개념정리 #4

딥러닝 파이토치 교과서 5장

5. 합성곱 신경망 I

5.3 전이 학습

CNN 기반의 딥러닝 모델을 제대로 훈련시키려면 **많은 양의 데이터**가 필요 BUT 충분히 큰 데이터 셋을 확보하려면 많은 돈과 시간이 필요함

→ (해결) **전이 학습**(transfer learning) : ImageNet처럼 아주 큰 데이터셋을 써서 **사전 훈련된 모델**의 가중치를 가져와 우리가 해결하려는 과제에 맞게 보정해서 사용하는 것

> 💡 **ImageNet**
>
> 영상 인식 기술의 성능을 평가하는 주된 이미지 데이터셋
>
> 클래스 2만 개 이상, 이미지 1419만 7122장

특성 추출 기법

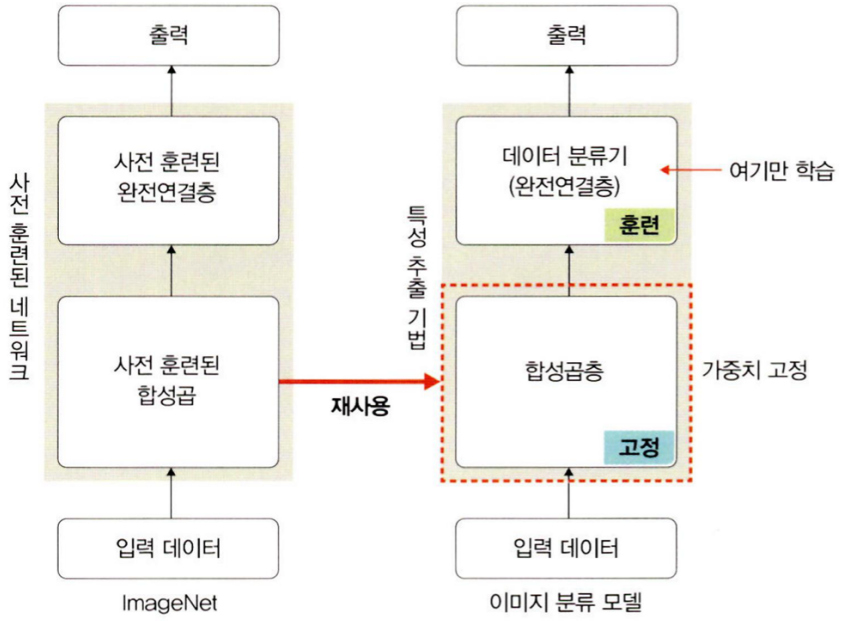
특성 추출(feature extractor) **기법** : ImageNet 데이터셋으로 사전 훈련된 모델을 가져온 후 마지막에 fully-connected layer 부분만 새로 만듦. ⇒ 학습할 때는 이미지의 카테고리를 결정하는 마지막 fully-connected layer만 학습하고, 나머지 계층들은 학습되지 않도록 함.

- **합성곱층** : 합성곱층 + 풀링층

- **데이터 분류기 (fully-connected layer)** : 추출된 특성을 입력받아 최종적으로 이미지에 대한 클래스를 분류하는 부분

사전 훈련된 모델의 합성곱층(가중치 고정)에 새로운 데이터를 통과시키고, 그 출력을 데이터 분류기에서 훈련시킴

> 💡 (ImageNet 데이터셋을 가지고 사전 훈련된 모델의 가중치를 이용한) 전이 학습에 **사용 가능한 이미지 분류 네트워크**
>
> - Xception
> - Inception V3
> - ResNet50
> - VGG16
> - VGG19
> - MobileNet



특성 추출 기법

Note ≡ **사전 훈련된 모델**

파이토치는 다음과 같은 방법으로 무작위의 가중치로 모델을 구성할 수 있습니다.

```
import torchvision.models as models
resnet18 = models.resnet18()
alexnet = models.alexnet()
vgg16 = models.vgg16()
squeezenet = models.squeezenet1_0()
densenet = models.densenet161()
inception = models.inception_v3()
googlenet = models.googlenet()
shufflenet = models.shufflenet_v2_x1_0()
mobilenet_v2 = models.mobilenet_v2()
mobilenet_v3_large = models.mobilenet_v3_large()
mobilenet_v3_small = models.mobilenet_v3_small()
resnext50_32x4d = models.resnext50_32x4d()
wide_resnet50_2 = models.wide_resnet50_2()
mnasnet = models.mnasnet1_0()
```

