COMP 4107 - Assignment#4

Due: 5th April 2020 at 11:55 pm

Name: Kirk Zhen
Student ID: 101087006

Using the provided TensorFlow model (cnn.py) implementation of a convolutional neural network, modify it to classify the CIFAR-10, 10 class image data. You must modify this model. You must use this codebase as it is within scope for the final exam. These modifications include:

1. Modification of the dataset loaded (currently MNIST). **[20 marks]**
2. You are to investigate changing the number of convolutional layers and sizes of max pooling layers along with the number of feature maps.
3. You must investigate:
    1. Number of convolutional layers in the range 1-3 inclusive. Document your accuracy for a varying number of convolutional layers **[25 marks]**
    2. Using LeNet-5 as inspiration, choose an appropriate number of feature maps for each CNN multi-layer scenario investigated. Document your accuracy for a varying number of feature maps. **[25 marks]**
4. Document the optimal configuration found. **[5 marks]**
5. Provide plots of a typical feature found in layers 1, 2 and 3. Document the CNN parameters for the plots provided. See slide 63 and 64 of notes for example plots. **[15 marks]**
6. Provide a chart of the accuracy of your network for 1-15 epochs for the various scenarios investigated. Note: you may train your CNN models for more epochs than 15. **[5 marks]**
7. Provide a table of the accuracy of your network for the various scenarios investigated. This is the testing accuracy after training. **[5 marks]**

1. Modification of the dataset loaded (currently MNIST). **[20 marks]**

The CIFAR-10 dataset contains totally 60000 samples (50000 for training and 10000 for testing) associated with their corresponding labels (encoded as 10 numbers).

Feature Scaling
My pre-training process encoded each image into a 3 channels image (red, green, blue respectively), and Min-max Normalization would be performed on the whole dataset, all pixels in the images would be rescaled in the range [0, 1], according to the following formula:

$$f\left(x_i\right) = \frac{x_i - \min x}{\max x - \min x}$$

One-hot Encoding
The categorical labels in the dataset would be encoded into a one hot format using keras.np_utils.to categorical() function. The one hot encoded labels is better for machine learning algorithm to use when dealing with multi-class classification problem.

2. You are to investigate changing the number of convolutional layers and sizes of max pooling layers along with the number of feature maps.
3. You must investigate:
    1. Number of convolutional layers in the range 1-3 inclusive. Document your accuracy for a varying number of convolutional layers **[25 marks]**
    2. Using LeNet-5 as inspiration, choose an appropriate number of feature maps for each CNN multi-layer scenario investigated. Document your accuracy for a varying number of feature maps. **[25 marks]**

Model Description:

There are totally 6 CNN models to be explored in my experiment (Input layer would not be describe in this part). These 6 CNN models would be used to investigate Question 2 and Question 3.1, 3.2.

The result of the experiment would be shown in the latter parts of this report.

Model A:

1^{st} Layer: Convolutional layer, 64 feature maps with filter size 3×3×3 and a stride of 1
2^{nd} Layer: Pooling layer, with a filter size 2×2 and a stride of two
3^{rd} Layer: Fully connected convolutional layer with 128 feature maps each of size 1×1
4^{th} Layer: Fully connected Relu layer with 64 units with
Output Layer: Fully connected softmax output layer with 10 class.

Model B:

1^{st} Layer: Convolutional layer, 64 feature maps with filter size 3×3×3 and a stride of 1
2^{nd} Layer: Pooling layer, with a filter size 2×2 and a stride of two
3^{rd} Layer: Convolutional layer, 128 feature maps with filter size 3×3×64 and a stride of 1
4^{th} Layer: Pooling layer, with a filter size 2×2 and a stride of two
5^{th} Layer: Fully connected convolutional layer with 128 feature maps each of size 1×1
6^{th} Layer: Fully connected Relu layer with 64 units with
Output Layer: Fully connected softmax output layer with 10 class.

Model C:

1^{st} Layer: Convolutional layer, 64 feature maps with filter size 3×3×3 and a stride of 1
2^{nd} Layer: Pooling layer, with a filter size 2×2 and a stride of two
3^{rd} Layer: Convolutional layer, 128 feature maps with filter size 3×3×64 and a stride of 1
4^{th} Layer: Pooling layer, with a filter size 2×2 and a stride of two
5^{th} Layer: Convolutional layer, 128 feature maps with filter size 3×3×128 and a stride of 1
6^{th} Layer: Pooling layer, with a filter size 2×2 and a stride of two
7^{th} Layer: Fully connected convolutional layer with 128 feature maps each of size 1×1
8^{th} Layer: Fully connected Relu layer with 64 units with
Output Layer: Fully connected softmax output layer with 10 class.

