YOLOv3-tiny

YOLOv3-tiny 구현을 위해 쓰는 문서임.

YOLOv3-tiny.cfg (config 파일) 의 구조를 살펴본다.

hipen@hipen-VirtualBox:~/Desktop/darknet/cfg$ cat yolov3-tiny.cfg

[net]

# Testing

batch=1

subdivisions=1

# Training

# batch=64

# subdivisions=2

width=416

height=416

channels=3

momentum=0.9

decay=0.0005

angle=0

saturation = 1.5

exposure = 1.5

hue=.1

learning\_rate=0.001

burn\_in=1000

max\_batches = 500200

policy=steps

steps=400000,450000

scales=.1,.1

[convolutional]

batch\_normalize=1

filters=16

size=3

stride=1

pad=1

activation=leaky

[maxpool]

size=2

stride=2

[convolutional]

batch\_normalize=1

filters=32

size=3

stride=1

pad=1

activation=leaky

[maxpool]

size=2

stride=2

[convolutional]

batch\_normalize=1

filters=64

size=3

stride=1

pad=1

activation=leaky

[maxpool]

size=2

stride=2

[convolutional]

batch\_normalize=1

filters=128

size=3

stride=1

pad=1

activation=leaky

[maxpool]

size=2

stride=2

[convolutional]

batch\_normalize=1

filters=256

size=3

stride=1

pad=1

activation=leaky

[maxpool]

size=2

stride=2

[convolutional]

batch\_normalize=1

filters=512

size=3

stride=1

pad=1

activation=leaky

[maxpool]

size=2

stride=1

[convolutional]

batch\_normalize=1

filters=1024

size=3

stride=1

pad=1

activation=leaky

###########

[convolutional]

batch\_normalize=1

filters=256

size=1

stride=1

pad=1

activation=leaky

[convolutional]

batch\_normalize=1

filters=512

size=3

stride=1

pad=1

activation=leaky

[convolutional]

size=1

stride=1

pad=1

filters=255

activation=linear

[yolo]

mask = 3,4,5

anchors = 10,14, 23,27, 37,58, 81,82, 135,169, 344,319

classes=80

num=6

jitter=.3

ignore\_thresh = .7

truth\_thresh = 1

random=1

[route]

layers = -4

[convolutional]

batch\_normalize=1

filters=128

size=1

stride=1

pad=1

activation=leaky

[upsample]

stride=2

[route]

layers = -1, 8

[convolutional]

batch\_normalize=1

filters=256

size=3

stride=1

pad=1

activation=leaky

[convolutional]

size=1

stride=1

pad=1

filters=255

activation=linear

[yolo]

mask = 0,1,2

anchors = 10,14, 23,27, 37,58, 81,82, 135,169, 344,319

classes=80

num=6

jitter=.3

ignore\_thresh = .7

truth\_thresh = 1

random=1

실제로 파일을 실행시키면

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

레이어는 0번부터 23번 yolo 까지 나오게 된다.

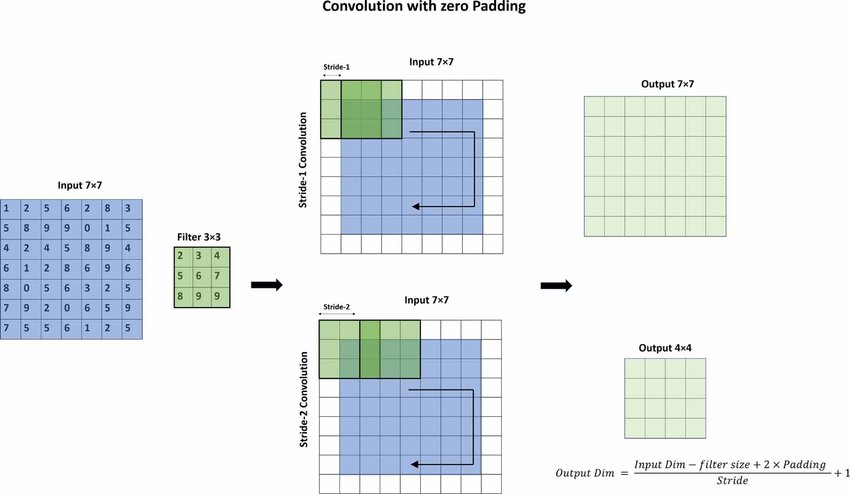
그 전에 conv 레이어에서는 무슨 일이 일어나는지 본다.

