Keras로 cnn을통한 cifar-100 구현하기

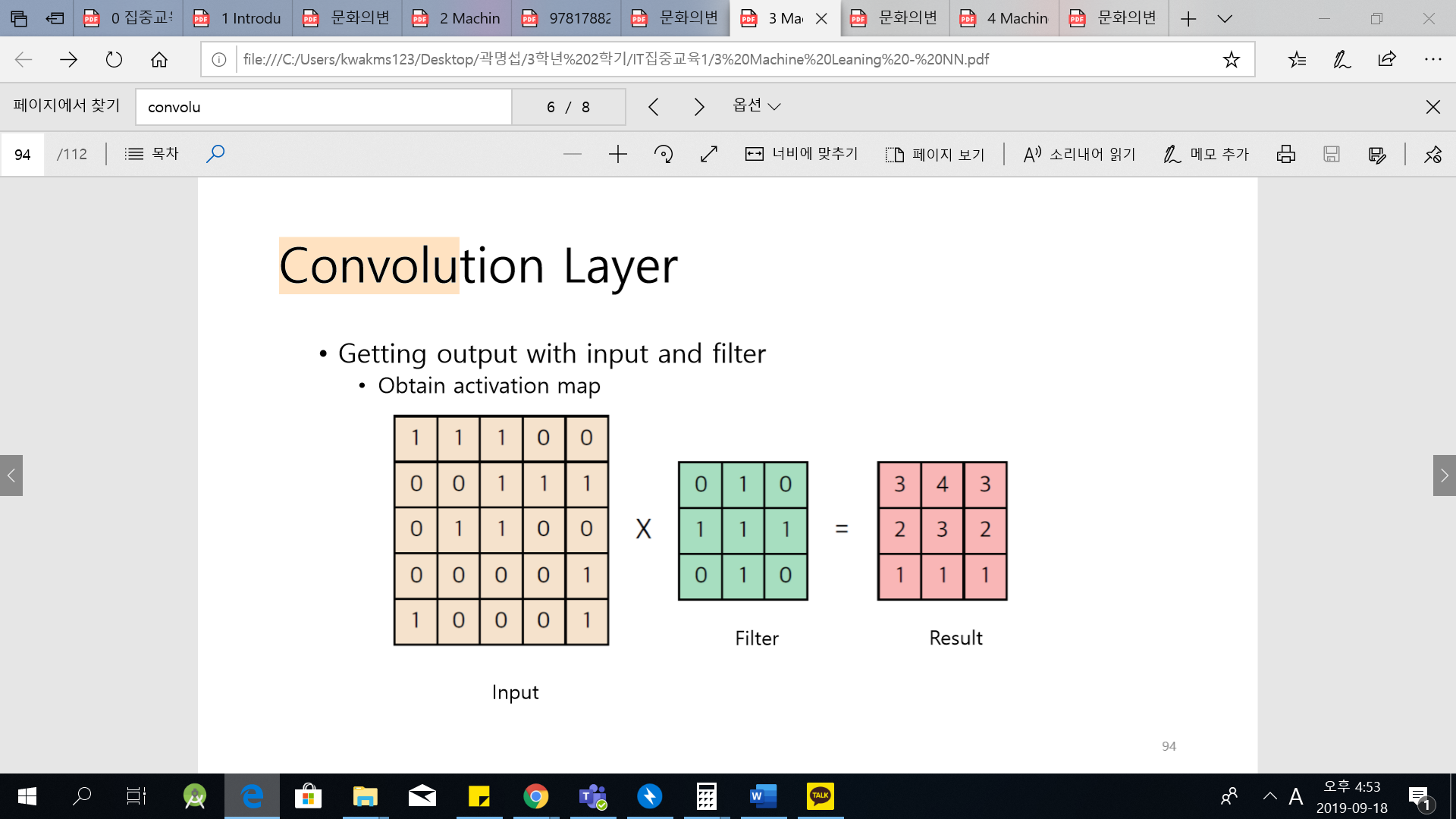
201520991

곽명섭

저번 과제에 이어 이번에는 cifar-100을 CNN을 이용해서 구현을 해보았습니다. DNN에서와는 달리 CNN은 몇 가지 기법들이 더 있어서 그 기법들에 대해서 이론적으로도 좀더 조사를 해보았습니다.

