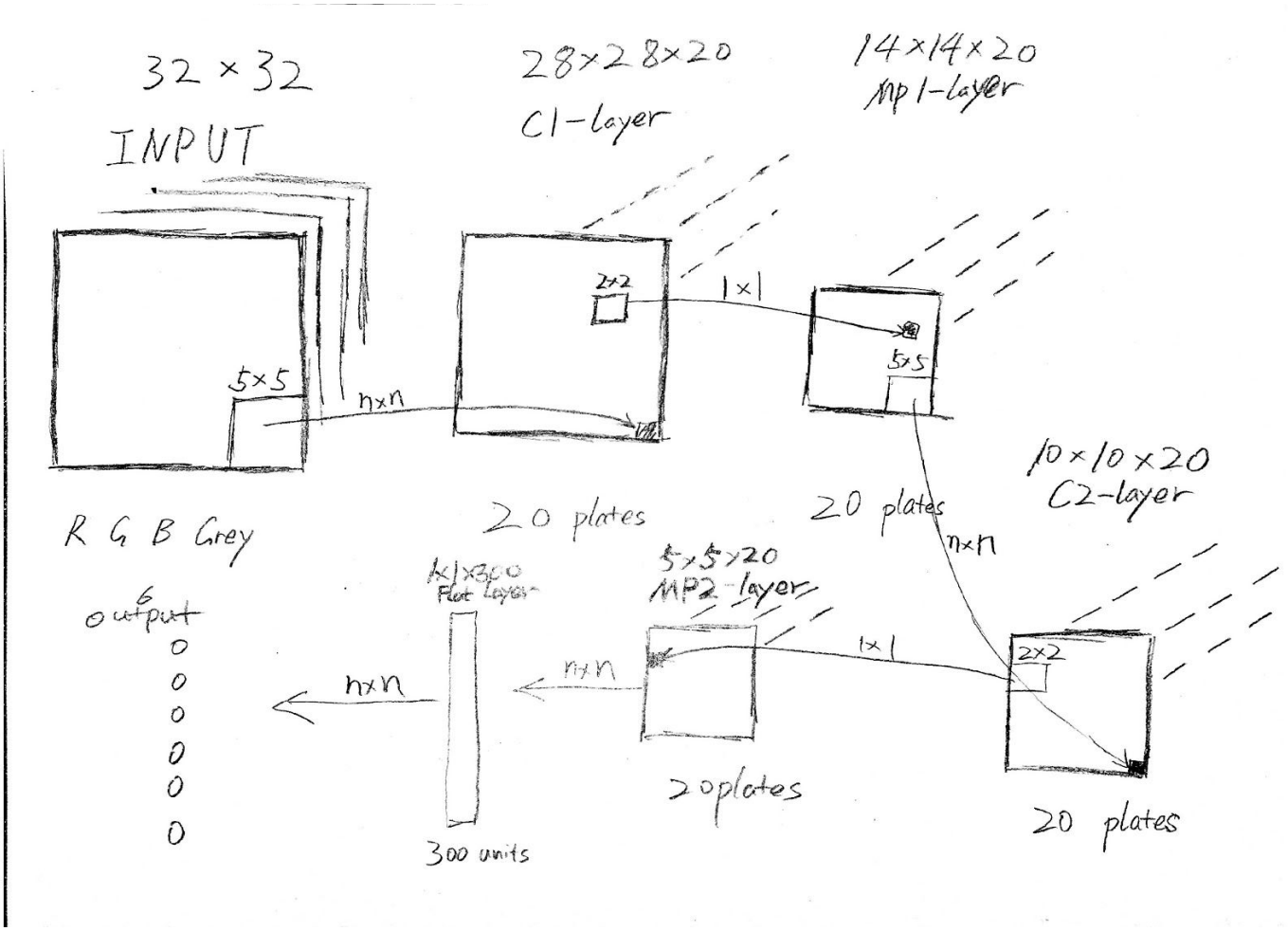
Lab 3 Report

Xiaofei Liu Renjie Tang Hangjun Piao Zhenda Lu

Final Configuration:



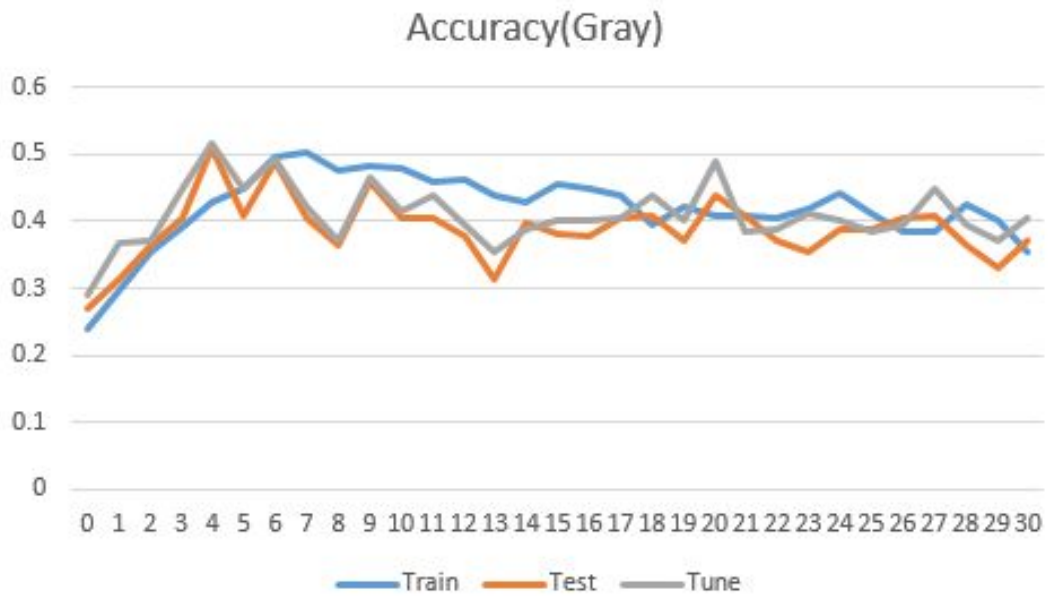
We use similar design as lecture slides, and the only difference is that we do not have the last convolution layer.
Learning Rate: 0.01
Activation function: Sigmoid for last two layers and Leaky ReLU for convolutional layers.
Create extra training examples: false.
Dropout: false.
useRGB: false

After using different configurations and debugging, we cannot improve the accuracy anymore within the limited time. And the accuracy is not good, which we will discuss our thoughts about it in the further thoughts section.

Early Stopping:
By simply following the early stop, we decide stop at epoch 4.

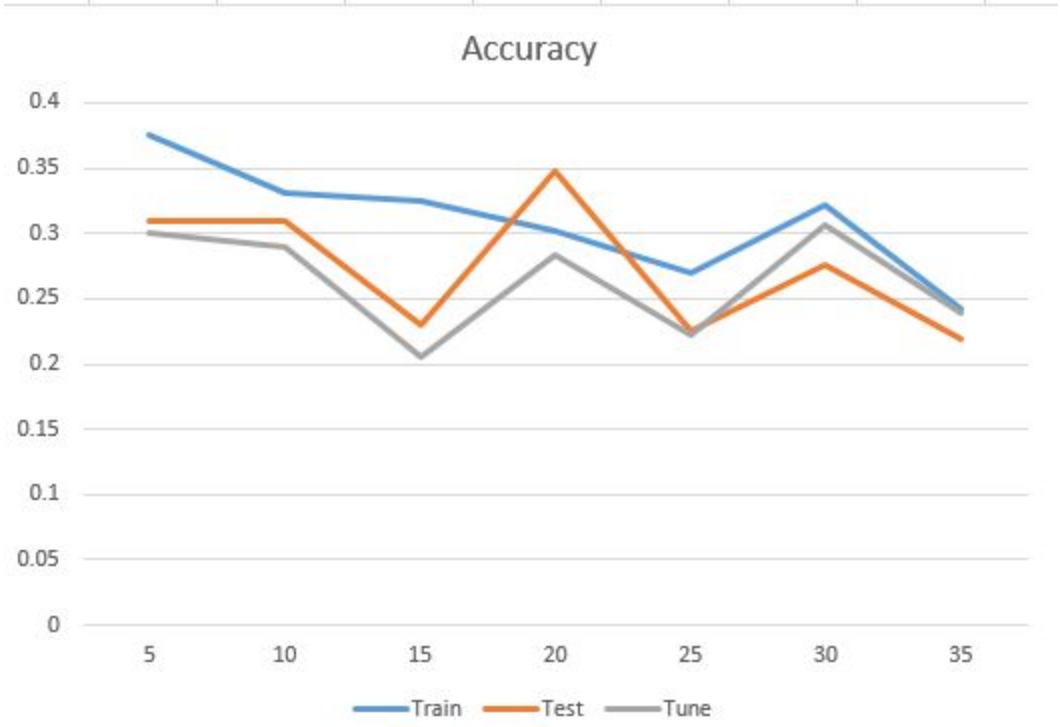
As for four channels, if we strictly following the algorithm, we should stop at epoch 5. However, after observing the learning curve, we found out that tune set accuracy rate went through a peak at epoch 20, which might give higher test set accuracy. Therefore, we decide to report both test accuracy and confusion matrix.

