Project Report

Course Name: Pattern Recognition Laboratory

Course Code: CSI 416 (A)

Project Name

Fashion Cloth Detection using CNN

Team Members

Name	Student ID	Email
Md. Al Emran Hossain	011181144	mhossain181144@bscse.uiu.ac.bd
Mashuke Alam Jim	011181145	mjim181145@bscse.uiu.ac.bd
Azizul Hakim	011181149	ahakim181149@bscse.uiu.ac.bd
Mohammad Mostafizur Rahman	011181140	mrahman181140@bscse.uiu.ac.bd

Problem Definition

MNIST is one of the famous datasets used by members of the AI/ML/Data Science community. They love this dataset and use it as a benchmark to validate their algorithms. Fashion-MNIST is a direct drop-in replacement for the original MNIST dataset. In our project we want to predict real life images through the labels from this trending dataset in Kaggle.

Dataset Description

Fashion MNIST dataset contains a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 grayscale image, associated with a label from 10 classes. Each image is 28 pixels in height and 28 pixels in width, for a total of 784 pixels in total. Each pixel has a single pixel-value associated with it, indicating the lightness or darkness of that pixel, with higher numbers meaning darker. This pixel-value is an integer between 0 and 255. The training and test data sets have 785 columns. The first column consists of the class labels (see above), and represents the article of clothing. The rest of the columns contain the pixel-values of the associated image. The dataset is used as a benchmark to validate different algorithms.

Figure: Dataset sample image and corresponding label.

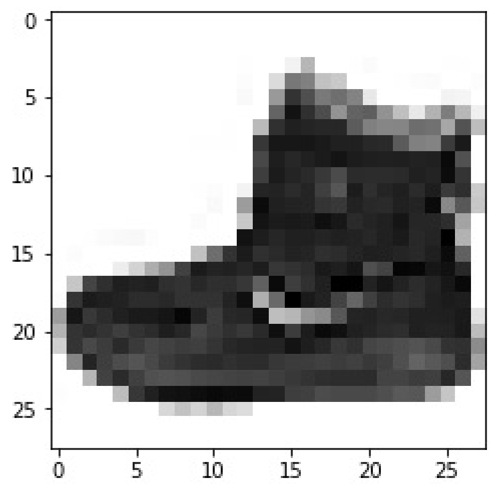


Figure: A single image of size 28 * 28

Attribute Information

i. Label: Unique identifier

Total 10 labels (0-9) in the label column.Each training example is assigned to one of the following labels:

- 0 T-shirt/top
- 1 Trouser
- 2 Pullover
- 3 Dress
- 4 Coat
- 5 Sandal
- 6 Shirt
- 7 Sneaker
- 8 Bag
- 9 Ankle boot

ii. Pixel [1-784]: The image has a corresponding 784 columns.

Note: There is no class column for the test dataset.

Dataset Credit: Zalando

Our proposed Approach

We have used Convolutional Neural Network (CNN) over Fashion-MNIST dataset to establish our model. Though this dataset is almost processed, we had to make slight changes or we'd preprocessed for the input to our model. Tensorflow is the state of the art library that we've used here.

(I) To create our CNN model we have initialized two convolutional layers (Conv2D) and applied a MaxPool (MaxPooling2D) function after each convolutional layer.

(II) After learning features from the CNN model we've flattened it so that we can make an input for the neural network. Multi-dimensional shapes can't be direct inputs to Neural Networks. And that is why we need flattening. Flattening is converting the data into a 1-dimensional array for inputting it to the next layer. We flatten the output of the convolutional layers to create a single long feature vector.

(III) This flattened vector is added to the dense function which has got a 64 fully connected dense layer in the model. The rectified linear activation function or ReLU for short is a piecewise linear function that will output the input directly if it is positive, otherwise, it will output zero. That is why we've used it in our model.

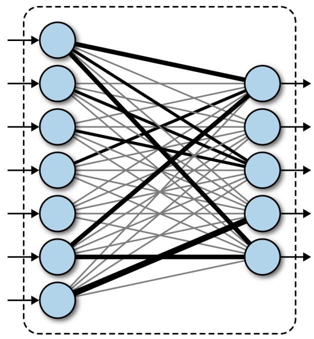


Figure: Densely connected network

(IV) In the similar way we added another fully connected dense layer where the input was from the output of the previous layer and it got output of 32 fully connected dense layers.

(V) After adding the same activation function we passed it to the output layer which defined 10 individual labels. We've used the softmax activation function for multiclass classification. It gives the probability of each individual label and the maximum probability is defined as the output.

