Training Fashion-MNIST Dataset With Deep Learning

İsmail Baş

Ege University, Institute of Science, Computer Engineering Department

Abstract

In the computer science image classification is a very important concept to be able to identify images for corporate purposes, global purposes like internet and many other areas. Deep learning architectures like CNN and fully connected networks generally can be used to classify images. In this paper I am planning to implement neural networks with Keras library which eases creating a neural network model then observe the results of deep learning models which are trained on Fashion-MNIST dataset which contains images about clothing related items which has 28x28 pixels in grayscale format.

1 Introduction

Image classification is the process of distinguishing images in a certain category by the computer. The most traditional image classification methods in computer science are machine learning and deep learning methods. Machine learning methods feed from the structured data which contains only limited features like image's textures, shapes, colour histograms etc. which are derived from pixel data. (https://developers.google.com/machine-learning/practica/image-classification). Machine learning methods tries to understand these labeled data features then creates outputs to predict which object the data belongs to. After a certain point machine learning falls short on complex huge amount of data which requires so many features to make detailed inference about inputs to predict the output corresponding to inputs. This means numerous features must be added and selected correctly and manually by humans. This approach requires a good feature engineering and a lot of workload. But deep learning methods feed from raw data directly. Deep learning removes a significant burden on researchers by creating the feature set over raw data. Deep learning models process these features through layers with built-in functions without human effort and improves its structure up to the optimal threshold. With numerous and detailed input features deep learning methods can distinguish more detailed and huge amount of objects with good accuracy (Krishna, Neelima, Harshali & Rao 2018).

In deep learning there are various network architectures like fully connected network (ANN), CNN, LSTM, RNN, DQN. In this paper I am planning to implement deep learning methods on ANN and CNN which is mostly used architecture generally in image classification, to achieve the image classification process on Fashion-MNIST dataset and analyze the results with experiment. Firstly, I will train ANN architecture with Fashion-MNIST dataset which contains 60,000 training images and 10,000 test images with 28x28 pixels in grayscale. Secondly I will train CNN architecture on same dataset then compare the results of these two architectures. In Section 2, I will explain in detail the problem and methods which planned to be used in this research. In Section 3, I will refer to related works. In Section 4, I will summarize the architectures. In Section 5, I will carry out experiment with results in detail. In Section 6, I will conclude the whole research with brief summary.

2 Problem and Used Methods

MNIST is a famous dataset which includes digits from 0 to 9. On MNIST dataset accuracies can reach maximum %99 with deep learning models. Fashion-MNIST which includes varying clothes with grayscale images is upgraded version of MNIST dataset. On Fashion-MNIST dataset accuracies can reach maximum %93-%94 according to structure of the architecture. The problem is training various CNN and ANN models on this dataset and comparing the results with various built-in parameters of keras library.