Model D (Same with model B, but with max pooling of size 4×4 and a stride of 4):

1^{st} Layer: Convolutional layer, 64 feature maps with filter size 3×3×3 and a stride of 1
2^{nd} Layer: Pooling layer, with a filter size 4×4 and a stride of 4
3^{rd} Layer: Convolutional layer, 128 feature maps with filter size 3×3×64 and a stride of 1
4^{th} Layer: Pooling layer, with a filter size 4×4 and a stride of 4
5^{th} Layer: Convolutional layer, 128 feature maps with filter size 3×3×128 and a stride of 1
6^{th} Layer: Pooling layer, with a filter size 4×4 and a stride of 4
7^{th} Layer: Fully connected convolutional layer with 128 feature maps each of size 1×1
8^{th} Layer: Fully connected Relu layer with 64 units with
Output Layer: Fully connected softmax output layer with 10 class.

Model E:

1^{st} Layer: Convolutional layer, 64 feature maps with filter size 3×3×3 and a stride of 1
2^{nd} Layer: Pooling layer, with a filter size 6×6 and a stride of 6
3^{rd} Layer: Convolutional layer, 128 feature maps with filter size 3×3×64 and a stride of 1
4^{th} Layer: Pooling layer, with a filter size 6×6 and a stride of 6
5^{th} Layer: Convolutional layer, 128 feature maps with filter size 3×3×128 and a stride of 1
6^{th} Layer: Pooling layer, with a filter size 6×6 and a stride of 6
7^{th} Layer: Fully connected convolutional layer with 128 feature maps each of size 1×1
8^{th} Layer: Fully connected Relu layer with 64 units with
Output Layer: Fully connected softmax output layer with 10 class.

LeNet:

1^{st} Layer: Convolutional layer, 6 feature maps with filter size 5×5 and a stride of 1
2^{nd} Layer: Pooling layer, with a filter size 2×2 and a stride of two
3^{rd} Layer: Convolutional layer, 16 feature maps with filter size 5×5 and a stride of 1
4^{th} Layer: Pooling layer, with a filter size 2×2 and a stride of two
5^{th} Layer: Fully connected convolutional layer with 120 feature maps each of size 1×1
6^{th} Layer: Fully connected Relu layer with 84 units
Output Layer: Fully connected softmax output layer with 10 class.

The result of the experiments would be shown in the later parts of this report. (part 6 and 7)

4. Document the optimal configuration found. **[5 marks]**

According to the result shown in part 6 and part 7, Model E gives the best Testing accuracy among the 6 models we explored.

