Contents

1 MNIST Classification Using CNN

In this section, we consider three architectures of the CNN which are presented below:

- Model 1 - 1 Convolutional Layer
- Model 2 - 2 Convolutional Layer
- Model 3 - 2 Convolutional Layer + 1 hidden fully connected layer

The fully description of the architecture is given in each corresponding section.

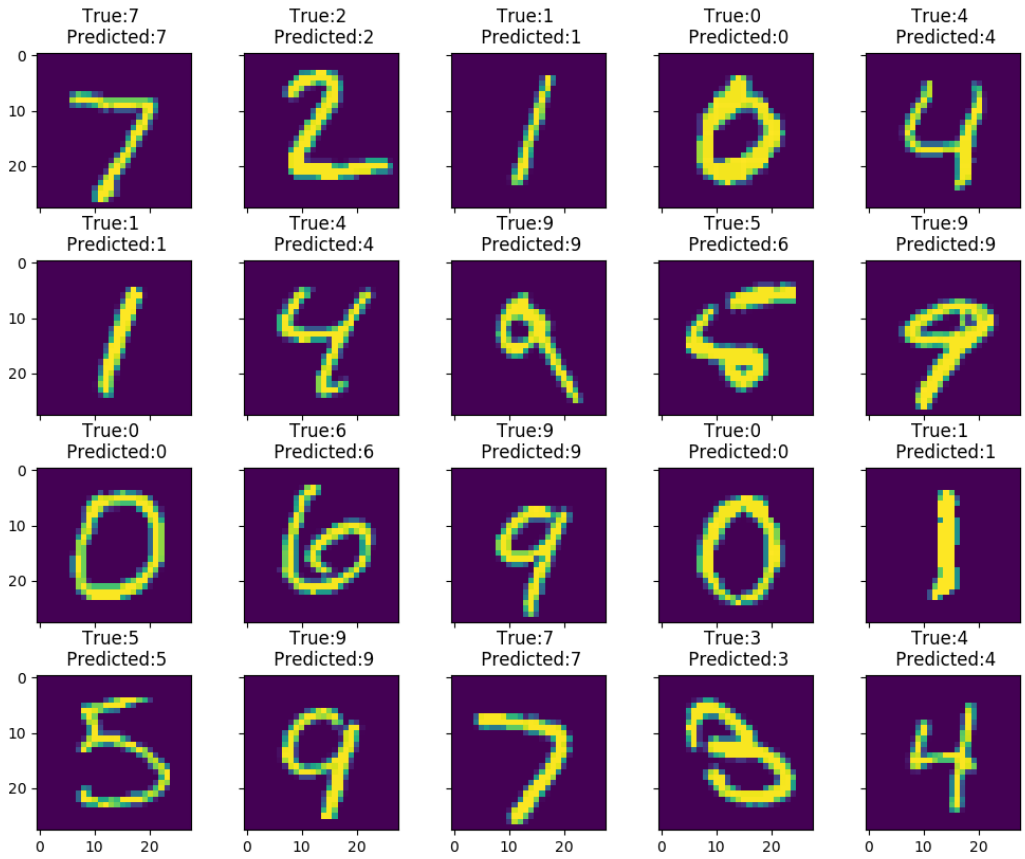
1.1 Model 1 - 1 Convolutional Layer

1.1.1 Detailed Architecture

The 1 Convolutional Layer CNN has the following architecture:

- Input
- Conv Layer
  - Number of Filters = 32
  - Filter Size - 3×3
  - Stride = 1
  - Zero Padding of 1
- Max Pool Layer
  - 2×2 Max Pooling
  - Stride = 2
- Fully Connected (10 Outputs)
- Softmax Classifer

