Knowledge Distillation

Index

Knowledge Distillation이란?

KD는 2014년에 힌튼이 처음으로 제시한 개념으로, 작은 네트워크로 좋은 성능을 내기 위해서 고안된 방법입니다.

기본적인 학습 방법 다음과 같습니다.

이미 학습된, 큰 네트워크를 Teacher라고 부르며, 학습되지 않은, 지식을 전달받을 작은 네트워크를 Student라고 부릅니다.

Teacher는 Student보다 상대적으로 더 많은 파라미터를 가지고 있으므로, 학습과정에서 당연히 더 많은 것을 배웠을 겁니다.

그러므로 Student가 이를 잘 모방할 수 있다면 보다 적은 연산량으로도 좋은 성능을 낼 수 있을 것입니다.

이러한 생각 때문에, KD는 초기에 큰 모델로 학습한 다음에 mobile이나, 서비스 배포 측면에서 이점을 갖기 위해서, 경량화 측면에서 각광을 받게 되었습니다.

Knowledge Distillation이란?

그렇다면 KD는 어떻게 진행될까요?

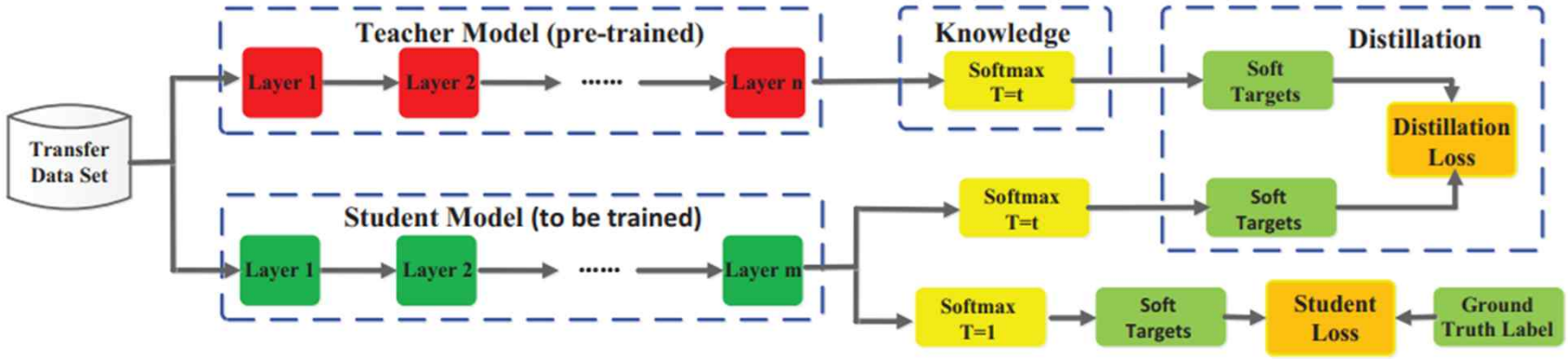
“Teacher의 지식을 Student가 따라하는 것이 목적이다.” 라는 말을 들었을 때 아마 어느정도 예상 하셨겠지만, Vanila KD의 경우에는 단순히 Teacher의 출력값을 Student가 근사하도록 만듭니다.

$$L_{ResD}(z_t, z_s) = \mathcal{L}_R(z_t, z_s) \ ,$$

여기서 얼마나 Teacher의 값을 잘 따라가는 것이 목적인지에 따라 Temperature를 넣어서 계산합니다.

$$p(z_i, T) = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)} \ ,$$

Knowledge Distillation이란?



위에 있는 그림은 초기에 제안된 KD으로, Vanilla KD, Responsed based KD라고도 부릅니다.
Responsed based KD라고 부르는 이유는 target을 Teacher의 Output에만 신경써서 그렇습니다.
사전에 학습된 Teacher Model이 있고, 학습을 진행해야할 Student Model이 있습니다.
Teacher Model의 prediction값과, Student Model의 prediction값을 줄이기 위한 **Distillation Loss**라는 것이 존재하고, 추가적으로 Student가 GT와의 격차를 줄이도록 설계된 **Student Loss**라는 것이 존재하게 됩니다.

Distillation Loss는 Student가 Teacher를 따라가도록 만드는 Loss지만, 사실 Teacher를 단순히 근사하면 성능이 잘 나오지 않습니다.

그림 참조 : Knowledge Distillation: A Survey

Knowledge Distillation이란?

이건 생각해보면 간단한 이유인데, 우리가 이미지넷의 데이터들을 다 학습한다고 해서, 이미지넷의 분포를 정확하게 예측할 수 있는 건 아닙니다.

매우 큰 모델에서, 오랫동안 학습을 진행해도 현재로는 90%의 성능이 나오죠.

마찬가지로, KD도 Student가 관측 데이터들을 기반으로 Teacher의 행동을 예측하는 문제라서, Teacher를 100% 근사하는 건 쉽지 않은 일입니다.

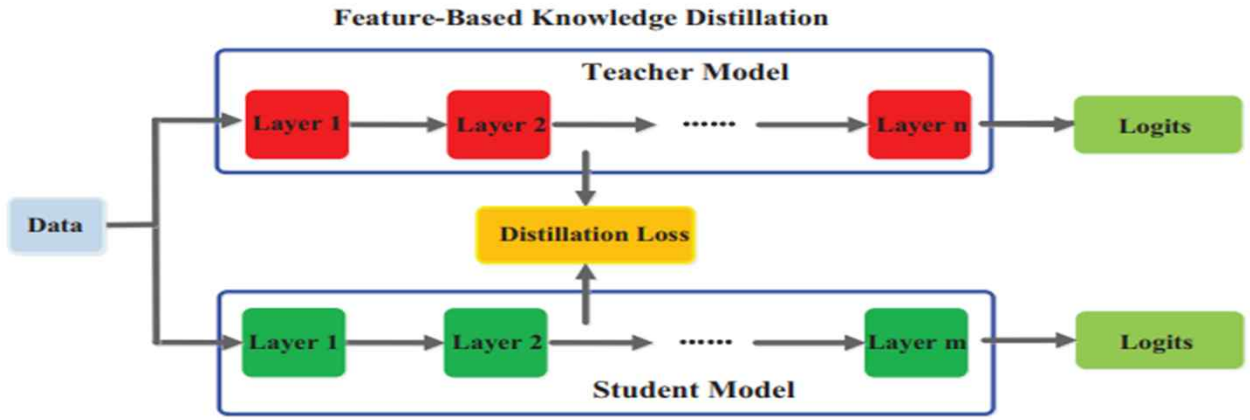
데이터가 많고, 네트워크가 적다면 몰라도, 데이터가 적으며 네트워크가 크다면 KD가 안되는 경우도 많습니다.