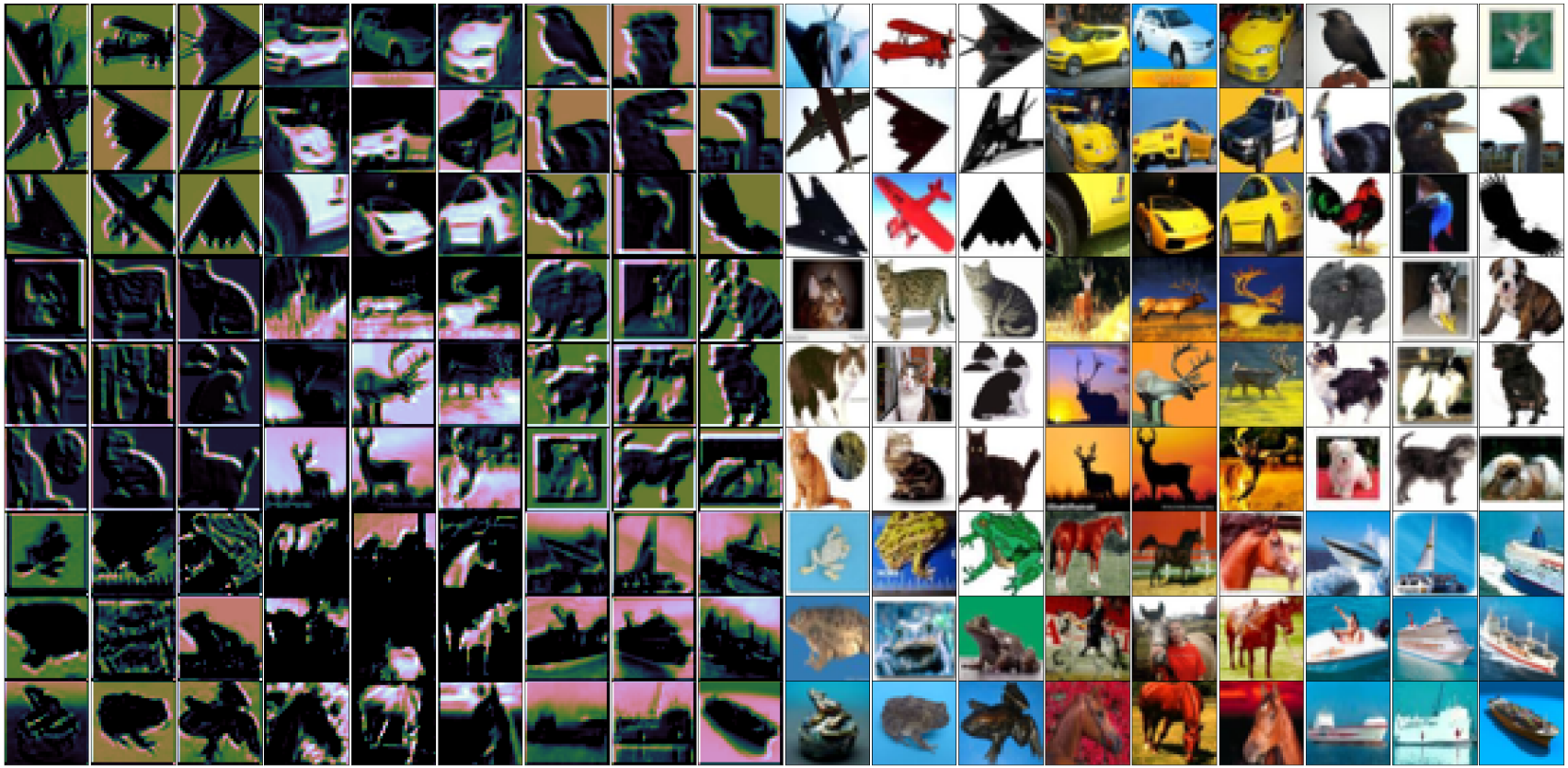
The architecture of Model E has been mentioned in the previous part.

The graph and table would be shown in part 6 and part 7.