1.1.2 Plot of Randomly Selected Test Images



1.2 Model 2 - 2 Convolutional Layer

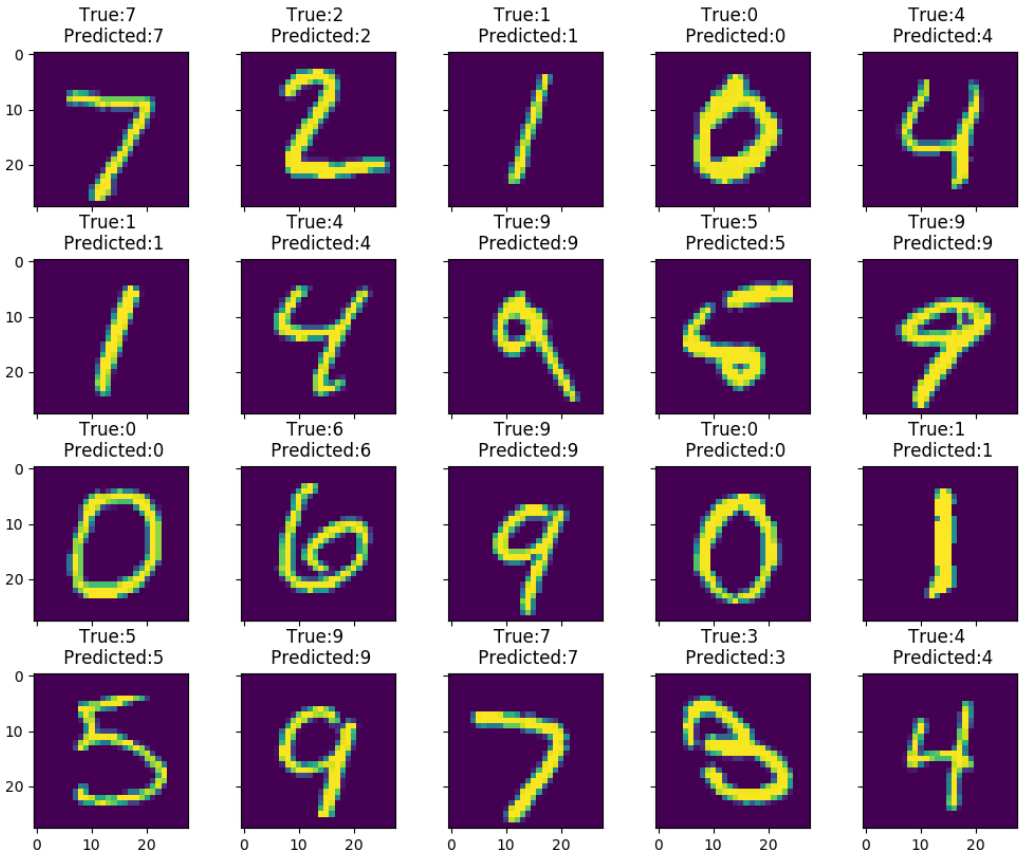
1.2.1 Detailed Architecture

The 2 Convolutional Layer CNN has the following architecture:

- Input
- Conv Layer 1
  - Number of Filters = 32

  - Filter Size - 3×3
  - Stride $= 1$
  - Zero Padding of 1
- Max Pool Layer 1
  - 2×2 Max Pooling
  - Stride $= 2$
- Conv Layer 2
  - Number of Filters $= 32$
  - Filter Size - 3×3
  - Stride $= 1$
  - Zero Padding of 1
- Max Pool Layer 2
  - 2×2 Max Pooling
  - Stride $= 2$
- Fully Connected (10 Outputs)
- Softmax Classifer

1.2.2 Plot of Randomly Selected Test Images



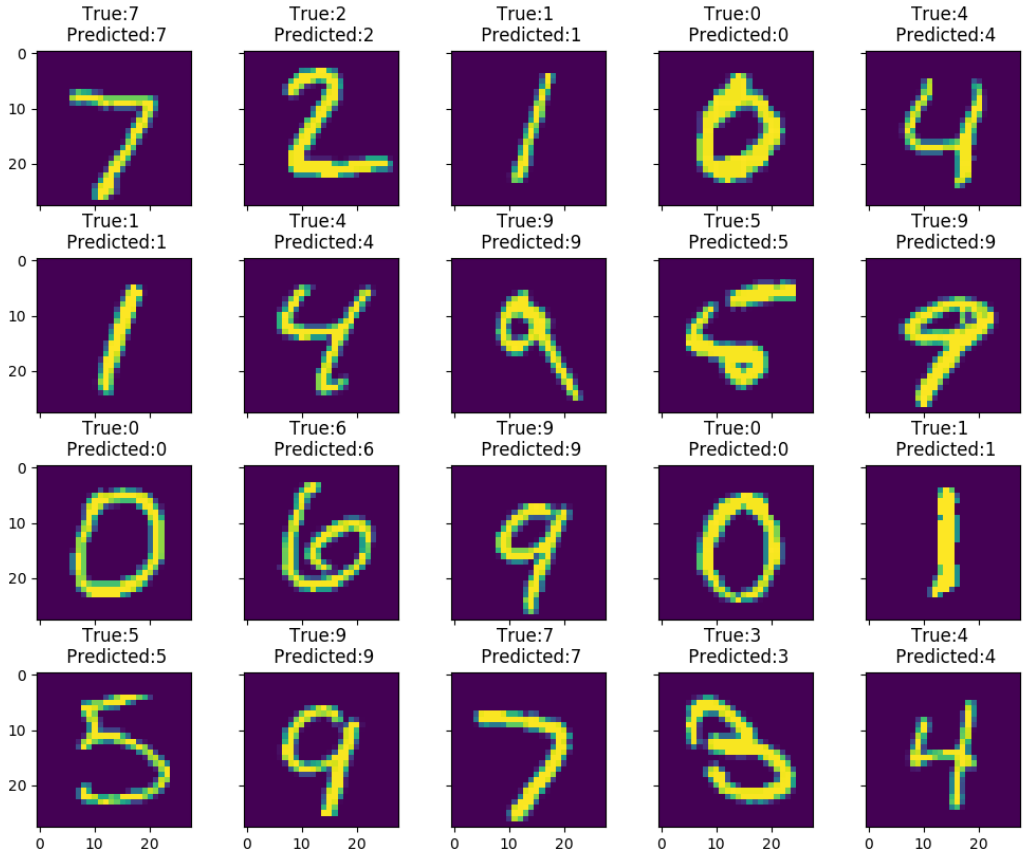
1.3 Model 3 - 2 Convolutional Layer + 1 hidden fully connected layer

1.3.1 Detailed Architecture

The 2 Convolutional Layer + 1 hidden fully connected layer CNN has the following architecture:

- Input
- Conv Layer 1
  - Number of Filters = 32
  - Filter Size - 3×3
  - Stride = 1
  - Zero Padding of 1
- Max Pool Layer 1
  - 2×2 Max Pooling
  - Stride = 2
- Conv Layer 2
  - Number of Filters = 32
  - Filter Size - 3×3
  - Stride = 1
  - Zero Padding of 1
- Max Pool Layer 2
  - 2×2 Max Pooling
  - Stride = 2
- Fully Connected (500 Outputs)
- Fully Connected (10 Outputs)
- Softmax Classifer

1.3.2 Plot of Randomly Selected Test Images

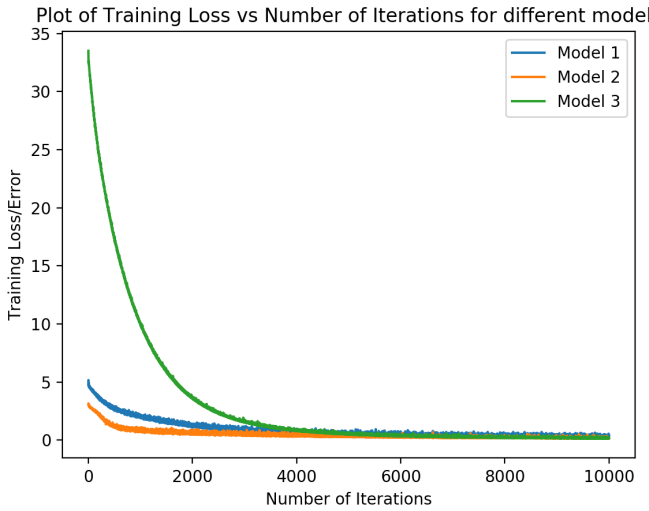


1.4 Comparison of the three models

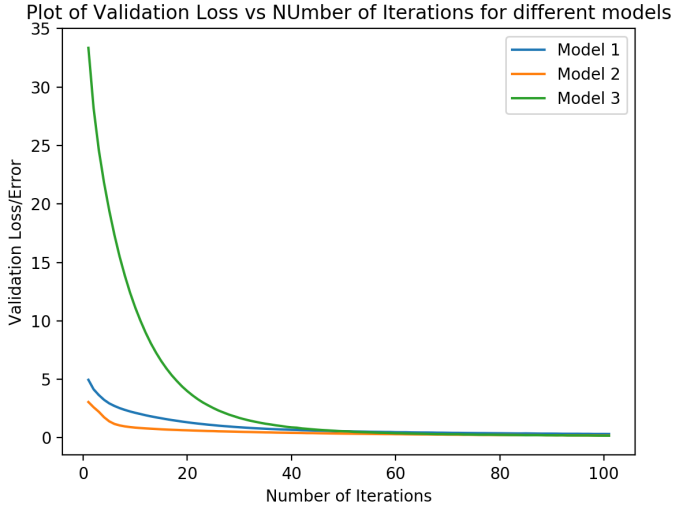
The three models are compared on the following three metrics :

- Training Loss Plot
- Validation Loss Plot
- Test Accuracy

1.4.1 Training Loss Plot

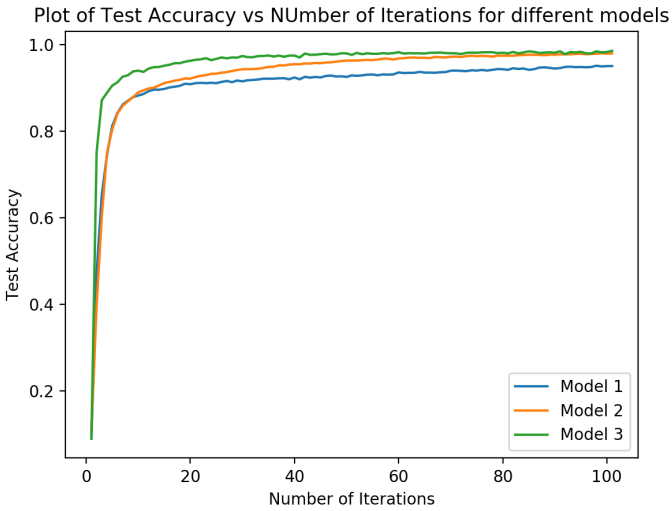


1.4.2 Validation Loss Plot



Note that since the validation loss plot is coming down with the increasing number of iterations, we can conclude that none of the models are over-fitting and regularization seems to be working. Also, note that even though it may not be very apparent from the graph, but Model 3 gives the least validation loss/error after 10000 iterations of training.

1.4.3 Test Accuracy



Model	Test Accuracy
Model 1 - 1 Convolutional Layer	95.06%
Model 2 - 2 Convolutional Layer	97.96%
Model 3 - 2 Convolutional Layer + 1 Hidden Layer	98.5%