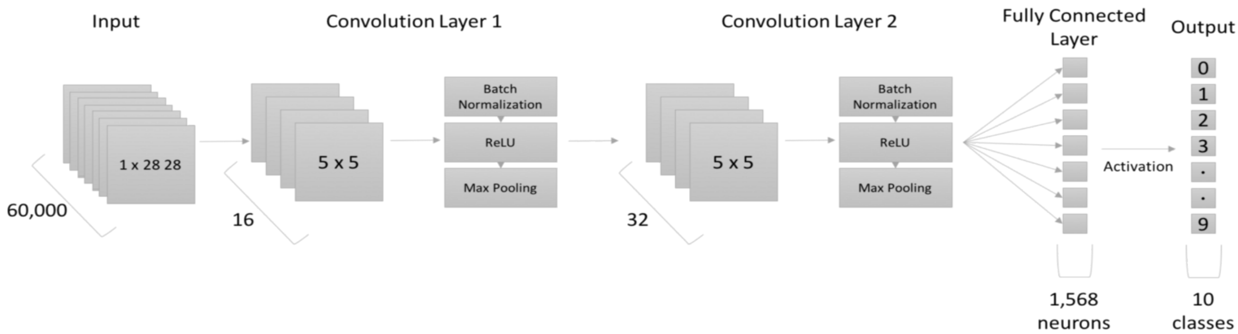


Figure: Built CNN Model

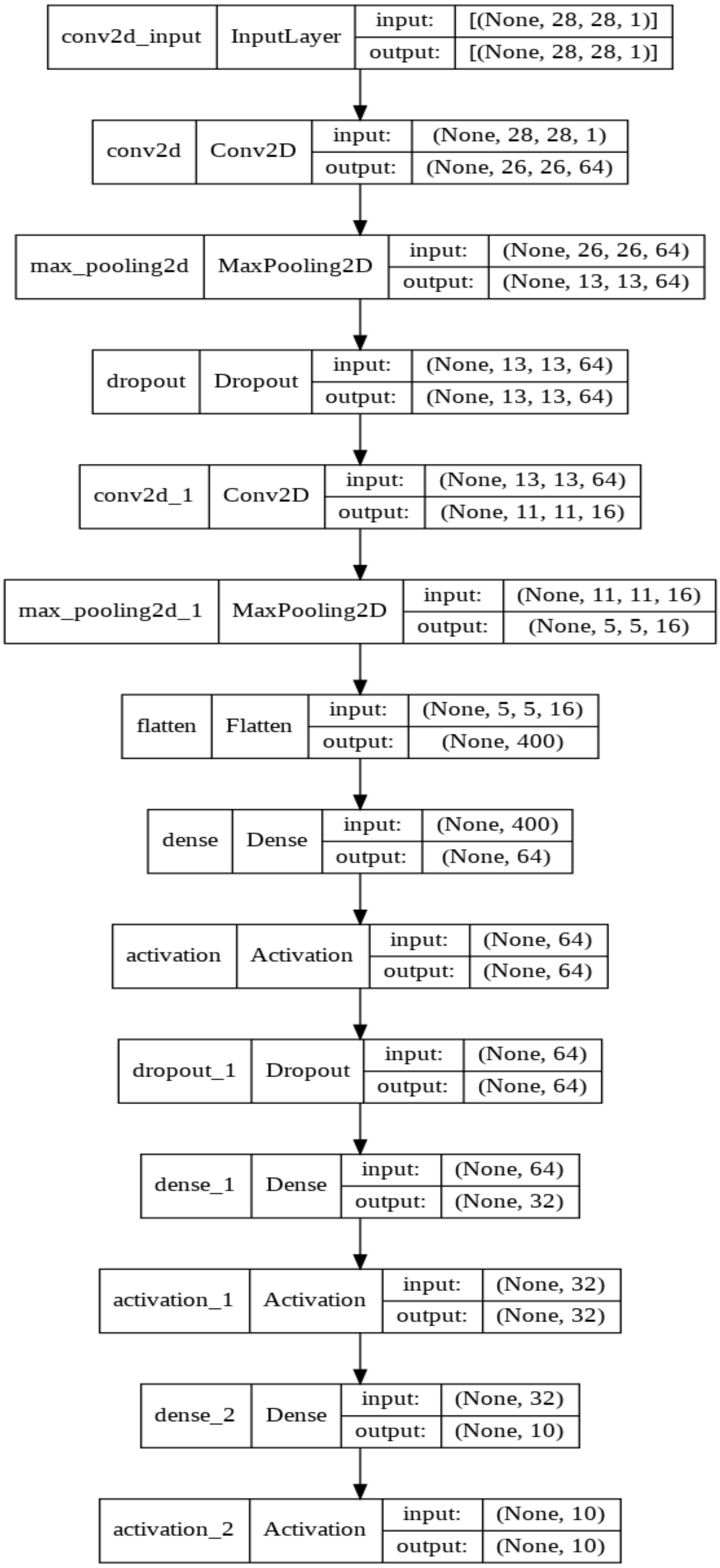


Figure: Convolutional Neural Network Architecture

This is how we've built our model.

