Machine Learning

Assignment #3



수업 일시 : 금 5, 6

담당 교수 : 이 혁 준 교수님

학 과 : 컴퓨터 공학과

학 번 : 2013722095

이 름 : 최 재 은

1. 제목

Recognizing finger pic. By using CNN

1. 목적

직접 1~5를 나타내는 손을 찍어서 data set을 마련하고, Convolution Neural Network를 이용하여 이를 구분해 내는 model을 구현해 본다. 이를 통해 어떤 parameter를 조정하는 것이 결과 Accuracy에 영향을 미치는지 확인해 보고 정확도를 더 높일 수 있는 방법을 구상해 본다.

1. 원리

1~5에 해당하는 사진들, 즉 Data Set들이 각 폴더 별로 구분되어 있기 때문에 이들을 하나의 list에 합하는 코드로 시작된다. 각각 분류된 데이터들을 list에 넣고 그에 각각 매칭되는 label또한 하나의 list로 관리한다. 각 클래스 별로 데이터가 들어갈 때에는 기존의 3depth(RGB)를 평균으로 나누어 1depth로 normalize하여 들어가게 된다.

하나의 list에 각 500개씩 마련된 data가 들어가 있기 때문에 이를 4:1 비율로 Train / Test data와 label로 재 분류 해 주었다. 물론 하나의 list에 들어가 있던 2500개의 데이터들과 label들은 분류 전에 shuffle되어 있다.

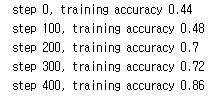
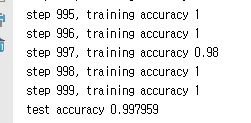
그 이외의 과정에서는 앞서 구현하였던 assignment#2의 모듈 진행을 따랐다.

다만 진행함에 몇가지 수정사항이 있었다. 먼저 convolution과 pooling에서 padding을 사용하도록 수정하였다. MNIST에서는 data set 자체가 key value들이 이미지의 중앙에 위치하도록 되어있고 테두리를 사용할 일이 없어 padding을 사용하지 않았으나, 본인이 마련한 data set에는 손의 위치를 옮겨 가면서 사진을 찍었기 때문에 테두리를 살려줄 필요가 있었기 때문이다. 또한 기존의 MNIST는 28\*28\*1의 이미지를 사용했으나 이번 과제에서는 72\*40의 data를 사용하였기 때문에 place holder의 size 또한 2880의 형태로 변경해 주었다.

Nextbatch func. 에서는 동일하게 batchsize를 50씩 주고 진행하였고, 기존에 마련한 data가 binary data가 아니라 다중 list 형태의 구조 였기 때문에 이를 다시 [50, 72\*40] 형태로 형 변환 해주었다.

Convolution과 Pooling을 진행하는 \_CNNModel func. 에서는 기존의 C -> P ->C ->P 구조에서 벗어나 보았다. 이를 위하여 단순하게 layer를 쌓는 model중 괜찮은 accuracy를 보였던 VGGNet을 활용해 보았다. Convolution filter size 또한 3x3으로 축소 시켰다. 본인이 쌓은 layer의 구조는 두 번의 convolution을 거친 뒤 bias를 더해주고 ReLU activation function 을 적용한다. 활성화 함수를 거치고 나온 result에 2x2 size의 filter로 pooling을 해주었다. 위와 같은 과정을 총 4번을 반복하게 된다. 이 과정에서 처음 1depth였던 image는 1layer에서 32depth로 늘어나고 size는 36\*20이 된다. 2layer에서는 depth를 64로 늘리고 결과는 18\*10이 된다. 3layer에는 depth를 늘리지 않고 Convolution과 Pooling을 진행하며 결과는 9\*5가 된다. 4layer에서는 depth를 128까지 늘리고 결과는 최종적으로 5\*3\*128이 되며 최후에 local response normalization으로 정규화 해주었다. 이후 \_FlatModel을 거치면서 [1, \*\*\*] 형태의 list로 형을 변환해준 뒤 ReLU를 적용해 준다. 그렇게 나온 결과를 softmax를 거쳐 5개의 class에 대한 확률 값을 얻어내고 그를 통해서 예측되는 class와 label을 비교하여 true false를 비교할 수 있다. 그 외의 과정에서는 Assignment#2와 동일하다.

1. 결과



* 위와 같은 정확도가 나왔다.
* 최종 결과를 보면 1000스텝만에 거의 99퍼센트 정도의 정확도를 보인다.
* 아무리 봐도 이미지 프로세싱에 있어서 99퍼센트의 정확도는 말이 안된다.

내가 그런 알고리즘을 짠 것 같지는 않은데 아무래도 overfitting 된 듯 하다.

* 아무래도 데이터 셋을 distortion하고 좀 더 다양한 변화를 준 test set으로

Accuracy를 측정해야 할 듯싶다.

1. 고찰

예전에 공부했던 모델이 Tensorflow에서 제안한 Cifar-10모델이었다. 처음 구상을 할 때에 3개 depth로 시작할 것이라고 생각해서 cifar-10과 유사한 모델로 구성하면 될 것이라고 생각했다가 아주 그냥 시원하게 몇시간을 말아먹었다. 그냥 assignment#2에 몇가지 parameter만 조정해서 넣어주면 되는 것을 …. 아무튼 python을 거의 다뤄보지 않은 터라 placeholder가 뭔지 list랑 array랑 다른 것이 뭔지, tensor는 또 뭔지, 에러는 뜨는데 이게 어디서 나오고 있는 에러인지 뭐가 문제인지 등등등 의문투성이인 터라 생각보다 시간을 오래 잡아먹었다. 게다가 처음에 layer를 통과해서 나온 값들이 계속 0.1~0.3 사이를 오가며 wave쳐서 멘탈이 아스라지는 줄 알았다. 이것 저것 parameter를 건드려 보면서 몇시간을 보냈었고, 결국에는 layer를 늘려보는 방향으로 가닥을 잡았다. 그동안 잠깐이나마 보았던 모델들에 대해서 떠올리던 중 간단한 방법으로 높은 정확도를 보였었던 VGGNet을 떠올리게 되었고, 이와 비슷한 방향으로 model을 쌓았다. 그래도 정확도가 다를 게 없어서 cifar10에서 2번의 컨볼루션 layer를 마치고 normalization을 해주었던 것에 감명을 받고 local response regulation을 적용시켜 보았다.

결과는 매우 양호하게 나오는 편이라고 생각하나 사실은 overfitting 되어있는 것이라 생각된다. Data set을 마련할 때에 다양한 data를 마련하기 위해 (사실은 distortion에 대한 코드를 추가하는게 좀 귀찮아서)일부러 손을 이리 저리 움직이고 손등을 보이기도 하고 뒤집어서 손가락을 들기도 하고 자체적으로 distortion을 진행해 보았으나 결과가 거의 0.99에 가까운 것을 보아하니 overfitting이 분명하다. 차라리 testset을 아예 다른 배경에서 찍은 다양한 크기의 다양한 색의 손이었다면 좀더 정확한 model accuracy를 뽑아낼 수 있었을 것 같다. 결국 모델도 중요하지만 그를 판단하기 위해서 정확한 data set을 마련해야 한다는 것을 다시 한번 느꼈고, deep learning은 이것 저것 바꿔보는 노가다의 산물이라는 것을 또 느꼈다.

\*참고

<http://pythonkim.tistory.com/52>

<https://ko.wikipedia.org>