 천천히 학습됨

high learning rate:

손실 감소가 지수적인(exponential) 형태를 보이며, 구간에 따라 빠른 학습 혹은 정체가 보임

very high learning rate:

매우 높은 학습률은 경우에 따라, 손실을 오히려 증가시키는 상황을 발생

good learning rate:

적절한 학습 곡선의 형태로, Learning rate를 조절하면서 찾아내야 함

Cost Function (Loss Function)

보통 예측값과 실제값의 차이로 정의된다.

$$C(w, b) \equiv \frac{1}{2n} \sum_x \|y(x) - a\|^2$$

n : 훈련에 사용하는 입력의 수

$y(x)$: 입력 x 를 가했을 때의 기대(target) 출력

a : 입력 x 를 신경망에 넣었을 때의 실제 출력

cost function?

: 신경망에 훈련 데이터 x 를 넣고

실제 값과 예측 값 간의 차에 대한 MSE(Mean Square Error)를 구하는 것

훈련 데이터를 이용해 가중치(w)와 바이어스(b)를 변화시키는 과정을 반복적으로 수행,
cost function이 최소값이 되도록 하는 것이 신경망 학습의 목표

Backpropagation (역전파)

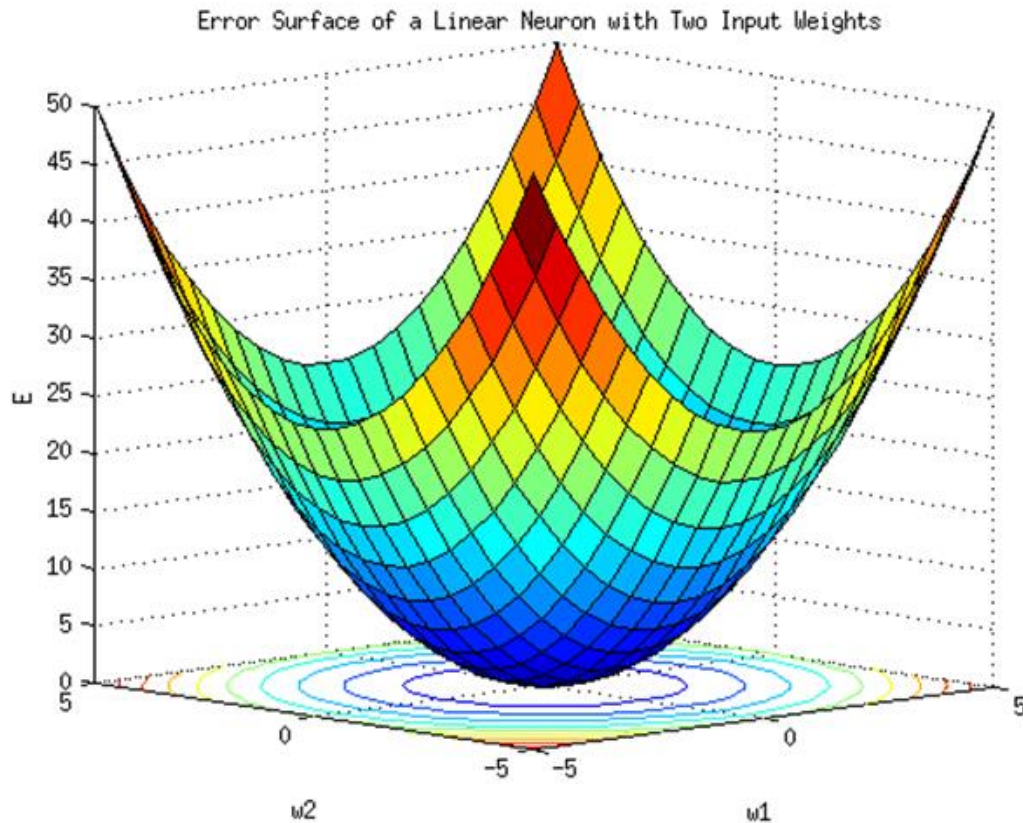
역방향으로 에러를 전파 (backward propagation of error) → 최적의 학습 결과 찾음

- feed forward: 입력 → 출력 (최종 출력단에서 error와 cost function을 구함)
- back propagation: 출력부터 반대 방향으로 순차적으로 편미분을 수행해가면서
뉴런의 가중치(w)와 바이어스(b)값들을 갱신
(가중치 w에 대해서 편미분하여 기울기 계산)

※에포크(epoch)는 전체 데이터에 대해 순전파(feed-forward)와 역전파(back-propagation)가 끝난 상태를 뜻함

Gradient Descent (경사 하강법)

안개가 끼어있는 산정상에서 한치 앞이 보이지 않는다면, 어떻게 내려가야 할까?
질은 안개로 인해 앞이 잘 보이지 않는다면,
자기의 현재 위치에서의 정보(local information)만을 활용해야 하며,
그 정보로는 가장 경사가 큰 쪽으로 내려가는 길을 택하면 된다.



경사 하강법?

: 미분의 개념을 최적화 문제에 적용된 대표적 방법 중 하나
함수의 **local minimum**을 찾음

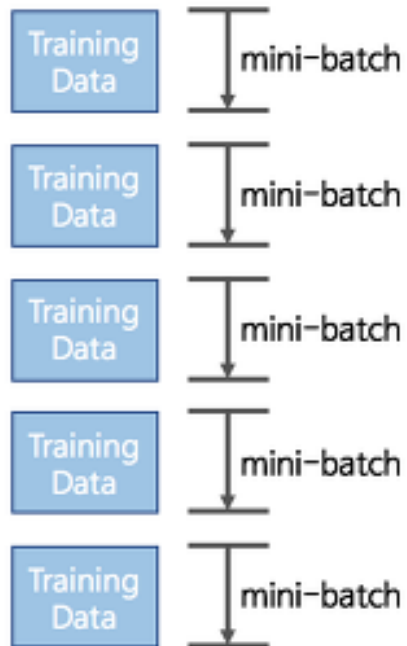
gradient(경사)가 음의 최대값 쪽으로 움직이면
결국 최솟값에 도달할 수 있다.

확률적 경사 하강법 (SGD : Stochastic Gradient Descent)

Gradient Decent



Stochastic Gradient Decent



GD

모든 트레이닝 데이터에 대한 손실함수를 구하고 기울기를 구함
but 데이터가 용량이 큰 경우 계산이 오래 걸림

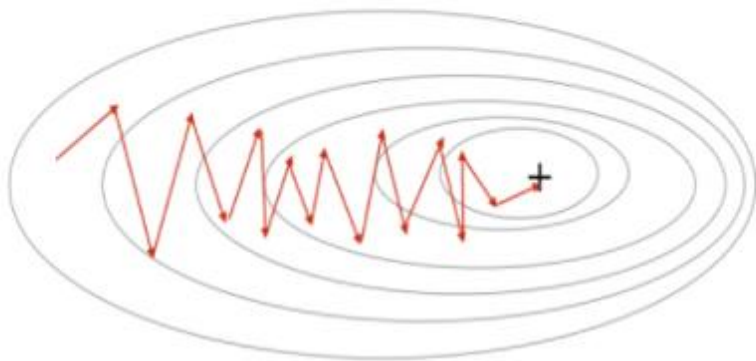
SGD

트레이닝 데이터를 **mini-batch(size=1)**단위만큼 **무작위**로 추출하여
이 데이터만으로 학습을 하는 것

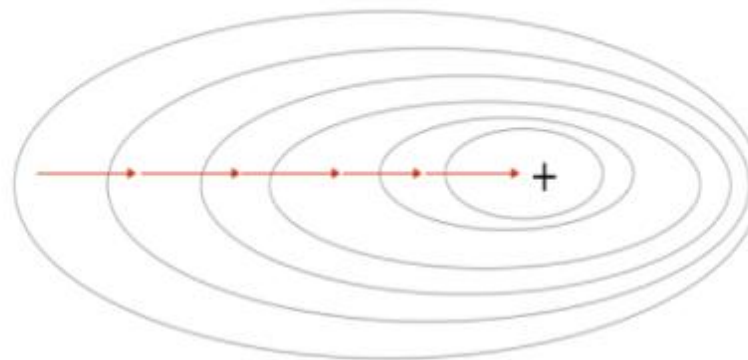
- 장점 : 수렴 속도가 빠르다
- 단점 : 오차율이 크다 (**global minimum**을 찾지 못할 가능성 있음)

확률적 경사 하강법 (SGD : Stochastic Gradient Descent)

Stochastic Gradient Descent



Gradient Descent



무작위로 표본집단을 계속 추출하여 최적해를 구한다면
최적해의 값은 통계학적으로 **GD**와 별 차이가 없지만,
계산 시간을 훨씬 단축시킬 수 있는 장점이 있어서 많이 쓰인다.

