

[11차시]

1. 적은 데이터로 학습하는 방법

소량의 데이터로 학습하는 기술이 중요해지는 시점이다.

데이터의 양이 적으면 데이터를 모으는 노력과 클렌징의 수고, 학습에 걸리는 시간과 부하도 크게 절약할 수 있다.

적은 데이터로 학습하는 방법에는 다음과 같은 것이 있다.

가) 품질 좋은 데이터를 사용

나) Data Augmentation

다) 전이 학습(Transfer learning)

2. 데이터 어그먼테이션(Data Augmentation)

참조자료:

https://m.blog.naver.com/PostView.nhn?blogId=4u_olion&logNo=221437862590&proxyReferer=https:%2F%2Fwww.google.com%2F

<https://www.kakaobrain.com/blog/64>

1) 이미지 어그먼테이션(Image Augmentation)이란?

원본 이미지에 인위적인 변화를 주는 것이다. 그리고 인위적으로 변화를 준 이미지는 충분히 학습에 활용될 수 있는 데이터가 된다.

- 목적은; 적당한 힘으로 학습 면적을 아주 조금 골고루 넓히자는 의미이다. 따라서 고유 정보가 학습될 때, 해당 정보가 맵핑된 공간의 영역이 조금 넓히면서 동시에 크게 벗어나지 않도록 학습하게 된다.

- 어그먼테이션의 필요성:

훈련 과정에서 왼쪽의 고양이 이미지들을 학습한 후에 오른쪽의 누워있는 고양이의 이미지를 인식하지 못할 수 있기 때문



2) 어그먼테이션의 기능;

어그먼테이션이라는 것은, 기존의 데이터의 정보량을 보존한 상태로 노이즈를 주는 방식이다.

- 내가 가지고 있는 정보량은 변하지 않는다는 것이다. 단지 정보량에 약간의 변화를 주는 것으로, 딥러닝으로 분석된 데이터의 강력하게 표현되는 고유의 특징을 느슨하게 만들어는 것이다. 이는 결과적으로 오버피팅을 막아줄 수 있고, 예측 범위를 약간 넓혀줄 수 있다.

- Augmentation의 역할: 학습될 수 있는 특징 공간을 조금이나마 더 넓혀 줄 수 있다는 점

3) 데이터 어그먼테이션 기술의 대표적인 방법에는 Scaling과 Rotation이 있는데

각각은 일반적으로 10, 20, 30의 비율 내에서 조절한다.

- 어그멘테이션 방법(Augmentation Method)으로는 Resize, Scale, Flip, Translate, Rotate, add a noise 이라는 6개의 작업이 존재한다.

그리고 이를 각각 독립적으로 진행할 경우 1개의 데이터로 7개 이상의 데이터를 얻을 수 있다.

- 어그멘테이션 방법으로는

노이즈 증가(가우스 노이즈와 임펄스 노이즈)

콘트라스트 조정

밝기 조정(감마 변환)

smoothing(평균화 필터)

스케일링

반전(좌우 / 상하)

회전

이동(수평 / 수직)

부분 마스크(Cut out 및 Random Erasing)

트리밍 (Random Crop)

변형

변색

배경 교체(이것은 라이브러리의 기능이 아닌 별도의 작업)등이 있다.

※ 오버피팅(overfitting)이란?

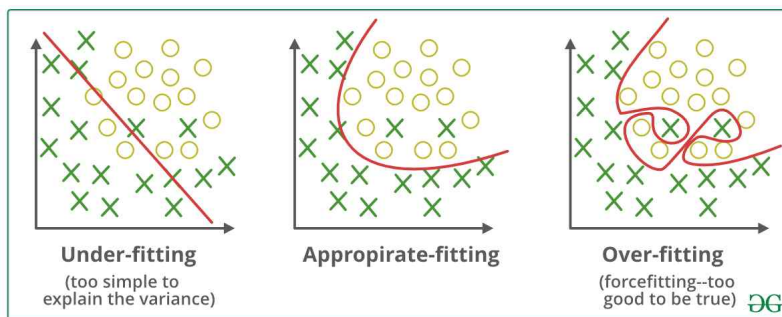
매개변수를 훈련할 **충분한 학습 데이터를 확보하지 않으면** 모델의 성능을 저해하는 과적합(overfitting) 문제가 상대적으로 더 쉽게 발생한다.

