

연구지도 및 연구윤리 14주차 보고서

(Chapter 7, 8)

경북대학교 전자공학부
2016113566 김남영

1) VGG, googlenet, resnet 장단점 및 각 모델 분석

1. VGG

AlexNet은 11*11, GoogLeNet은 7*7 크기의 필터를 사용하지만 VGG는 처음부터 끝까지 3*3 크기의 필터를 여러 번 사용하여 좋은 성과를 얻는다. 필터의 크기를 줄여 매개변수의 수가 줄어들었기 때문에 학습의 속도가 빨라지고, 층을 깊게 했기 때문에 풍부한 표현력을 가진다. CNN에서는 계층마다 다른 정보를 표현하고 있으므로, 층이 깊은 VGG의 경우 복잡한 사물 또는 사람에 대한 풍부한 표현력을 가진다(비선형성이 커진다). 하지만 완전 연결 계층에 의해서 파라미터 수가 급격히 늘어난다는 단점이 있다. (GoogLeNet의 경우 5 million, VGG의 경우 완전 연결 계층에서만 122 million). 따라서 메모리 소요량이 매우 크다.

2. GoogLeNet

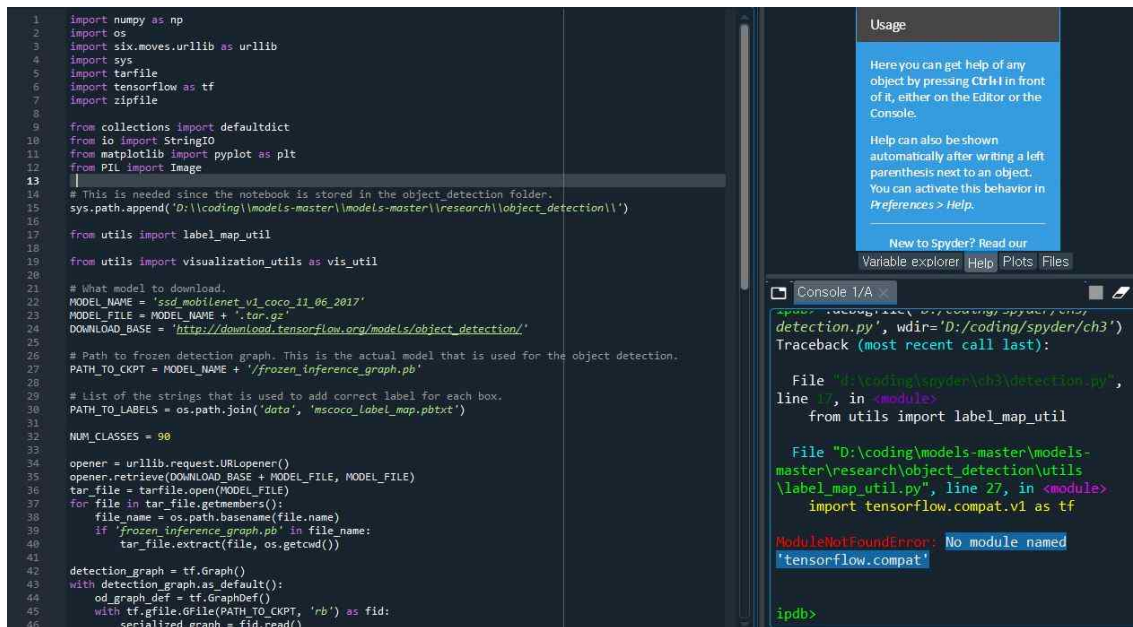
망의 깊이를 늘리게되면 풍부한 표현력을 보장받지만 파라미터의 수가 증가하게 되고 망이 오퍼피팅이 될 가능성이 높아진다. 따라서 GoogLeNet에서는 인셉션 구조를 두어 이러한 문제를 해결하였다. 인셉션 구조는 크기가 다른 필터를 여러 개 적용하여 그 결과를 결합하고, 이 결과를 하나의 블록으로 보는 것이다. 인셉션 구조에서는 1*1 Convolution 필터를 사용하여 매개변수를 줄이는 작업을 통해 고속으로 연산할 수 있게끔 한다(망이 깊어질 여지가 높아짐).