확률적 경사 하강법 (SGD : Stochastic Gradient Descent)

$$W \leftarrow W - \eta \frac{\partial L}{\partial W}$$

W : 갱신할 가중치 매개변수

η : 학습률 (learning rate) >0

이때 너무 크지도, 작지도 않은 값으로 세팅해야 한다.

$\ell L / \ell W$: 매개변수(W)에 대한 손실함수(L)의 기울기

Data Augmentation (데이터 확장)

이미 존재하는 데이터 샘플을 일정하게 **가공**하여 양을 늘리는 방법

ex.

- 이미지의 평행이동, 대칭이동, 회전 등의 변환
- 픽셀의 명암값, 색에 변동을 가한 결과 이미지
- 가우스 분포를 따른 랜덤 노이즈 추가
- 배경 음악의 일부를 랜덤 샘플링하여 음성과 합성하는 것

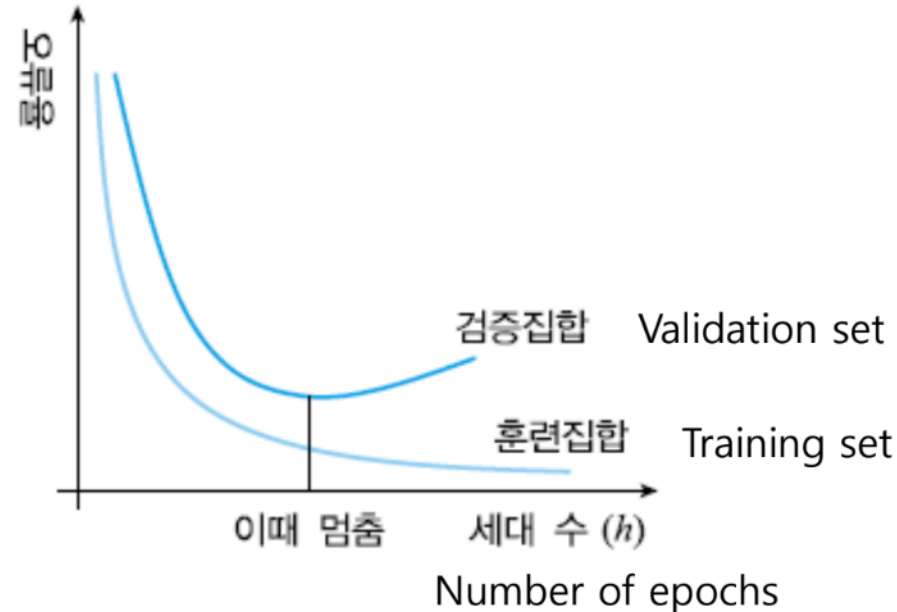
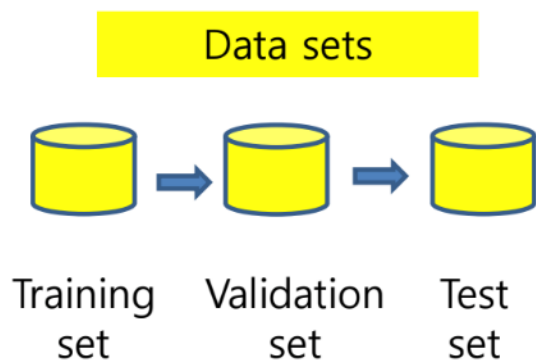
학습 데이터 셋이 부족할 경우 **과적합**(overfitting)이 발생하는데,
새로운 데이터 셋을 가져오기 보다는 **기존의 데이터 셋에서 확장**하여 **과적합을 방지**하는 방법

Overfitting (과적합)

Overfitting

: 파라미터에 비해 학습 데이터가 너무 적은 경우,
학습 데이터 외의 데이터에 대해 **잘못된 학습 결과**(예측에 오류가 많이 일어남)를 보이는 현상

- 극복 방법 : Regularization, Cross Validation, Add more data as much as possible
- 이를 방지하기 위해 학습 데이터에서 Test set과 Validation set을 샘플로 뽑아내서 두 개의 데이터 셋을 제외하고 학습 시킴
- 이렇게 **Training set, Validation set, Test set**으로 데이터 셋을 3개로 나누어 학습 진행 : **일반화(Generalization)**
 - 목적 : 학습에 사용되지 않은 데이터에 대한 정확도를 높임



Regularization (정규화)

- 과적합을 막는 방법
- 지금까지 연구된 정규화 방식 : L2 정규화, L1 정규화, Max norm constraints, 드롭아웃

L2 정규화

- 가장 일반적으로 사용되는 정규화 기법
- 모든 파라미터 제곱 만큼의 크기를 손실함수에 더하는 방식
- 큰 값이 많이 존재하는 파라미터에는 제약이 걸리며, 작은 값이 널리 퍼지는 효과

L1 정규화

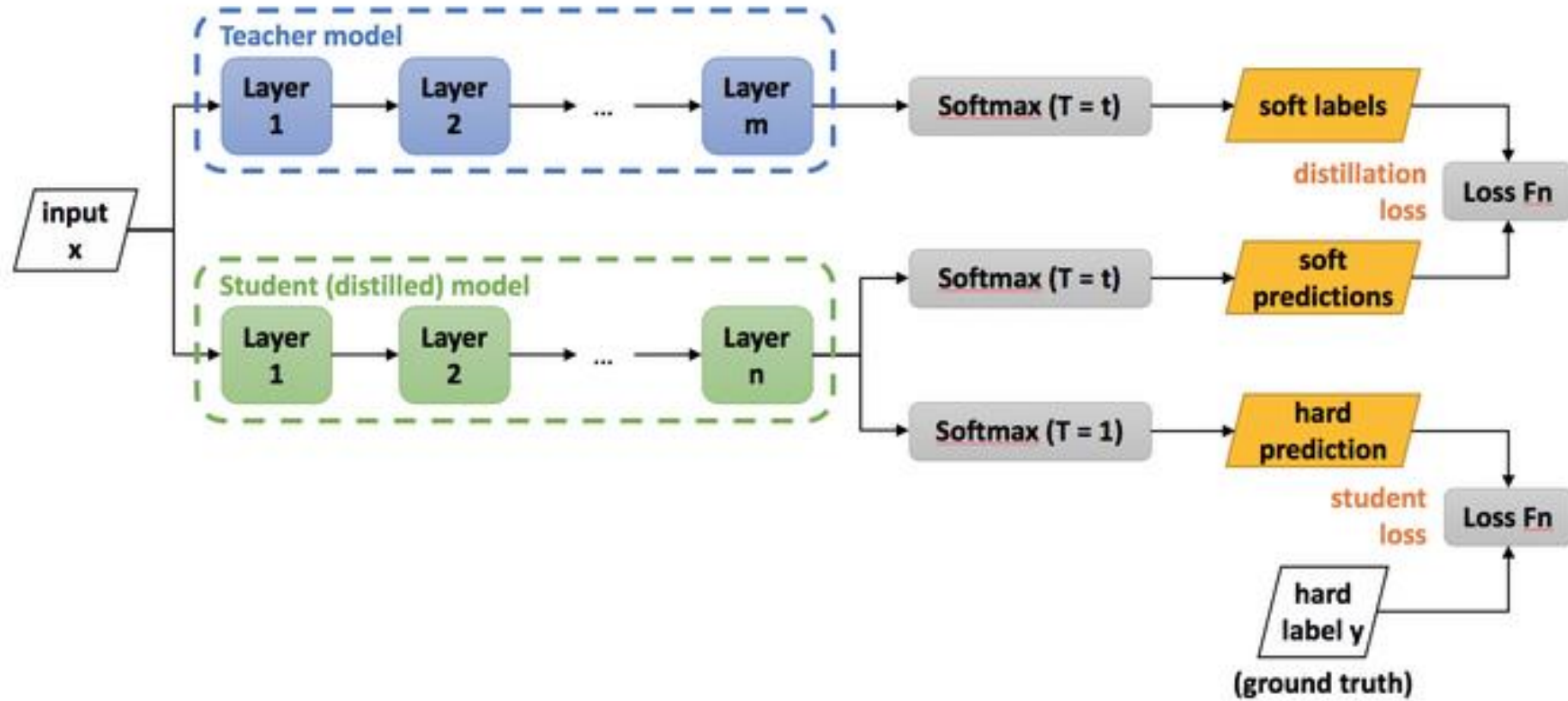
- 파라미터의 L1 norm을 손실함수에 더하는 방식
- 파라미터 행렬을 sparse하게 (0에 가깝게) 만드는 특성
- 노드들은 입력 데이터의 sparse한 부분만 사용하고, noisy한 입력 데이터에는 영향을 받지 않음
- 두 정규화 방식을 합쳐서 쓰는 경우도 있다.