Learning Curve and Confusion Matrix:
UseRGB = False



Epoch=4	GrayOnly					
	airplanes	butterfly	flower	grand_pia	starfish	watch
airplanes	31	0	4	0	0	6
butterfly	6	0	6	0	0	6
flower	7	0	19	2	0	9
grand_pia	4	0	6	7	0	2
starfish	5	0	6	1	0	5
watch	4	0	6	2	0	34

UseRGB = True



Epoch=5		Accuracy = 0.3089887640449438				
		CONFUSION		MATRIX		
	airplanes	butterfly	flower	grand_pia	starfish	watch
airplanes	4	0	4	0	0	33
butterfly	0	0	4	0	0	14
flower	0	0	7	1	0	29
grand_pia	1	0	2	4	0	12
starfish	0	0	2	0	0	15
watch	0	0	6	0	0	40

Epoch=20		Accuracy = 0.30144404332129965				
		CONFUSION		MATRIX		
	airplanes	butterfly	flower	grand_pia	starfish	watch
airplanes	4	0	4	4	0	29
butterfly	0	0	5	1	0	12
flower	0	1	17	1	0	18
grand_pia	0	0	4	6	0	9
starfish	0	0	5	1	0	11
watch	0	0	10	1	0	35

Dropout:
We tried to implement dropout with 0.5 dropout rate. We drop every units by randomize a number, which gives 50% probability for each unit to be dropped. However, the method does not improve our accuracy but decrease it. After applying dropout, the train set and test set accuracy will be lowered around 20%. The reason why dropout does not work could be followings according to our discussion:
(1) The dropout method might not help the convolution-max pooling set up. Using dropout will discard units in convolution layer. If all the units of the "pool" are dropped, we decide to assign 0 for the value. This will cause "over" dropping units in max pooling layer, which can have bad result on overall training.
(2) We use probability to achieve dropout, instead of drop exactly half of the units. This can have potential effect on the training, since our method can drop too many or too few units in some extreme case. However, we tried to use different random seed to make sure we do not drop too many or too few units, the result did not improve. Combining our knowledge of dropout, we think the problem of dropout is likely not because of this.
(3) We might implement something wrong with dropout. The bug is likely located in convolution layers and max pooling layers, because input, flay layer and output are rather simple to implement. In addition, we believe there are still bugs in our simplest configuration, and that might also have bad impact on dropout.

Experiments:
Activation Function: We use Sigmoid for last two layers and Leaky ReLU for rest of the layers. And we tested many different combination of activation functions.
(1) Sigmoid for last two layers and Leaky ReLU for rest of the layers: This will cause all deltas from some specific plates be zero. The reason behind can be too many units output negative values, and after deviation(F') of ReLU, they became zero. So we switched to Leaky ReLU to solve the problem.
(2) Leaky ReLu for all the layers: In our analysis, we believed that using Leaky ReLU for all the layers can improve finding local minimum. However, according to our experiment, it does not help.

UseRGB: We use only grey image for final result. We also tried use all four channels, which decreases our accuracy. We would likely to believe that there are some bugs in our RGB settings.

Learning Rate: The final learning rate is 0.01. We have tried 0.3, 0.1 and 0.001. Someone claims that lower learning rate improves accuracy a lot, but 0.01 gives the best result in our case.

Weight Initialization: The final weight initialization is [-0.3,0.3]. We implemented [-0.01,0.01], xavier and fanin-fanout sqrt setup. None of them help, and we will further discuss this later.

Further thoughts:

If we only use grey image, the result is rather good. However, if we use all four channels, the result is not optimal and potential buggy. Here are our thoughts about bug:
(1) Weight initialization: We still think our weight initialization is too simple and not serve the best interest. According to most of people who posted on piazza, a very low weight initialization helps a lot. However, this is not true for our case. …
(2) Four channels: Our grey performs better than four channels set up, which is in contrast of most of people. It is likely that we implemented something wrong about the four channels set up. However, we also believe that it is possible that only using grey image can have better result. RGB can only provide the ratio between them in each pixel, and that might be confusing to our model. On the contrary, we can detect edge and other spatial parameters by grey image solely.
(3) Back Propagation: After observing our confusion matrix, we found out that our model tends to predict more 0 and 5, which is airplane and watch. We tends to believe that this is not a coincidence, since this is the first and last output node. Our forward is rather simple and being reviewed a number of times, so we believe there might have bugs in our back propagation.

Perceptron and One Hidden layer: We tried to implement perceptron at the very beginning, and it did not perform good. We did not further change configuration for it and then focused on deep neural network. And we did not try one hidden layer, which is actually good according to some posts on Piazza. We want to further develop a one hidden layer method and try to compare it with our deep neural network to get more insights.