스크린샷, 텍스트, 도표, 직사각형이(가) 표시된 사진

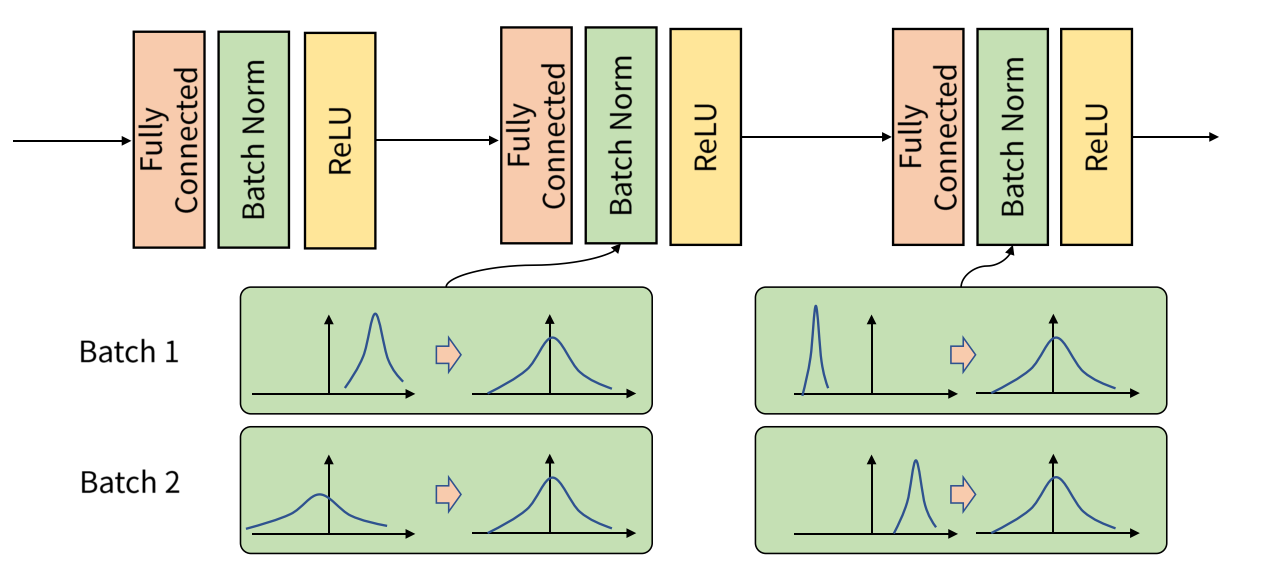
자동 생성된 설명

컨벌루션 레이어는 입력 이미지를 특정 필터를 이용해 탐색하면서 이미지의 특징을 추출하고, 추출한 특징들을 Feature Map 으로 생성한다. 이 Feature map 은 전체 이미지를 Stride 하며 모든 픽셀과 연산을 거쳐 나온 결과값이고, 필터의 크기와 같다. 필터의 개수 만큼 Feature Map 을 생성해내는 셈이다.

(참고) Stride 가 1이면 한 칸씩 움직이게 된다.



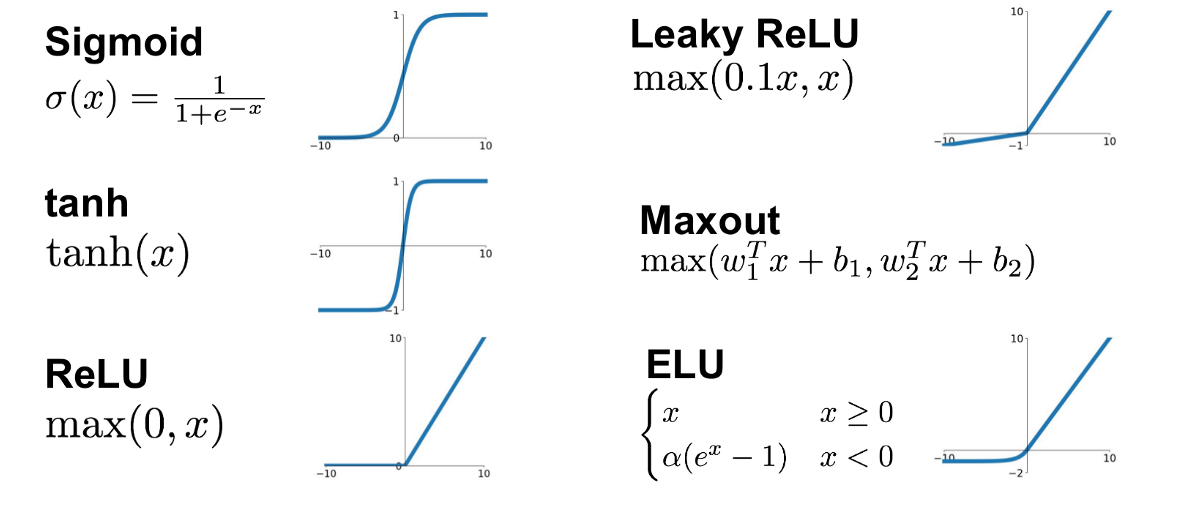
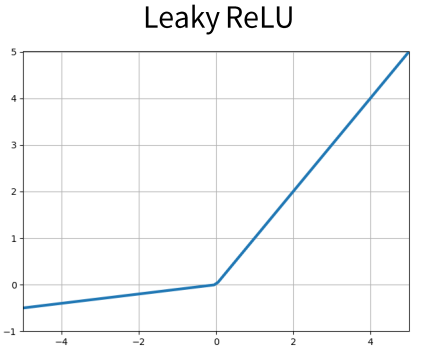
(참고) Batch normalize