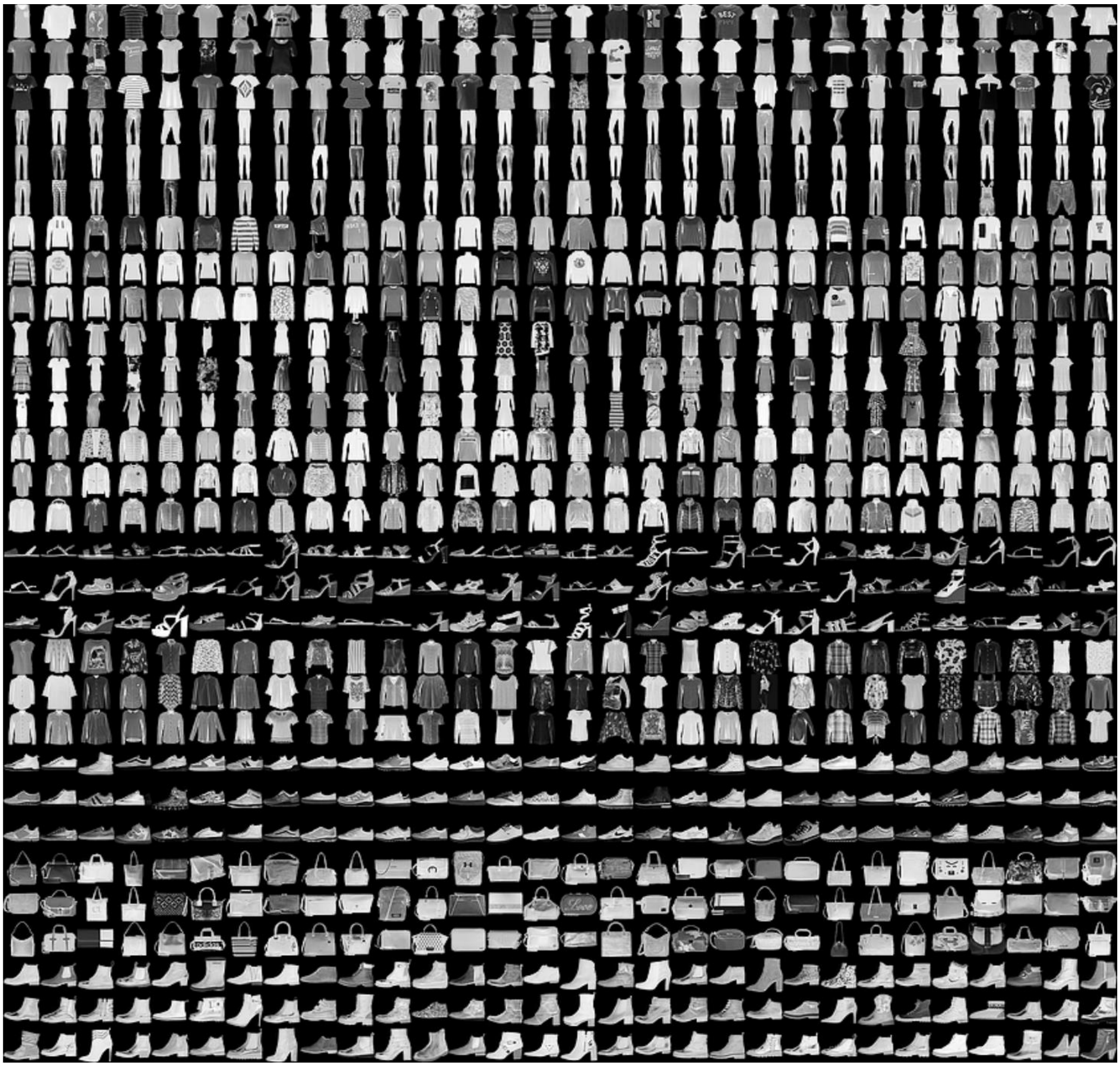


Figure 1: Some sample images from Fashion-MNIST Dataset from https://github.com/zalandoresearch/fashion-mnist

2.1 Artificial Neural Network

Artificial Neural Network (ANN) is a deep learning model which inspired by a biological brain. Brain has synapses and neurons. Signals can be transmitted with synapses from neurons to other neurons. Like biological brain ANN includes nodes which corresponds to neurons and connections which corresponds to synapses. ANN consists of layers. This layers are input, hidden and output. There can be no hidden layers or many hidden layers according to problem or dataset. Every layer consists of nodes. Node size of each layer is called density. Values of input layer for each sample are processed through network with various formulas which called forward propagation process. After this process the outputs are created. Then output layer's values are updated with backpropagation process through network to the input layer. This iteration which called epoch continues until the network trains.

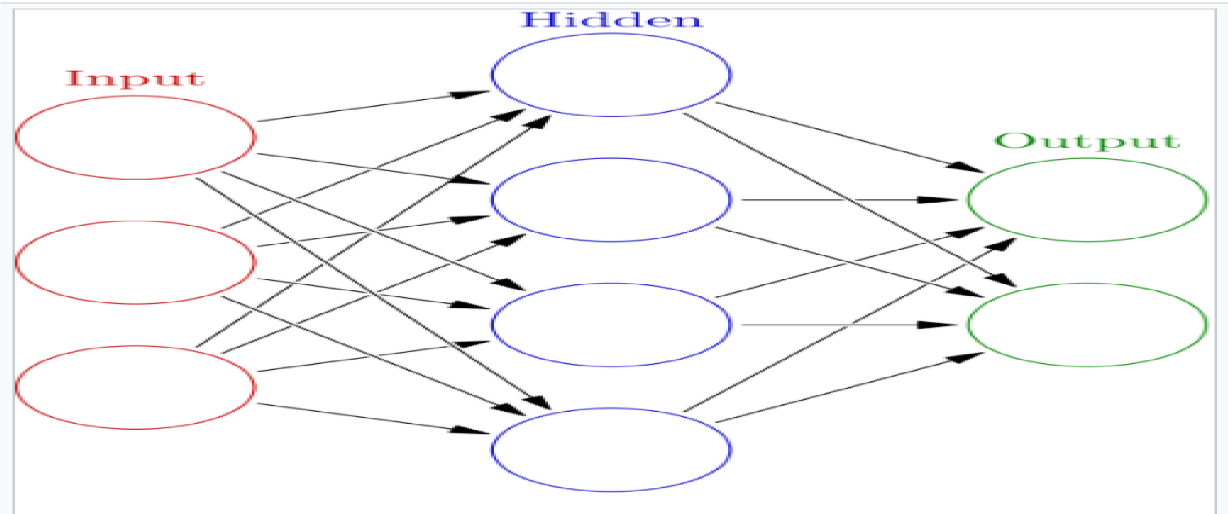


Figure 2: Artificial Neural Netwok (ANN) Architecture from https://en.wikipedia.org/wiki/Artificial_neural_network

2.2 Convolutional Neural Network

Convolutional Neural Network (CNN) is a deep learning model which includes convolutional layers, pooling layers, flatten layer and fully connected network. CNN is widely used for tasks like image classification and natural language processing. In image classification process convolutional layer takes each image's pixels (like 28x28 grayscale or RGB pixels) as input matrices then applies given size of filters (like 8, 16, 32, 64, 128 filters) with kernels (like 3x3, 5x5 matrice) on each image then creates feature maps for each image. Pooling layer (generally uses 2x2 matrice) takes these feature maps then summarizes features to smaller sized matrices. So, pooling operation speeds up CNN's running time reducing the details on feature maps. This operation has disadvantadges like information loss but these losses can be minimized by some algorithms like same padding. After convolution and pooling operations flatten layer takes processed features' pixels of each image then converts them to 1D pixels of array. For each image these pixels are given to fully connected layer and processed through network. CNN architecture runs these operations through epochs.

