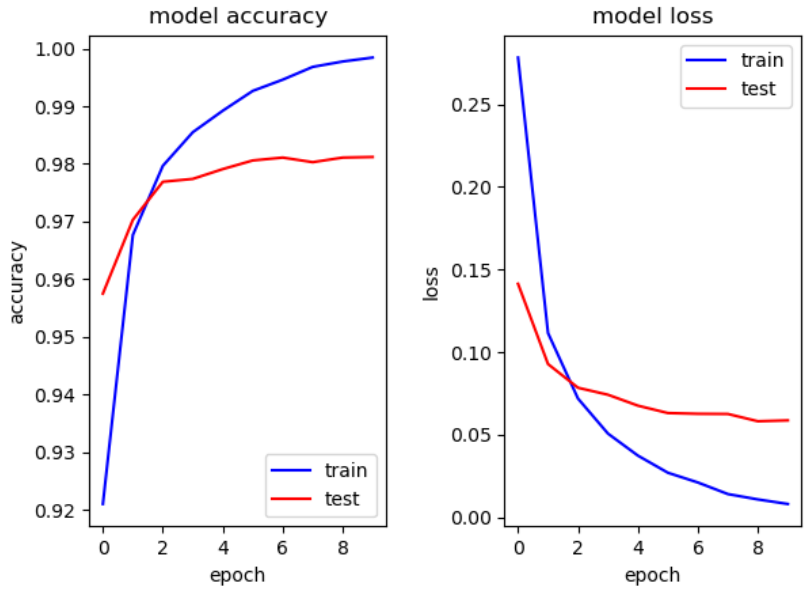
Deep Learning for Advanced Robot Perception Assignment 4
Nathan Stallings
10/18/2021

**Multi-layer Perceptron**

I began by running the mnist_mlp_baseline.py to see what prediction error it reported. I found that without any modifications, the base model produced the following results:

Baseline Error: 1.88%




I started by modifying the batch size to see how that impacted the model:

| | |
|---|---|
| Batch_size=500 | Baseline Error: 1.82% |
| Batch_size=32 | Baseline Error: 2.36% |

I reset the batch_size to see how the number of epochs affected the error.

| | |
|---|---|
| nb_epoch=5 | Baseline Error: 2.14% |
| nb_epoch=30 | Baseline Error: 1.63% |

A larger number of epochs increases performance, so the rest of my testing uses 30 epochs. The next thing I tried was increasing the number of layers. I found that adding layers of 1000, 256, and 128 worked well.

```
model.add(Dense(1000, input_dim=num_pixels, init='normal',
activation='relu'))
model.add(Dense(num_pixels, init='normal', activation='relu'))
model.add(Dense(256, init='normal', activation='relu'))
model.add(Dense(128, init='normal', activation='relu'))
model.add(Dense(num_classes, init='normal', activation='softmax'))
```

**Baseliner Error: 1.49%**

I then added drop out layers and batch normalization layers to stabilize the training process and prevent overfitting.

```
model.add(Dense(1000, input_dim=num_pixels, init='normal',
activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(.5))
model.add(Dense(num_pixels, init='normal', activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(.5))
model.add(Dense(256, init='normal', activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(.5))
model.add(Dense(128, init='normal', activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(.5))
model.add(Dense(num_classes, init='normal', activation='softmax'))
```