2. 본론

Knowledge Distillation

- 교사 네트워크(**teacher network**)라고 하는 사전 훈련된 복잡한 네트워크의 지식을 전달하여 더 간단한 네트워크, 즉 학생 네트워크(**student network**)를 효과적으로 훈련하는 것
- **knowledge** : hidden layer의 **feature**이나 final layer의 **logits** 등을 포함

Knowledge Distillation



self-knowledge distillation

- 교사 네트워크의 사전 훈련 과정이 필요없다.
- 크게 두 가지로 구분된다.
 - (1) **data augmentation based approach**
 - ex. DDGSD, CSKD, SLA
 - 데이터 증강은 반드시 공간 정보를 보존하지 않는다.
→ 특징 증류를 활용하기가 어렵다.
 - (2) **auxiliary network based approach**
 - 분류기 네트워크 중간에 추가 분기를 활용하고
지식 전달을 통해 추가 분기를 유도하여 유사한 출력 생성 (상향식, 하향식 경로)
 - ex. FPN, PANet, BiFPN
 - 분류기 네트워크에 대한 컨볼루션 계층의 출력인 기능이나
soft label로 정제된 지식을 생성하기 어렵다.

- 보조 자가 교사 네트워크를 활용하여 분류기 네트워크에 정제된 지식 전달
 - 자가 지식 증류를 위해 **soft label**과 **feature-map** 증류 모두 사용 가능
- **local information**의 보존을 강조하는 분류 및 의미 분할에 적용될 수 있음
- 데이터 증강 뿐만 아니라 다른 자가 지식 증류 방법과도 호환됨

FRSKD 개요

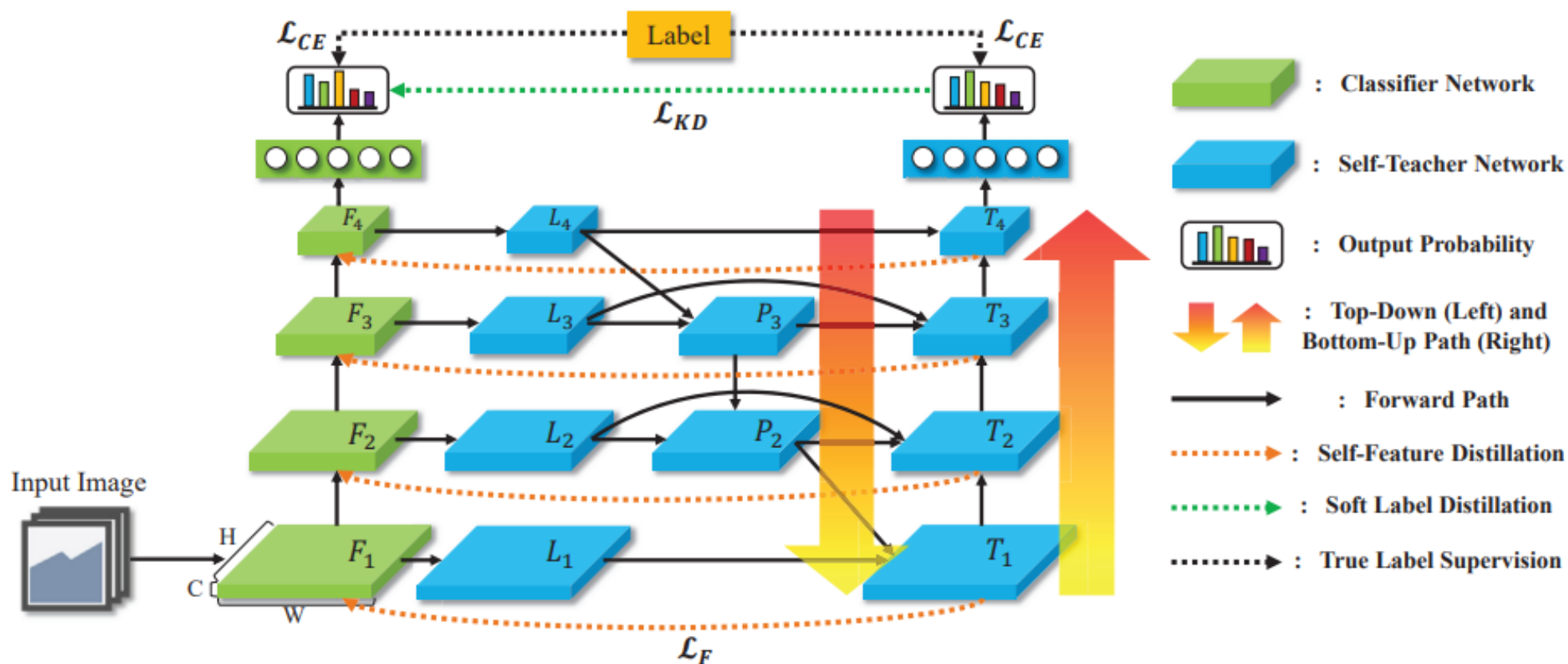
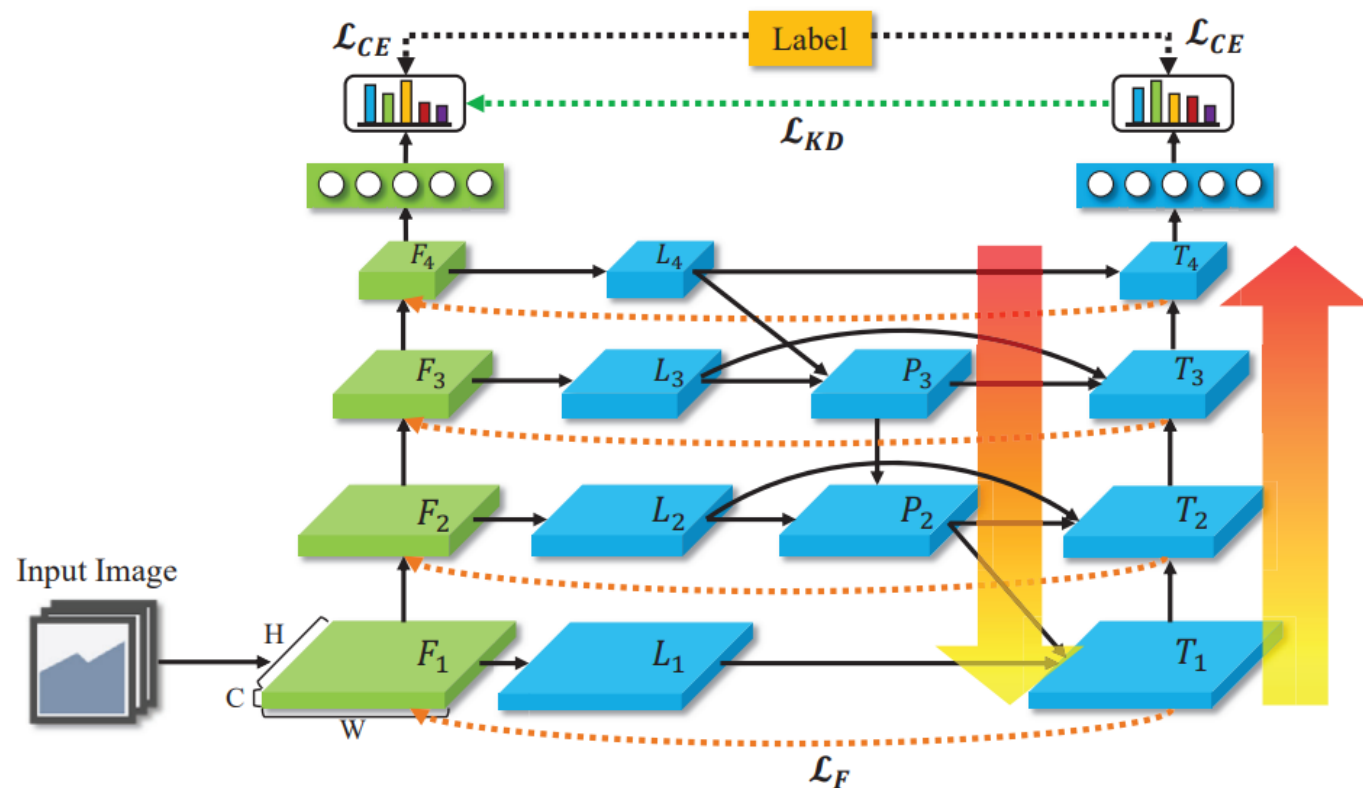
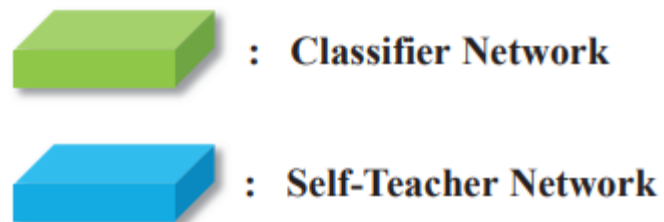