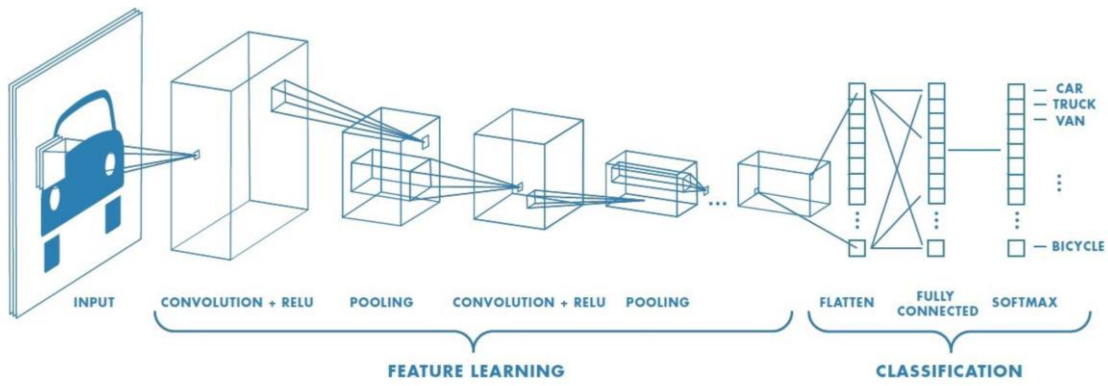


Figure 3: Convolutional Neural Network (CNN) Architecture from https://towardsdatascience.com/

3 Related Works

There are a lot of works about image classification. Firstly you can review the work that covers the creation of fashion-MNIST dataset (Xiao, Rasul & Vollgraf 2017).

One of the detailed work about image classification on fashion-MNIST dataset offers three different architectures of CNN, explains numerous important formulas in detail which are used on architectures and compares the various output metrics of these architectures (Bhatnagar, Ghosal & Kolekar 2017).

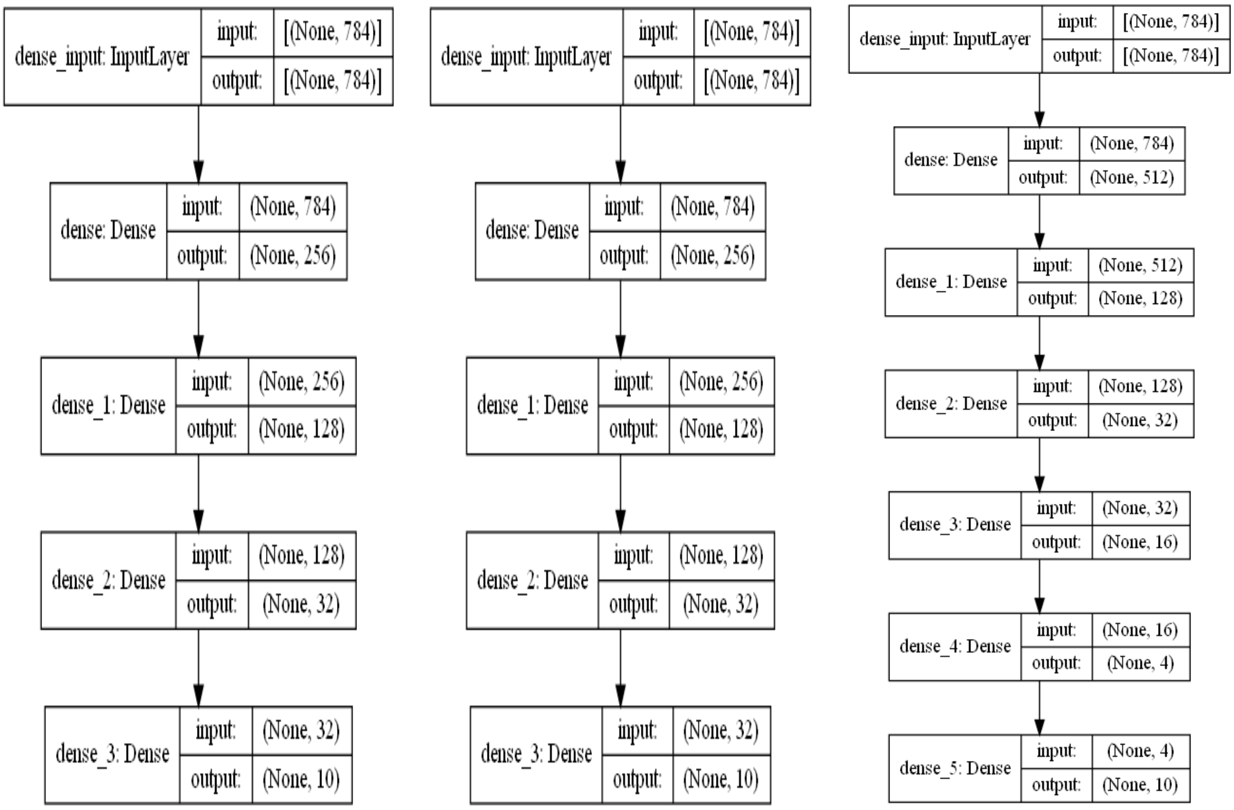
Architectures like LeNET-5, AlexNET, VGG-16, ResNet are common CNN architectures for image classification. AlexNET is a CNN architecture which is designed by Alex Krizhevksy in collaboration with Ilya Sutskever and Geoffrey Hinton, who was Krizhevsky's Ph.D. advisor. Too much work that make image classification using AlexNET can be seen in the literature. One of these works is classification of sea anemone, barometer, stethoscope and radio interferometer images. In this study results show that AlexNET is making effective classifications for test images (Krishna, Neelima, Harshali & Rao 2018). Also you can review a work which is using VGG-11 architecture on fashion-MNIST dataset and explaining batch normalization in detail (Duan, Yin, Zhi, Li 2019).

Numerous works can be found about image classification with deep learning. In literature there are various detailed works. In this paper I will not study on AlexNET or other specific CNN architectures but I will just use my own ANN and CNN architectures and observe the results with changing the architectures and hyperparameters.