네트워크들은 보통 비선형적인 방식으로 구성되기 때문에, 단순히 Weight들의 선형적인 조합으로 관계를 나타낼 수 없습니다.

그러므로 Student가 보다 큰 Teacher의 추상적인 특징정보들을 학습하는 과정에서는 충분한 Capacity가 필요하며, 추상적인 특징정보들을 예측하기 위한 충분한 관측데이터들이 필요합니다.

그림 참조 : Knowledge Distillation: A Survey

Knowledge Distillation이란?



상기의 학습 구조는 Feature Base Knowledge Distillation입니다.

네트워크에서 layer들은 각자 깊이에 따라, Stage에 따라 가진 추상적인 정보량이 달라지게 됩니다.

그렇기 때문에, Model의 Ouptut값에 대해서만 Distillation을 진행하는 것이 아니라,
Layer 혹은 stage별로 나오는 Teacher의 출력값을 Student가 모방하도록 학습하는 방식입니다.

그림 참조 : Knowledge Distillation: A Survey

Knowledge Distillation이란?

또한 KD는 학습 전략에 따라서 Offline, Online, Self-Distillation으로 나눌 수 있습니다.

1) Offline Distillation이란 오프라인에서 작동되는 Distillation입니다. 사전에 학습된 teacher model에서 student 모델로 지식을 옮기는 것을 의미합니다.

1) Online Distillation의 경우에는 잘 학습된 좋은 Teacher 모델이 없는 경우, 여러개의 네트워크들이 서로 Teacher와 Student가 되서 서로 학습을 도와주는 방식입니다. 앙상블과 유사합니다.

1) Self-Distillation은 Teacher와 Student가 동일한 네트워크를 사용하는데, 네트워크의 더 깊은 layer에서 얕은 layer로 지식을 증류하는 등의 방식들이 있습니다.

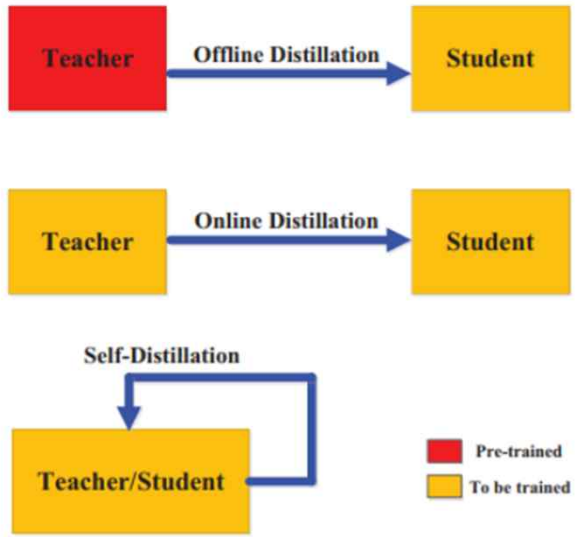


그림 참조 : Knowledge Distillation: A Survey

Does Knowledge Distillation Really Work?

2021 NIPS
Google Research

Does Knowledge Distillation Really Work?

한 줄 요약 :
“Teacher를 잘 모방하는 Student가, 모두 좋은 Student는 아닙니다.”
(Teacher를 잘 모방하는 Student는 Teacher와 유사한 성능을 보여주는 편이지만, 좋은 성능을 낸다는 것을 보장하지는 않습니다.)

KD는 Teacher의 Model을 Emulate하기 위해서, 소규모의 Student Model들을 훈련하는데 널리 사용되는 기술입니다.

KD를 많은 사람들이 사용하지만, 작동 원리에 대해서는 사람들이 알고 있는 것과 많은 괴리가 있습니다.
이 논문에서는 KD가 정말 사람들이 알고 있는 것과 동일하게 작동하는지를 주로 검증하려고 노력하였습니다.

Does Knowledge Distillation Really Work?

보통 KD는 Student의 Generalization 성능을 향상시킨다고 알려져 있습니다.
왜 그러냐면,
Network는 Large하고 Deep할수록 General한 representation을 배우기에 좋습니다.
이는 반대로 말하면, Network가 작아진다면, General한 representation를 배우기는 어렵다는 의미입니다.
그러므로, Student가 Teacher를 잘 모방한다면, Student는 보다는 General한 Representation을 학습할 수 있는 것이죠.
그렇기에 잘 학습된 Student는 Generalization 성능이 향상되는 것이지요.

하지만, Network가 작다면, 혹은 훈련 데이터가 적다면, Teacher를 모방하는 쉽지는 않습니다.
Network가 작다면 추상적인 특징정보를 학습하기에는 파라미터가 너무 부족할 수도 있고,
훈련 데이터가 적다면 관측횟수가 부족하기 때문에 Teacher를 모방하기 어려워지겠죠.
하지만, 작은 네트워크라도 훈련 데이터들을 암기하기에는 충분한 capacity를 가질 수 있습니다.
그렇기 때문에 Teacher를 잘 모방했다고 생각했는데 사실은 단순히 훈련 데이터들에 대한, Teacher의 예측값을 암기한 것일수도 있습니다.