Figure 2: Overview of our proposed self-knowledge distillation method, Feature Refinement via Self-Knowledge Distillation (FRSKD). The top-down path and the bottom-up path aggregates different sized features and provide the refined feature-map to the original classifier network. Exploiting feature-map of the self-teacher network, FRSKD performs distillation on refined feature-map and soft label.

Self-Teacher Network

- 목적 : 분류기 네트워크에 대한 **refined feature-map**과 **soft label** 자체 제공
- 입력 : 분류기 네트워크의 **feature-map** (F_1, \dots, F_n)
- 본 논문에서는
BiFPN의 구조를 수정해서 **Self-Teacher Network** 모델링함
- PANet 및 BiFPN에서의 하향식&상향식 경로 적용



Self-Teacher Network

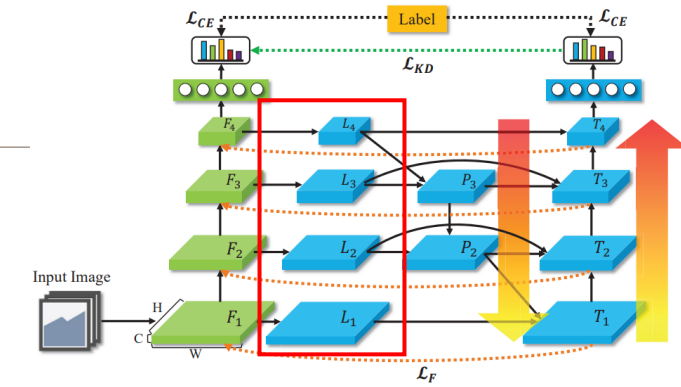
self-teacher network가 제공하는 **Soft Label**은 다음과 같다.

$$\hat{p}_t = \text{softmax}(f_t(\mathbf{x}; \theta_t))$$

where f_t denotes the self-teacher network,
parameterized by θ_t .

Self-Teacher Network

lateral convolutional layers



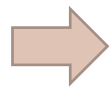
$$L_i = \text{Conv}(F_i; d_i) \quad (1)$$

Conv : 컨볼루션 연산 (convolutional operation)

d_i : 출력 차원 (output dimension)

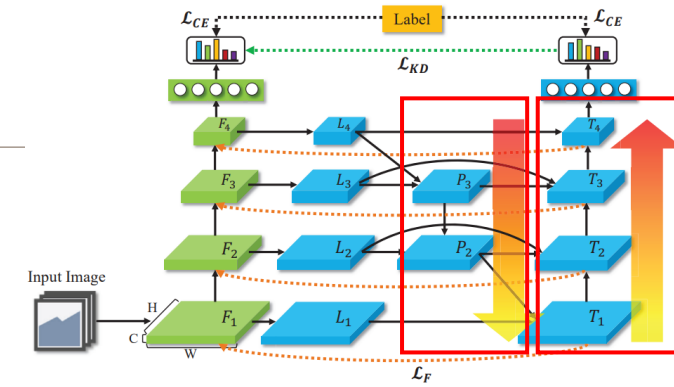
$$d_i = w \times c_i$$

feature-map의 채널 차원 c_i 에 의존하는 d_i 설계
채널 폭 매개변수 w 사용



각 레이어의 채널 차원을 조정하여 **feature-map** 깊이 정보를 포함시킴
(분류화 작업의 경우, 깊은 계층에 대해 더 높은 채널 차원이 설정됨)
이는 측면 컨볼루션 레이어의 **계산을 줄임**

Self-Teacher Network



하향식 경로의 i번째 레이어

$$P_i = \text{Conv}(w_{i,1}^P \cdot L_i + w_{i,2}^P \cdot \text{Resize}(P_{i+1}); d_i) \quad (2)$$