과적합은 모델이 훈련 데이터에만 지나치게 적응해 테스트 데이터 또는 새로운 데이터에는 제대로 반응하지 못하는 현상을 가리킨다.

예를들면, 고양이의 정면 사진만 배운 네트워크는 고양이의 옆면 사진을 입력받으면 이를 고양이로 인식하지 못하는 것과 같다.

어떤 사진으로든 고양이를 잘 인식할 수 있도록 하기 위해선 다양하고 많은 데이터로 네트워크를 훈련하는 게 무엇보다 중요한 이유이다.

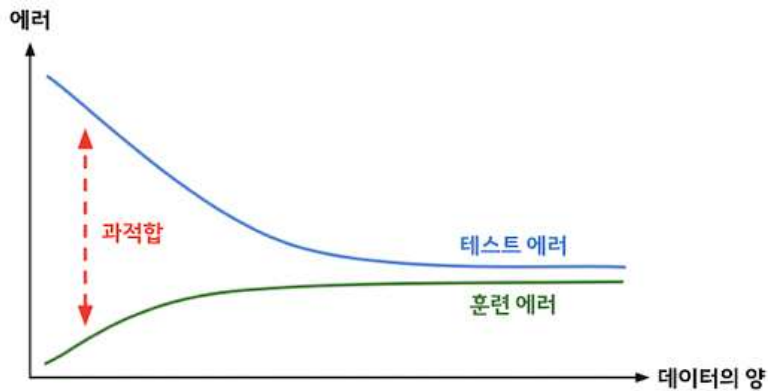
일반적으로 딥러닝 모델은 훈련 데이터의 양이 적고 단순할수록 해당 데이터셋에서만 나타나는 패턴이나 노이즈까지도 학습해버립니다. 이렇게 모델이 훈련 데이터를 너무나 완벽하게 추론하게 되면 새로운 데이터에는 제대로 반응하지 못하는 과적합(overfitting)이 더 쉽게 발생하게 된다. 따라서 모델의 일반화 성능을 높이기 위해서는 충분한 데이터를 확보하는 게 중요하다고 볼 수 있다.



- 하지만 많은 학습 데이터의 확보는 쉽지 않다.

이처럼 딥러닝 모델을 충분히 훈련하는 데 필요한 데이터를 확보하는 기법 중 하나로 어그먼테이션(Augmentation)이 필요하다.

- 아래그림처럼 데이터가 많아질수록 과적합 문제가 발생할 가능성이 낮아진다.



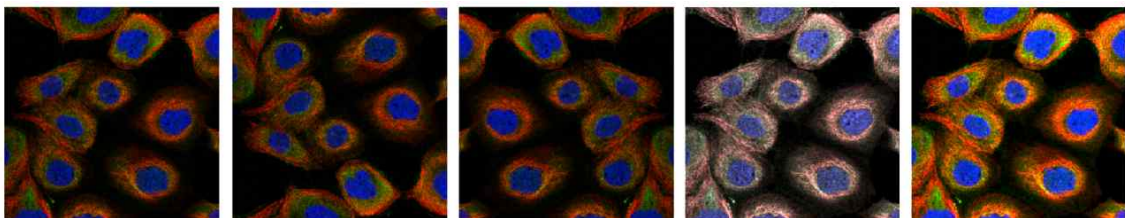
- 데이터어그먼테이션의 예; 다양한 방식으로 변형해 새로 획득한 이미지를 학습 데이터로 활용한다.



4) 데이터 어그먼테이션의 중요성;

데이터 전처리(preprocessing)나 어그먼테이션 기법에 따라 성능이 10~30% 정도 오를 수 있다.

예) 단백질 이미지 분류 대회(Protein Image Classification Challenge)



(원본 이미지(1)를 위아래로 뒤집고(2), 좌우로 뒤집고(3), 색깔을 바꾸거나(4) 그 밝기를 변경(5)했다.)

※ 데이터 전처리(preprocessing)

■ 데이터 전처리의 필요성

실제데이터는 그 자체를 바로 활용하는데 있어 다소 불편하다.

이는 데이터가 불완전(incomplete)하며, 잡음(noisy)이 있고 불일치(inconsistent)하기 때문이다.