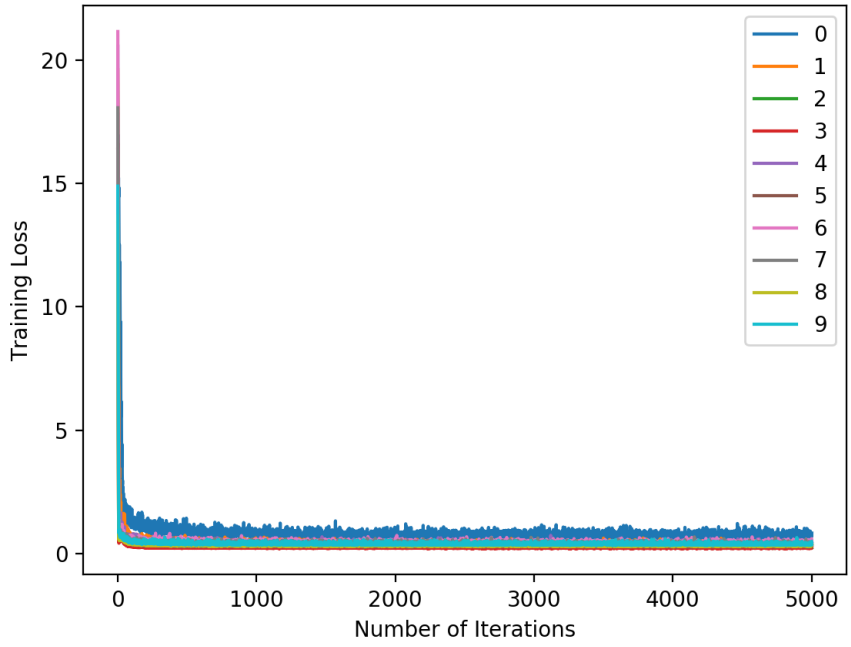
We can clearly see that Model 3 outperforms the other models in terms of **Test Accuracy** and hence is the best model learnt. We make use of this learnt model in our future experimentation.

2 Generating Adversarial Examples

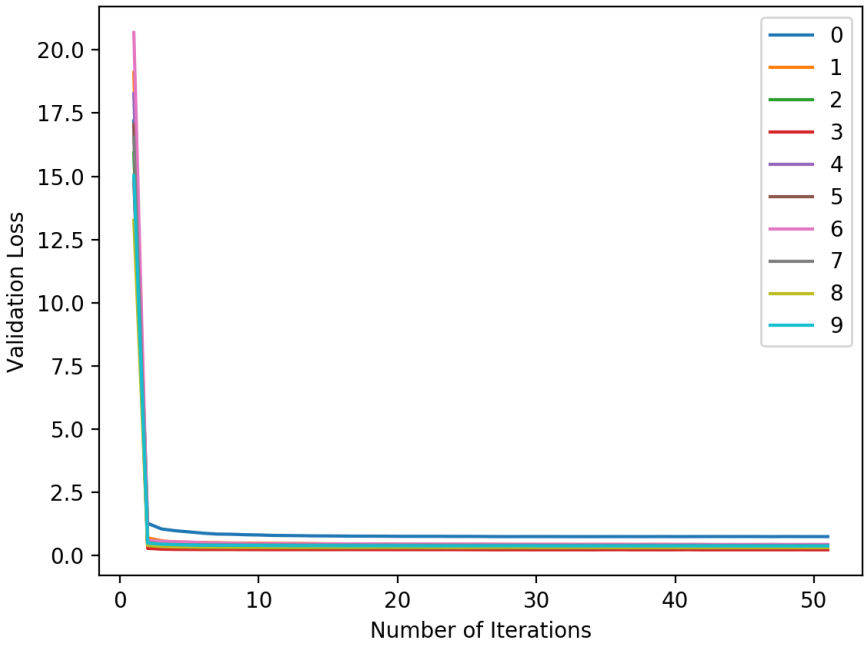
2.1 Learning Curves

Presented below are the learning curves for generating the noise masks for each of the classes from 0 to 9.

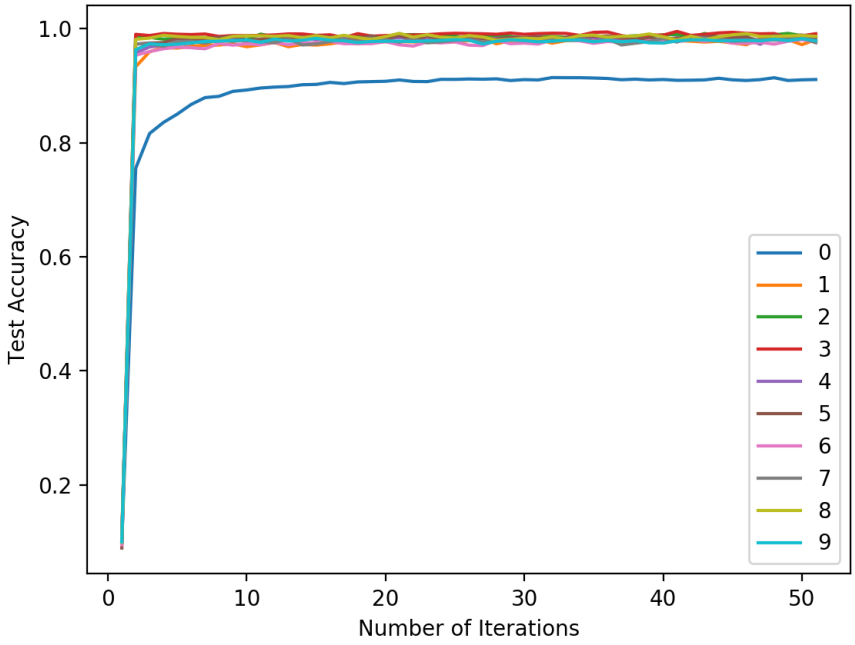
2.1.1 Plot of Training Loss for all 10 classes



