# Implementation of simple CNN architecture on MNIST dataset

Hridoy Pal

110146466

School of Computer Science

University of Windsor

### CNN Architecture proposed as follows (Model - 1):

Input Convolution layer (Conv2D), 32 filters, kernel size 5, activation Relu followed by a MaxPooling2D layer, and BatchNormalization.

Convolution layer (Conv2D), 64 filters, kernel size 5, activation Relu followed by a Maxpooling2D layer, and BatchNormalization.

Convolution layer (Conv2D), 128 filters, kernel size 3, activation Relu followed by a MaxPooling2D layer, and BatchNormalization.

Flatten layer.

Dropout probability 0.5

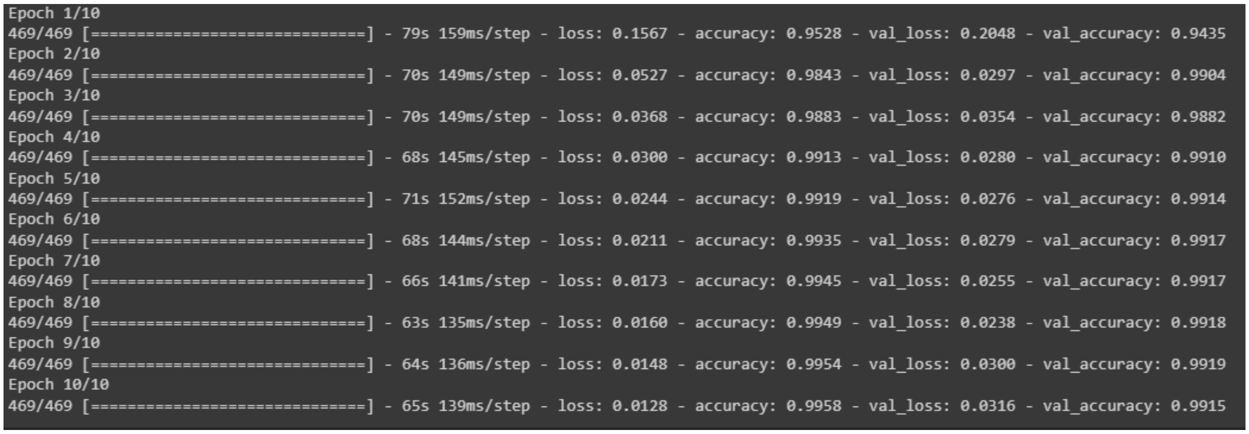
Dense layer with 10 classes.

Implemented the model using Keras API which runs over TensorFlow.

### Results Evaluation:

Training accuracy: 99.58%

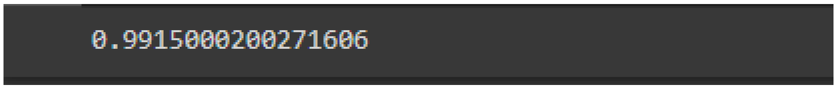
Validation accuracy: 99.15%



```
Epoch 1/10
469/469 [==============================] - 79s 159ms/step - loss: 0.1567 - accuracy: 0.9528 - val_loss: 0.2048 - val_accuracy: 0.9435
Epoch 2/10
469/469 [==============================] - 70s 149ms/step - loss: 0.0527 - accuracy: 0.9843 - val_loss: 0.0297 - val_accuracy: 0.9904
Epoch 3/10
469/469 [==============================] - 70s 149ms/step - loss: 0.0368 - accuracy: 0.9883 - val_loss: 0.0354 - val_accuracy: 0.9882
Epoch 4/10
469/469 [==============================] - 68s 145ms/step - loss: 0.0300 - accuracy: 0.9913 - val_loss: 0.0280 - val_accuracy: 0.9910
Epoch 5/10
469/469 [==============================] - 71s 152ms/step - loss: 0.0244 - accuracy: 0.9919 - val_loss: 0.0276 - val_accuracy: 0.9914
Epoch 6/10
469/469 [==============================] - 68s 144ms/step - loss: 0.0211 - accuracy: 0.9935 - val_loss: 0.0279 - val_accuracy: 0.9917
Epoch 7/10
469/469 [==============================] - 66s 141ms/step - loss: 0.0173 - accuracy: 0.9945 - val_loss: 0.0255 - val_accuracy: 0.9917
Epoch 8/10
469/469 [==============================] - 63s 135ms/step - loss: 0.0160 - accuracy: 0.9949 - val_loss: 0.0238 - val_accuracy: 0.9918
Epoch 9/10
469/469 [==============================] - 64s 136ms/step - loss: 0.0148 - accuracy: 0.9954 - val_loss: 0.0300 - val_accuracy: 0.9919
Epoch 10/10
469/469 [==============================] - 65s 139ms/step - loss: 0.0128 - accuracy: 0.9958 - val_loss: 0.0316 - val_accuracy: 0.9915
```