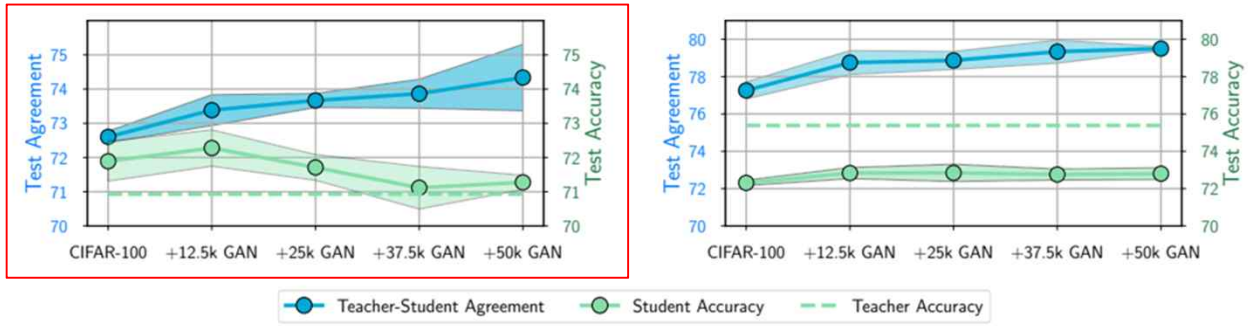
그럼 여기까지는 이해가 되는데, Student가 충분히 커서 Teacher를 모방할 수 있는 Capacity가 있음에도 불구하고 생각보다 많은 경우 Teacher를 모방하는데 실패하는 이유는 무엇일까요? 그리고 왜 이러한 문제들이 발생하는 걸까요?

Does Knowledge Distillation Really Work?

이러한 궁금증을 해결해보고자 다음과 같이 실험 세팅을 준비했습니다.

Teacher와 Student를 동일하게 Resnet-56으로 구성합니다.
그리고 해당 네트워크들을 사용해서 Self Distillation과, Ensemble Distillation을 진행합니다.

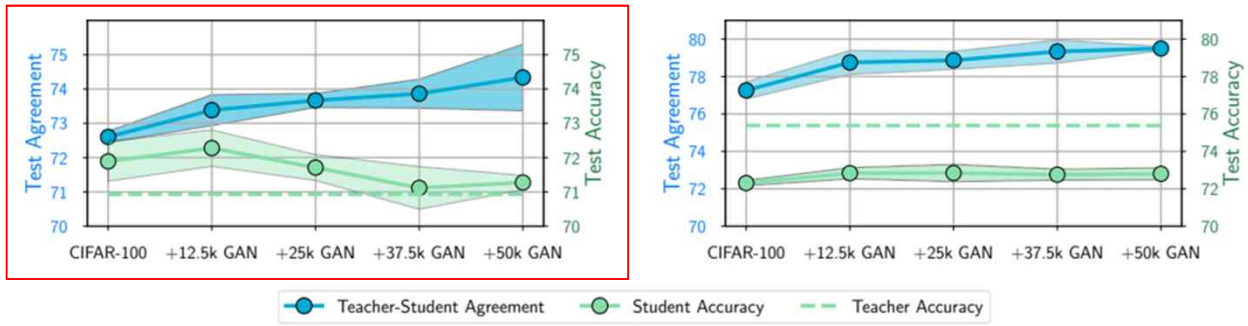
Self Distillation은 말 그대로 Teacher와 Student가 동일한 경우의 Distillation이며,
Ensemble Distillation은 3개의 Teacher가 출력한 Ensemble 값을 Student가 전이받는 것을 의미합니다.



(a) Self-distillation (b) Ensemble distillation

Does Knowledge Distillation Really Work?

Self distillation 실험을 먼저 보시겠습니다.
CIFAR-100에서 Teacher와 Student의 성능을 3번씩 측정하는데, GAN으로 생성된 데이터를 늘리면서 Generalization 성능을 측정한 것입니다.(데이터가 바뀌어도 잘 모방할 수 있는지!)
Teacher의 Accuracy는 초록색 실선으로 71%이고, 데이터가 늘어남에 따라서 Teacher의 Feature를 모방하는 fidelity(Teacher와 Student간 Feature의 유사도)는 높아지게 되지만, 오히려 Student의 Accuracy는 줄어드는 모습을 보입니다.
즉, fidelity가 Accuracy와 반대되는 경향을 보인 겁니다.
Student는 Teacher를 잘 모방했는데, 성능은 떨어지는 겁니다.

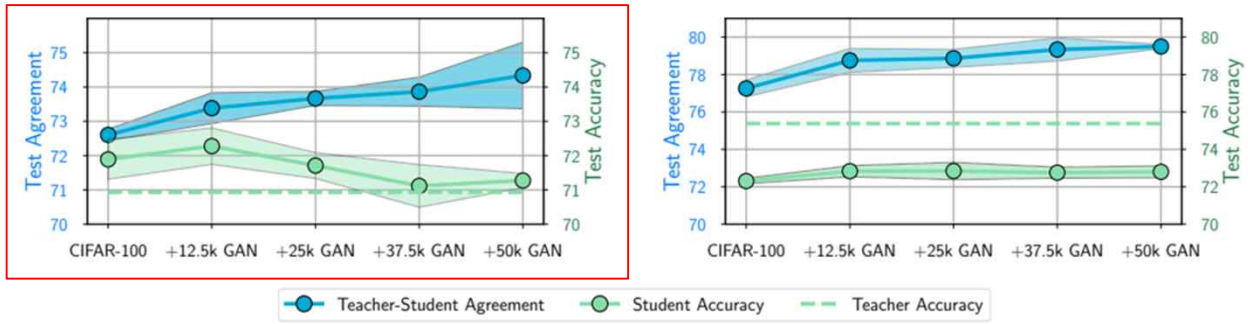


(a) Self-distillation (b) Ensemble distillation

Does Knowledge Distillation Really Work?

Self-Distillation에서는, Student가 Teacher를 모방하는 것에 실패해도 역설적으로 그 성능이 향상된다고 널리 알려져 있습니다.
하지만, fidelitry가 향상됨에 따라서 Generalization(매 실험 성능이 유사하게 나오는지) 성능은 오히려 향상된다는 것을 저희가 그래프를 통해서 알 수가 있습니다.

무슨 말이냐면, 초기에 fidelity가 낮을 때, Student의 성능이 Teacher보다 높았지만, fidelity가 높아짐에 따라서, 매번 실험을 진행해도 Teacher의 성능을 유사하게 모방할 수 있었다는 의미입니다.
즉! Fidelity가 높다면, Teacher와 유사한 성능을 보여줄 가능성이 높아지게 되는 것이죠.
즉, Teacher보다 좋은 성능을 내고 싶다면 fidelity를 높이면 안된다는 의미입니다.
그렇기 때문에, Student가 Teacher를 모방에 실패해도 역설적으로 성능이 향상되는 것입니다.