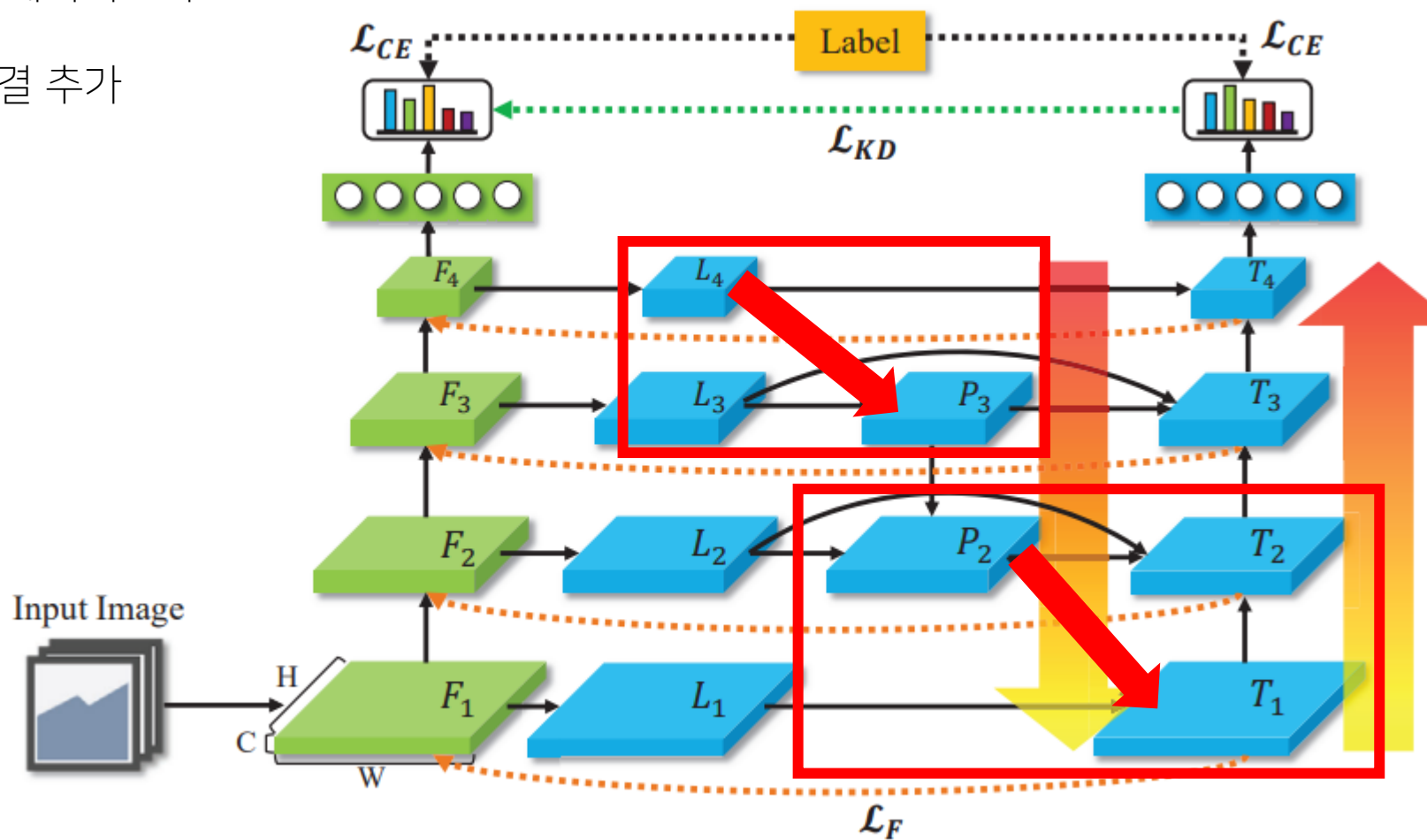
상향식 경로의 i번째 레이어

$$T_i = \text{Conv}(w_{i,1}^T \cdot L_i + w_{i,2}^T \cdot P_i + w_{i,3}^T \cdot \text{Resize}(T_{i-1}); d_i)$$

Self-Teacher Network

하향식, 상향식 경로 레이어

가장 얇은 레이어, 중간 레이어, 깊은 레이어 모두 연결하는 **하향식 구조**를 만들기 위해
순방향 전파를 위한 **2개의 대각선 연결** 추가



Self-Feature Distillation

Feature Distillation

: classifier network가 self-teacher network에서 refined feature-map을 모방(위치 학습)하도록 유도

- 본 논문에서는 **attention transfer**을 적용
- Loss 함수는 아래와 같다.

$$\mathcal{L}_F(T, F; \theta_c, \theta_t) = \sum_{i=1}^n \|\phi(T_i) - \phi(F_i)\|_2 \quad (3)$$

Self-Feature Distillation

“Paying More Attention to Attention: Improving the Performance of Convolutional Neural Networks via Attention Transfer” [36]

attention transfer



- **Attention map** : 이미지에서 집중적으로 학습해야 하는 부분
- **Layer**이 깊어질수록 데이터를 학습하여 이미지의 **Attention**이 뚜렷하게 표현됨
- 사람의 얼굴에 **Attention**됨

Self-Feature Distillation

FRSKD는 다른 자가지식증류 방법들과 비슷하게 **soft label**을 통해 증류를 한다.

$$\hat{p}_t = \text{softmax}(f_t(\mathbf{x}; \theta_t))$$

$$\begin{aligned} \mathcal{L}_{KD}(\mathbf{x}; \theta_c, \theta_t, K) \\ = D_{KL}(\text{softmax}(\frac{f_c(\mathbf{x}; \theta_c)}{K}) || \text{softmax}(\frac{f_t(\mathbf{x}; \theta_t)}{K})) \end{aligned} \quad (4)$$

f_c is the classifier network

K is the temperature scaling parameter

Self-Feature Distillation

\mathcal{L}_{CE}

classifier network와 self-teacher network은
교차 엔트로피 손실 (cross-entropy loss)을 사용하여
ground-truth 레이블을 학습

Self-Feature Distillation

$$\mathcal{L}_F(T, F; \theta_c, \theta_t) = \sum_{i=1}^n \|\phi(T_i) - \phi(F_i)\|_2$$

$$\mathcal{L}_{KD}(\mathbf{x}; \theta_c, \theta_t, K)$$

$$= D_{KL}(\text{softmax}(\frac{f_c(\mathbf{x}; \theta_c)}{K}) \parallel \text{softmax}(\frac{f_t(\mathbf{x}; \theta_t)}{K}))$$

$$\mathcal{L}_{CE}$$

➡ 위의 손실함수를 통합하여 다음 **최적화 목표**를 구성

Self-Feature Distillation

최적화 목표는 다음과 같다.

$$\begin{aligned} \mathcal{L}_{FRSKD}(\mathbf{x}, y; \theta_c, \theta_t, K) & \quad (5) \\ &= \mathcal{L}_{CE}(\mathbf{x}, y; \theta_c) + \mathcal{L}_{CE}(\mathbf{x}, y; \theta_t) \\ &\quad + \alpha \cdot \mathcal{L}_{KD}(\mathbf{x}; \theta_c, \theta_t, K) + \beta \cdot \mathcal{L}_F(T, F; \theta_c, \theta_t) \end{aligned}$$

α and β are hyperparameters

we choose $\alpha \in [1, 2, 3]$ and $\beta \in [100, 200]$

Self-Feature Distillation

$$\begin{aligned}\mathcal{L}_{FRSKD}(\mathbf{x}, y; \theta_c, \theta_t, K) & \quad (5) \\ &= \mathcal{L}_{CE}(\mathbf{x}, y; \theta_c) + \mathcal{L}_{CE}(\mathbf{x}, y; \theta_t) \\ & \quad + \alpha \cdot \mathcal{L}_{KD}(\mathbf{x}; \theta_c, \theta_t, K) + \beta \cdot \mathcal{L}_F(T, F; \theta_c, \theta_t)\end{aligned}$$

최적화는 분류기 네트워크, 자가교사 네트워크 모두에 대해 동시에 역전파(**backpropagation**)에 의해 **initialize**된다.

모델의 붕괴 문제를 방지하기 위해

FRSKD는 학생 네트워크에만 적용되는 증류 손실 $\mathcal{L}_{KD}, \mathcal{L}_F$ 에 의해 매개변수가 업데이트 된다.

Experiments

1. Classification
2. Semantic Segmentation
3. Further Analyses on FRSKD

Experiments – 세 가지 설정

FRSKD\F

utilizing
the distillation of
soft label only

FRSKD\F

optimized by
 L_{FRSKD}
with distillation of
refined feature-map
and its soft label

FRSKD\F

attaching
the data augmentation
based on
the self-knowledge
distillation, SLASD

(1) Classification [Data Set]



- 작은 크기의 이미지로 구성됨
- **CIFAR-100 : 32x32**
- **TinyImageNet**의 이미지 크기를 **CIFAR-100**과 동일한 크기로 조정



- **FGVR (Finegrained Visual Recognition)**
작업을 위한 데이터 세트
- **CIFAR-100, TinyImageNet**에 비해
클래스당 더 적은 데이터 인스턴스를 포함



- 모델을 실제로 테스트하는 방법을
검증하기 위한 대규모 데이터 세트

(1) Classification [Implementation Details]

CIFAR-100,
TinyImageNet

- ResNet18
- WRN16-2

ResNet18의 첫번째 컨볼루션 레이어를
커널 크기 3*3, 단일 보폭,
단일 패딩으로 구성,
max-pooling 작업 제거

FGVR 작업

- ResNet18

ImageNet

- ResNet18
- ResNet34

(1) Classification [Implementation Details]

