Final Project

Classification of Clothing Items

Jalan, Pragya

CSCI E-89 Deep Learning, Spring 2019
Harvard University Extension School
Prof. Zoran B. Djordjević

Objective

- To detect and classify images of different types of clothing into their respective classes

Real World Impact

- A lot of people have turned to online retailers to browse or buy clothing - therefore there is a huge demand. These retailers need to organise their catalog thereby correctly tagging the labels. It take a lot of man power of correctly classify the clothing into its correct labels.
- People put up a lot images on social media, which they want classified.

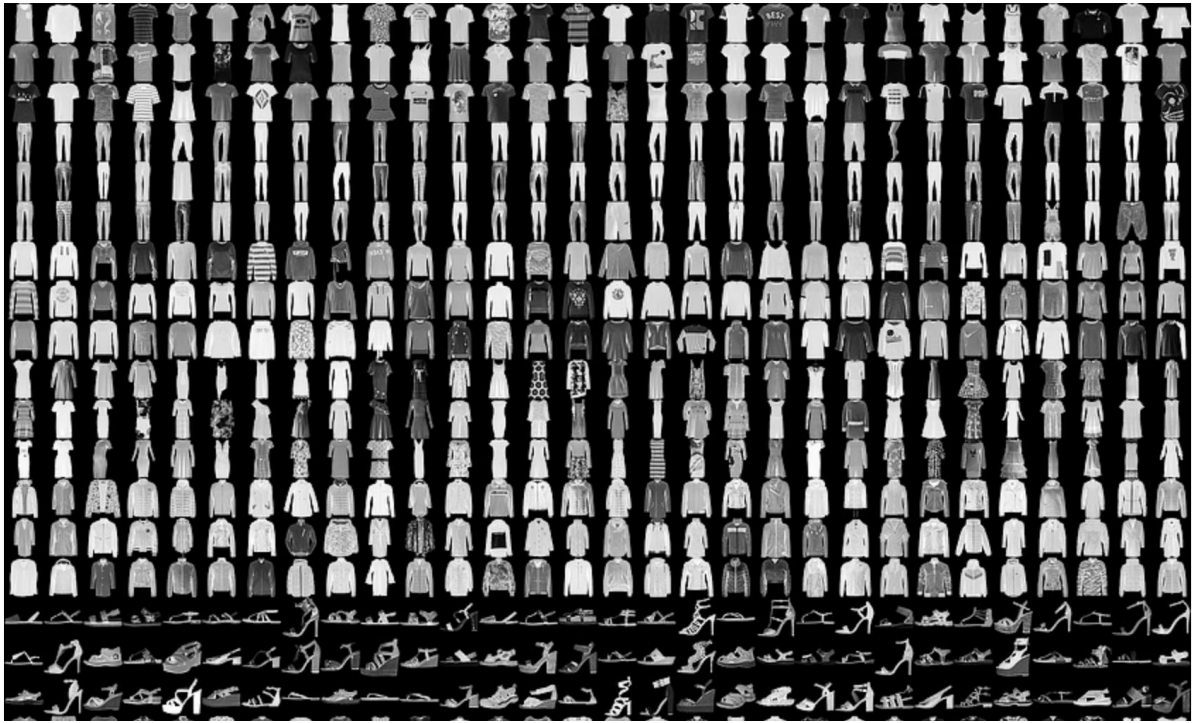
  This classifier will help business correctly classify their images into their respective groups

Dataset

- Fashion MNIST dataset
- Consists of 70,000 28x28 pixel grey-scale images derived from the online retailer Zalando
- Consists of 10 different labels of the fashion items
- The dataset promises to the more diverse
- This dataset is publicly available to Kaggle and Zalando Fashion MNIST repository on github
- The Keras library also includes it as a built in dataset

Dataset

Here is an example of how the data looks (each class takes three rows)



Technology

- This model is written in Python using Keras
- The code was written and run using Jupyter Notebook
- The model was run on Google Colab since I could take advantage of the Graphical Processing Unit (GPU)

Model

- I used a deep Convolutional Neural Network (CNN) to build this image classifier
- I used a pattern of Convolution, Dropout, Convolution and Max Pooling.
- This pattern was repeated 3 times with 32, 64, 128 feature maps.
- Finally an additional and larger Dense layer was used at the end of the network to translate the large number of feature maps to class values.

Model

- Convolutional Layer - this is used to extract features from the input image
- Pooling Layer - this reduces the dimensionality of each feature map but retains most of the important information
- Dense/Fully Connected Layer - implies that every neuron in the previous layer is connected to every neuron in the next layer

- Dropout Layer - here random neurons are dropped during training, therefore other neurons have to step in and handle the representation required to make the prediction. This helps to overcome Overfitting.

Model

The training process:

- The network takes a training image as input, goes through the forward propagation step - which is the convolution and pooling operations in the fully connected layer, finds the output probability for each class
- It then calculates the total error at the output layer
- Finally it uses backpropagation to calculate the gradients of the error with respect to all the weights in the network and use the gradient descent to update all the filter values and parameters values to minimise the output error