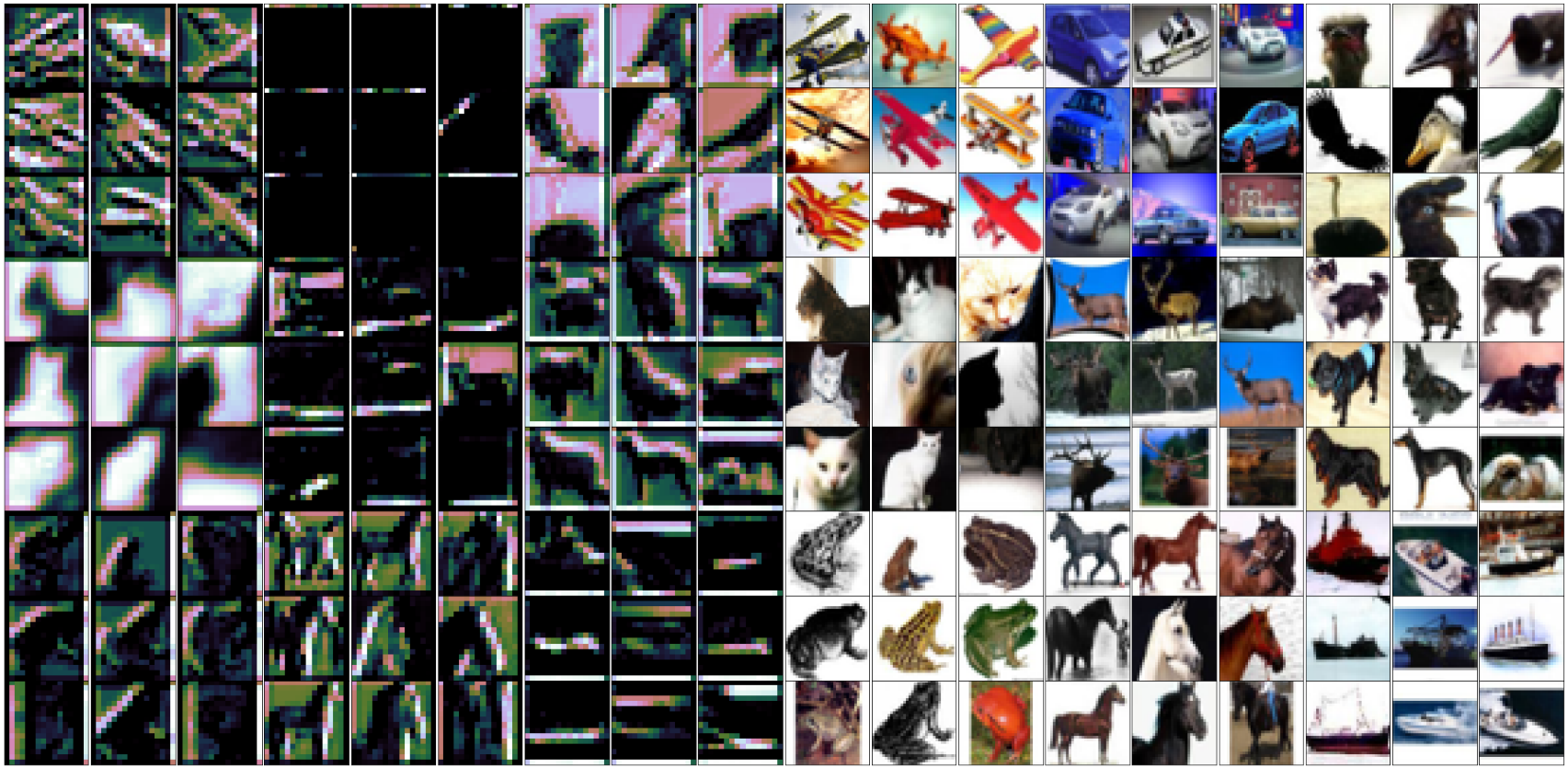
5. Provide plots of a typical feature found in layers 1, 2 and 3. Document the CNN parameters for the plots provided. See slide 63 and 64 of notes for example plots. **[15 marks]**

Model C is chosen to be investigated.
The parameters of Model C is shown in the previous parts.