모든 Classification 실험에 대해 **확률적 경사 하강법** (SGD) 사용

- **learning rate** : 0.1
- **weight decay**: 0.0001
- **CIFAR-100, TinyImageNet 및 FGVR**
 - 총 epoch : 200
 - epoch 100과 150 : 학습률을 10으로 나눔 ($\rightarrow 0.01 \rightarrow 0.001$)
- **ImageNet**
 - 총 epoch : 90
 - epoch 30과 60 : 학습률을 10으로 나눔 ($\rightarrow 0.01 \rightarrow 0.001$)
- 온도 스케일링 매개변수 $K = 4$
- 채널 너비 매개변수 $w = 2$

(1) Classification [Baselines]

FRSKD와 비교할 classifier 총 7개의 baselines
: Baseline(증류X), 아래의 6가지 자가지식증류 방법들

- **ONE** [43] exploits an ensembled prediction of additional branches as the soft label.
- **DDGSD** [32] generates different distorted versions of a single instance, and DDGSD trains to produce consistent prediction for the distorted data.
- **BYOT** [39] applies auxiliary classifiers utilizing the outputs of intermediate layers, and BYOT trains auxiliary classifiers by ground-truth labels and signals from network itself, such as the predicted logit or the feature-map.
- **SAD** [10] focuses on a lane detection by a layer-wise attention distillation in network itself.
- **CS-KD** [35] forces a consistent prediction for the same class by utilizing the prediction of other instances within the same class as the soft label.
- **SLA-SD** [18] trains a network with an original classification task and a self-supervised task jointly by utilizing the label augmentation. SLA-SD exploits an aggregated prediction as the soft label.

(1) Classification [Performance comparison]

Methods	CIFAR100		TinyImageNet	
	WRN-16-2	ResNet18	WRN-16-2	ResNet18
Baseline	70.42±0.08	73.80±0.60	51.05±0.20	54.60±0.33
ONE	73.01±0.23	76.67±0.66	52.10±0.20	57.53±0.39
DDGSD	71.96±0.05	76.61±0.47	51.07±0.24	56.46±0.24
BYOT	70.22±0.26	76.68±0.07	50.33±0.03	56.61±0.30
SAD	70.31±0.45	74.65±0.33	51.26±0.39	54.45±0.06
CS-KD	71.79±0.68	77.19±0.05	50.08±0.18	56.46±0.10
SLA-SD	73.00±0.45	77.52±0.30	50.77±0.33	58.48±0.44
FRSKD\F	73.12±0.06	77.64±0.12	<u>52.91±0.30</u>	59.50±0.15
FRSKD	<u>73.27±0.45</u>	<u>77.71±0.14</u>	53.08±0.33	<u>59.61±0.31</u>
FRSKD+SLA	75.43±0.21	82.04±0.16	51.83±0.37	63.58±0.04

Table 1: Performance comparison on CIFAR-100 and TinyImageNet. Experiments are repeated three times, and we report average and standard deviation of the accuracy of the last epoch. The best performing model is indicated as bold-face. The second-best model is indicated as underline.

- 자가 교사 네트워크의 soft label이 데이터 증대 기반 방법보다 성능이 우수함
- 특징 증류를 사용하지 않는 FRSKD\F는 다른 baselines보다 성능이 우수함
- FRSKD는 다른 자가지식 증류법보다 일관되게 더 나은 성능을 보임
- 데이터 증강 기반의 자기 지식 증류 방법인 SLA-SD와 데이터 증강에 의존하지 않는 FRSKD를 통합한 FRSKD+SLA는 대부분의 실험에서 큰 마진으로 성능 향상을 보임

(1) Classification [Performance comparison]

Methods	CUB200	MIT67	Dogs	Stanford40
Baseline	51.72 \pm 1.17	55.00 \pm 0.97	63.38 \pm 0.04	42.97 \pm 0.66
ONE	54.71 \pm 0.42	56.77 \pm 0.76	65.39 \pm 0.59	45.35 \pm 0.53
DDGSD	58.49 \pm 0.55	59.00 \pm 0.77	69.00 \pm 0.28	45.81 \pm 1.79
BYOT	58.66 \pm 0.51	58.41 \pm 0.71	68.82 \pm 0.15	48.51 \pm 1.02
SAD	52.76 \pm 0.57	54.48 \pm 1.30	63.17 \pm 0.56	43.52 \pm 0.06
CS-KD	64.34 \pm 0.08	57.36 \pm 0.37	68.91 \pm 0.40	47.23 \pm 0.22
SLA-SD	56.17 \pm 0.71	61.57 \pm 1.06	67.30 \pm 0.21	54.07 \pm 0.38
FRSKD\F	62.29 \pm 1.65	61.32 \pm 0.67	69.48 \pm 0.84	53.16 \pm 0.44
FRSKD	<u>65.39\pm0.13</u>	<u>61.74\pm0.67</u>	<u>70.77\pm0.20</u>	<u>56.00\pm1.19</u>
FRSKD+SLA	67.80\pm1.24	66.04\pm0.31	72.48\pm0.34	61.96\pm0.57

Table 2: Performance comparison on FGVR. Experiments are repeated three times, and we report average and standard deviation of the accuracy of the last epoch. The best performing model is indicated as boldface. The second-best model is indicated as underline.

- FRSKD는 다른 self-knowledge distillation보다 우수한 성능을 보임
- FRSKD+SLA는 큰 마진으로 다른 모든 방법보다 성능이 우수함

(1) Classification [Performance comparison]

Model	Method	Top-1	Top-5
ResNet18	Baseline	69.76	89.08
	FRSKD	70.17	89.78
ResNet34	Baseline	73.31	91.42
	FRSKD	73.75	92.11

Table 3: Performance comparison on ImageNet. The best performing model is indicated as boldface.

- 대규모 데이터 세트에서 FRSKD를 시연하기 위해 ResNet18, ResNet34를 사용하여 ImageNet에서 FRSKD평가
- FRSKD는 ImageNet에서 성능을 향상시킴

(2) Semantic Segmentation

[Dataset]

- Training set : VOC2007, VOC2012 *trainval* 결합
- Validation set : VOC2007

[실험]

- 스택형 BiFPN 구조를 기준으로 EfficientDet을 사용
- 3개의 BiFPN 레이어를 쌓고 추가로 2개의 BiFPN 레이어를 자가 교사 네트워크로 사용

[파라미터 설정]

- learning rate : 0.01
- 총 epoch : 60
 - epoch 40에서 학습률을 10으로 나눔 (0.001)

(2) Semantic Segmentation

Model	Method	mIOU
EfficientDet-d0	Baseline	79.07
	FRSKD	80.55
EfficientDet-d1	Baseline	81.95
	FRSKD	83.88

Table 4: Performance comparison on semantic segmentation task. The best performing model is indicated as bold-face.

→ FRSKD는 self-teacher 네트워크의 self-knowledge distillation을 활용하여 의미론적 분할 모델의 성능을 향상시킴