Fully Connected Layer 만으로 구성된 인공 신경망의 입력 데이터는 1차원(배열) 형태로 한정됩니다. 한 장의 컬러 사진은 3차원 데이터입니다. 배치 모드에 사용되는 여러 장의 사진은 4차원 데이터입니다. 사진 데이터로 Fully Connected 신경망을 학습시켜야 할 경우에, 3차원 사진 데이터를 1차원으로 평면 화 시켜야 합니다. 사진 데이터를 평면화 시키는 과정에서 공간 정보가 손실될 수밖에 없습니다. 결과적으로 이미지 공간 정보 유실로 인한 정보 부족으로 인공 신경망이 특징을 추출 및 학습이 비효율적이고 정확도를 높이는데 한계가 있습니다. 이러한 한계를 극복하기위해 이미지의 공간 정보를 유지한 상태로 학습이 가능한 모델이 CNN(Convolutional Neural Network)입니다.

그리고 CNN의 여러 layer기법들에 대해서 조금의 설명을 하자면 우선 layer을 Convolution layer하는 기법인데 이것은 수업시간에도 다루고 강의 노트에도 있듯이 layer을 압축시켜서 filtering 해주는 기법입니다.



강의 노트에 있는 예를 들어서 간단히 설명해보면 input과 Filter가 이런 식으로 존재한다고 하면 filter을 3x3짜리를 5x5의 왼쪽 맨 위로 겹치게 해주어서 filter에 1이있는 부분의 input값들만 추출하여 더해주면 그게 결과값이 되는 그런 과정입니다. 그러한 과정을 1칸씩 옮겨가면서 해주면 5x5짜리 matrix를 3x3짜리 matrix로 압축이 됩니다. keras에서는 Conv2D()라는 함수로 구현을 해주는데 제가 코드에 사용한 예시를 보면

Conv2D(32, (3,3), padding='same', kernel\_regularizer=regularizers.l2(weight\_decay), input\_shape=x\_train.shape[1:])