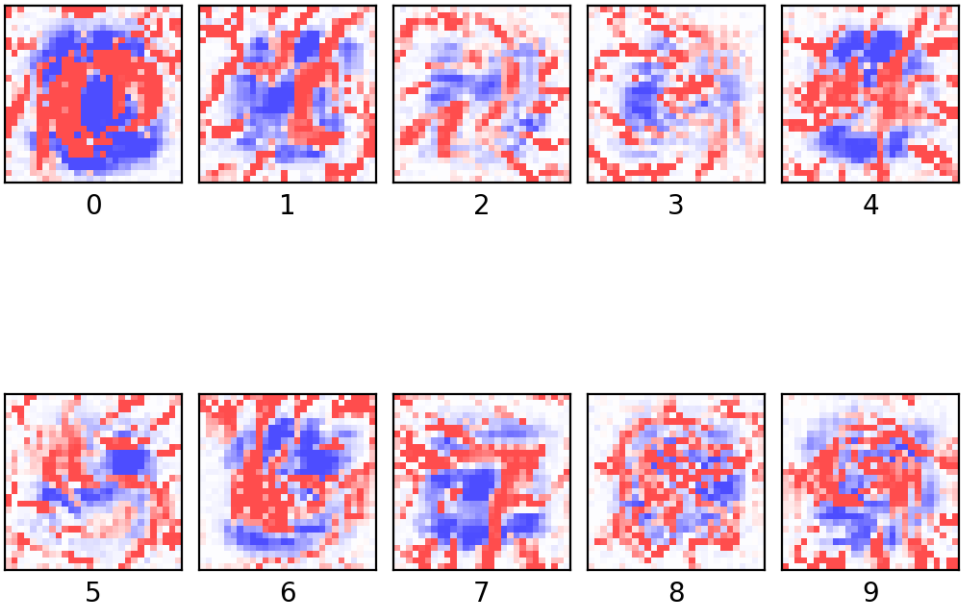
2.1.2 Plot of Validation Loss for all 10 classes