(a) Self-distillation (b) Ensemble distillation

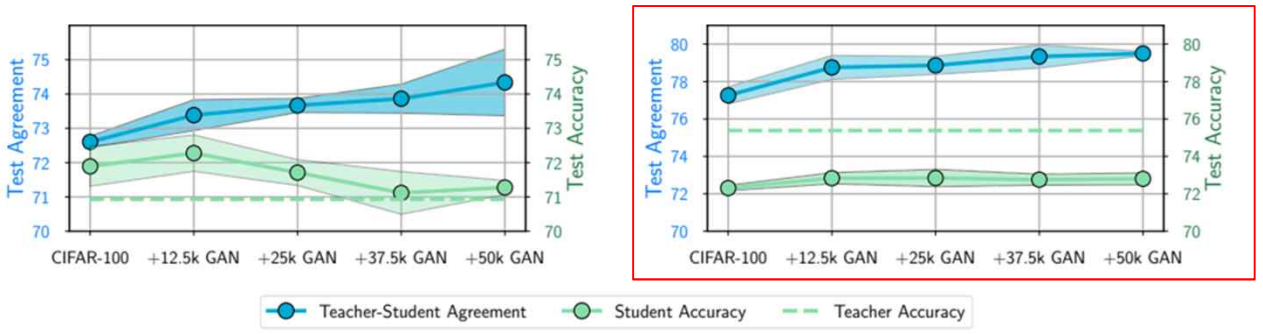
Does Knowledge Distillation Really Work?

다음으로는, Ensemble Distillation 실험을 보도록 하겠습니다.

Student는 Resnet-56으로, Teacher는 앙상블한 모델(Resnet-56 3개를 앙상블)입니다.
Teacher가 큰 모델(Student와 동일한 모델 3개)일 때, fidelity(Teacher와 Student간 Feature의 유사도) 가 높을수록 Genearlization 성능이 오르는 것을 알 수 있습니다.

Teacher의 성능은 76%에 가까운 성능인데 반해서, Student의 성능은 그에 크게 못 미치는 72%에 근접한 성능이라는 것을 알 수 있습니다.

음.. 그럼, Self Distillation에서는 74%정도만 모방해도 성능이 비슷했는데, 왜 Ensemble 모델에서는 80%모방했는데 성능이 크게 낮을까요?



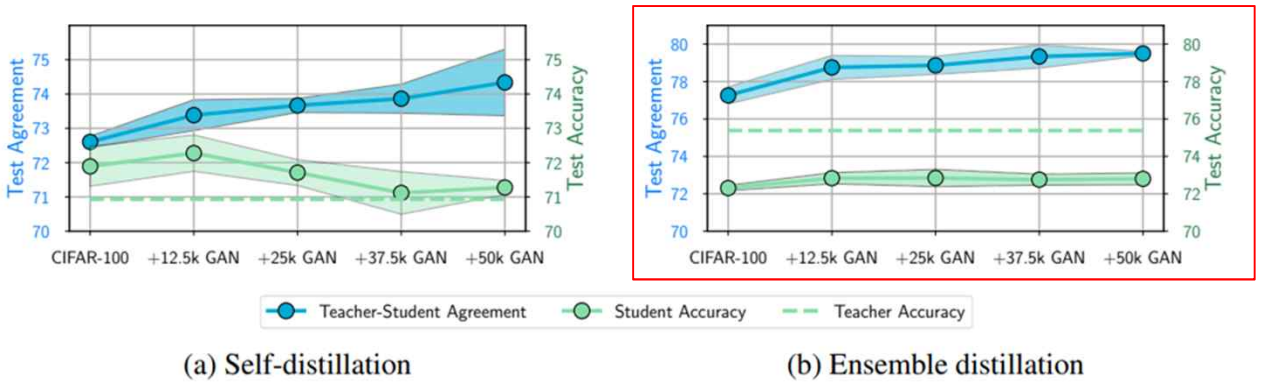
(a) Self-distillation (b) Ensemble distillation

Does Knowledge Distillation Really Work?

그 이유는, fidelity가 성능을 보장하는 지표가 아니기 때문입니다. 그동안 사람들이 잘못 알고 있었던 거지요.

Fidelity가 높다는 것은 훈련 데이터 안에서, Student가 Teacher의 출력값들을 외워버린 것인지, 정말로 Teacher를 잘 학습해서 General한 Representation을 학습한 것인지 구분해서 알려주는 지표가 아닙니다.

단순히, 학습 데이터 상에서 Teacher를 잘 모방했다는 지표이고, Teacher를 잘 모방한 경우에는 Teacher와 유사한 성능을 낼 수 있다는 의미에 불과합니다.



(a) Self-distillation (b) Ensemble distillation

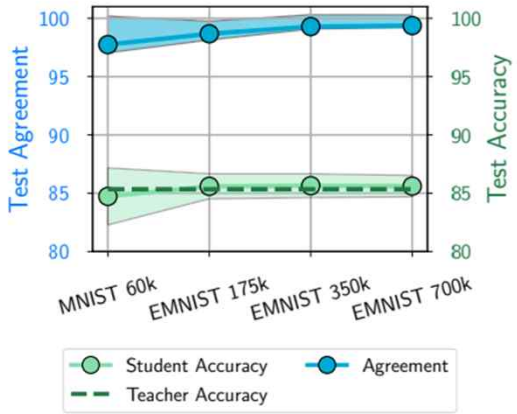
Does Knowledge Distillation Really Work?

상기 실험들을 통해서 Teacher와 Student의 fidelity는 80%정도로
Student Network가 Teacher를 모방하려는 능력(fidelity)을 얻는것이 놀라울정도로 어렵다는 것을 알 수 있습니다.

그러면, 대규모 네트워크를 KD 할 때, 높은 fidelity를 갖는것이 왜 이렇게 힘들까요?

이에 대해서 알아 보기 위해 간단한 Self Distillation 실험을 준비해 보겠습니다.
Teacher와 Student를 동일하게 LeNet-5로 설정합니다.

100 epoch 동안 MNIST + EMNIST(MNIST와 유사한 추가 데이터셋) 데이터 셋으로 Teacher를 훈련해서 84~86%의 Accuracy를 얻습니다.
그 후에, Teacher와 Student는 99%이상 일치하며 성능 또한 거의 동일하다는 것을 보여주게 됩니다.