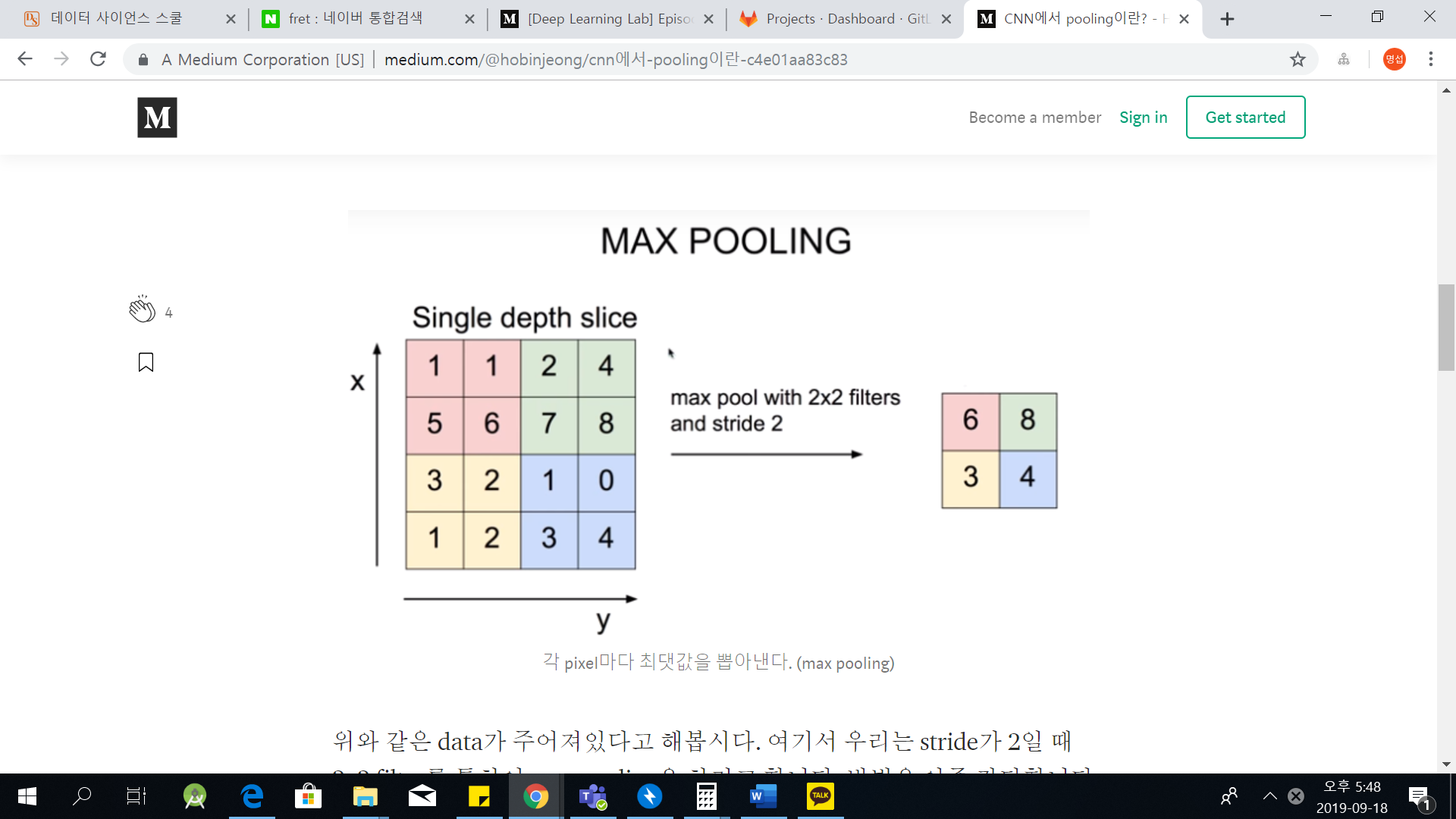
이런 식으로 사용을 해주는데 parameter들을 간단히 설명하면 여기서 32는 convolution의 필터의 수가 32개임을 의미하며 (3,3) 같은 경우엔 커널의 행과 열을 의미, 그리고 kernel\_regularizer는 kernel을 표준화시키는 방식을 의미합니다. Regularization이란 모델이 저장할 수 있는 정보의 양과 종류에 제약을 부과하는 방법입니다. 네트워크가 소수의 패턴만 기억할 수 있다면 최적화 과정 동안 일반화 가능성이 높은 가장 중요한 패턴에 초점을 맞출 수 있게 해주는 방식입니다.

그 다음 기법은 Batch Normalization기법인데 Batch Normalization은 Gradient Vanishing같은 문제들을 해결하기 위한 기법인데 말그대로 Batch를 정규화해주는 것을 말한다. Activation function의 값을 정규화 하는 작업인데 정규화를 하면서 입력분포가 일정하게 되고, 이에 따라 학습속도가 빨라지게 됩니다.

model.add(BatchNormalization())

코드에는 이런식으로 사용을 해주었고 parameter는 딱히 넣어주지 않았습니다.