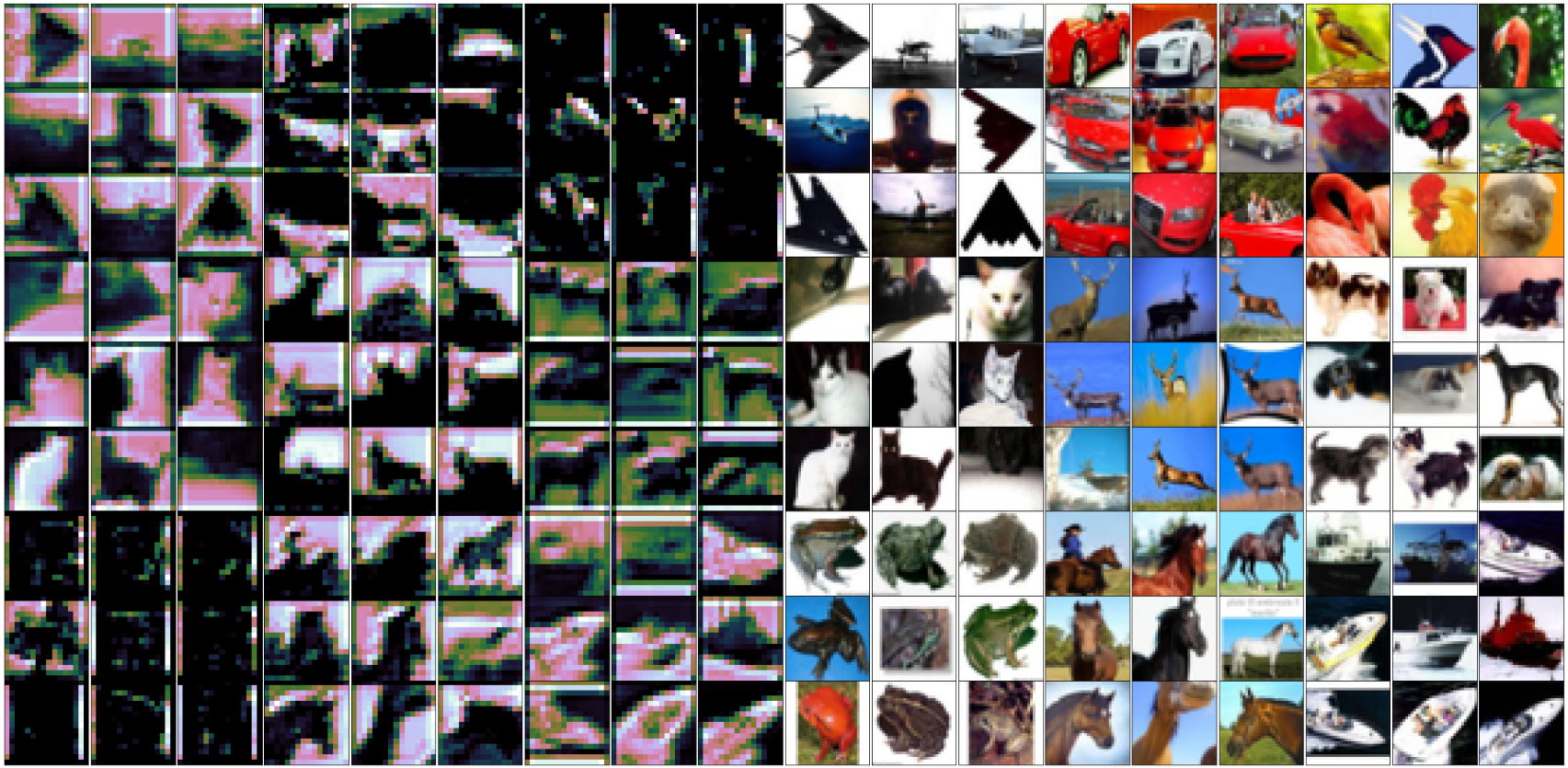
Visualization of feature maps in the first Convolutional Layer in Model C:



Visualization of feature maps in the second Convolutional Layer in Model C:



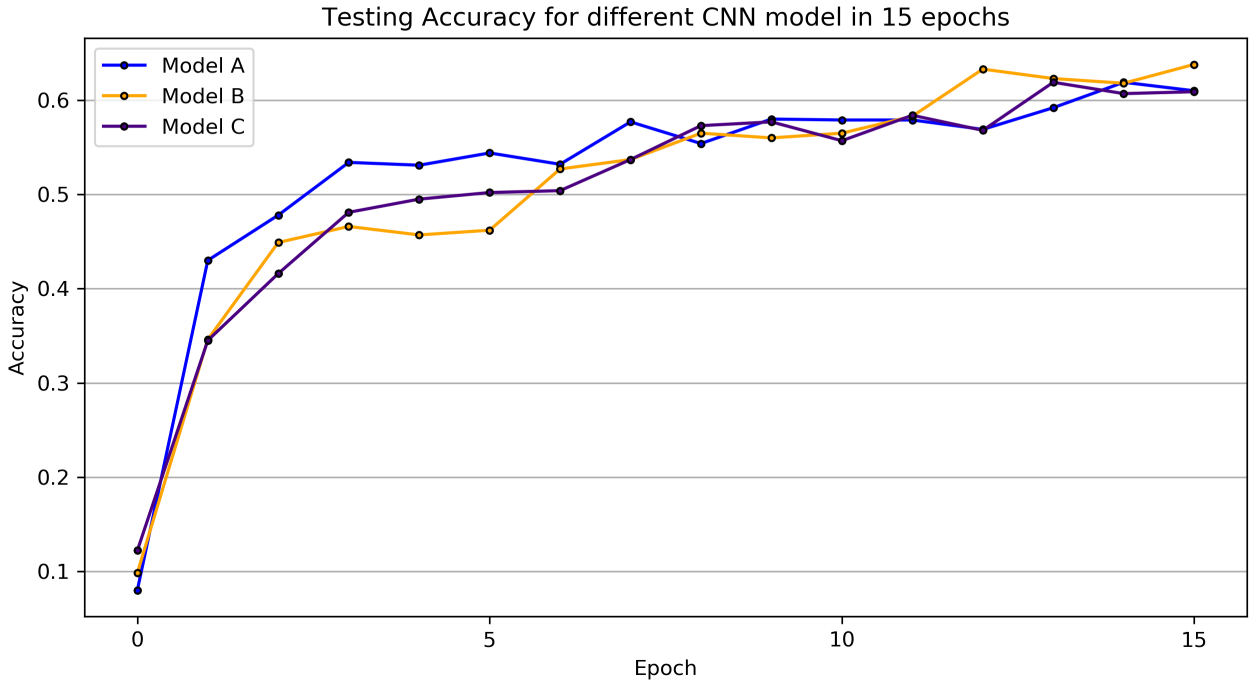
Visualization of feature maps in the third Convolutional Layer in Model C:



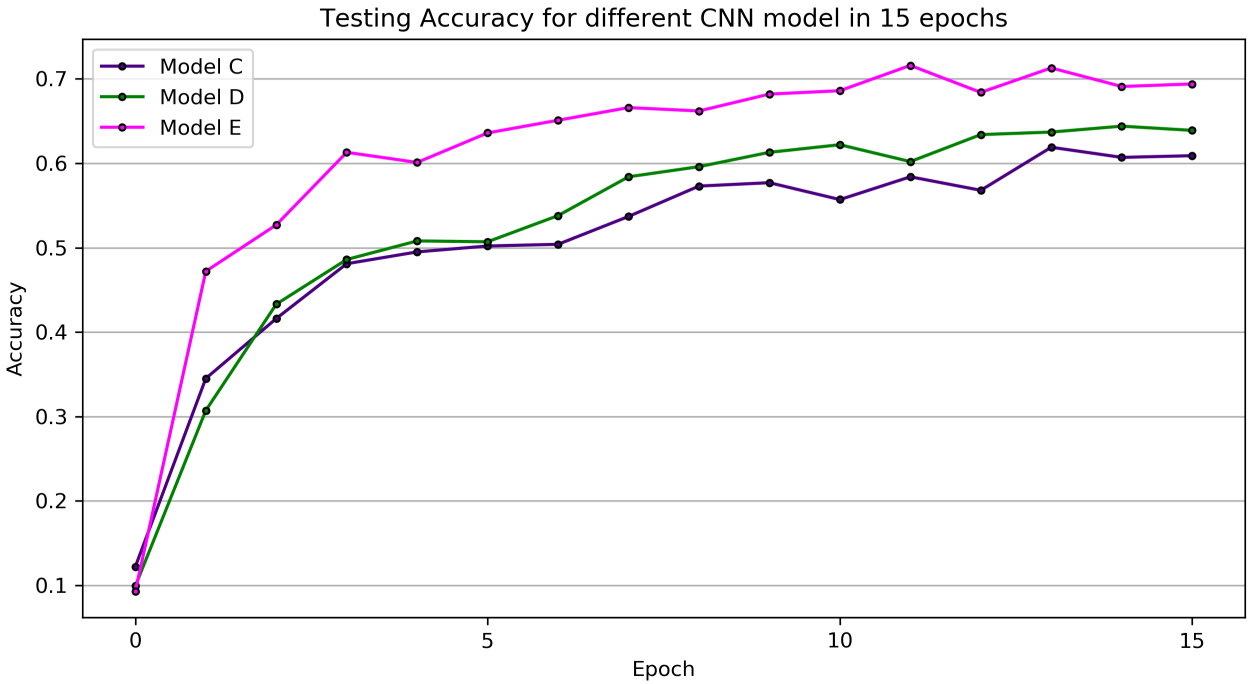
6. Provide a chart of the accuracy of your network for 1-15 epochs for the various scenarios investigated. Note: you may train your CNN models for more epochs than 15. **[5 marks]**
7. Provide a table of the accuracy of your network for the various scenarios investigated. This is the testing accuracy after training. **[5 marks]**

The following parts also illustrate the answer for Question 2, Question 3.1, Question 3.2, and Question 4