■ 일반적인 데이터 전처리 방법

1. 데이터 클리닝(Cleaning)

- 결측치 대체
- 잡음 데이터의 평활
- 이상치의 확인 및 제거
- 불일치 해결

2. 데이터 통합(Integration)

- 다양한 로그 파일 및 데이터베이스의 통합
- 일관성 있는 데이터 형태로 변환

3. 데이터 변환(Transformation)

- 정규화(normalization)
- 집합화(Aggregation)
- 요약(summarization)
- 계층 생성

4. 데이터 축소(Reduction)

- 축소된 데이터도 원래 데이터와 같은 분석 결과를 얻을 수 있어야 함.
- 컴퓨팅 시간 등 고려 위해 데이터 축소가 필요
- 방대한 로그 데이터의 경우 일정 시간 단위로 데이터 축소 필요

5. 데이터 이산화(Discretization)

- 데이터 축소의 일종이나 중요시 됨
- 수치 값을 속성 값으로 변환
예) [0~0.5) ⇨ Low, [0.5~1.0) ⇨ High
- 많은 알고리즘은 데이터 이산화 과정이 요구됨

6. 데이터 표현 특징 추출(Descriptive Characteristics Mining)

- 데이터를 더 잘 이해하기 위해 대표 특징을 이해하는 과정
- 데이터 축소의 일종이기도 함
- 실제 도메인을 고려한 방법이 많이 사용됨
예) 가속도 센서: 가속도 특성에 따른 연산 필요
예) GPS 센서: GPS 데이터 특성에 따른 연산 필요

5) 데이터 어그멘테이션에서 지나친 불필요한 외곡은 오히려 성능향상에 저해요인이 된다; 즉 데이터에 최적화된 어그멘테이션 방법론이 필요하다.

현실과 너무 동떨어지거나 기존 특징을 왜곡할 수도 있는 기법은 오히려 학습 난이도와 성능에 좋지 않은 영향을 끼칠 수도 있죠. 예를 들어, 글자를 인식하는 문제에서는 글자 이미지를 좌우 또는 상하로 뒤집는 기법은 적당하지 않는다. 글자가 지닌 의미가 왜곡될 가능성이 크기 때문이다. 마찬가지로 이유로 **심장을 촬영한 영상도 뒤집기 기법은 적절치 않는다**. 반면, 일반 이미지에서는 이미지가 지닌 의미를 바꾸지 않는 좌우로 뒤집기 기법은 매우 효과적이다.

한편, 공장의 최종 공정에서 흘러 나오는 제품의 품질검사 같은 경우는 정위치 카메라로 촬영한 영상의 크기와 품질이 안정되어 있기 때문에, 노이즈 추가 및 밝기 감소 등 비정상적으로 robust성을 높이는 작업이 필요 없다. 오히려 서투른 변형을 가하여 실제로 발생하지 않는 학습자료를 입력하면 정답률이 떨어지게 된다.

※ robust성 이란?



















외란이나 장애에 강하다는 의미로, 자동차로 비유하면 '비포장길에 강하다', 사람으로 비유하면 '맷집이 강하다'같은 의미이다. 이미지 인식은 대상 이미지가 깨끗한 것만 있는 게 아니고, 일부가 숨어 있거나 각도가 안좋다던지, 심지어 굵혀 있기도 하다. 운영 데이터의 이미지 품질이 불안정한 경우, 그런 이미지도 인식할 수 있는 robust성이 높은 분류기가 필요하다.

6) 데이터어그멘테이션의 자동화

- 머신러닝으로 데이터 수를 늘리는 자동화 방법을 찾는 연구
- 구글(Google)이 발표한 AutoAugment

AutoAugment는 강화학습(reinforcement learning)을 통해 이미지 데이터셋에 가장 적합한 어그멘테이션 정책(policy)을 자동으로 찾아주는 알고리즘이다.

대단히 많은 GPU 자원을 토대로 어그멘테이션 방법을 최적화했을 때 다양한 태스크에서 SOTA 성능을 내는 결과를 보였다.

	Original	Sub-policy 1	Sub-policy 2	Sub-policy 3	Sub-policy 4	Sub-policy 5
Batch 1						
Batch 2						
Batch 3						
		Equalize, 0.4, 4 Rotate, 0.8, 8	Solarize, 0.6, 3 Equalize, 0.6, 7	Posterize, 0.8, 5 Equalize, 1.0, 2	Rotate, 0.2, 3 Solarize, 0.6, 8	Equalize, 0.6, 8 Posterize, 0.4, 6

- AutoAugment란?