그 다음은 maxpooling layer인데 너무 간단한 filter 방법으로서 강의노트의 예시를 들어서 설명을 하자면



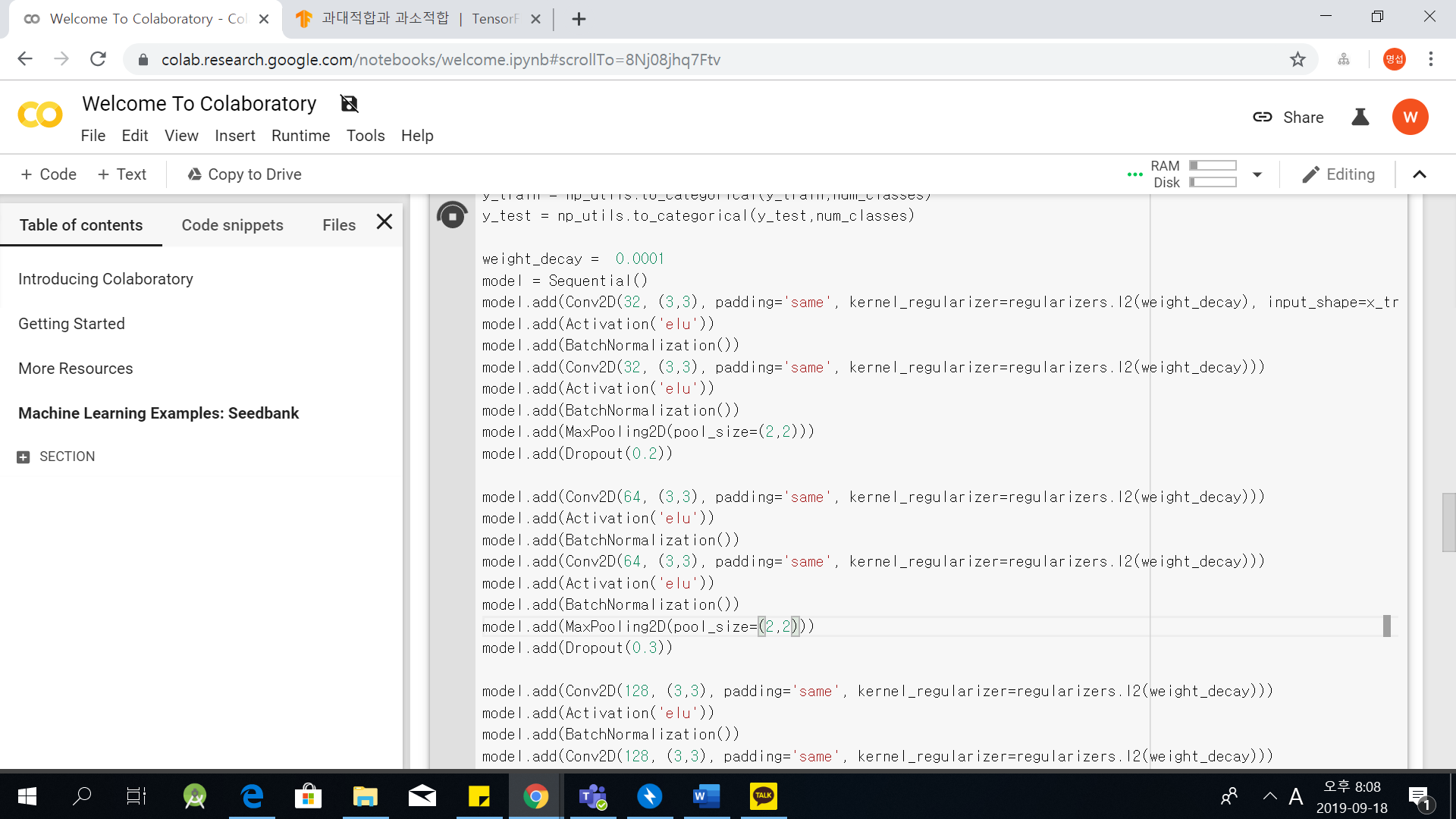
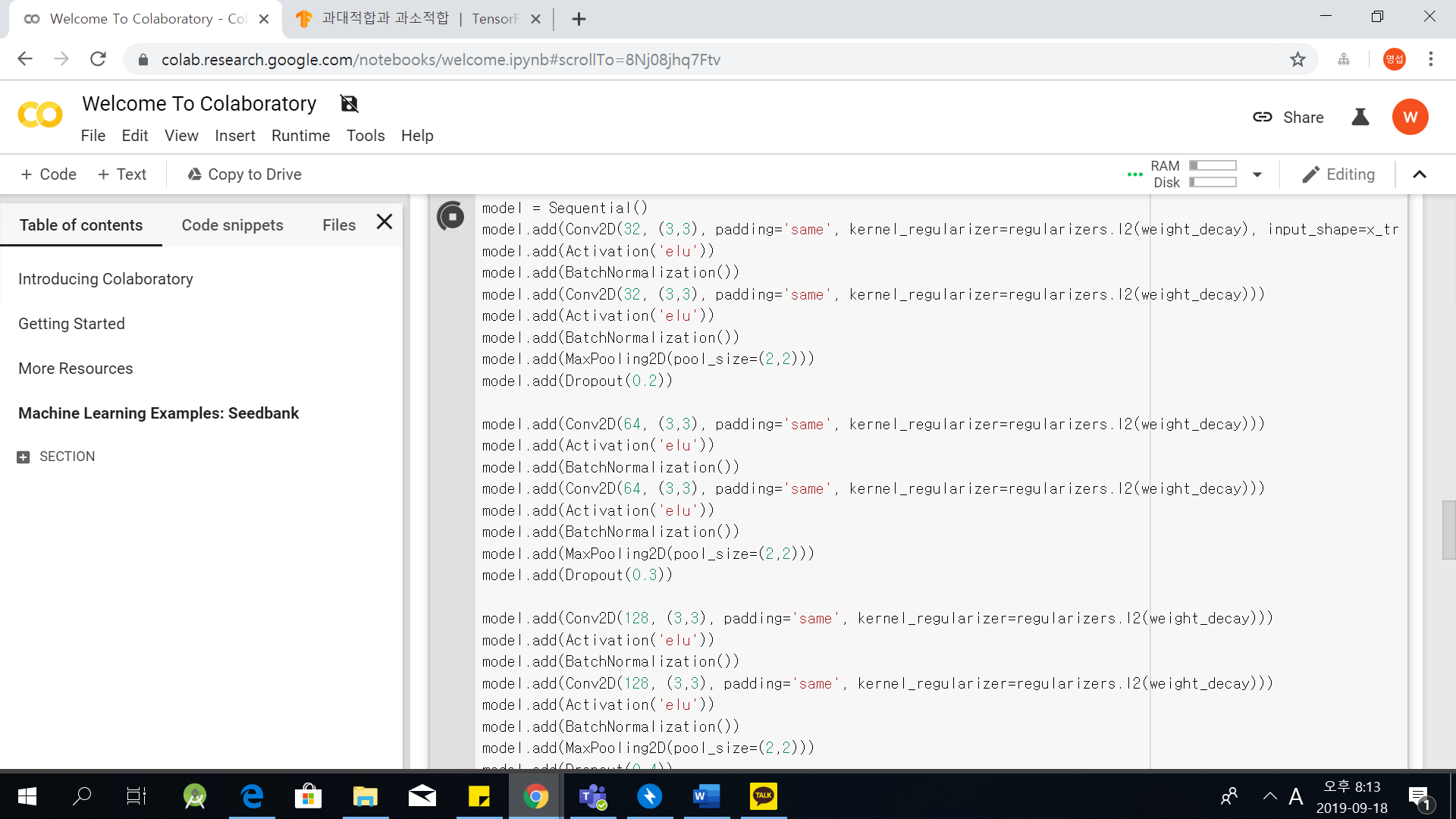
4x4 matrix에서 4부분에있는 각각의 max값만으로 pooling을 해주는 것입니다. 보면 각 색별로 나누어 놓은 cell의 최대값만을 뽑아서 filtering해주고 압축시켜 준 것을 볼 수 있습니다.