**Baseliner Error: 1.35%**

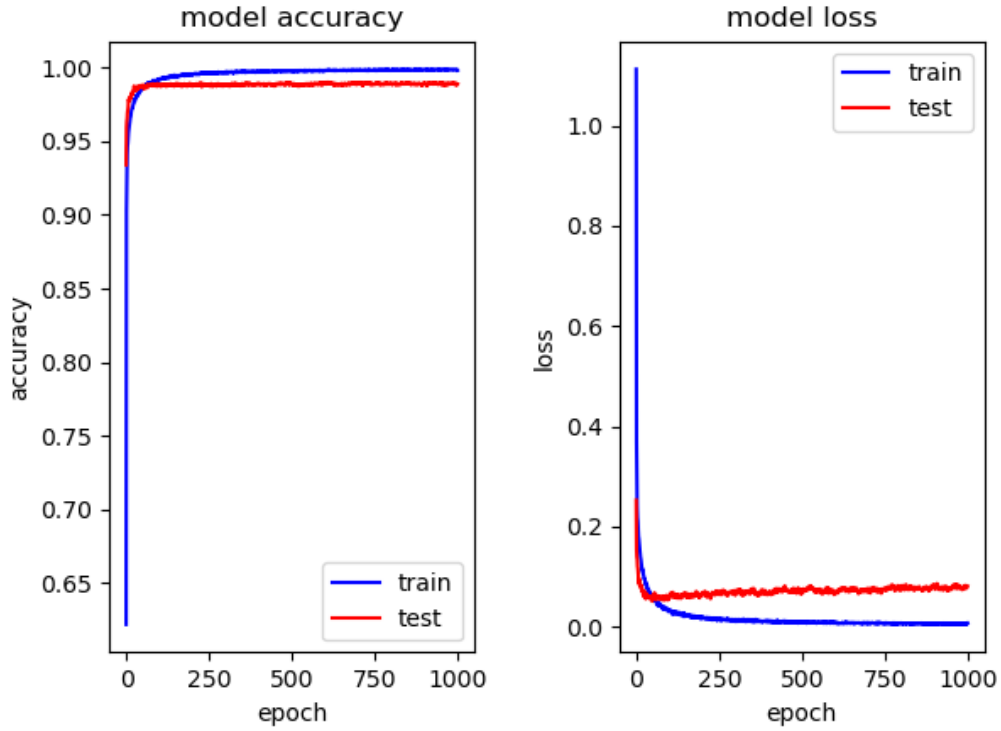
Deep Learning for Advanced Robot Perception Assignment 4
Nathan Stallings
10/18/2021

I continued to add layers and modify the batch_size through a trial-and-error process to improve the model. I had a hard time improving the model past an average error of 1.10%. I continued to modify batch_size and drop out rates until I saw it reach 1.07%. This occurred with a **batch_size of 104** and I let it run for **1000 epochs** overnight. The final model and results were as follows:

```
model.add(Dense(1000, input_dim=num_pixels, init='normal',
activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(.57))
model.add(Dense(num_pixels, init='normal', activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(.3))
model.add(Dense(num_pixels, init='normal', activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(.57))
model.add(Dense(256, init='normal', activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(.56))
model.add(Dense(256, init='normal', activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(.15))
model.add(Dense(256, init='normal', activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(.72))
model.add(Dense(256, init='normal', activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(.35))
model.add(Dense(128, init='normal', activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(.6))
model.add(Dense(num_classes, init='normal', activation='softmax'))
```

**Baseliner Error: 0.99%**

Deep Learning for Advanced Robot Perception Assignment 4
Nathan Stallings
10/18/2021




The final baseline error was **0.99%,** which completes the assignment. I found the lowest error occurred at epoch 608 which was an error of **0.97%.** While these errors are good, they are not consistent and didn't appear on every run of the model.

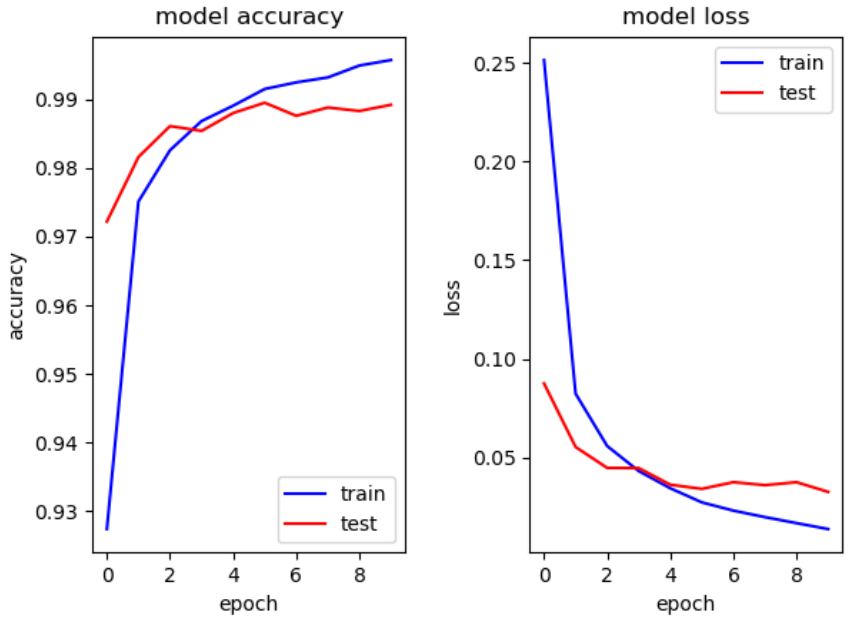
```
Epoch 608/1000
 - 10s - loss: 0.0084 - accuracy: 0.9982 - val_loss: 0.0668 - val_accuracy: 0.9903
Epoch 609/1000
 - 10s - loss: 0.0055 - accuracy: 0.9988 - val_loss: 0.0734 - val_accuracy: 0.9897
Epoch 610/1000
 - 10s - loss: 0.0065 - accuracy: 0.9983 - val_loss: 0.0724 - val_accuracy: 0.9901
```