Fig – 1: Training Results

Test accuracy: 99.15%



```
0.9915000200271606
```

Fig -2: Test Result

## Learning Curves:

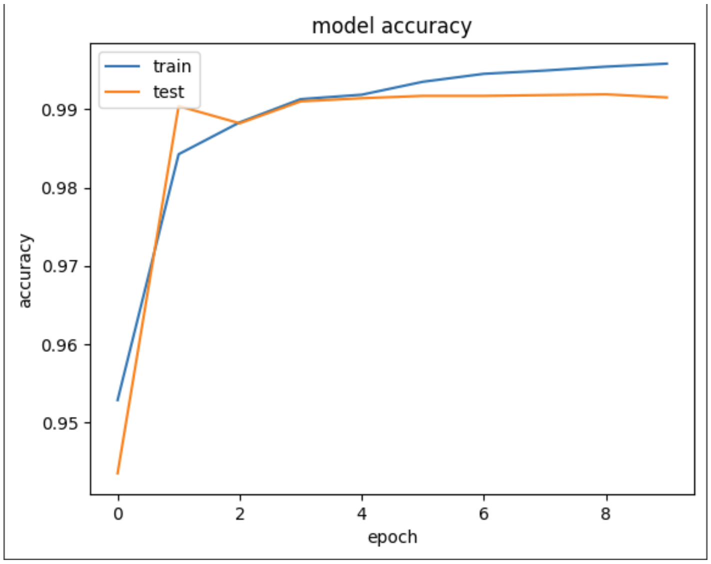


Fig – 3: Model -1 Accuracy

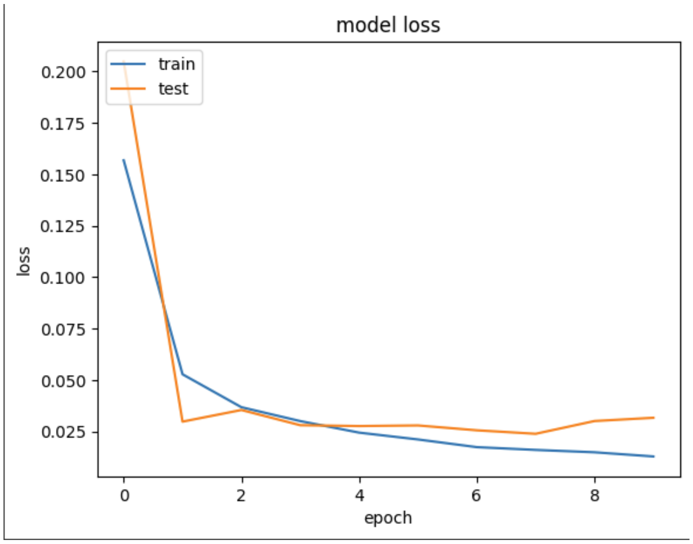
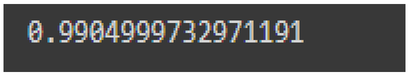


Fig – 4: Model -1 Loss

The model is performing very well on both train and test data with 99.58% and 99.15% accuracy.