(VI) As the model requires a lot of time to run and huge computational power, we used a callback method of keras named **ModelCheckpoint.**

(VII) Then we've trained the model by fit function and predict the accuracy of the model.

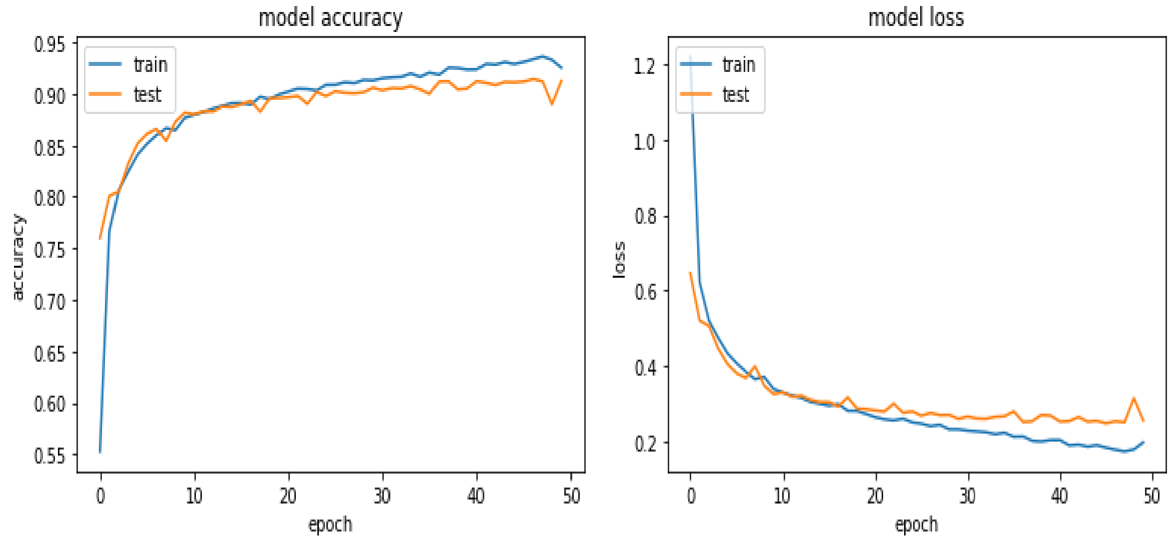


Figure: Model accuracy (right), Model loss(left) for 50 epochs

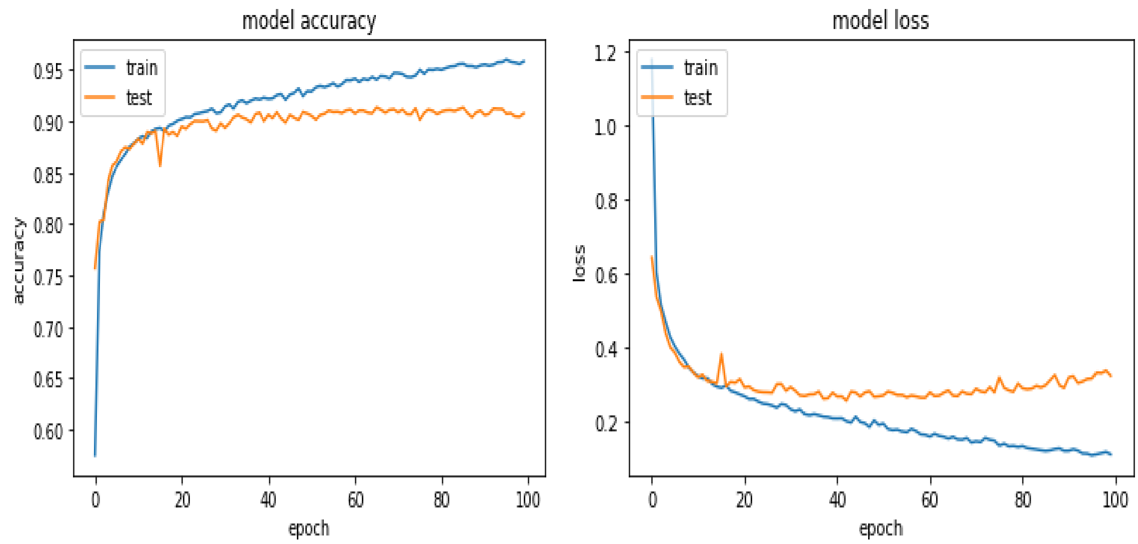
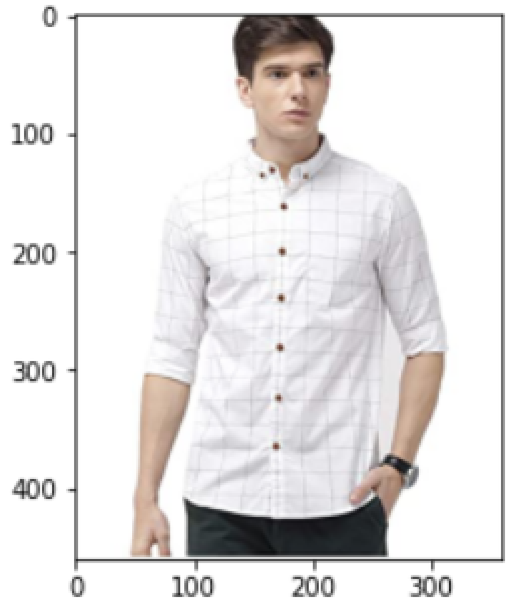
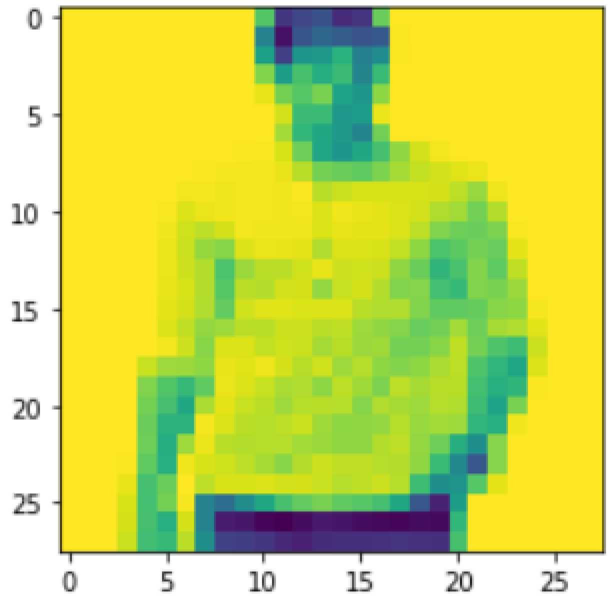


Figure: Model accuracy (right), Model loss(left) for 100 epochs

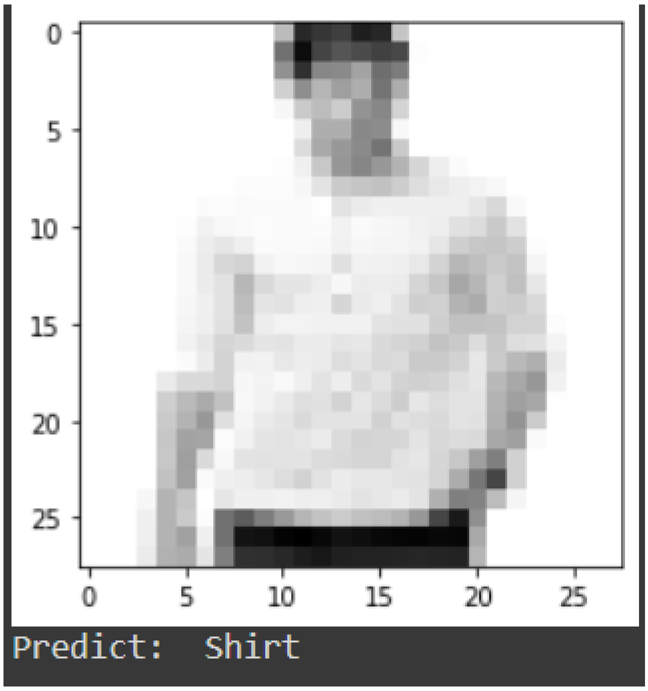
(VIII) We also have tried to predict the model by comparing a real life example of fashion image.



(a)



(b)



(c)

Figure: (a) Raw RGB image of size 450*350

(b) Converted 28 * 28

(c) Gray scale image with prediction.

Problem Faced:

- **I.** The dataset was large and this caused more time to run the model.
- **II.** It also needed huge computational power and GPU support.

Result/Comparison Metrics

We'll use accuracy to compare our model with the others existing state-of-the art models.

SL	Model Name	Error (Percentage)	Accuracy (Percentage)
1	**Fine-Tuning DARTS (State-of-the art)**	3.09	96.91
2	Shake-Shake (SAM)	3.59	96.41
3	PreAct-ResNet18 + FMix	3.64	95.93
4	VGG8B(2x)	4.14	94.54
5	**Our Proposed Model**	**26.77**	**90.79**