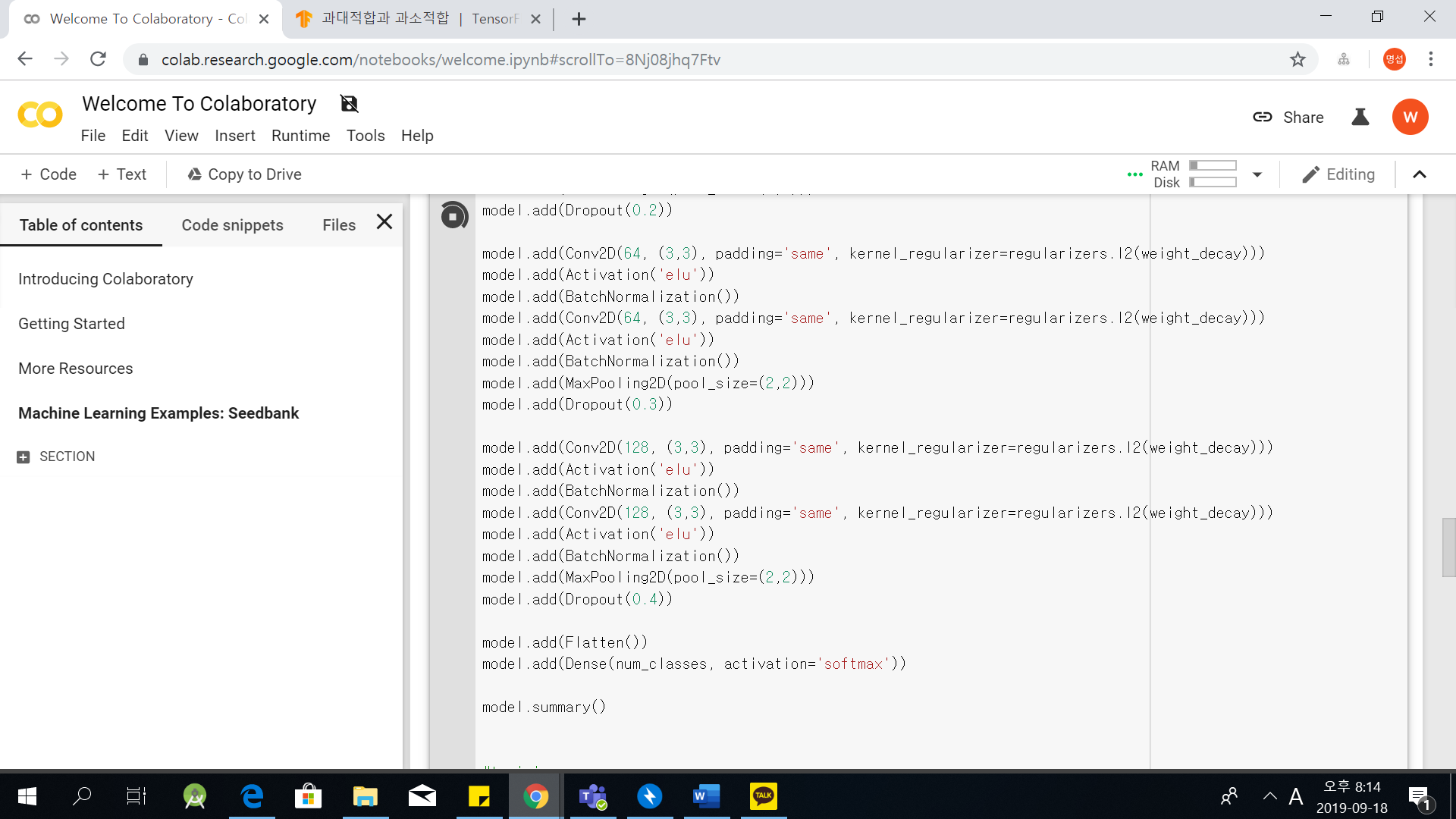
다음으로는 Flatten layer 방식인데 CNN에서 Convolution layer나 MaxPooling layer를 반복적으로 거치면 주요 특징만 추출되고, 추출된 주요 특징은 전 결합층에 전달되어 학습됩니다. Convolution layer나 MaxPooling layer는 주로 2차원 자료를 다루지만 전결합층에 전달하기 위해선 1차원 자료로 바꿔줘야 합니다. 이 때 사용되는 것이 Flatten layer입니다.

제 코드에서 flatten은 model.add(Flatten()) 같이 아무런 parameter없이 써주었습니다.

이 정도면 CNN의 주요 layer의 기법에 관한 이론적인 설명이었고 제 코드에 관해서 설명을 하면 저는 이번에는 실습 때 쓰던 모델은 도저히 accuracy가 높아지지가 않아서 실습 때 쓰던 모델 말고 제가 따로 찾은 다른 모델을 기반으로 하여 accuracy를 최대한으로 높여 주기 위해서 파라미터들과 layer, optimizer, learningrate를 수정하는 방식으로 진행을 하였습니다.

대략의 모델을 보면



이게 제가 사용한 모델이고 설명은 위에 있습니다.

이 모델로 돌려본 결과 accuracy는 대략 0.647정도까지 나왔습니다. 첨부파일을 jupyter로 열어 보시면 코드와 accuracy가 전부 표기 되어있습니다.