3. ResNet

층을 무한정 늘리면 학습이 잘 되지 않고 오히려 성능이 떨어지는 실험적 결과를 얻을 수 있다. 이러한 문제를 해결하고 층의 깊이에 비례하여 성능이 향상되도록 만드는 기술이 스킵 연결(residual learning이라고도 함) 기술이다. 이는 입력 데이터를 합성곱 계층을 거치지 않고 shortcut을 통해 출력에 바로 더하는 구조를 말한다. 이 구조에 의해 출력은 기존의 출력데이터에 입력데이터가 더해진 형태로 나타나는데, 이 때문에 역전파 때 신호가 작아지거나(기울기가 작아지거나) 지나치게 커지는 등의 문제가 줄어들어 가는 것이다. 이러한 구조로 ResNet은 152층의 신경망을 구현했고 top-5 error 비율은 3.57%로, 이는 사람이 이미지를 분류할 때 보다 더 좋은 결과라고 한다.

2) object detection 대표 예제 구현 및 결과 작성

tensorflow object detection API를 사용하여 임의의 이미지에서 detection을 할 수 있도록 하려했지만 다음과 같은 오류가 발생했다.



```
1 import numpy as np
2 import os
3 import six.moves.urllib as urllib
4 import sys
5 import tarfile
6 import tensorflow as tf
7 import zipfile
8
9 from collections import defaultdict
10 from io import StringIO
11 from matplotlib import pyplot as plt
12 from PIL import Image
13
14 # This is needed since the notebook is stored in the object_detection folder.
15 sys.path.append('D:\\coding\\models-master\\models-master\\research\\object_detection\\')
16
17 from utils import label_map_util
18
19 from utils import visualization_utils as vis_util
20
21 # What model to download.
22 MODEL_NAME = 'ssd_mobilenet_v1_coco_11_06_2017'
23 MODEL_FILE = MODEL_NAME + ".tar.gz"
24 DOWNLOAD_BASE = 'http://download.tensorflow.org/models/object_detection/'
25
26 # Path to frozen detection graph. This is the actual model that is used for the object detection.
27 PATH_TO_CKPT = MODEL_NAME + "/frozen_inference_graph.pb"
28
29 # List of the strings that is used to add correct label for each box.
30 PATH_TO_LABELS = os.path.join('data', 'mscoco_label_map.pbtxt')
31
32 NUM_CLASSES = 90
33
34 opener = urllib.request.URLopener()
35 opener.retrieve(DOWNLOAD_BASE + MODEL_FILE, MODEL_FILE)
36 tar_file = tarfile.open(MODEL_FILE)
37 for file in tar_file.getmembers():
38     file_name = os.path.basename(file.name)
39     if 'frozen_inference_graph.pb' in file_name:
40         tar_file.extract(file, os.getcwd())
41
42 detection_graph = tf.Graph()
43 with detection_graph.as_default():
44     od_graph_def = tf.GraphDef()
45     with tf.gfile.GFile(PATH_TO_CKPT, 'rb') as fid:
46         serialized_graph = fid.read()
```

Usage

Here you can get help of any object by pressing Ctrl+I in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in [Preferences > Help](#).

New to Spyder? Read our [Variable explorer](#) | [Help](#) | [Plots](#) | [Files](#)

Console 1/A x

Traceback (most recent call last):

File "D:\\coding\\spyder\\ch3\\detection.py", line 17, in <module>
from utils import label_map_util

File "D:\\coding\\models-master\\models-master\\research\\object_detection\\utils\\label_map_util.py", line 27, in <module>
import tensorflow.compat.v1 as tf

ModuleNotFoundError: No module named 'tensorflow.compat'

ipdb>

tensorflow의 경우 성공적으로 설치가 된 것을 확인했지만, API의 proto 파일을 컴파일하려고 했을 때 오류가 발생했다. 컴파일 이후에 해당 모델에 대한 py파일이 생성되어야 하지만 오류 때문에 py파일이 생성되지 않아, module이 없다는 오류가 발생하는 것 같다. protobuf의 버전의 문제라고 생각이 되어 버전을 다운그레이드 해보았지만 그래도 실행이 되지 않았다. 환경변수나 tensorflow의 버전 등 여러 방향으로 원인을 탐색해봤지만 알 수 없었다.

3) object detection, segmentation, RNN, GAN, Reinforcement Learning 각 주제에 대한 설명 및 최신 모델 조사

1. object detection

detection이란 이미지 속에 담긴 사물의 위치와 클래스를 알아내어 bounding box로 구분하여 검출하는 일이다. 즉 물체를 분류하는 일(classification)과 위치를 파악하는 일(localization)이 합쳐진 개념이다. 크게 이 두 문제(classification, localization)를 동시에 처리하는 방법과(1-stage detector), 순차적으로 처리하는 방법(2-stage detector)으로 나뉜다. YOLO 계열의 방법이 1-stage detector의 대표적 예시이고, 처음으로 detection 분야에 CNN을 적용시킨 R-CNN이 2-stage detector 방법의 대표적 예시다. 우선 사물처럼 보이는 물체를 추출하고 그 영역에 대해 CNN을 처리하여 클래스를 분류하는 방법이다. 최근에는 물체에 핵심이 되는 Keypoint를 detect하여 bounding box를 얻어내는 cornerNet, extremeNet, centerNet 등이 연구되고 있다.