4 Structure of Architectures

Firstly I am going to explain three ANN models which were used on three different experiments. These models have output layers with 10 nodes. This means that these classifier models are multiclassifier models. Because there are 10 categories in this dataset like t-shirt, trouser, pullover and so on. That's why these models use softmax activation function on output layer. Also except last layer, other layers use relu activation function. The train data was divided into train and validation data to observe training process and the ratio of validation data to all data is 0.33 which is called validation_split rate. These models were trained with this training dataset. Then test data which includes 10,000 images were tested on these trained models.

First ANN model has three hidden layers, second ANN model is same with first ANN model in terms of layers and finally third ANN model has five hidden layers.

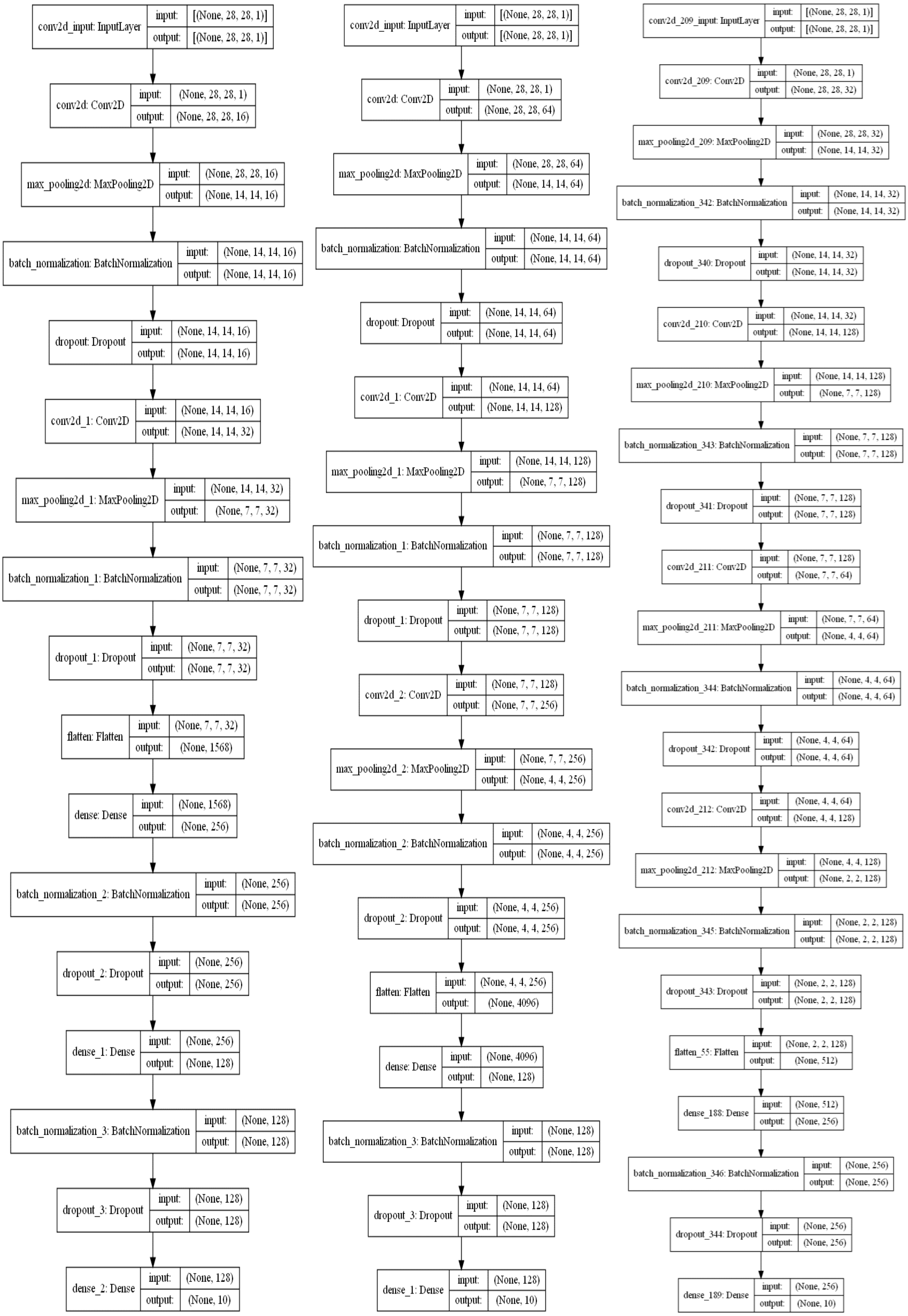


ANN Model 1 ANN Model 2 ANN Model 3

Figure 4: ANN models which are drawn by plot_model function in keras.utils.vis_utils

Secondly I am going to explain three CNN models which were also used on three different experiments. Ratio of validation data to all data is 0.20 in these experiments which has been made with these models. Again these models were trained with this training dataset. Then test data which includes 10,000 images were also tested on these trained models. All of these three models have 3x3 kernel size with stride value 1 for each convolutional layer. They have relu activation function for each layer except the last layer which has softmax activation function. They have 2x2 pooling size for each pooling layer. Also the dropout layers in these models usually have a value around 0.20.

First CNN model has fourteen hidden layers, second CNN model has fifteen hidden layers and finally third CNN model has nineteen hidden layers. After determining the structures you can review the results of these deep learning models which has been trained in different conditions in experimental studies section.