Deep Learning for Advanced Robot Perception Assignment 4
Nathan Stallings
10/18/2021

**Convolutional Neural Network**

I began by running the mnist_cnn.py to see what prediction error it reported. I found that without any modifications, the base model produced the following results:

CNN Error: 1.08%




I began by modifying the batch size and found the following:

| | |
|---|---|
| Batch_Size = 300 | CNN Error: 1.05% |
| Batch_Size = 128 | CNN Error: 1.31% |

While it looks like a larger batch size is better, I know from experience that increasing the number of epochs will change these results. I increased the number of epochs to 50 and found the following results:

nb_epoch = 50

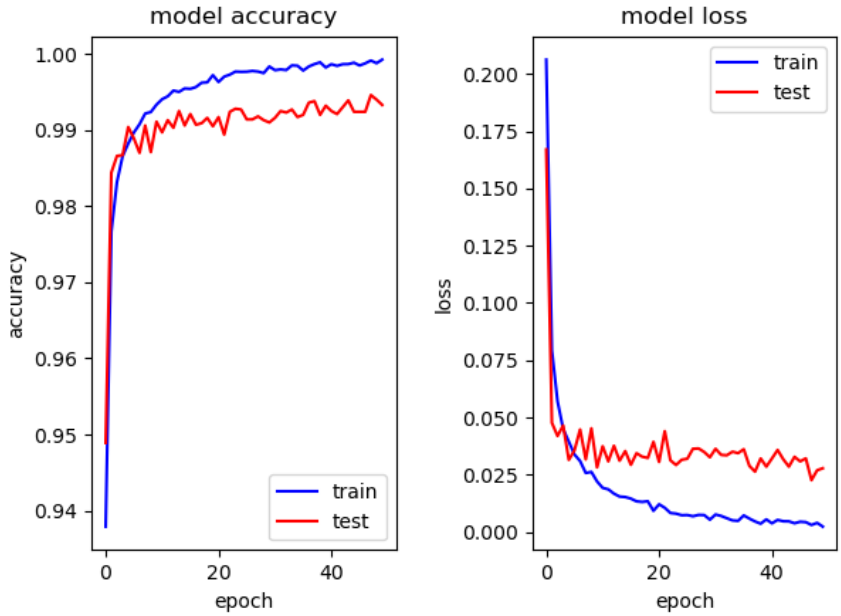
| | |
|---|---|
| Batch_Size = 300 | CNN Error: 1.02% |
| Batch_Size = 128 | CNN Error: 0.97% |

Deep Learning for Advanced Robot Perception Assignment 4
Nathan Stallings
10/18/2021

While this completed the assignment, I knew I could improve the model. I added a convolution layer and two dense layers. I modified the drop out value and found that 0.45 worked the best. The next thing I added was batch normalization to stabilize the training process. I chose 500, 250, 100 for the neurons in the dense layer because I have used thes values for past discussion board problems and found they had good performance. Putting it all together and running the model for 50 epochs with a batch_size of 128 produced the following results:

```
model = Sequential()
model.add(Convolution2D(32, 5, 5, border_mode='valid', input_shape=(1, 28, 28),
activation='relu',  dim_ordering ='th'))
model.add(Convolution2D(32, 5, 5, activation='relu', border_mode='valid'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(.45))
model.add(Flatten())
model.add(Dense(500, activation='relu'))
model.add(BatchNormalization())
model.add(Dense(250, activation='relu'))
model.add(BatchNormalization())
model.add(Dense(100, activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(.45))
model.add(Dense(num_classes, activation='softmax'))
```

**CNN Error: 0.67%**



Deep Learning for Advanced Robot Perception Assignment 4
Nathan Stallings
10/18/2021

A CNN error of 0.67 is very good. Below is a screenshot of the last few epochs. We can see that the model's best performance was on epoch 48 where it had a **cnn_error of 0.54%**

```
Epoch 48/50
 - 14s - loss: 0.0030 - accuracy: 0.9991 - val_loss: 0.0226 - val_accuracy: 0.9946
Epoch 49/50
 - 14s - loss: 0.0039 - accuracy: 0.9988 - val_loss: 0.0269 - val_accuracy: 0.9940
Epoch 50/50
 - 14s - loss: 0.0023 - accuracy: 0.9992 - val_loss: 0.0278 - val_accuracy: 0.9933
CNN Error: 0.67%
```