2. Segmentation

segmentation은 위 detection처럼 특정 keypoint 또는 edge에 기반한 단순한 구별이 아니라 픽셀 수준에서 의미있는 영역을 구분하여 정확하게 객체의 경계선까지 추출하는 기술이다. 기술적으로 semantic segmentation, instance level segmentation, panoptic segmentation으로 분류된다. semantic segmentation은 이미지 각 픽셀을 클래스 단위로 분

류하는 것이고 인스턴스 단위로 구분되지 않는다. instance level segmentation은 객체를 인스턴스 단위로 구분하는 것이다. 인스턴스 단위란 같은 클래스일지라도(같은 dog라도) 서로 구분되어야(ex>다른 색으로 구분) 한다는 것이다. 마지막으로 panoptic segmentation은 semantic과 instance level이 결합된 형태로, 이미지 각 픽셀을 클래스로 분류하면서, 객체에 대해서는 인스턴스 단위로 구분 짓는 방법이다. semantic segmentation에는 대표적으로 모든 픽셀을 한번에 예측할 수 있는 FCN 기반 모델들이 주류를 이룬다. 현재는 저사양의 환경에서도 동작이 가능하도록 처리량을 줄여 경량화 및 고속화 문제를 해결해 나가는 연구들이 수행되고 있다(ENet, RNet, DeepLab 등).

3. RNN(Recurrent Neural Network, 순환 신경망)

순환적 관계를 갖도록 신경망을 구현하여, 이전에 생성한 정보에 영향을 받으며 동작한다는 점이 특징이다(내부에 과거의 상태를 저장하는 메모리를 갖고 있음). 따라서 순차적인 문제를 해결할 때 적합하기 때문에, 자연어 처리 등에 많이 활용된다. 하지만 장시간 과거의 데이터를 저장해야하는 문제의 경우 연산의 길이가 매우 길어져, 기울기 소실, 발산 문제로 학습하기가 어려워진다는 단점이 있다. 이러한 단점 때문에 장기간 메모리 역할을 수행하는 cell state를 추가한 LSTM(3개의 gate)으로 발전하게 되었으며, 최근에는 2개의 gate로 구성되어 더 효율적인 GRU 기술을 사용하기도 한다.

4. GAN(Generative Adversarial Networks)

데이터를 생성해내는 모델 Generator와 그 데이터를 판별하고자 하는 Discriminator를 대립시켜 서로의 성능을 올리는 개념이다. Generator는 진짜같은 이미지를 생성하는 방향으로 발전하고, Discriminator는 진짜와 가짜 이미지를 구별해내는 방향으로 발전한다. 정답이 있는 데이터셋을 활용하는 것이 아니라 단지 대량의 이미지만 주어진 상태에서 학습하므로 비지도 학습의 핵심 개념이라고 볼 수 있다. 하지만 고해상도의 이미지 생성 불가, 학습 불안정 문제가 있어 심층 컨볼루션 네트워크로 보완한 DCGAN 등이 개발되었다.

5.Reinforcement learning

주어진 상황에서 보상을 최대화 할 수 있는 방향으로 학습하는 것을 말한다. 수많은 시도와 실패를 통해 최적의 결과를 낸다. 하지만 통제된 상황이 아닌 실제 상황에서는 다양한 변수가 존재하므로 최적의 결과를 내지 못할 수도 있다. 따라서 다양한 환경에서 반복적으로 알고리즘을 수행하여 다양한 상황에서 최상의 결과를 낼 수 있도록 해야한다. 대표적으로 딥마인드에서 개발한 DQN 알고리즘이 있다. DQN은 과거와 현재의 입력을 통해 최선의 행동을 수행한다. 최근 Google에서 개발한 world model은 미래의 입력을 예측하고 이를 근거로 현재의 행동을 결정하므로 보다 복잡한 추론이 가능하다.

4) 고찰

1. API를 사용해서 detection을 구현하려 했는데, 그마저도 쉽지 않았다. 버전과 경로 등 고려해야 할 것들이 많았다. 학부연수생을 하면서 가장 해보고 싶었던 분야 중 하나인데, 제대로 구현하지 못해서 아쉽다. 시험이 끝나면 확실히 이해할 수 있도록 계획을 짜야겠다.
2. 딥러닝, 컴퓨터비전 연구 분야가 학기 초반에 생각했던 것보다 훨씬 세부적으로 나뉘어져 있다는 것을 알게 되었다.