CNN Model 1 CNN Model 2 CNN Model 3

Figure 5: CNN models which are drawn by plot_model function in keras.utils.vis_utils

5 Experimental Studies

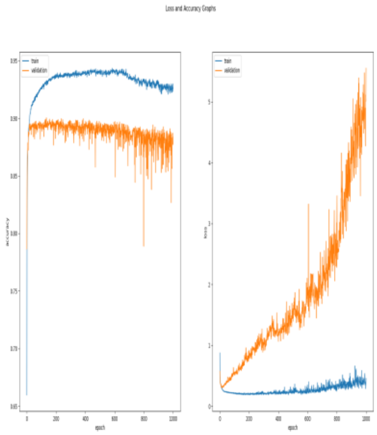
The dataset which is used in this study is Fashion-MNIST. Fashion-MNIST is upgraded version of MNIST dataset. Fashion-MNIST contains images about clothing related items with 28x28 pixels in grayscale. These items are t-shirt/top, trouser, pullover, dress, coat, sandal, shirt, sneaker, bag and ankle boot. This dataset has 60,000 train images and 10,000 test images. You can see the details about this dataset at https://github.com/zalandoresearch/fashion-mnist.

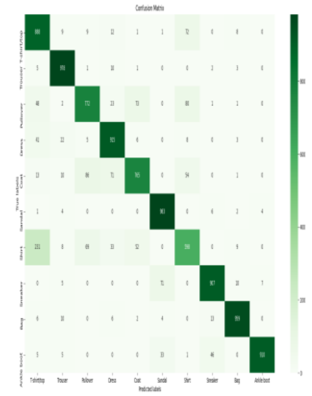
ANN and CNN models were implemented with Keras 2.4.3 library. ANN models were trained on train dataset which has been splitted by 0.33 ratio of validation data. However CNN models were trained with 0.2 validation data on same train dataset. Validation data were used to observe the accuracies of datas which has no label values while training process. Splitting some data for validation on training dataset helps observing the accuracy of unknown labeled data for each iteration while training. Thus, an impression can be obtained about how the model performs on different untrained data. After the training has been completed prediction was performed on test dataset which contains 10,000 test images to observe the model's performance on test dataset.

First ANN model was trained under conditions of RMSprop optimizer with default learning rate, 1000 epochs and 100 batch sizes. First experiment's results were not good. On training dataset accuracy was around 0.92 and loss was around 0.4. But on validation dataset accuracy was around 0.87 and loss was increasing from 2 to 5 while loss was decreasing on training dataset. That was very bad because there were too much overfitting and on validation dataset accuracy was unstable and couldn't make any progress. On test dataset pullover, coat and especially shirts had poor accuracy. Second ANN model which has same architecture as the first was trained under conditions of SGD optimizer with 0.025 learning rate, 1000 epochs and 200 batch sizes. The results were very good according to the first experiment. On training dataset accuracy was around 0.96 and loss was around 0.082. On validation dataset accuracy was around 0.90 and loss was around 0.38. On test dataset pullover, coat and shirt had better accuracy than the first experiment. But the accuracy of shirts were still not at the desired level. Overfitting was minimized. In the second experiment, the model had a much more stable learning curve. It seems that changing the optimizer to SGD and changing the learning rate has made the model incomparably better than the previous one. Finally third ANN model was trained under conditions of Adam optimizer with default learning rate, 100 epochs and 50 batch sizes. In third experiment results were not bad but little worse than the previous one. On training dataset accuracy was around 0.95 and loss was around 0.11. On validation dataset accuracy was around 0.89 and loss was around 0.45. On test dataset the only good thing was that the model had a slightly better accuracy in shirts than previous one. In later iterations some overfitting has occurred.

In ANN models, accuracy reached a maximum of 0.90 and the loss reached a minimum of 0.34 on the validation dataset. The next stage of the experiment will be to train this dataset on CNN models and observe in which direction the results change.

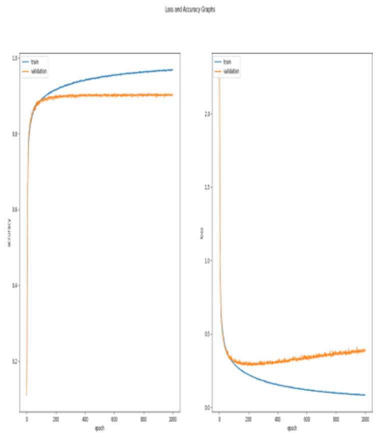
Results of first experiment

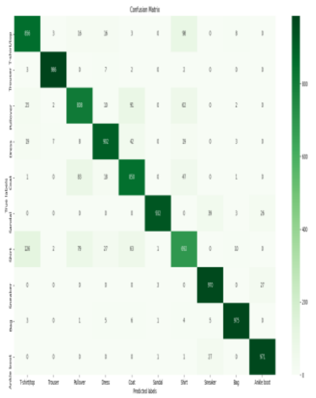




```
Confusion Matrix
[[888   9   9  12   1   1  72   0   8   0]
 [  5 978   1  10   1   0   0   2   3   0]
 [ 48   2 772  23  73   0  80   1   1   0]
 [ 41  22   5 915   6   0   8   0   3   0]
 [ 13  10  86  71 765   0  54   0   1   0]
 [  1   4   0   0   0 983   0   6   2   4]
 [231   8  69  33  52   0 598   0   9   0]
 [  0   5   0   0   0  71   0 907  10   7]
 [  6  10   0   6   2   4   0  13 959   0]
 [  5   5   0   0   0  33   1  46   0 910]]

              precision    recall  f1-score   support

 T-shirt/top       0.72      0.89      0.79      1000
     Trouser       0.93      0.98      0.95      1000
    Pullover       0.82      0.77      0.80      1000
       Dress       0.86      0.92      0.88      1000
        Coat       0.85      0.77      0.81      1000
      Sandal       0.90      0.98      0.94      1000
       Shirt       0.74      0.60      0.66      1000
     Sneaker       0.93      0.91      0.92      1000
         Bag       0.96      0.96      0.96      1000
  Ankle boot       0.99      0.91      0.95      1000

    accuracy                           0.87     10000
   macro avg       0.87      0.87      0.87     10000
weighted avg       0.87      0.87      0.87     10000
```