AutoAugment는 임의로 선택한 어그멘테이션 기법으로 훈련 데이터를 증강하고 네트워크를 훈련한다.

초기에는 성능이 좋은 기법과 그렇지 않은 것 모두를 골고루 탐색하고 모델의 성능을 높이는 방향으로 보상을 획득해가며 AutoAugment는 점점 더 좋은 성능을 내는 어그멘테이션 기법을 찾아가게 된다.

- 5개 데이터셋에 대한 오류율(%)을 비교한 도표. 서로 다른 데이터셋과 다양한 모델에서

AutoAugment로 찾은 데이터 증강 기법만으로도 좋은 성능을 내고 있다.

Dataset	Model	Baseline	Cutout [12]	AutoAugment
CIFAR-10	Wide-ResNet-28-10 [67]	3.9	3.1	2.6±0.1
	Shake-Shake (26 2x32d) [17]	3.6	3.0	2.5±0.1
	Shake-Shake (26 2x96d) [17]	2.9	2.6	2.0±0.1
	Shake-Shake (26 2x112d) [17]	2.8	2.6	1.9±0.1
	AmoebaNet-B (6,128) [48]	3.0	2.1	1.8±0.1
	PyramidNet+ShakeDrop [65]	2.7	2.3	1.5 ± 0.1
Reduced CIFAR-10	Wide-ResNet-28-10 [67]	18.8	16.5	14.1±0.3
	Shake-Shake (26 2x96d) [17]	17.1	13.4	10.0 ± 0.2
CIFAR-100	Wide-ResNet-28-10 [67]	18.8	18.4	17.1±0.3
	Shake-Shake (26 2x96d) [17]	17.1	16.0	14.3±0.2
	PyramidNet+ShakeDrop [65]	14.0	12.2	10.7 ± 0.2
SVHN	Wide-ResNet-28-10 [67]	1.5	1.3	1.1
	Shake-Shake (26 2x96d) [17]	1.4	1.2	1.0
Reduced SVHN	Wide-ResNet-28-10 [67]	13.2	32.5	8.2
	Shake-Shake (26 2x96d) [17]	12.3	24.2	5.9

- 기존의 SOTA(state-of-the-art, 현재 최고 수준의)방법론이 주로 네트워크 구조나 최적화 방법으로 성능 개선을 이뤘다고 한다면, AutoAugment는 동일한 모델 구조와 최적화 기술을 활용할 때 학습에 적합한 데이터 증강 기법을 적용하는 것만으로 성능을 개선했다는 점에서 의의가 있다고 볼 수 있다.
- AutoAugment의 단점; 계산복잡도가 높다는 점에서 AutoAugment를 일반적인 연구 환경에 적용하기 어렵다는 한계가 존재한다.

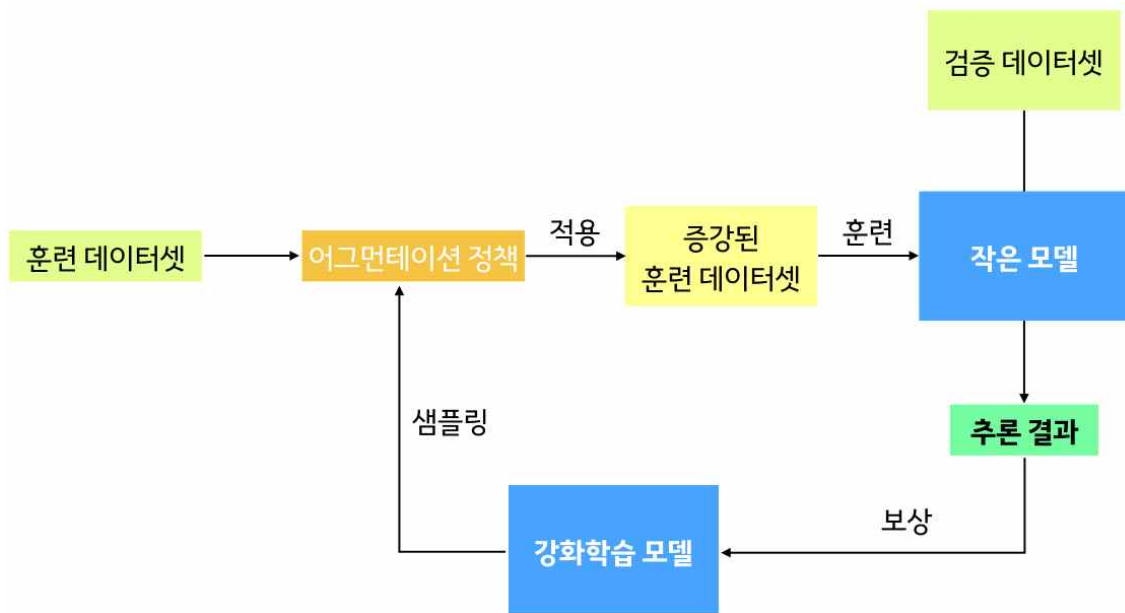
수많은 어그먼테이션 기법 중 하나로 증강한 데이터로 모델을 훈련하고 그 성능을 평가하는 데에는 대단히 많은 컴퓨팅 자원(그래픽처리장치(GPU))이 소모되기 때문이다.