## Without BatchNormalization & Dropout (Experimental Model):

Accuracy:



```
0.9904999732971191
```

Fig – 5: Experimental Model Accuracy

Learning Curves:

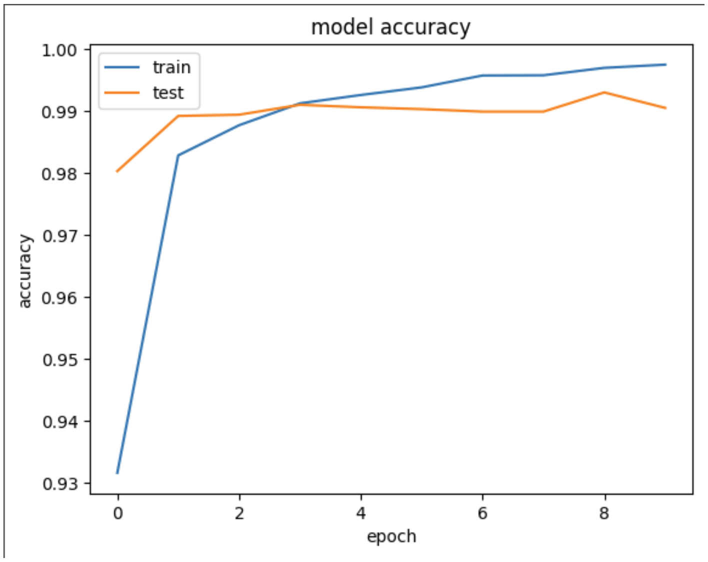


Fig – 6: Experimental Model Accuracy Learning Curve

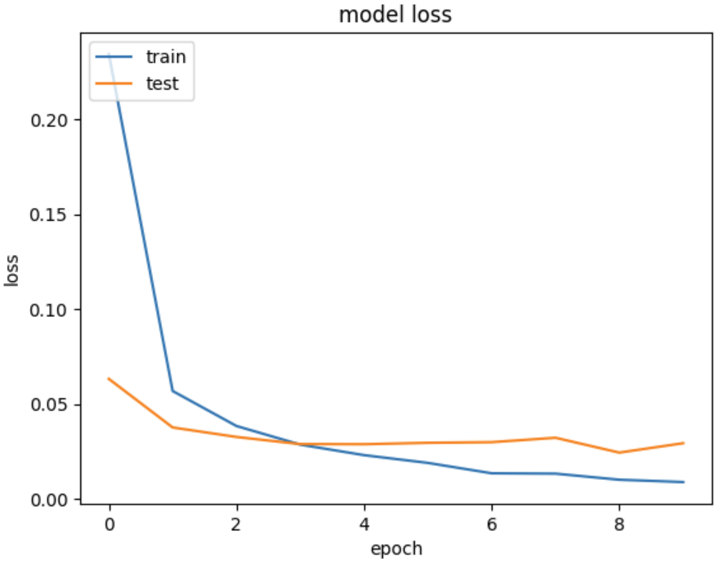


Fig – 7: Experimental Model Loss Curve

Here, the distance between the learning curves for train and test data is bigger compared to Model – 1 (Fig - 3 & Fig – 4). This means that experiment model is overfitting the data. In the experiment model, test accuracy is lower at 99.04%.

## Summary:

By adding BatchNormalization and Dropout (0.5) to Model - 1, data overfitting was slightly reduced, and the accuracy improved to 99.15%. BatchNormalization and Dropout helped to gain higher accuracy and less overfitting.

## Model Comparison:

| CNN Implementation (Model-1) | Existing CNN Implementation |
|---|---|
| Accuracy: 99.15 | SOPCNN [Yahia Saeed Assiri et al. 2020]<br>Accuracy: 99.83 |
| | Branching/Merging CNN + Homogeneous Vector Capsules [Adam Byerly et al. 2020]<br>Accuracy: 99.87 |
| | Ensemble learning in CNN augmented with fully connected subnetworks [ Daiki Hirata et al. 2020]<br>Accuracy: 9984 |
| | Regularization of Neural Networks using DropConnect [ Li Wan et al. 2013]<br>Accuracy: 99.77 |