2.1.3 Plot of Test Accuracy for all 10 classes

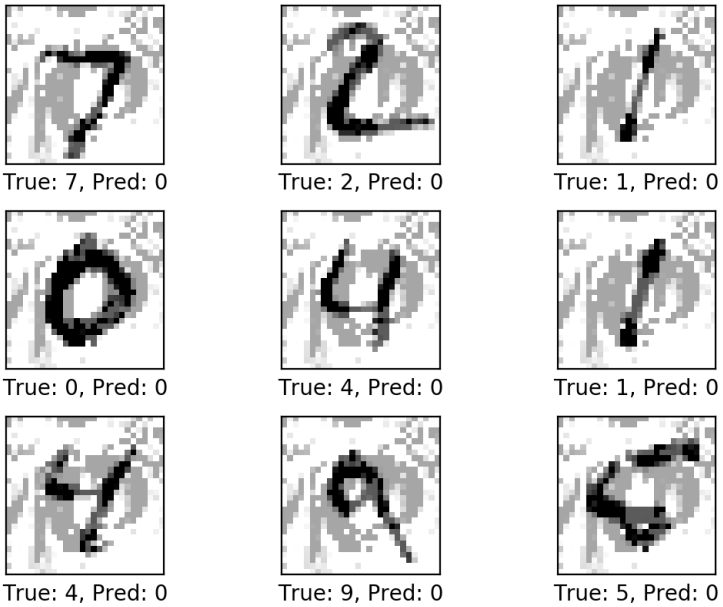


2.2 Plot of All Noise Patterns

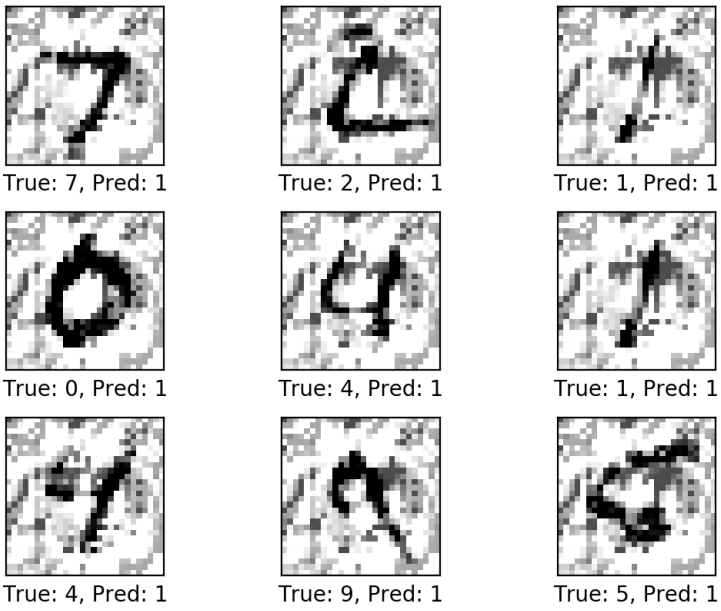


2.3 Examples Showing Misclassification Due to Addition of Noise

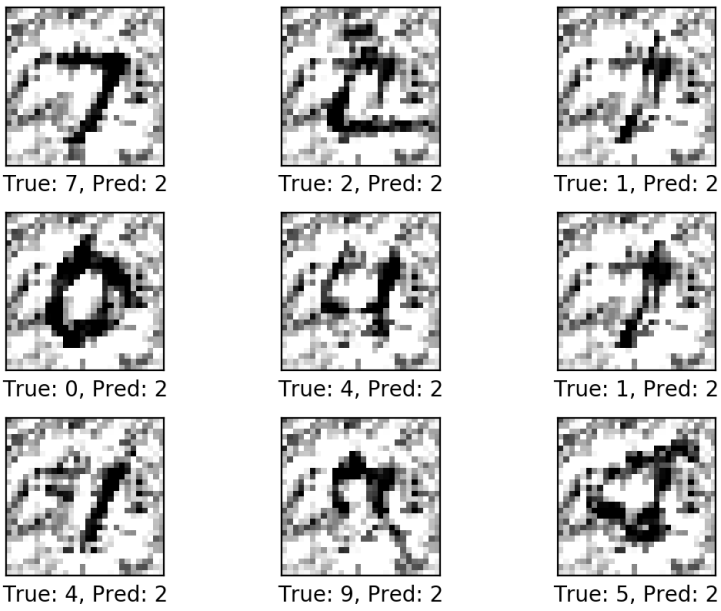
2.3.1 Misclassification with Class 0 Noise



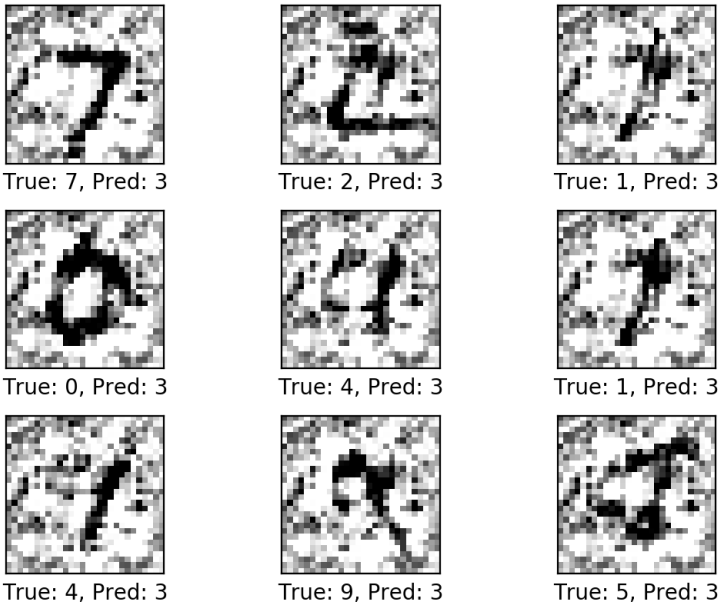
2.3.2 Misclassification with Class 1 Noise



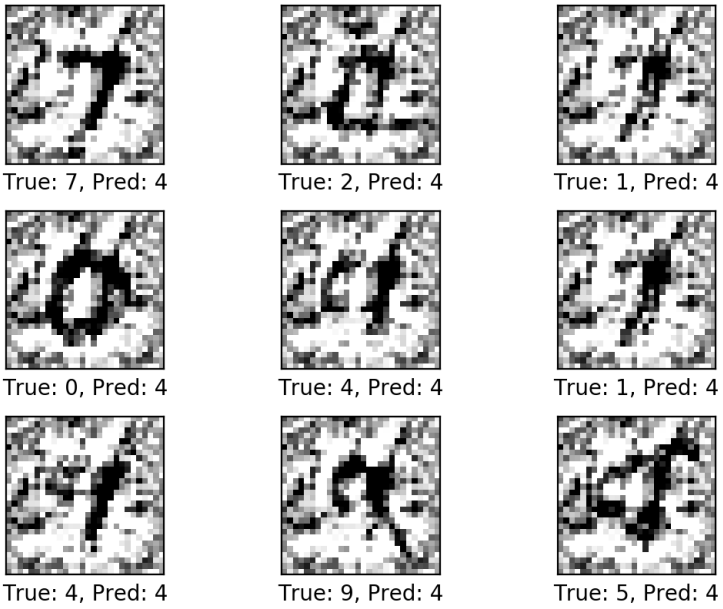
2.3.3 Misclassification with Class 2 Noise



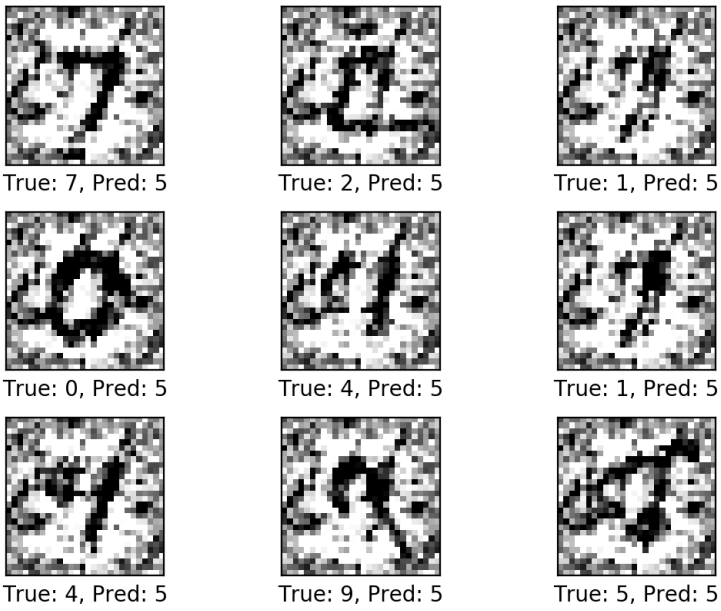
2.3.4 Misclassification with Class 3 Noise