아래 [도표]를 보면 엔비디아 P100 제품을 기준으로 이미지넷에 적합한 어그먼테이션 기법을 찾는데 15,000 GPU 시간이 소요된다. 비용으로 추산하면 대략 3,000만원 규모

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

(5가지 데이터셋에서 최상의 결과값(오류율)을 비교한 표. 이미지넷에서 잘 동작하는 데이터 어그먼테이션 기법을 찾는데 15,000 GPU 시간이 소모됨)

- 어그먼테이션 정책 탐색 방식을 도식화



※ 카카오브레인팀에서 나온 **Fast AutoAugment**

카카오브레인 연구팀은 누구나 쉽게 활용할 수 있으며 연산에 드는 GPU 자원 수를 획기적으로 줄이고, 병렬 계산을 고도화해 더 빠르게 어그멘테이션 기법을 탐색하는 알고리즘인 Fast AutoAugment를 제안.

논문: 국제기계학습학술대회(International Conference on Machine Learning, ICML) 워크샵
<https://kakaobrain.com/blog/116>

7) 코드로 살펴보는 데이터 전처리

```

from tensorflow.keras.preprocessing.image import ImageDataGenerator

# train_datagen = ImageDataGenerator(rescale = 1.0/255.)
train_datagen = ImageDataGenerator(rescale = 1.0/255.,
                                   rotation_range=40,
                                   width_shift_range=0.2,
                                   height_shift_range=0.2,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   horizontal_flip=True,
                                   fill_mode='nearest')

test_datagen = ImageDataGenerator(rescale = 1.0/255.)

```

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

```
# train_datagen = ImageDataGenerator(rescale = 1.0/255.)
train_datagen = ImageDataGenerator(rescale = 1.0/255.,
                                    rotation_range=40,
                                    width_shift_range=0.2,
                                    height_shift_range=0.2,
                                    shear_range=0.2,
                                    zoom_range=0.2,
                                    horizontal_flip=True,
                                    fill_mode='nearest')
test_datagen = ImageDataGenerator(rescale = 1.0/255.)
```

=> ImageDataGenerator 클래스의 rescale 파라미터의 값을 1.0/255로 지정하면 모든 값을 255로 나누게 된다.

- rotation_range는 이미지를 임의로 회전시키는 각도를 지정한다. 0~180 사이의 값을 입력
- width_shift, height_shift는 이미지를 임의로 수직 또는 수평 방향으로 이동시키는 범위를 지정. 이미지의 너비 또는 높이에 대한 비율로 지정한다.
- shear_range는 전단변환 (shearing transformation)을 위한 파라미터, 이미지를 어긋나 보이도록 변환한다.
- zoom_range는 이미지를 임의로 확대하는 정도를 지정한다.
- horizontal_flip은 이미지를 임의로 (수평 방향으로) 뒤집을지 여부를 결정한다.
- fill_mode는 회전 또는 이동 변환 후 빈 픽셀을 채우는 방식을 지정한다. 디폴트는 'nearest'이며, {'constant', 'nearest', 'reflect', 'wrap'} 중 하나의 값으로 지정한다.