(3) Further Analyses on FRSKD

- Qualitative Attention map comparison

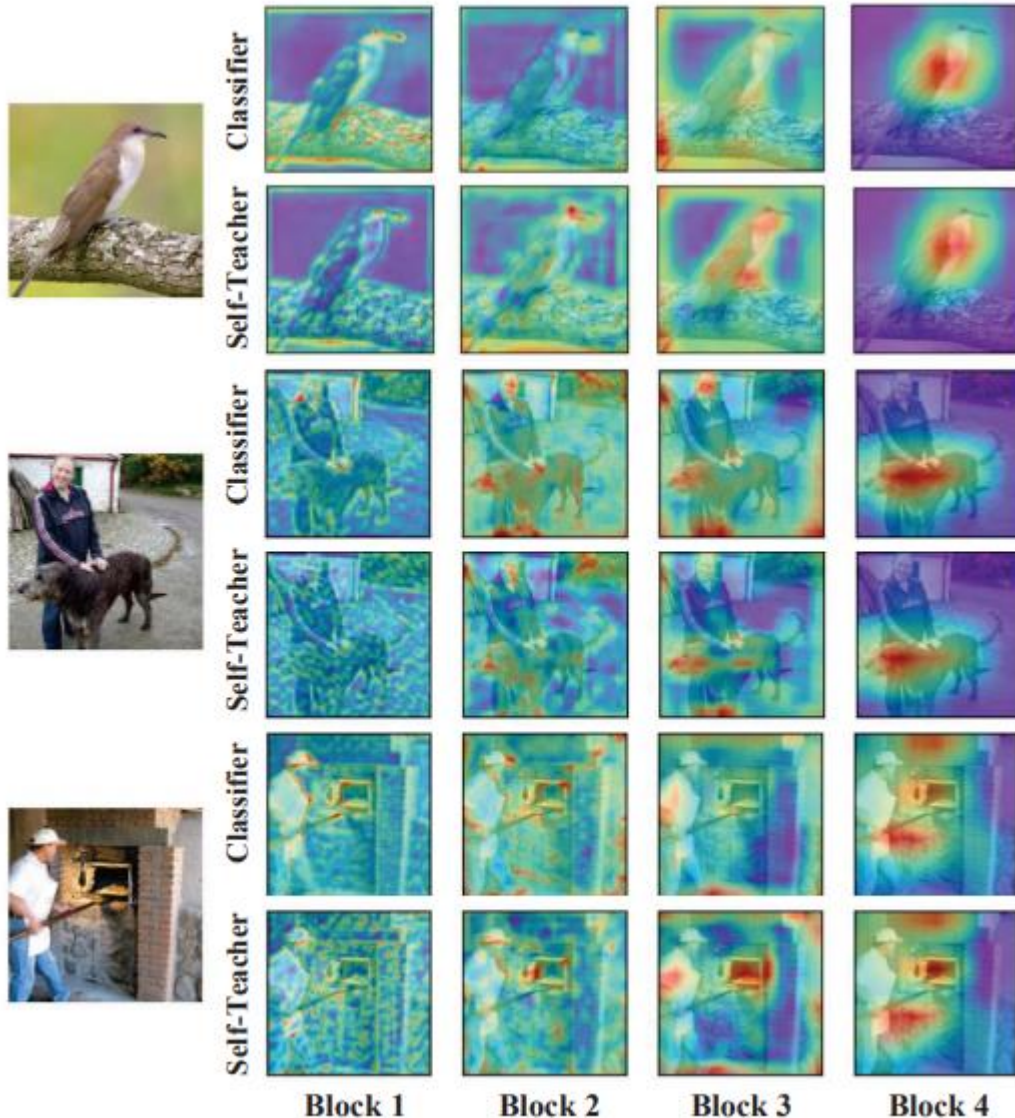


Figure 3: The block-wise attention map comparison between classifier network (first row from each data) and self-teacher network (second row from each data). From above, each data was taken from CUB200, Dogs and MIT67.

classifier network가 self-teacher network에서 의미 있는 증류를 받고 있는지 확인하기 위한 실험
→ 네트워크에서 각 블록의 attention map을 비교, 정성 분석

- [Data set] CUB200, MIT67, Dogs
- 데이터 세트에서 각 블록의 기능 맵에 채널 별 풀링을 적용하여 attention map을 얻음
- 증류를 관찰하기 위해 50번째 epoch에서 맵 선택

(3) Further Analyses on FRSKD

- Qualitative Attention map comparison

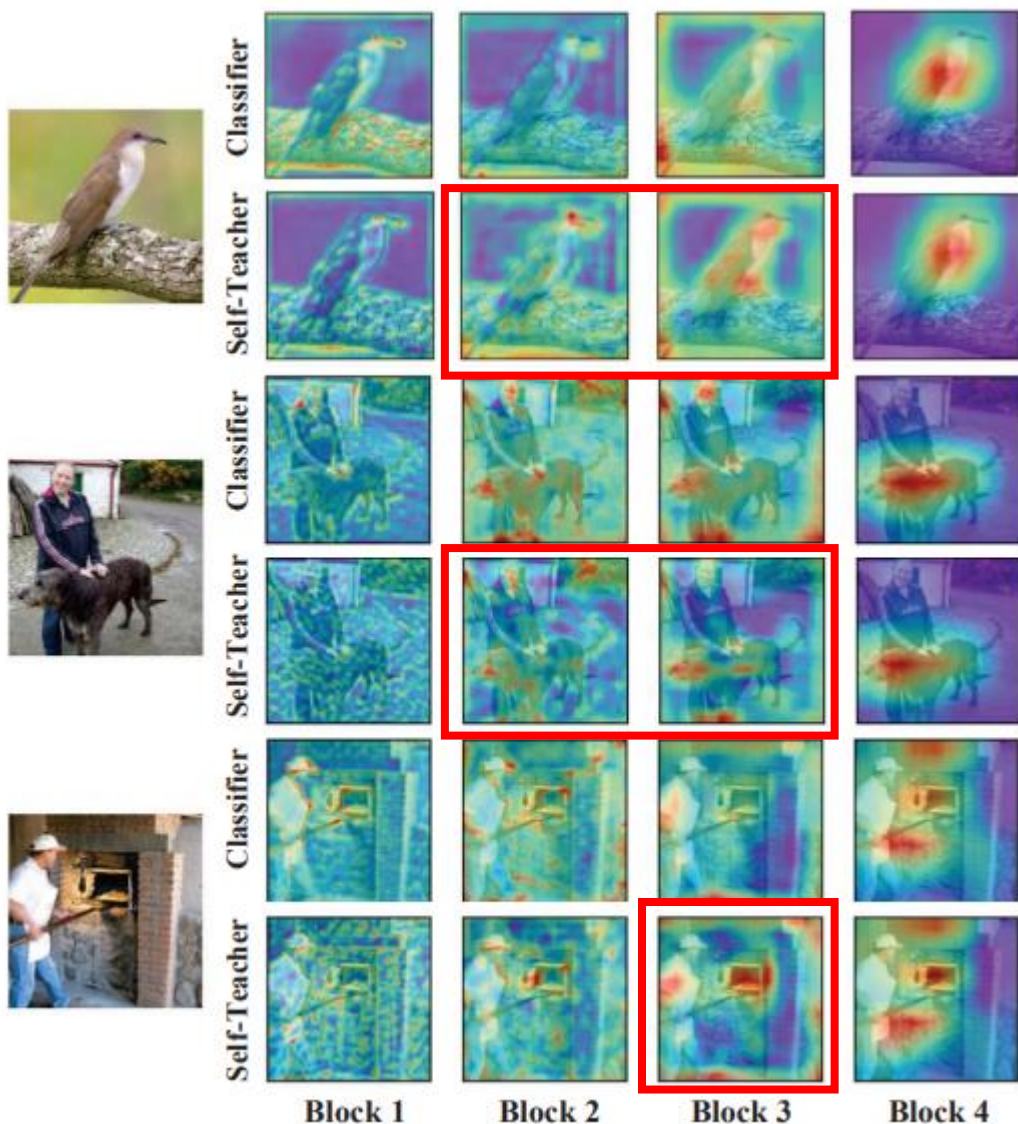


Figure 3: The block-wise attention map comparison between classifier network (first row from each data) and self-teacher network (second row from each data). From above, each data was taken from CUB200, Dogs and MIT67.

[CUB200]

- 블록 2와 3을 보면, Classifier 네트워크는 주요 대상 '새'에 대한 적절한 attention을 포착하지 못했으나 Self-Teacher 네트워크는 집계된 기능을 활용하여 일괄된 주의 맵을 표시함

[Dogs]

- Classifier 네트워크는 주 대상 '개'가 아닌 사람에게 편향되어 있지만 Self-Teacher 네트워크는 상대적으로 주 대상을 집중적으로 잘 가리킴

[MIT67]

- 블록 3의 경우, Classifier 네트워크와 달리 Self-Teacher 네트워크는 '빵'에 집중함 → 데이터의 장면 클래스 '빵집'을 성공적으로 인식할 수 있는 중요한 단서

(3) Further Analyses on FRSKD

- Ablation with the feature distillation methods

Method	CIFAR-100	TinyImageNet	CUB200	MIT67	Dogs	Stanford40
Baseline	73.80 \pm 0.60	54.60 \pm 0.33	51.72 \pm 1.17	55.00 \pm 0.97	63.38 \pm 0.04	42.97 \pm 0.66
Fit+SKD	77.03 \pm 0.05	59.06 \pm 0.20	<u>61.05\pm1.05</u>	<u>57.69\pm0.28</u>	<u>67.50\pm0.32</u>	<u>51.66\pm1.32</u>
OD+SKD	<u>77.12\pm0.09</u>	<u>59.14\pm0.20</u>	57.44 \pm 0.92	54.83 \pm 2.63	66.51 \pm 0.87	49.09 \pm 0.47
FRSKD	77.71\pm0.14	59.61\pm0.31	65.39\pm0.13	61.74\pm0.67	70.77\pm0.20	56.00\pm1.19

Table 5: Performance comparison according to feature distillation method of FRSKD. The feature distillation method of Fit+SKD is based on FitNet [26]; OD+SKD is based on overhaul distillation [8]; and FRSKD is based on attention transfer [36]. ResNet18 is used as classifier network. The best performing model is indicated as boldface. The second-best model is indicated as underline.