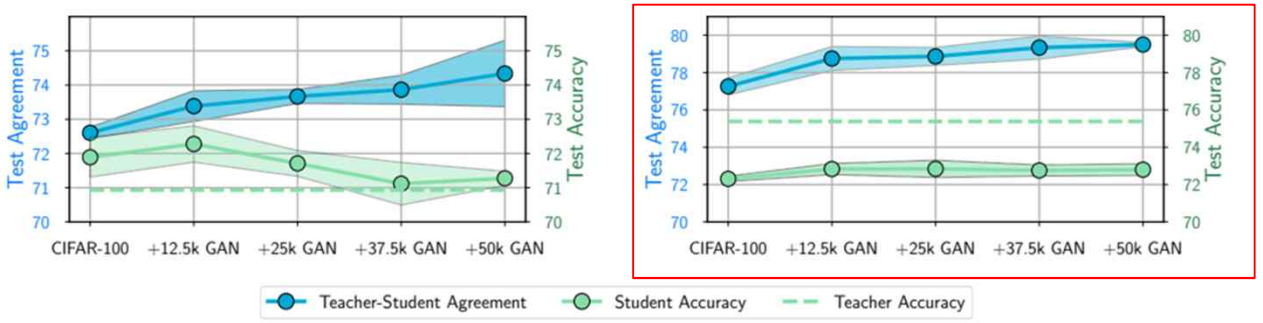
Does Knowledge Distillation Really Work?

이제, KD에서 모델을 조금 더 키워서 실험을 진행해 보겠습니다.
Teacher와 Student를 동일하게 ResNet-56 모델로 설정합니다.

CIFAR-100 데이터셋에서 학습을 진행하는데, CIFAR-100은 증류 데이터를 보강하기 위한 데이터가 따로 없으므로, SN-GAN으로 CIFAR-100에 대해서 데이터를 추가적으로 증강해서 사용하도록 합니다.

데이터를 추가함에 따라 fidelity는 80%까지 증가하지만, 매우 미미하게 증가했을 뿐이며, LeNet 실험의 fidelity인 99%에는 한참 떨어집니다.

즉, 모델의 크기에 비해서 학습데이터가 부족하면, 모방 하기가 어렵다는 것을 의미합니다.
그리고 Self-Distillation 실험과 Ensemble 실험을 통해서 fidelity가 성능과는 영향이 크게 없다는 것도 알 수가 있는데, fidelity가 높아져도 성능은 제자리 걸음이거든요.



(a) Self-distillation (b) Ensemble distillation

Does Knowledge Distillation Really Work?

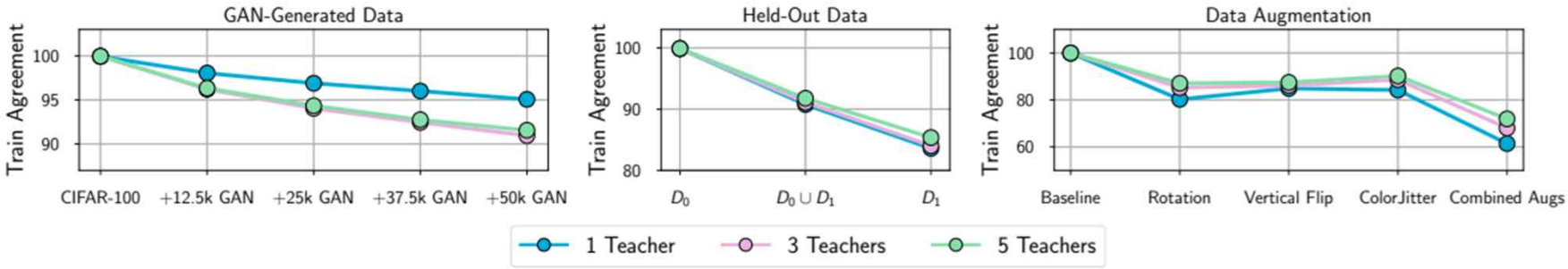
그럼 여기서 잠깐 !
잡다한 지식 하나 알려드리려고 합니다..

Furlanello 논문에 따르면, KD에서 증류 절차에 실패를 해야,(fidelity가 낮아야) 학생이 교사의 성능을 뛰어 넘을 수 있음을 보였습니다.
즉, Student의 fidelity가 높다는 가정 하에서 Student는 절대로 Teacher의 성능을 능가할 수 없습니다.

그렇기 때문에, Student가 Teacher의 성능을 따라가려면 fidelity가 중요하지만, Student가 Teacher의 성능을 뛰어 넘기 위해서는 fidelity에 집중하면 안됩니다.

Does Knowledge Distillation Really Work?

KD를 진행할 때, 광범위한 Data Augmentation이나 GAN의 사용은 fidelity를 낮추게 됩니다.



즉, Self-Distillation에서 Student 모델의 성능을 더 올리기 위해서는 GAN이나 Augmentation들을 많이 진행할수록 좋겠죠.
또한, Batch Norm 대신에 Layer Norm을 사용해야 하는데,
그 이유는 Teacher Model과 Student Model이 정확히 동일한 weight를 가지고 있다고 가정해도, 누적된 통계의 차이로 인해서 Teacher와 Student가 서로 다른예측을 진행할 수 있기 때문입니다.
또한, 학생이 Teacher의 Weight에서 시작하든 From scratch로부터 학습하든 KD에서 큰 성능의 차이가 나지 않는 것을 발견하였습니다.

이 외에도 다양한 실험들이 있지만, 크게 중요한 내용은 없으므로 결론으로 넘어가겠습니다.

Does Knowledge Distillation Really Work?

결론 :

fidelity가 높다고 accuracy가 높은 것은 아니고, fidelity만 잘 따라가면 되!라는 기존의 관념들은 너무 naive한 생각이었다는 것.

학습 데이터에 대해서 fidelity가 높다고 test 데이터에서 fidelity가 높다는 것을 의미하지 않습니다.

1. Fidelity는 데이터의 질과 양에 영향을 많이 받습니다.
2. Student Model의 Capacity와 fidelity의 연관성은 거의 없습니다.
3. Network의 아키텍처에 무관하게 나타나는 증상들입니다.
4. 데이터셋에 상관없이 무관하게 나타나는 증상들입니다.
5. 이미지 뿐만 아니라 텍스트 데이터에서도 같은 증상이 나타납니다.
6. Fidelity가 높을 수록 Teacher와 Student의 성능이 유사해지는 경향이 있음.
7. 하지만, Fidelity가 높다고 좋은 모델인 것은 아님!
8. Student가 Teacher의 training data에 대한 Representation을 학습한다고, Teacher의 Test data에 대한 Representation도 학습한 것이 아님!