3. 전이 학습 (Transfer learning)

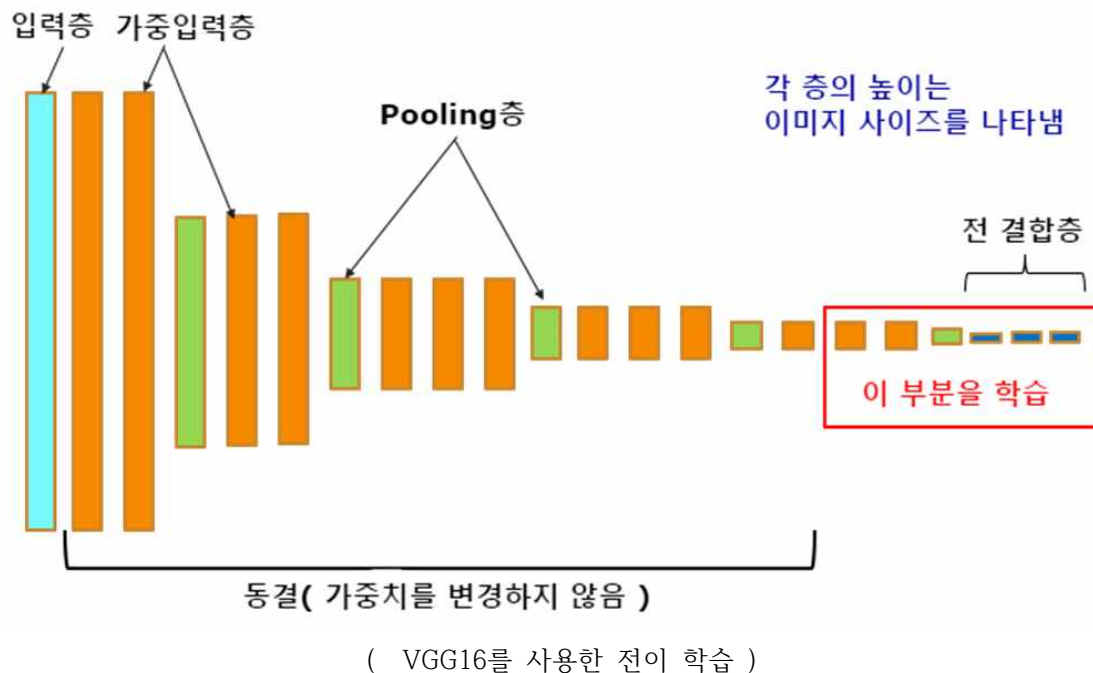
이미지 인식에 있어서의 소량 데이터 학습 방법.

전이 학습은 영역(도메인)에서 학습한 모델을 다른 영역(도메인)에 사용하여 일반적으로 학습하는 것보다 적은 데이터로 추가 학습이 가능한 방법이다.

즉, '저쪽에서 배운 기학습 모델을 활용하여, 이쪽의 학습은 적은 데이터로 끝내는 방법'을 뜻한다.

만일 외국 꽃의 이름을 학습한 모델이 있다면, 우리나라 꽃을 추가로 가르쳐 주면 쉽게 한국 꽃의 이름도 판별하는 분류기가 완성된다. 전이 학습은 이러한 유사한 도메인(꽃)이 아닌 다른 도메인(동물이나 탈것 등)의 모델을 유용하고 통용하는 것이 핵심이다.

1) 전이학습의 방법



전이 학습의 기본은 기존 모델이 열심히 공부한 결과(가중치)를 이어받는 것이다. 즉, 오차 역전파를 반복하여 조정된 각 노드의 가중치(weight)를 재사용하는 것이다.

위 그림에서 16번째 레이어까지 동결(가중치를 변경하지 않음)하여 합성곱층의 마지막 2층과 전결합층에서 학습하는 방법을 보여준다. 동결하지 않은 부분을 다시 생성하여 그 부분만 새로 꽃의 이미지를 추가 학습하는 것이다. 데이지 밖에 꽃의 이름을 기억하지 못했던 학습 모델이지만, 아마 16층까지의 가중치는 좋다고 상정하여 동결하고 추가 학습에 의해 꽃의 이름을 출력 계층에서 꺼내는 분류기를 만드는 것이다.

전이 학습에서 여러 층까지 동결하거나 지정할 수 있으므로 더 동결범위를 늘려 전결합층만 변경하여 학습시키는 방법도 있다. 위에 비해 다소 정확성은 떨어지지만, 학습시간을 더 단축할 수 있다.