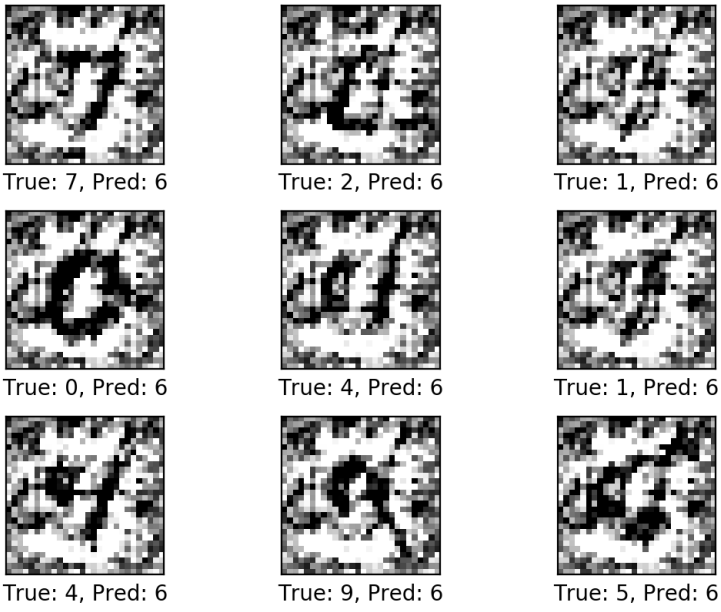
2.3.5 Misclassification with Class 4 Noise



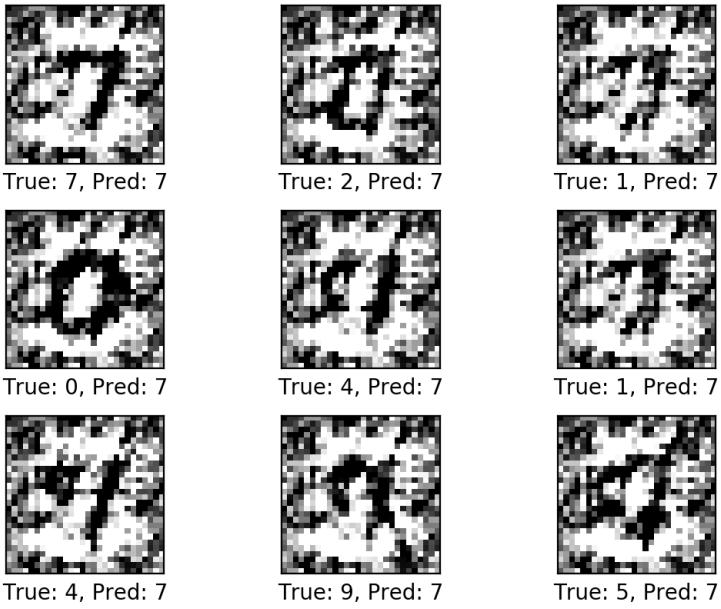
2.3.6 Misclassification with Class 5 Noise



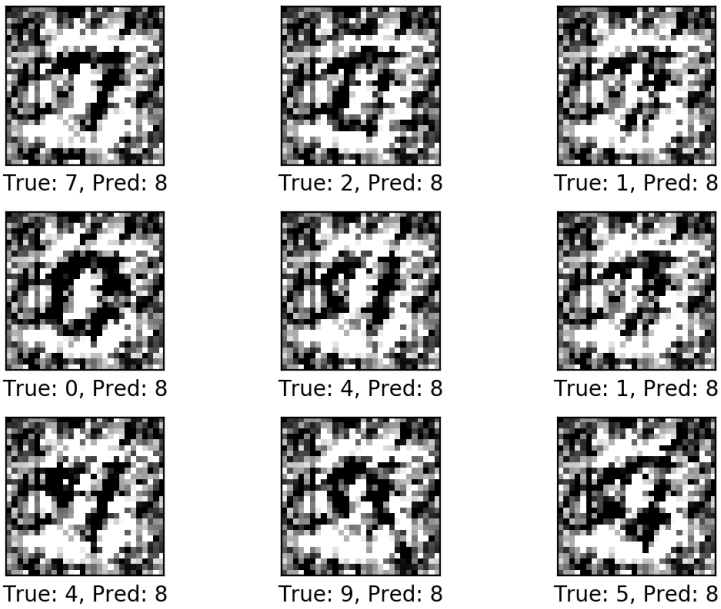
2.3.7 Misclassification with Class 6 Noise



