Data 640 Summer 2022

Professor Steve Knode

Tiania Chan

tiania.chan@gmail.com

Assignment 4: Deep Learning

**Introduction**

Introduction:

The objective of the analysis is to use the fashion MNIST dataset that is part of the Python library, Keras, to train a convolutional neural network (CNN) to be able to identify each article of clothing based on a 28 by 28 pixel picture. The libraries TensorFlow and Keras in Python will be to take the fashion MNIST dataset and run through a convolutional neural network. The libraries Pandas, Numpy, and Matplotlib is used to load and visualize the data.

Dataset:

The dataset, fashion MNIST has a training set of 60,000 examples of 28x28 pixel grayscale images and the test set has 10,000 of these images. An example of an image in the data set can be found in Appendix C which shows a section of the code used to display the image in python. There are 10 different classes or categories of clothing that are identifiable in the dataset.

The 10 different classes or categories of clothing are: tshirt or top, trouser, pullover, dress, coat, sandal, shirt, sneaker, bag, and ankle boot. These are labeled from 0 to 9 in the first column of the dataset. The rest of the columns in the dataset are numeric values of each pixel in the images with a total of 784 pixels with values ranging from 0 to 255. The value of 0 through 255 denote the “darkness” of the pixel whether it is more black or whiter in its position of the image.

**Predictive Models**

Several variations of CNNS were developed and the parameters changed to see which model would produce the most accurate results from training the model on the 60,000 rows of training data to the 10,000 rows of data. The specifics are outlined in Appendix A the model table.

Model 1 is a CNN that has 2 layers. The first layer has a pool size of 2, with 236 layers, a kernel size of 2 and a dropout ratio of .2. The second layer in Model 1 uses 1258 filters, kernel size of 2, pool size of 2 and a drop out ratio of .2. Finally, the layers are flattened to have 10 units with a softmax activation. When the model is trained on the dataset, 10 epochs are used with a batch size of 128. For all the models, the batch size will remain the same.

Model 2 is the second model developed. Model 2 is a CNN that contains 2 layers. The first layer has 236 filters, a kernel size of 2, using a relu activation as the first variation from Model 1 to see if the activation change will make a difference in the model being able to label images correctly. Next, the first layer has a pool size of 3 to see if a larger pooling size would make a difference with a drop out ratio of .3. The second layer of Model 2 has 128 filters, kernel size of 2, relu activation, pool size of 3, and drop out ratio of .3. Finally, the two layers were flattened to 10 units and finished with a softmax activation. The similarity Model 2 to Model 1 is that they both have 10 epochs. The variation in Model 2 has the same number of layers as Model 1 but with a relu activation, pool size of 3, and a drop out ratio of .3 to see how this affects the model when training on the fashion MNIST dataset.

Model 3 is the third model developed. Model 3 has 3 layers but like Model 1, uses the default activation which, when not declared is linear. The reasoning for Model 3 is to see if the increased number of layers will help the model in better categorizing the different articles of clothing better than Model 1. Model 3’s first layer has 238 filters, kernel size of 2, pool size of 2, and a drop out ratio of .2. The second layer has 128 filters instead of 238 filters, but all other variables are the same. The third layer has 64 filters with all other variable the same. Finally, as with Model 1 and 2, the layers are flattened to 10 units and a softmax activation. Also like Model 1 and 2, Model 3 is trained using 10 epochs.

The next three models are like the first three models but instead of 10 epochs, Models 4, 5, and 6 all have 15 epochs. The reasoning behind using 15 epochs instead of 10 epochs is to see if the number of iterations that the CNNs go through make a difference in how well the model performs.