Solving Imagenet

2022 arxiv

Alibaba Group

Solving Imagenet

결론 : “KD는 안정적인 학습을 보장하며, Augmentation을 사용하면 부정확한 라벨을 정제해서 성능 향상에도 도움이 된다.”

해당 논문에서는 대부분의 모델에서 하이퍼 파라미터에 무관하게, SOTA 성능을 찍을 수 있는 방법을 제시합니다.

하이퍼 파라미터는 매우 중요한데, 모델별로 3.3%에서 최대 6%까지도 성능 차이가 발생할 수 있습니다.

예를 들어서 ResNet50의 하이퍼파라미터로 EfficientNetV2를 학습하면 3.3%의 성능하락을 발견할 수 있고,

ViT-L의 경우에는 코드에 따라 6%까지도 성능 하락이 발생 할 수 있습니다.

Solving Imagenet

모델에 따른 학습 방법들은 backbone에 따라 다음과 같이 4가지의 분류로 나눌 수 있습니다.

1. **ResNet like Model** : TResNet이나, SEResNet, Renet-D등 다양한 ResNet 기반의 모델들은, 대부분의 traning scheme에 상관없이 좋은 성능을 보여주는 것으로 알려져 있습니다.
2. **Mobile oriented Model** : 해당 모델들의 경우에는 depth-wise convolution과, 효율적인 cpu 지향적 구조에 전적으로 의존하고 있습니다. 그리고 이러한 방법들은 보통 RMSProp Optimizer를 사용하거나, waterfall learning rate schedular를 사용하기도 하고, EMA를 사용하기도 합니다.
3. **Transformer based Model** : Transformer의 경우에는 학습이 불안정한 편에 속하고, 기본적으로 많은 에폭(1000 epochs)을 필요로 하며, 많은 데이터가 필요합니다.
4. **MLP only Model** : Transformer와 마찬가지입니다.

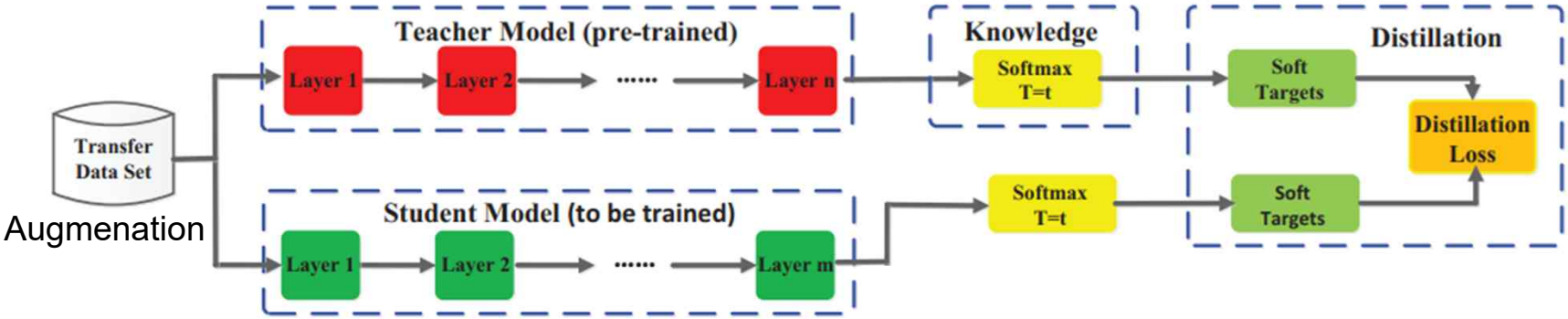
최근 대부분의 학습 방법들은 ImageNet-21K에서 ImageNet-1K로 transfer learning을 진행했습니다.

하지만 이는 사전에 Imagenet-21K에서 학습된 모델이 있어야 하고, 모델의 구조가 동일해야 하며, 학습에 오랜 시간이 걸리는 문제들이 있었습니다.

이 논문에서는 USI(Unified Scheme for Imagenet)이라는 모델에 관계없이 좋은 성능을 얻을 수 있는, 학습 방법을 제안합니다.

Solving Imagenet

학습 방법은 설명할 게 있나? 쉬울 정도로 간단한데, Teacher에 Augmentation을 주고, Distillation Loss를 줄이도록 학습한게 전붑니다.



보시면, Augmentation 된 데이터에 대해 Teacher가 예측한 결과를 Student가 모방하도록 Distillation Loss를 사용한게 전붑니다.

Augmentation이 가장 중요한 부분인데, Student가 Teacher를 잘 모방한다면 (fidelity가 높다면) Teacher 성능을 절대로 뛰어넘을 수 없습니다. Augmentation된 데이터는 정제된 라벨로서, 라벨 노이즈를 제거해줬기 때문에 성능이 오르는 것입니다.

Solving Imagenet

아래 그림에서 볼 수 있듯, USI는 대부분의 모델에서 Architecture에 상관없이 SOTA를 달성합니다.

