Fashion-MNIST Challenge

fashionistas

Omar Wagih
Jessie Liu
Sarah Hafez

Confession: At the very start of the competition, we thought that we MUST have 3 models: CNN, Random Forest and Random Baseline. We cleared up that confusion but we still did 3 models, even though we knew that Random baseline will perform abysmally.

Implementation of the model:

A. **CNN:** The implementation of the model was kind of complicated as there were lots of possibilities to have. I started with normalizing the variance and mean centring the data. At the beginning, I had many convolution layers with the first ones starting at 6 filters and then growing to 8 filters. That had high training accuracy but left the testing accuracy at 85%. I then played with dropout layers and dense layers. I also tried doing conv, conv, pooling, conv, conv , pooling as I read that it was a popular method but it wasn't working out well for me. Eventually, I ended up with two convolutions with 8 filters, with a drop out of 0.5 after each one. Then, I did some pooling. I found it successful to slowly decrease the density layers up until they reach 10.

B. **Random Forest:** Used the randomforestclassifier function from sklearn.ensemble and used the method fit on it with X_train and y_train as parameters. Then, obtain the y_pred from using this model with method predict and parameter X_test.

C. **Random Baseline:** The implementation of this model was pretty simple, and that is because it is not very accurate in general. To implement this model, I simply created two numpy arrays: X_train and y_train, which had the relevant data. I then imported DummyClassifier from sklearn.dummy and used that to implement the model. As there are multiple strategies that I could use within this model, I tried them all out and later found that "stratified" yielded the best results.

Results:

A. **CNN:** Training accuracy could reach 92% if the epochs were high enough. Last time I checked, our testing accuracy was 87.8%!

B. **Random Forest:** The accuracy of this model was around 0.82 which is above the baseline of Random Forest which is 0.80.

C. **Random Baseline:** As this model is the simplest, it is expected that it would perform the worst. Therefore, the model performed pretty horribly, with a score of 0.102.

Challenges:

A. **CNN:** Everything was a challenge in a way. Just trying to get an intuition behind how these convolution layers, pooling layers and density layers work and most importantly what it means to add or reduce them took some time. However, with trial I feel like I got a bit of an intuition of what they do and was able to be more creative and smart about my next steps!

B. **Random Forest:** There were few challenges since this is a simple model to implement, but I had to convert the test_images and train_images numpy arrays with dimension 3 to dimension 2 in order for the methods to work properly. Otherwise I got a valueError since the estimater expects dim <=2 .

C. **Random Baseline:** There were very few challenges in implementing this model because it is quite simple. What I found most challenging was deciding which strategy to implement within the model itself because there were a few to pick from. It also appeared that there is a considerable difference between them, as they all yielded very different results from each other. After experimenting with the training set, I was then able to decide that using the stratified strategy was the best, in terms of accuracy.

Conclusion:

CNN had the highest testing accuracy so it was our best hope in competing at the competition! Random Forest also showed very impressive results. We still don't know why we did random baseline.

Individual Contribution:

Sarah did CNN, Jessie did Random Forest and Omar did Random Baseline.