또한, 다음과 같은 방법을 이용하여 사전 학습된 모델(사전 학습된 모델의 가중치 값)을 사용할 수 있습니다. 다음의 모델들이 사용하는 것과 같이 pretrained=True로 설정하여 사용하면 됩니다.

```
import torchvision.models as models
resnet18 = models.resnet18(pretrained=True)
alexnet = models.alexnet(pretrained=True)
squeezenet = models.squeezenet1_0(pretrained=True)
vgg16 = models.vgg16(pretrained=True)
densenet = models.densenet161(pretrained=True)
```

```
inception = models.inception_v3(pretrained=True)
googlenet = models.googlenet(pretrained=True)
shufflenet = models.shufflenet_v2_x1_0(pretrained=True)
mobilenet_v2 = models.mobilenet_v2(pretrained=True)
mobilenet_v3_large = models.mobilenet_v3_large(pretrained=True)
mobilenet_v3_small = models.mobilenet_v3_small(pretrained=True)
resnext50_32x4d = models.resnext50_32x4d(pretrained=True)
wide_resnet50_2 = models.wide_resnet50_2(pretrained=True)
mnasnet = models.mnasnet1_0(pretrained=True)
```

- 만약 모델의 네트워크를 처음부터 직접 구현하고 최적의 파라미터 값을 찾는다면 꽤 오랜 시간이 소요될 것 BUT 사전 훈련된 모델 사용 → 손쉽게 모델을 학습시킬 수 있음 (실무에서도 많이 사용)
- 정확하지 않은 예측 결과 → Train data 더 늘리고, 에포크 횟수도 늘리면 성능 ⬆ 가능

미세 조정 기법

미세 조정(fine-tuning) **기법** : 특성 추출 기법에서 더 나아가 사전 훈련된 모델과 합성곱층, 데이터 분류기의 가중치를 업데이트하여 훈련시키는 방식

- 특성이 잘못 추출되었다면 미세 조정 기법으로 새로운 이미지 데이터를 사용하여 네트워크의 가중치를 업데이트해서 특성을 다시 추출할 수 있음
- 사전 학습된 모델을 목적에 맞게 재학습시키거나, 학습된 가중치의 일부를 재학습
- 사전 훈련된 네트워크를 미세 조정 → 분석하려는 데이터에 잘 맞도록 모델의 파라미터를 조정

훈련시키려는 데이터셋의 크기와 사전 훈련된 모델에 따라—

데이터셋 크기	사전 훈련된 모델과의 유사성	재학습
큼	**작음**	모델 전체
큼	**큼**	합성곱층 뒷부분 + 데이터 분류기
작음	**작음**	합성곱층 일부분 + 데이터 분류기
작음	**큼**	데이터 분류기

합성곱층의 뒷부분 : 강한 특징이 나타남

데이터 분류기 : fully-connected layer

5.4 설명 가능한 CNN

Explainable CNN : 딥러닝 처리 결과를 사람이 이해할 수 있는 방식으로 제시하는 기술 — CNN 처리 과정 시각화

- CNN을 구성하는 각 중간 계층부터 최종 분류까지, 입력된 이미지에서 특성이 어떻게 추출되고 학습하는지를 시각적으로 설명할 수 있어야 결과에 대한 신뢰성 확보 가능
    - 필터에 대한 시각화
    - 특성 맵에 대한 시각화

특성 맵 시각화

특성 맵(feature map, 활성화 맵) : 입력 이미지 또는 다른 특성 맵처럼 필터를 입력에 적용한 결과 (합성곱 계산을 통해 얻은 출력)

> "특정 입력 이미지에 대한 특성 맵을 시각화" → 특성 맵에서 입력 특성을 감지하는 방법을 이해할 수 있도록 도움

입력층과 가까운 계층 : 입력 이미지의 형태가 많이 유지됨

출력층과 가까운 계층 : 원래 형태는 찾아볼 수 없고, 이미지 특징들만 전달됨

딥러닝 결과에 대한 신뢰성 문제 → **XAI** 이슈됨

CNN : filter와 feature map을 시각화하여 결과의 신뢰성 확보 가능!