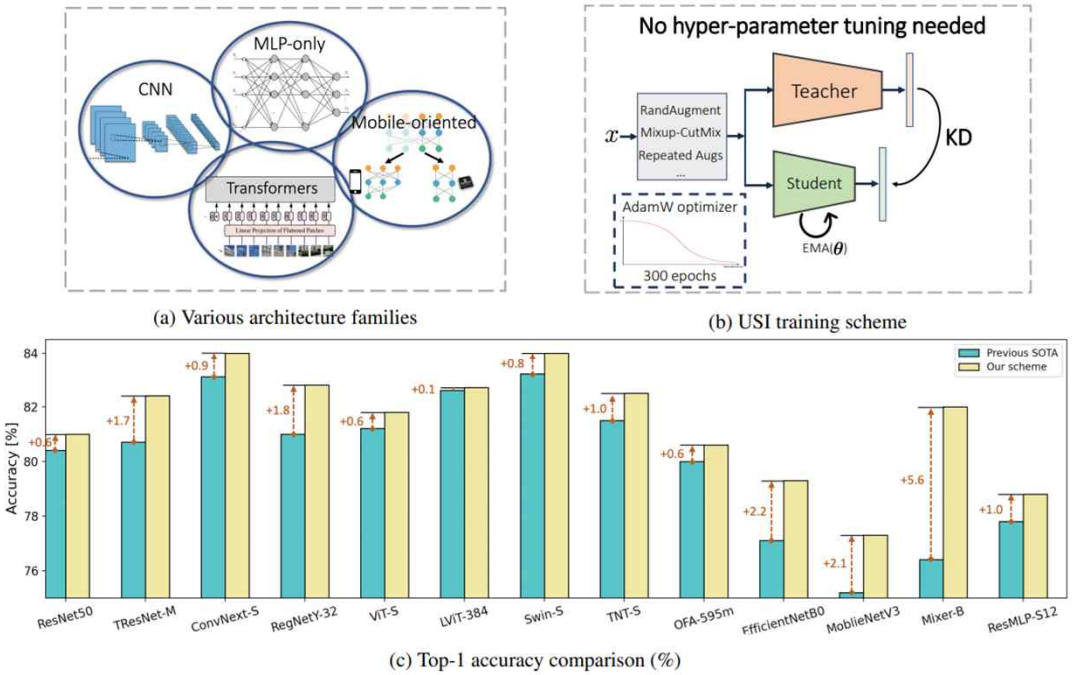


Figure 1: **Our unified training scheme for ImageNet, USI**. With USI, we can train any backbone to top results on ImageNet, without any hyper-parameter tuning or adjustments per architecture.

Solving Imagenet

처음에 저자들이 이 논문을 작성할 때, 주목한 점은 KD가 backbone에 상관없이 어떤 모델에서도 안정적으로 동작한다는 점이었습니다.

그러면 KD는 왜 모델에 상관없이 잘 작동하지?에 대해서 고민하다보니, KD는 기존 Label에서는 없던 정보들이, Teacher Model로부터 추가되었다는 점입니다.

이는 Teacher가 만들어낸 Soft Label이 Hard Label(GT)보다 더 많은 정보를 포함하고 있고, 라벨의 오류 또한 보정하며, 클래스들간의 상관관계 또한 포함하기 때문입니다.

그리고 KD 과정에서 성능이 올랐던 원인이 Label을 Refine하는 과정에서 발생한 것이고, 이를 사용해서 Student 모델이 기존 모델보다 좋은 성능을 달성하는데 큰 기여를 하게 됩니다.

이렇게 만들어진 KD는 augmentation과 매우 궁합이 좋고, label Noise를 제거할 수 있고, 더 적은 training trick과, training epoch(maximum 300epoch)에서 작동되며 심지어 regularization 성능마저 좋아지게 됩니다.

일반적으로 전이학습이 아닌, from scratch로부터의 학습은 일반적으로 더 어렵고, 더 높은 learning rate와, 강력한 정규화 및 더 많은 epoch을 통해 학습을 진행해야 합니다. 이는 모델들 간 학습과정에서 하이퍼파라미터가 상당부분 달라지는 원인이었습니다.

Solving Imagenet

우선, KD가 가진 Label을 Refine하는 과정의 장점을 설명하기 위해 아래 그림을 첨부합니다.

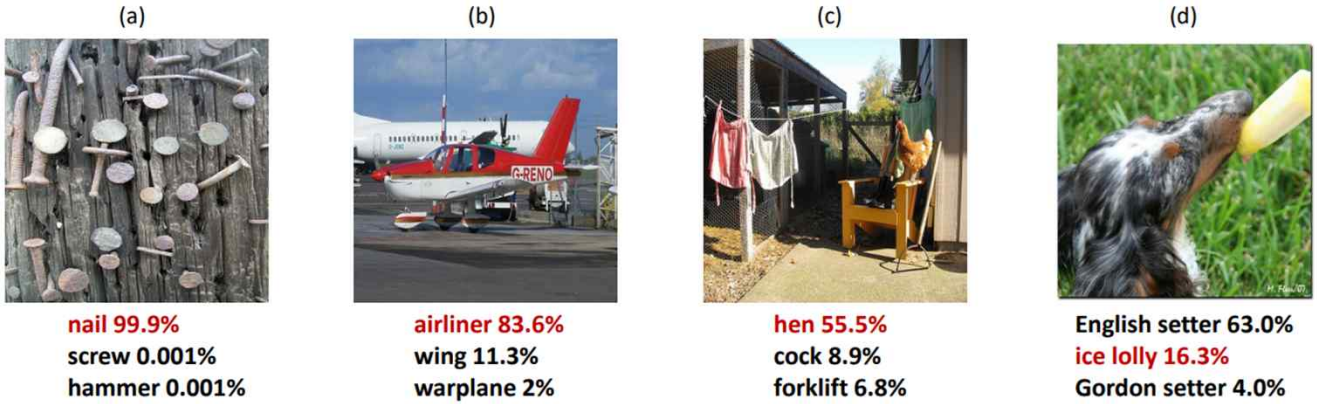


Figure 2: **Examples for teacher predictions**. ImageNet ground-truth labels are highlighted in red. Unlike the ground-truth, the teacher predictions account for similarities and correlations between classes, objects' saliency, pictures with several objects, and more. The teacher predictions would also better represent the content of images under strong augmentations.

Solving Imagenet

Solving Imagenet에서는 다음과 같은 하이퍼 파라미터에서 다양한 모델들을 가지고 실험을 진행했습니다.

주의 깊게 볼 건, epoch이 상대적으로 적고, Augmentation으로 mixup-cutmix를 사용하며 Teacher는 TResNet-L을 썼다? 말고는 크게 볼 건 없는거 같습니다.

이 학습 방법은 backbone에 무관하게, batch_size에 무관하게, Learnign rate에 적은 영향을 받습니다.

또한, Batch size가 커짐에 따라 Learning rate의 조절이나 큰 Batch size를 튜닝하기 위한 전용 Optimizer를 사용하지 않아도 됩니다.

Procedure	Value
Train resolution	224
Test resolution	224
Epochs	300
Optimizer	AdamW
Weight decay	2e-2
Learning rate	2e-3
LR decay	One-cycle policy
Mixup alpha	0.8
Cutmix alpha	1.0
Augmentations	Rand-augment (7/0.5)
Test crop ratio	0.95
Repeated Augs	3
Base loss	Cross entropy
KD loss	KL-divergence
KD temperature	1
α_{kd}	5
Teacher	TResNet-L
Batch size	512 to 3456

Solving Imagenet

제안된 방법은 batch size에 상관없이 좋은 성능을 보여주게 됩니다.