(참고) 활성화 함수

시그모이드 함수, 하이퍼볼릭 탄젠트, 소프트맥스 함수, ReLu ...

여기서는 leaky\_ReLu 를 사용함



ReLu가 갖는 뉴런이 죽는 현상을 해결하기 위해 나온 함수임. x가 0보다 작거나 같으면 0.1 \* x 를 반환.

첫 번째 conv 레이어부터 본다.

[convolutional]

batch\_normalize=1

filters=16

size=3

stride=1

pad=1

activation=leaky

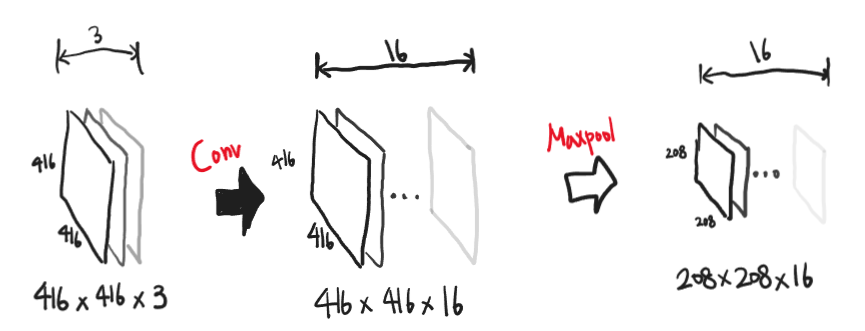
[maxpool]

size=2

stride=2

Convolution 에서 size 가 3이므로 3x3 커널 사용, / 1 이므로 Stride 는 1임.

패딩은 1개 추가하기 때문에 width, height 의 변함 없음.



처음 input image 가 416 x 416 x 3 에서, Conv 레이어를 거치고 416 x 416 x 16(Feature map 개수, 필터 수와 동일) 이 되었고, Maxpool 을 거쳐 208 x 208 x 16 이 되었다.

(참고) Maxpool

커널에서(현재는 2x2 사이즈, stride 2)

텍스트, 도표, 스케치, 그림이(가) 표시된 사진

자동 생성된 설명

(참고?) 줄어드는 부분



1x1 커널, Stride 값 1을 사용해서 연산한다.

padding 은 1이다.

일반적으로 생각해 봤을 때

13x13 에서 패딩을 추가해 15x15 가 되고, 커널이 1x1 이니까 15x15x256이 되어야 하지 않나..?

cfg 파일은 padding 을 1로 설정하던지 0으로 설정하던지 동일한 결과가 나옴. (?)

바로 아래 14번 : 15x15, 커널 3x3, stride 1 --> 13x13 output (ok)

나머지 conv, maxpool layer에 대해서는 동일한 연산의 반복을 해 주면 된다.

yolo 레이어 전에는 conv 로 n x n x 255 사이즈를 맞춰주는 모습이 보임.

남은 yolo, route, upsample layer 에 대해 알아본다.

첫 번째 yolo, route layer 이다.