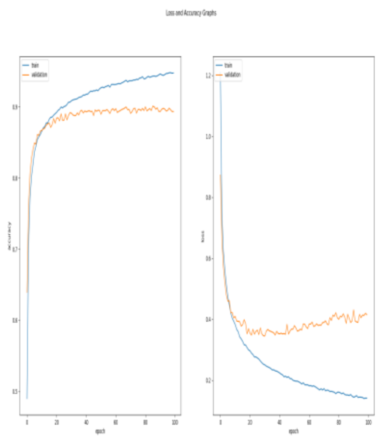
Results of second experiment

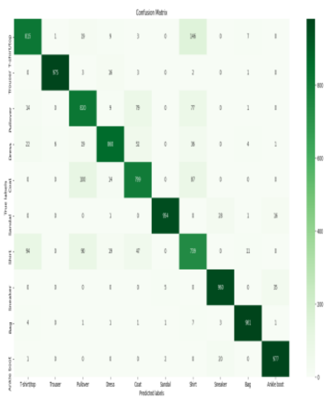




```
Confusion Matrix
[[856   3  16  16   3   0  98   0   8   0]
 [  3 986   0   7   2   0   2   0   0   0]
 [ 25   2 808  10  91   0  62   0   2   0]
 [ 19   7   8 902  42   0  19   0   3   0]
 [  1   0  83  18 850   0  47   0   1   0]
 [  0   0   0   0   0 932   0  39   3  26]
 [126   2  79  27  63   1 692   0  10   0]
 [  0   0   0   0   0   3   0 970   0  27]
 [  3   0   1   5   6   1   4   5 975   0]
 [  0   0   0   0   0   1   1  27   0 971]]

              precision    recall  f1-score   support

 T-shirt/top       0.83      0.86      0.84      1000
     Trouser       0.99      0.99      0.99      1000
    Pullover       0.81      0.81      0.81      1000
       Dress       0.92      0.90      0.91      1000
        Coat       0.80      0.85      0.83      1000
      Sandal       0.99      0.93      0.96      1000
       Shirt       0.75      0.69      0.72      1000
     Sneaker       0.93      0.97      0.95      1000
         Bag       0.97      0.97      0.97      1000
  Ankle boot       0.95      0.97      0.96      1000

    accuracy                           0.89     10000
   macro avg       0.89      0.89      0.89     10000
weighted avg       0.89      0.89      0.89     10000
```

Results of third experiment





```
Confusion Matrix
[[815   1  19   9   3   0 146   0   7   0]
 [  0 975   3  16   3   0   2   0   1   0]
 [ 14   0 820   9  79   0  77   0   1   0]
 [ 22   6  19 860  52   0  36   0   4   1]
 [  0   0 100  14 799   0  87   0   0   0]
 [  0   0   0   1   0 954   0  28   1  16]
 [ 94   0  90  19  47   0 739   0  11   0]
 [  0   0   0   0   0   5   0 960   0  35]
 [  4   0   1   1   1   1   7   3 981   1]
 [  1   0   0   0   0   2   0  20   0 977]]

              precision    recall  f1-score   support

 T-shirt/top       0.86      0.81      0.84      1000
     Trouser       0.99      0.97      0.98      1000
    Pullover       0.78      0.82      0.80      1000
       Dress       0.93      0.86      0.89      1000
        Coat       0.81      0.80      0.81      1000
      Sandal       0.99      0.95      0.97      1000
       Shirt       0.68      0.74      0.71      1000
     Sneaker       0.95      0.96      0.95      1000
         Bag       0.98      0.98      0.98      1000
  Ankle boot       0.95      0.98      0.96      1000

    accuracy                           0.89     10000
   macro avg       0.89      0.89      0.89     10000
weighted avg       0.89      0.89      0.89     10000
```

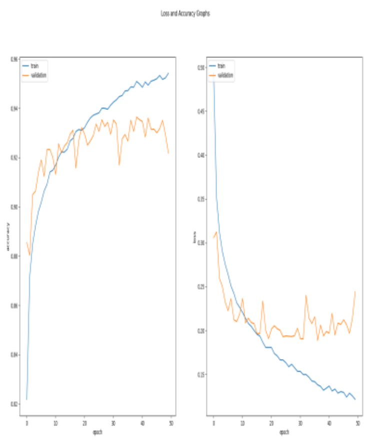
Figure 6: Results of ANN Models

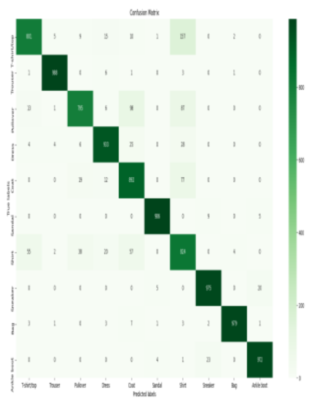
As we conclude the experiments on the ANN side we can move on to the experiments carried out on the CNN side. All of the three CNN models has been trained with 32 batch sizes and 50 epochs. Data augmentation operation was performed on these models. But while performing data augmentation, parameters like rotation_range, width_shift_range and height_shift_range were not used because these parameters has a negative effect on accuracy in this dataset. The reason for this situation may be that the positions of the images in the dataset do not contain too much variation, such as reversed or leaning.