이는 KD를 사용한 학습이 기존의 다른 학습 방법들에 비해서 안정적이라는 것을 의미합니다.

이과 관련해서는 Soft Label이 Hard Label에 비해서 학습을 안정적으로 진행할 수 있기 때문입니다.

Soft된 Label은 정제된 Label이므로, Hard Label에 비해서 잘못된 정보가 적습니다.

Batch Size	Top1 Acc. [%]	Training speed [img/sec]
512	82.3	1100
1024	82.5	1900
2048	82.3	3000
2752	82.4	4100
3456	82.4	4300
3456		4900 (no-KD reference)

Solving Imagenet

또한, 교사와 학생은 모델(CNN, Transformer)에 관계없이 Knowledge Distillation이 잘 적용되는 것을 관찰할 수 있었습니다.

또한 추가적으로 재미있는 결과인데, Teacher가 Student보다 작아도, 커도 관계없이 모두 성능이 오른다는 것을 의미 합니다.

Student	Student Type	Teacher	Teacher Type	Top1 Acc. [%]
ResNet50 (80.4%)	CNN	TResNet-L	CNN	81.0
		Volo-d1 (81.5%)	Transformer	80.9
LeViT384 (82.6%)	Transformer	TResNet-L (84.2%)	CNN	82.7
		Volo-d1 (81.5%)	Transformer	82.7
				(84.2%)

Solving Imagenet

USI를 적용해서 성능을 측정한 결과표는 다음과 같습니다.

Model Type	Model Name	USI Top1 Acc. [%]	Comparable Training Scheme			
			Top1 Acc. [%]	Epochs	KD	Additional Details
CNN	ResNet50	81.0	80.4 [42]	600	no	ResNet-strike-back, A1 config
			80.2 [47]	300	yes	Relabel-based KD
	TResNet-M	82.4	80.7 [30]	300	no	
	ConvNext-S	84.0	83.1 [25]	300	no	
	RegNetY-32	82.8	81.0 [28]	100	no	
Transformer	ViT-S	81.8	79.8 [8]	300	no	Original paper scheme
			81.2 [38]	1000	yes	DeiT scheme
	LeViT-384	82.7	82.6 [11]	1000	yes	DeiT scheme
	Swin-S	84.0	83.2 [24]	300	no	
	TNT-S	82.5	81.5 [12]	300	no	
Mobile-Oriented CNN	OFA-595m	80.6	80.0 [3]	255	yes	KD from super network
	EfficientnetB0	79.3	77.1 [34]	not stated	no	
	MobileNetV3	77.3	75.2 [17]	not stated	no	
MLP-Only	Mixer-B	82.0	76.4 [36]	255	no	
	ResMLP-S12	78.8	77.8 [37]	400	yes	

Table 2: Comparison of our proposed scheme, USI, to previous state-of-the-art results. Train and test resolution - 224.

Solving Imagenet

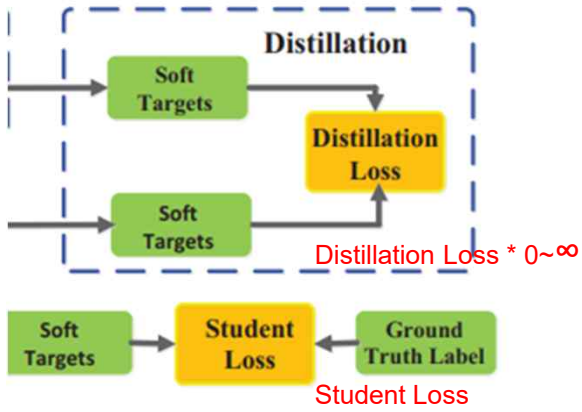
그리고 추가로, vanilla KD에서 학습 할 때, Soft Label만 가지고 Distillation Loss를 계산해도 학습이 잘 되나? 에대한 실험입니다.

hard label은 학습에 영향을 미치지 않습니다.

이는 teacher의 prediction value가 GT보다 좋다는 것을 의미합니다.

KD relative weight, α_{kd}	Top1 Acc. [%]
0 (no KD loss)	76.2
1	80.8
5	82.7
10	82.6
20	82.7
∞ (no CE loss)	82.7

Distillation Loss * ∞ + Student Loss = Distillation Loss



Solving Imagenet

구현이 매우 간단한게 장점입니다.

Cross entropy loss에, Teacher 모델의 probability와 student 모델의 probability간 KL loss만 추가해주면 끝납니다.

```
with amp_autocast():
    output = model(input)
    loss = loss_fn(output, target)

    # KD logic
    if model_KD is not None:
        # student probability calculation
        prob_s = F.log_softmax(output, dim=-1)

        # teacher probability calculation
        with torch.no_grad():
            input_kd = model_KD.normalize_input(input, model)
            out_t = model_KD.model(input_kd.detach())
            prob_t = F.softmax(out_t, dim=-1)

        # adding KL loss
        if not args.use_kd_only_loss:
            loss += args.alpha_kd * F.kl_div(prob_s, prob_t, reduction='batchmean')
        else: # only kid
            loss = args.alpha_kd * F.kl_div(prob_s, prob_t, reduction='batchmean')
```

Solving Imagenet

결론 :

KD 학습 방법은 매우 매우 안정적입니다.

더 이상 Hyper Parameter Search를 할 필요 없이 단 6줄 만으로 성능을 올리세요!

1. KD 과정 만들어진 Soft Label은 Label Noise를 줄여주게 됩니다.
2. Augmentation은 Label을 refine해주게 됩니다!
3. KD 과정에서 학습 데이터는 한정되어 있고, Student가 Teacher를 따라가기에 데이터는 항상 부족합니다. Augmentation은 Studnet가 Teacher를 따라가기에 부족한 관측 횟수를 늘려주는 역할을 합니다.

Ex: Cut-Out 예시

English setter 63.0%
Ice lolly 16.3%
Gordon setter 3.0%

English setter 0.6%
Ice lolly 99%
Gordon setter 0.4%

English setter 96%
Ice lolly 0.1%
Gordon setter 3.9%