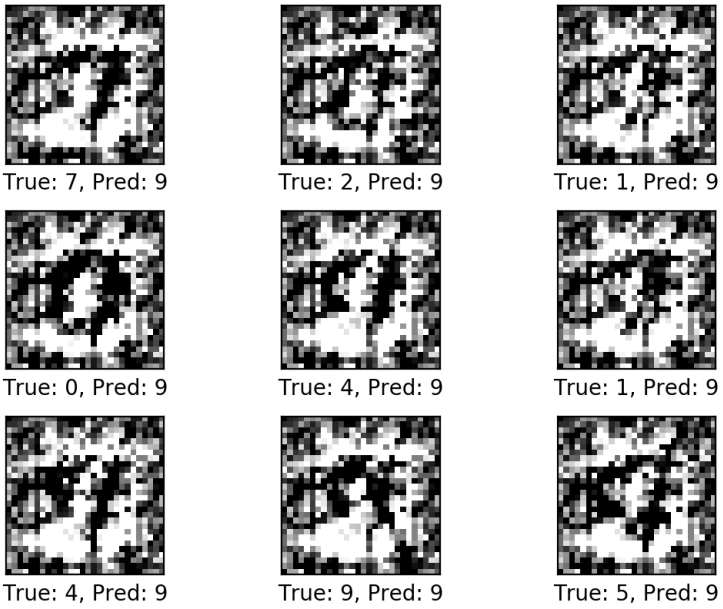
2.3.8 Misclassification with Class 7 Noise



2.3.9 Misclassification with Class 8 Noise

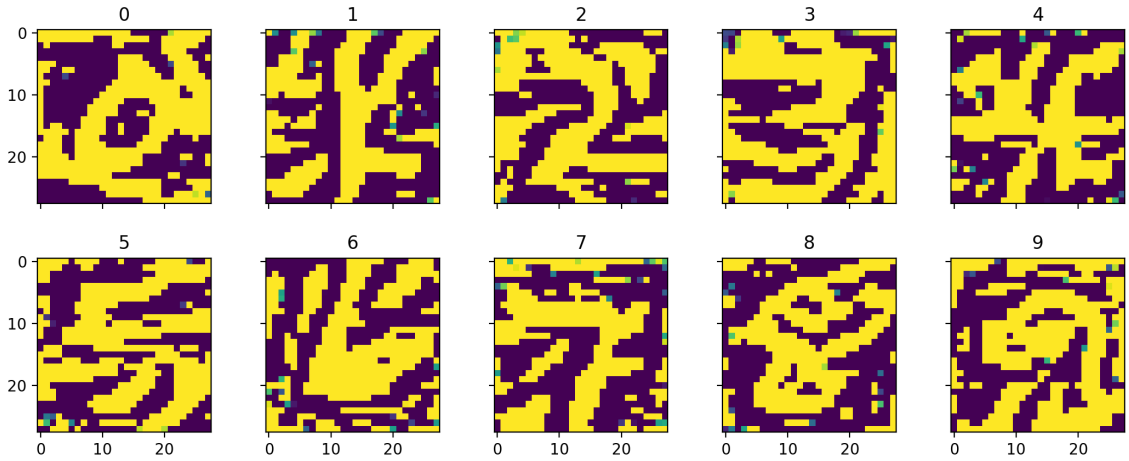


2.3.10 Misclassification with Class 9 Noise

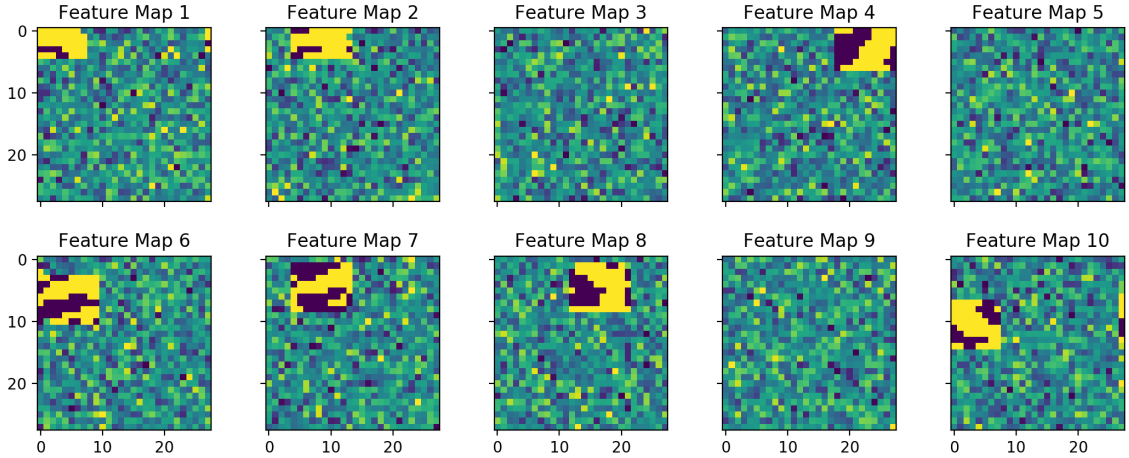


3 Visualizing the CNN

3.1 Plot of x_{init} for 10 Output Neurons



3.2 Plot of x_{init} for 10 Feature Maps after 2^{nd} Max Pooling


Feature Map 1
Feature Map 2
Feature Map 3
Feature Map 4
Feature Map 5
Feature Map 6
Feature Map 7
Feature Map 8
Feature Map 9
Feature Map 10
0
10
20