First CNN model was trained under conditions of Adam optimizer with default learning rate. First experiment's results were not bad but has demonstrated a fluctuating learning performance. This shows that the model is not very stable while learning. However, it is necessary to say that CNN saw higher accuracy than ANN in the first experiment. On training dataset accuracy was around 0.95 and loss was around 0.12. On validation dataset accuracy was around 0.92 and loss was around 0.21. On test dataset accuracy of shirts were not bad but the accuracy of t-shirts has decreased slightly. Second CNN model was trained under conditions of SGD optimizer with default learning rate. Results were better than the previous one. The model has displayed a somewhat more stable learning process. On training dataset accuracy was around 0.95 and loss was around 0.11. On validation dataset accuracy was around 0.93 and loss was around 0.18. Although the accuracy of the shirts were the lowest among other items on the test dataset in this experiment, the accuracy of all items were at the desired levels. Also unit_norm() kernel constraint function has been added on the model. This function made the model learn a little faster. Third CNN model was trained under conditions of RMSprop optimizer with default learning rate. Third CNN model had similar results with previous one. But the third model has learned faster than the previous one.

In CNN models, accuracy reached a maximum of 0.93 and loss reached a minimum of 0.18 on the validation dataset. In the Conclusions section, a general evaluation about the study will be made and a conclusion will be reached.

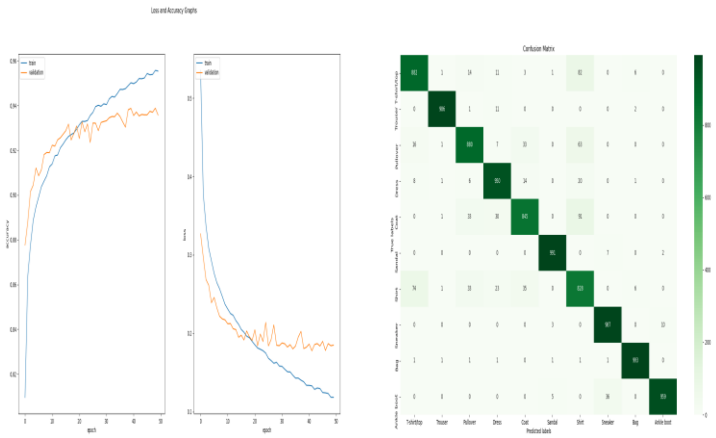
Results of first experiment





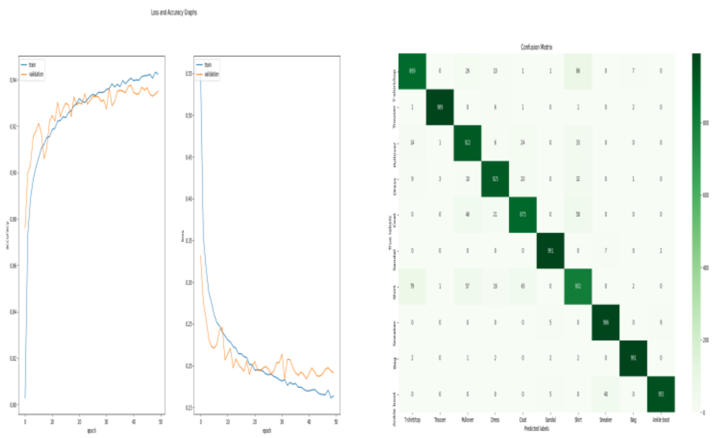
```
Confusion Matrix
[[801   5   9  15  10   1 157   0   2   0]
 [  1 988   0   6   1   0   3   0   1   0]
 [ 13   1 795   6  98   0  87   0   0   0]
 [  4   4   6 933  25   0  28   0   0   0]
 [  0   0  19  12 892   0  77   0   0   0]
 [  0   0   0   0   0 986   0   9   0   5]
 [ 55   2  38  20  57   0 824   0   4   0]
 [  0   0   0   0   0   5   0 975   0  20]
 [  3   1   0   3   7   1   3   2 979   1]
 [  0   0   0   0   0   4   1  23   0 972]]

              precision    recall  f1-score   support

 T-shirt/top       0.91      0.80      0.85      1000
     Trouser       0.99      0.99      0.99      1000
    Pullover       0.92      0.80      0.85      1000
       Dress       0.94      0.93      0.94      1000
        Coat       0.82      0.89      0.85      1000
      Sandal       0.99      0.99      0.99      1000
       Shirt       0.70      0.82      0.76      1000
     Sneaker       0.97      0.97      0.97      1000
         Bag       0.99      0.98      0.99      1000
  Ankle boot       0.97      0.97      0.97      1000

    accuracy                           0.91     10000
   macro avg       0.92      0.91      0.92     10000
weighted avg       0.92      0.91      0.92     10000
```

Results of second experiment



```
Confusion Matrix
[[882   1  14  11   3   1  82   0   6   0]
 [  0 986   1  11   0   0   0   0   2   0]
 [ 16   1 880   7  33   0  63   0   0   0]
 [  8   1   6 950  14   0  20   0   1   0]
 [  0   1  33  30 845   0  91   0   0   0]
 [  0   0   0   0   0 991   0   7   0   2]
 [ 74   1  33  23  35   0 828   0   6   0]
 [  0   0   0   0   0   3   0 987   0  10]
 [  1   1   1   1   0   1   1   1 993   0]
 [  0   0   0   0   0   5   0  36   0 959]]

              precision    recall  f1-score   support

 T-shirt/top       0.90      0.88      0.89      1000
     Trouser       0.99      0.99      0.99      1000
    Pullover       0.91      0.88      0.89      1000
       Dress       0.92      0.95      0.93      1000
        Coat       0.91      0.84      0.88      1000
      Sandal       0.99      0.99      0.99      1000
       Shirt       0.76      0.83      0.79      1000
     Sneaker       0.96      0.99      0.97      1000
         Bag       0.99      0.99      0.99      1000
  Ankle boot       0.99      0.96      0.97      1000

    accuracy                           0.93     10000
   macro avg       0.93      0.93      0.93     10000
weighted avg       0.93      0.93      0.93     10000
```