5.5 그래프 합성곱 네트워크

Graph Convolutional Network

그래프란

그래프 : 방향성이 있거나 없는 엣지로 연결된 노드의 집합

노드 Node : 원소

엣지 Edge : 두 노드를 연결한 선; 결합 방법을 의미

그래프 신경망

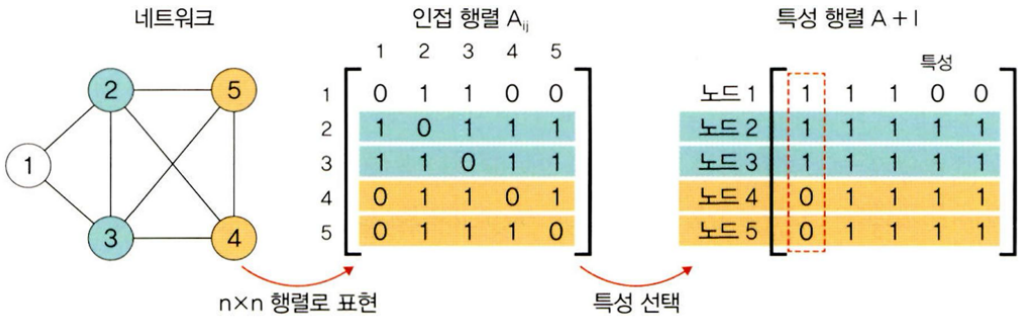
GNN (Graph Neural Network) : 그래프 구조에서 사용하는 신경망

그래프 데이터에 대한 표현 —

1. 인접 행렬 Adjacency matrix
    - 노드 n개를 nxn 행렬로 표현
    - 인접 행렬(nxn) 내의 값은 'A_{ij}는 i와 j의 관련성 여부(두 노드가 직접 연결되어 있는지)'를 만족하는 값으로 채움

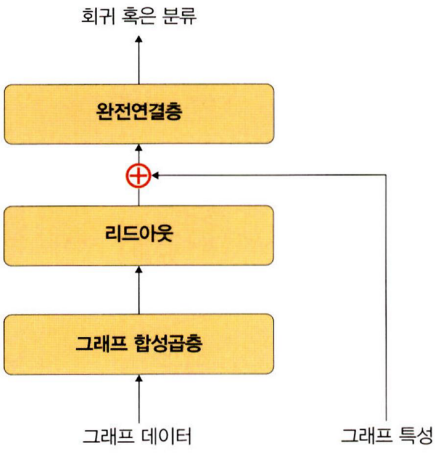
    ⇒ 컴퓨터가 이해하기 쉽게 표현하는 과정
2. 특성 행렬 Feature matrix
    - 인접 행렬에 단위 행렬(identity matrix)을 적용 (노드 자신과의 상관관계 고려)
    - 각 입력 데이터에서 이용할 특성을 선택
    - 특성 행렬에서의 각 행 : 선택된 특성에 대해 각 노드가 갖는 값 (e.g. 첫 번째 행 = 첫 번째 노드의 특성 값)

    ⇒ 특성 행렬 과정을 거쳐 그래프 특성 추출



그래프 합성곱 네트워크

GCN (Graph Convolutional Network) : 이미지에 대한 합성곱을 그래프 데이터로 확장한 알고리즘



- 리드아웃 readout : 특성 행렬을 하나의 벡터로 변환하는 함수. 전체 노드의 특성 벡터에 대해 평균을 구하고, 그래프 전체를 표현하는 하나의 벡터 생성
- 그래프 합성곱층 graph convolutional layer : GCN에서 가장 중요한 부분. 그래프 형태의 데이터가 행렬 형태로 변환되어 딥러닝 알고리즘을 적용할 수 있음

GCN 활용

- SNS에서 관계 네트워크
- 학술 연구에서 인용 네트워크
- 3D Mesh