**Results**

Each of the models were run using Google Colab and the results are detailed in Appendix B. For Model 1, the basic CNN with two layers and 10 epochs had a loss of .8893 and the accuracy of .8893 which is less than 1 so not too high but still high. Next Model 2 with different pooling size, used relu and had 10 epochs had a loss of 1.27498 and a accuracy rate of 1.27498 which are both higher than 1 so Model 2 is performing better than Model 1. Model 3 had 3 layers but otherwise the same values as Model 1 and has a loss and accuracy of .9705 which is good that these t values are consistent and since .97 > .89, Model 3 is the best performing model so far. Next, Model 4 is Model 1 but with 15 epochs and the loss is .6971 and the accuracy is the same at .6971 which is less than Model 1 so Model 4 is so far the model with the lowest loss of all the models. Next, Model 5 was Model 2 but with 15 epochs and the loss s .8934 while accuracy is.8934. Since .6971 < .8934, this means that Model 5 performed worse than Model 4 in loss but better in accuracy. Finally, Model 6 was the same as Model 3 but with 15 epochs and the loss was at .7808 while accuracy was also .7808. Since .7808 > .6971, Model 4 is still the best performing model in loss but not as well in accuracy.

**Conclusion**

From all 6 of the CNN models, the best performing model with the lowest values of loss and accuracy is Model 4. The highest in loss and accuracy is Model 2. However, since it is preferred to have the lowest loss and a high accuracy and none of the models are a clear winner, Model 5 was chosen since the results were in between all of the results but with a lower loss than Model 1 whose loss is higher. By taking the median value, the best performing model out of the 6 models is Model 5. Since this is the best performing model, it is recommended to use Model 5 for further analysis of the Fashion MNIST dataset.

Next steps in this project would be to try using a dense neural network as well to compare results. On a similar vein, more CNN models can be created with different numbers of layers as well as using all the variations of activations; relu. Sigmoid, softmax, softsign, tanh, selu, elu, and exponential to see what the differences the activations will have on the result of the CNNs. Also, a more in-depth comparison can be done by using Pandas to create a data frame and run statistical analysis on the results as well for more insight to the results. Also, creating a confusion matrix based on the results of the matrix for more statistical variables to look at each of the models to make a more informed decision as to which model would be better for further input after training the model on the fashion MNIST dataset.

**References**

Brownlee, J. (2016, June 24). Crash Course in Convolutional Neural Networks for Machine Learning. Machine Learning Mastery. Retrieved from <https://machinelearningmastery.com/crash-course-convolutional-neural-networks/>

Deshpande, A. (n.d.). A beginner’s Guide to Understanding Convolutional Neural Networks. Retrieved from <https://adeshpande3.github.io/>

Knode, S. (n.d). Week 7 Lesson Guidance – Deep Learning Models Retrieved from https://learn.umgc.edu/d2l/le/content/685790/viewContent/26248222/View

Keras. (n.d.). Keras API Reference. Retrieved from <https://keras.io/api/>

Karas. (n.d.) Fashion MNIST dataset, an alternative to MNIST. Retrieved from <https://keras.io/api/datasets/fashion_mnist/>

Xiao, H. Rasul, K., Vollgraf, R. (2017, September 15). Fashion-MNIST: a novel Image Dataset for Benchmarking Machine Learning Algorithms. Retrieved from <https://doi.org/10.48550/arXiv.1708.07747>

Zalando Research. (2017, Dec 6). Fashion MNIST: An MNIST-like dataset of 70,000 28x28 labeled fashion images. Kaggle.com Retrieved from https://www.kaggle.com/datasets/zalando-research/fashionmnist/metadata

**Appendix**

**Appendix A: Model Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model Number | # of layers | Pool Size | Filter #s | # Epochs | Activation |
| 1 | 2 | 2 | 236 128 | 10 | Linear |
| 2 | 2 | 3 | 236 128 | 10 | Relu |
| 3 | 3 | 2 | 238 126 64 | 10 | Linear |
| 4 | 2 | 2 | 236 128 | 15 | Linear |
| 5 | 2 | 3 | 236 128 | 15 | Relu |
| 6 | 3 | 2 | 238 126 64 | 15 | Linear |

**Appendix B: Model Results**

|  |  |  |
| --- | --- | --- |
| Model Number | Loss | Accuracy |
| 1 | 0.8993 | 0.8993 |
| 2 | 1.27498 | 1.27498 |
| 3 | 0.9705 | 0.9705 |
| 4 | 0.69708687 | 0.69708687 |
| 5 | 0.8934 | 0.8934 |
| 6 | 0.7808 | 0.7808 |

**Appendix C: First Image in Fashion MNIST with section of code**

**Graphical user interface

Description automatically generated**