Results of third experiment



```
Confusion Matrix
[[869   0  26  10   1   1  86   0   7   0]
 [  1 989   0   6   1   0   1   0   2   0]
 [ 14   1 922   6  24   0  33   0   0   0]
 [  9   3  10 925  20   0  32   0   1   0]
 [  0   0  46  21 875   0  58   0   0   0]
 [  0   0   0   0   0 991   0   7   0   2]
 [ 79   1  57  16  43   0 802   0   2   0]
 [  0   0   0   0   0   5   0 986   0   9]
 [  2   0   1   2   0   2   2   0 991   0]
 [  0   0   0   0   0   5   0  40   0 955]]

              precision    recall  f1-score   support

 T-shirt/top       0.89      0.87      0.88      1000
     Trouser       0.99      0.99      0.99      1000
    Pullover       0.87      0.92      0.89      1000
       Dress       0.94      0.93      0.93      1000
        Coat       0.91      0.88      0.89      1000
      Sandal       0.99      0.99      0.99      1000
       Shirt       0.79      0.80      0.80      1000
     Sneaker       0.95      0.99      0.97      1000
         Bag       0.99      0.99      0.99      1000
  Ankle boot       0.99      0.95      0.97      1000

    accuracy                           0.93     10000
   macro avg       0.93      0.93      0.93     10000
weighted avg       0.93      0.93      0.93     10000
```

Figure 7: Results of CNN Models

6 Conclusions

Image classification is an important field in Computer Science. In this study, it has been observed that deep learning models give very successful results in this field. CNN is specialized version of ANN which makes convolutional and pooling operations on images then gives these images to fully connected network which is a non-customized ANN model. So after the experiments, it has been observed that CNN gives much more successful results than ANN in the field of image classification. In addition, it is necessary to say that many factors such as hyperparameters, epochs, data augmentation, batch normalization, dropout, optimizer, learning rate which have been applied on models have a significant effect on the results.

References

1. krishna, M & Neelima, M & Mane, Harshali & Matcha, Venu. (2018). Image classification using Deep learning. *International Journal of Engineering & Technology*. 7. 614. 10.14419/ijet.v7i2.7.10892.

2. arXiv:1708.07747

3. S. Bhatnagar, D. Ghosal and M. H. Kolekar, "Classification of fashion article images using convolutional neural networks," *2017 Fourth International Conference on Image Information Processing (ICIIP)*, Shimla, 2017, pp. 1-6, doi: 10.1109/ICIIP.2017.8313740.

4. Duan, Chao & Yin, Panpan & Zhi, Yan & Li Xingxing. (2019). Image classification of Fashion-mnist Data Set Based on VGG Network. *2nd International Conference on Information Science and Electronict Technology (ISET 2019)*. DOI: 10.23977/iset.2019.004

5. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html

6. https://stackoverflow.com/questions/50920908/get-confusion-matrix-from-a-keras-multiclass-model

7. https://gist.github.com/RyanAkilos/3808c17f79e77c4117de35aa68447045

8. https://stackoverflow.com/questions/40427435/extract-images-from-idx3-ubyte-file-or-gzip-via-python

9. https://datascience.stackexchange.com/questions/38604/too-low-accuracy-on-mnist-dataset-using-a-neural-network

10. https://machinelearningmastery.com/multi-class-classification-tutorial-keras-deep-learning-library/

11. https://medium.com/swlh/fully-connected-vs-convolutional-neural-networks-813ca7bc6ee5

12. https://github.com/zalandoresearch/fashion-mnist

13. https://hackernoon.com/deep-learning-vs-machine-learning-a-simple-explanation-47405b3eef08

14. https://developers.google.com/machine-learning/practica/image-classification

15. https://machinelearningmastery.com/save-load-keras-deep-learning-models/

16. https://machinelearningmastery.com/display-deep-learning-model-training-history-in-keras/

17. https://www.researchgate.net/post/How_to_improve_CNN_accuracy

18. https://towardsdatascience.com/build-your-first-cnn-fb3aaad77038

19. https://machinelearningmastery.com/batch-normalization-for-training-of-deep-neural-networks/

20. https://machinelearningmastery.com/how-to-develop-a-cnn-from-scratch-for-fashion-mnist-clothing-classification/

21. https://medium.com/@mjbhobe/classifying-fashion-with-a-keras-cnn-achieving-94-accuracy-part-1-1ffcb7e5f61a

22. https://www.analyticsvidhya.com/blog/2020/09/overfitting-in-cnn-show-to-treat-overfitting-in-convolutional-neural-networks/

23. https://www.kaggle.com/gpreda/cnn-with-tensorflow-keras-for-fashion-mnist

24. https://ai.stackexchange.com/questions/5546/what-is-the-difference-between-a-convolutional-neural-network-and-a-regular-neur#:~:text=The%20class%20of%20ANN%20covers,just%20one%20kind%20of%20ANN.&text=A%20CNN%2C%20in%20specific%2C%20has,more%20layers%20of%20convolution%20units.

25. https://www.kaggle.com/kanncaa1/deep-learning-tutorial-for-beginners

26. https://www.kaggle.com/kanncaa1/convolutional-neural-network-cnn-tutorial