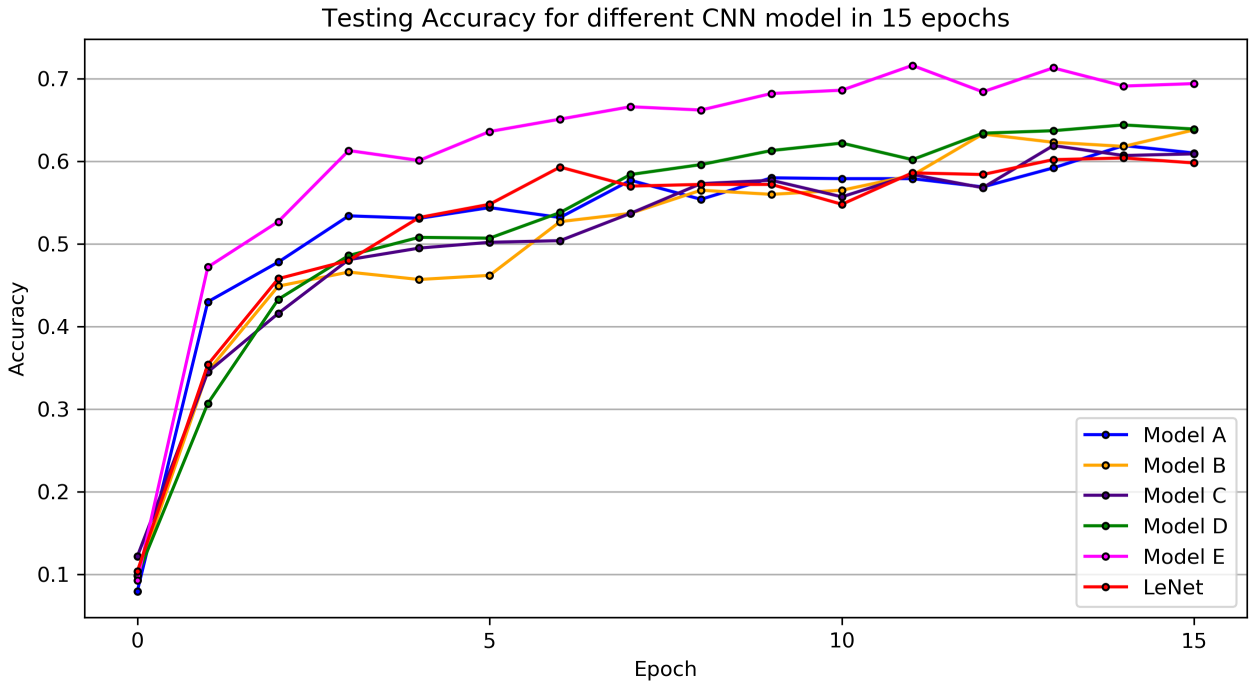
The first figure below shows the testing accuracy of Model A, B, C for the first 15 epoch.
According to the model description above, Model A, B, C has different number of hidden layers and feature maps.
I think it's hard to tell how number of hidden layers and feature maps influence the predicting accuracy from the blowing graph, since their performance in 15 epochs are very close. But from what we have for 50 epochs (table in part 7), the Model B hits the highest accuracy for both training and testing set. I inferred that the predicting accuracy does simply increase or decrease along with hidden layers and feature maps, it get optimized at some parameter combination.



The second figure below shows the testing accuracy of Model C, D, E for the first 15 epoch.
According to the model description above, we know that, Model C, D, E has almost the same architecture, but the size of their max polling layer are different. We can see how the predicting accuracy changes along with the size of polling layers.



The following graph shows the testing accuracy of the 6 different models for the first 15 epoch.



And the following table shows the performance of different CNN models with 50 epochs trained.

Model	**Training Accuracy**	**Training Loss**	**Testing Accuracy**	**Testing Loss**
Model A	72.1%	1.641	66.7%	1.798
Model B	79.1%	1.312	68.7%	1.712
Model C	72.1%	1.652	67.7%	1.741
Model D	81.1%	1.143	72.1%	1.672
Model E	**89.7%**	**0.744**	**74.0%**	**1.628**
LeNet-5	69.1%	1.891	63.7%	1.812

We can see from the table, Model E gives the best performance among the 6 models.

Reference

1. CIFAR-10 Image Classification in TensorFlow, https://towardsdatascience.com/cifar-10-image-classification-in-tensorflow-5b501f7dc77c.