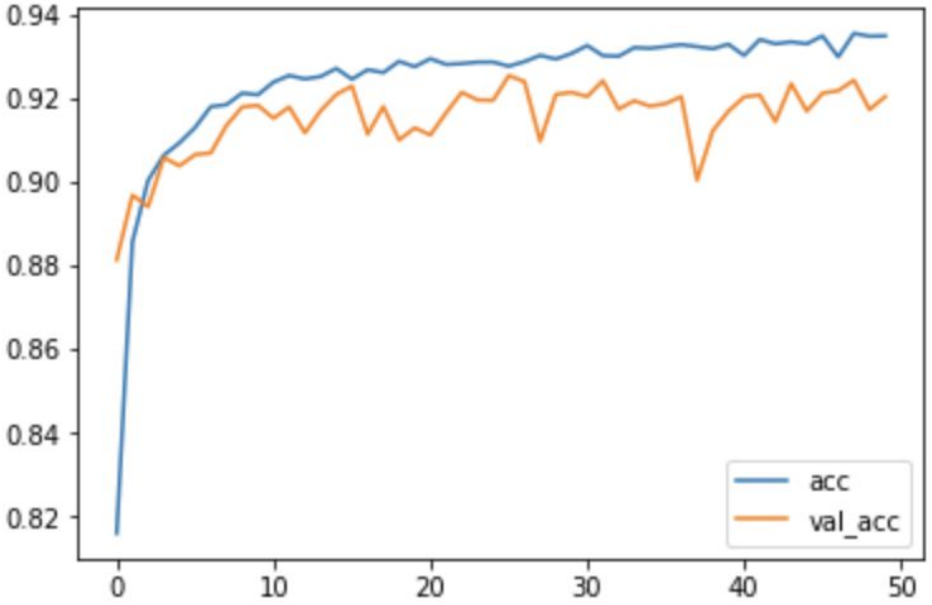
Modeling Strategy

The model was run :

- with a batch size of 32
- for 50 epoches
- on a training set of 60,000
- on a validation set of 10,000

Results

The model gave me an accuracy of ~91%



Results

We observe that it correctly identified the 25 images in the test set.

Results

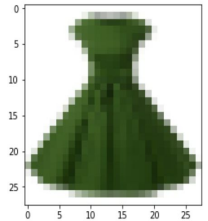
Given an image not it the dataset, it was correctly able to identify its class label

```
--2019-05-13 22:07:48--  https://images-na.ssl-images-amazon.com/images/I/412V6B5Fo4L._SY445_QL70_.jpg
Resolving images-na.ssl-images-amazon.com (images-na.ssl-images-amazon.com)... 52.222.171.116
Connecting to images-na.ssl-images-amazon.com (images-na.ssl-images-amazon.com)|52.222.171.116|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 12011 (12K) [image/jpeg]
Saving to: '412V6B5Fo4L._SY445_QL70_.jpg.3'

412V6B5Fo4L._SY445_ 100%[===================>]  11.73K  --.-KB/s    in 0s

2019-05-13 22:07:48 (323 MB/s) - '412V6B5Fo4L._SY445_QL70_.jpg.3' saved [12011/12011]
```

```
(1, 28, 28, 3)
```

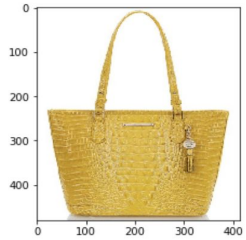


```
Predicted as Dress
```

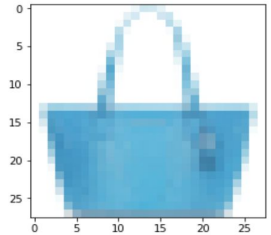
```
Resolving dimg.dillards.com (dimg.dillards.com)... 96.17.122.143
Connecting to dimg.dillards.com (dimg.dillards.com)|96.17.122.143|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 40268 (39K) [image/jpeg]
Saving to: '04774914_zi_sunflower.jpg'

04774914_zi_sunflow 100%[===================>]  39.32K  --.-KB/s    in 0.01s

2019-05-14 18:39:04 (3.59 MB/s) - '04774914_zi_sunflower.jpg' saved [40268/40268]
```



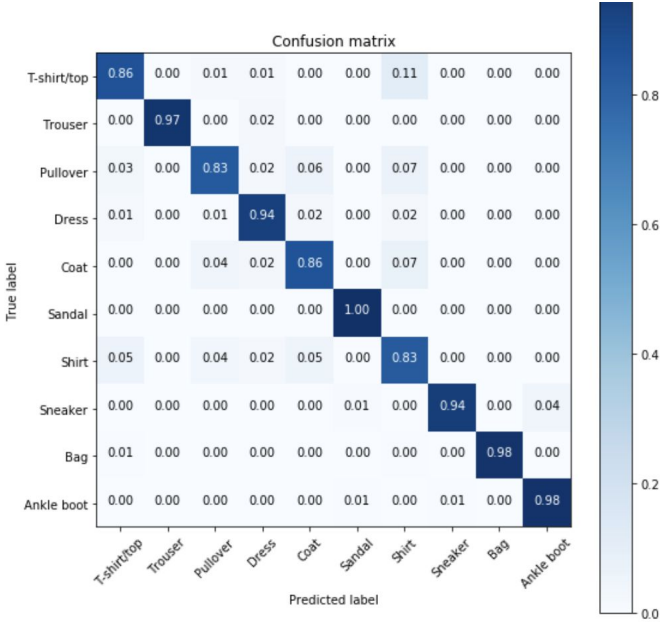
```
(1, 28, 28, 3)
```



```
Predicted as Bag
```

Results

Here we see that most of the miscalculations are happening between 4 of the class labels - T-shirt/top, Pullover, Coat and Shirt which is impacting the performance of the classifier



Conclusion

- The classifier gave a good accuracy of ~91%
- Most of the misclassification was due to the 4 classes - T-shirt/top, Pullover, Shirt and Coat
- Since deep learning needs a large number of data to learn, I would collect more samples and give more images, especially from the above 4 classes to help the classifier learn better.
- Added to this, I would tweak my model so as to add Regularization and maybe another Dropout Layer to improve overfitting.

Thank you

- 2-minute video Youtube link: https://youtu.be/gHL2ggJRzm8
- 15-minute video Youtube link: https://youtu.be/Ii9cFvA-EiQ