[특징 증류 방법]

- Fit+SKD : FitNet 증류 기반
- OD+SKD : Overhaul 증류 기반
- FRSKD : attention transfer 기반

성능 차이 분석을 위해 2가지 비교

(1) 정확한 특징 증류 방법 (FitNet, Overhaul) (2) attention transfer 방법

→ FRSKD의 attention transfer이 정확한 특징 증류를 통합한 것보다 다양한 데이터 셋에서 더 나은 정확도를 보여줌

(3) Further Analyses on FRSKD

– Structure of self-teacher network

Type	#channel	Parameters	FLOPs	CIFAR-100
BiFPN	128	$\times 0.30$	$\times 0.67$	72.64 ± 0.12
BiFPN	256	$\times 0.97$	$\times 2.38$	73.54 ± 0.41
BiFPNc	128	$\times 0.19$	$\times 0.21$	71.70 ± 0.19
BiFPNc	256	$\times 0.59$	$\times 0.68$	73.27 ± 0.45

Table 6: Performance and efficiency comparison between the self-teacher network structures. WRN-16-2 is used as classifier network on CIFAR-100. BiFPN is a structure with the same channel dimension for each layer, and BiFPNc is a structure in which the channel dimension is different depending on the depth of the layer as proposed in Section 3.1. #channel is the channel dimension of the deepest layer, i.e. L_n, P_n, T_n , of the self-teacher network. #channel of BiFPNc depends on the channel width parameter, w . Parameters and FLOPs are the ratio of those of the classifier network.

[self-teacher 네트워크의 효율성을 보여주기 위해 다양한 네트워크 구조를 실험]

- BiFPN(채널 256)은 최고의 성능을 달성하지만, 매개변수 및 FLOP는 classifier 네트워크보다 더 크거나 유사
- 효율성 측면에서 BiFPN은 매개변수 크기가 크기 때문에 self-teacher 네트워크에 적합하지 않지만 BiFPNc는 훨씬 적은 계산으로 BiFPN과 호환되는 성능을 보여줌

→ FRSKD는 classifier 네트워크를 중복으로 사용하는 데이터 증대 기반 자가 지식 증류 방법보다 효율적

(3) Further Analyses on FRSKD

- Compare to knowledge distillation

Method	CIFAR-100	TinyImageNet	CUB200	MIT67	Dogs	Stanford40
Baseline	73.80 \pm 0.60	54.60 \pm 0.33	51.72 \pm 1.17	55.00 \pm 0.97	63.38 \pm 0.04	42.97 \pm 0.66
FitNet	76.65 \pm 0.25	59.38 \pm 0.10	58.97 \pm 0.07	59.15 \pm 0.41	67.18 \pm 0.10	46.64 \pm 0.24
ATT	<u>77.16\pm0.15</u>	59.83\pm0.28	<u>59.21\pm0.34</u>	<u>59.33\pm0.22</u>	<u>67.54\pm0.18</u>	47.04 \pm 0.17
Overhaul	74.59 \pm 0.32	59.50 \pm 0.09	58.82 \pm 0.12	58.81 \pm 0.58	66.43 \pm 0.08	<u>47.06\pm0.26</u>
FRSKD	77.71\pm0.14	<u>59.61\pm0.31</u>	65.39\pm0.13	61.74\pm0.67	70.77\pm0.20	56.00\pm1.19

Table 7: Performance comparison on knowledge distillation. ResNet18 is used as classifier network. The best performing model is indicated as boldface. The second-best model is indicated as underline.

[지식 증류에 대한 성능 비교 실험]

- 비교 대상 : **Baseline**(사전 훈련된 teacher 네트워크O), FitNet과 overhaul 증류, attention transfer 방법
- 교사 네트워크 : 사전 훈련된 ResNet34
- 학생 네트워크 : 훈련되지 않은 ResNet18
- 각 지식 증류 방법은 특징 증류와 소프트 라벨 증류를 활용

→ **FRSKD**는 대부분의 데이터 세트에서, 사전 훈련된 교사 네트워크를 사용하여 실험된 지식 증류 방법보다 성능이 좋다.

(3) Further Analyses on FRSKD

- Training with data augmentation

Method	CIFAR-100	TinyImageNet	CUB200	MIT67	Dogs	Stanford40
Baseline	73.80 ± 0.60	54.60 ± 0.33	51.72 ± 1.17	55.00 ± 0.97	66.38 ± 0.04	42.97 ± 0.66
Mixup	76.26 ± 0.41	56.28 ± 0.24	57.60 ± 0.42	56.77 ± 1.45	65.96 ± 0.03	47.15 ± 0.60
FRSKD + Mixup	78.74 ± 0.19	<u>60.30 ± 0.38</u>	67.98 ± 0.58	<u>62.11 ± 0.81</u>	<u>71.64 ± 0.29</u>	56.50 ± 0.36
CutMix	<u>79.23 ± 0.23</u>	58.97 ± 0.29	51.54 ± 1.12	60.87 ± 0.30	67.71 ± 0.14	46.90 ± 0.29
FRSKD + CutMix	80.49 ± 0.05	61.92 ± 0.11	<u>65.92 ± 0.59</u>	66.19 ± 0.49	72.81 ± 0.23	<u>55.75 ± 0.43</u>

Table 8: Performance of data augmentation method with FRSKD. ResNet18 is used as classifier network. The best performing model is indicated as boldface. The second-best model is indicated as underline.

[FRSKD를 사용한 데이터 증대 방법의 성능 실험]

- Mixup : 두 이미지와 레이블 사이에 볼록한 조합 사용
- Cutmix : 한 쌍의 이미지와 레이블 혼합

→ FRSKD는 데이터 증대와 함께 사용될 때 성능이 크게 향상되었다.

3. 결론

Conclusion

- 하향식 및 상향식 경로를 사용하여 자기 지식 증류를 위한 특수 신경망 구조를 제시
- classifier network에 refined feature-maps와 해당 soft label을 제공할 것으로 예상된다.
- 채널 차원(channel dimensions)을 변경하여 기능 맵 미세 조정을 유지하면서 매개변수를 줄임
- FRSKD는 실험을 통해 분류 및 의미분할의 비전 작업에서 자기 지식 증류를 적용할 수 있음을 보인다.
- FRSKD는 성능이 크게 향상됨을 보여준다.

References

[Refine Myself by Teaching Myself : Feature Refinement via Self-Knowledge Distillation]

Knowledge distillation 설명과 이해 <https://light-tree.tistory.com/196>

컨볼루션 신경망 <https://sungjk.github.io/2017/04/27/Ch6-convolutional-nn.html>

CNN(Convolution Neural Network)이란? <https://22-22.tistory.com/26>

신경망 Part3 (Neural Networks) <https://taeu.github.io/cs231n/deeplearning-cs231n-Neural-Networks-3/>

학습에 영향을 주는 요소 <https://movefast.tistory.com/297>

Backpropagation (역전파) <https://m.blog.naver.com/PostView.naver?isHttpsRedirect=true&blogId=laonple&logNo=220507299181>

텐서플로우로 배우는 딥러닝 <https://www.slideshare.net/w0ong/ss-82372826>