[yolo]

mask = 3,4,5

anchors = 10,14, 23,27, 37,58, 81,82, 135,169, 344,319

classes=80

num=6

jitter=.3

ignore\_thresh = .7

truth\_thresh = 1

random=1

[route]

layers = -4

YOLO layer

* anchors : 6개의 anchors, mask tag의 속성으로 index 된 anchors 만 사용
  + 따라서, 3, 4, 5번 마스크 ((81,82), (135,169), (344,319)) 사용
* 6개의 박스를 예측(mask 인덱스당 2개, 총 6개)
* 왜 2개일까? (예측 박스 개수 num이 6, mask 가 3개 이므로)
* class : 분류할 클래스 개수(오브젝트 종류)
* jitter : augmentation 과정에서 이미지를 변형시키는 정도(0 < jit < 1)
* IOU 임계값 : 0.7보다 작은 경우 박스 무시
* truth\_thresh : 신뢰도가 1 이상이어야만 박스를 유효한 객체 감지 결과로 간주
* random : 데이터 augmentation 과정을 활성화

route layer

* 네트워크의 마지막 레이어에서 네 번째 레이어 전까지의 출력을 연결한다는 것을 의미, 연결된 출력은 다음 레이어에서 사용
* rout layer는 약간의 설명이 필요합니다. 이것은 하나 또는 두개의 값을 가질 수 있는 layers 속성을 갖습니다.
* layers 속성이 하나의 값을 갖을 때, 이것은 값으로 index된 layer의 feature maps을 출력합니다. 우리의 예제에서 이것은 -4 입니다. 따라서 layer은 Routh layer 부터 4 번째 뒤에 있는 layer이 feature map을 출력할 것입니다.
* layers가 두개의 값을 가질 때, 이것은 값으로 index된 연결된 layers의 feature maps를 반환합니다. 우리의 예제에서 이것은 -1, 61이고 layer은 이전 레이어와 61번째 layer에서 깊이 차원에 따라 연결된 feature map을 출력할 것입니다.

[참고 자료]

[[알고리즘] 컨볼루션 신경망(CNN, Convolution Neural Network) (tistory.com)](https://jarikki.tistory.com/26)

[Schematic of 2D CNN for stride-1 and stride-2 with zero padding (adding... | Download Scientific Diagram (researchgate.net)](https://www.researchgate.net/figure/Schematic-of-2D-CNN-for-stride-1-and-stride-2-with-zero-padding-adding-additional-layers_fig13_350386951)

[배치 정규화(Batch Normalization) - gaussian37](https://gaussian37.github.io/dl-concept-batchnorm/)

[CNN - Convolution 연산에서 Filter에 대한 이해 (tistory.com)](https://woochan-autobiography.tistory.com/883)

[[Deep Learning] CNN(Convolutional Neural Network) (tistory.com)](https://eehoeskrap.tistory.com/281)

[활성화 함수(Activation Function)의 종류 (tistory.com)](https://jaaamj.tistory.com/76)

[기울기 소실 문제와 ReLU 함수 (velog.io)](https://velog.io/@lighthouse97/%EA%B8%B0%EC%9A%B8%EA%B8%B0-%EC%86%8C%EC%8B%A4-%EB%AC%B8%EC%A0%9C%EC%99%80-ReLU-%ED%95%A8%EC%88%98)

[CNN(합성곱 신경망) 기본 (tistory.com)](https://yeong-jin-data-blog.tistory.com/entry/%EB%94%A5%EB%9F%AC%EB%8B%9D-%ED%95%A9%EC%84%B1%EA%B3%B1-%EC%8B%A0%EA%B2%BD%EB%A7%9DCNN-%EA%B0%9C%EB%85%90)

[[CS231n] 강의노트 : CNN (Convolutional Neural Networks) - Taeu](https://taeu.github.io/cs231n/deeplearning-cs231n-CNN-1/)

[[Object Detection] YOLO(v3)를 PyTorch로 바닥부터 구현하기 - Part 1 (tistory.com)](https://deep-learning-study.tistory.com/411)

[[라즈베리파이 4] yolo 학습 데이터셋 생성하기 (velog.io)](https://velog.io/@bae_mung/%EB%9D%BC%EC%A6%88%EB%B2%A0%EB%A6%AC%ED%8C%8C%EC%9D%B4-4-yolo-%ED%95%99%EC%8A%B5-%EB%8D%B0%EC%9D%B4%ED%84%B0%EC%85%8B-%EC%83%9D%EC%84%B1%ED%95%98%EA%B8%B0)