Q1.1.1

The reasons why ReLU activation function is preferred to sigmoid:

The first reason is the calculate of sigmoid is much more complicated than ReLU.

The second is less likely to vanish in gradient. The gradient of ReLU is constant while the gradient of sigmoid would become increasingly small as the absolute value of x increases.

The last is sparsity. The sigmoid would more likely to generate dense representations.

Q1.1.2

If both intermediate activation functions(g(x)) and output activation functions(f(x)) are linear, then f(g(x)) is also linear. For the first hidden layer, $h(1) = g(W(1)x + b(1))$, and for the second hidden layer, $h(2) = g(W(2)h(1) + b(2))$, this is also a linear transformation, in which case the first transformation and the second one could sum up to one single linear transformation. Likewise, for all these hidden layers and output layers, these transformations could be combined into one single transformation from the input layer to the output layer. That is the reason why the number of hidden layers has not effect on the representation capability of the network.

Q2.1.1

If a network is initialized with all zeros, then all the neurons would do the same calculation at the first step, which is none of use. The same condition with initializing a network with all ones or with all same constant value. The initialization of network should break the symmetry.

Q2.1.3

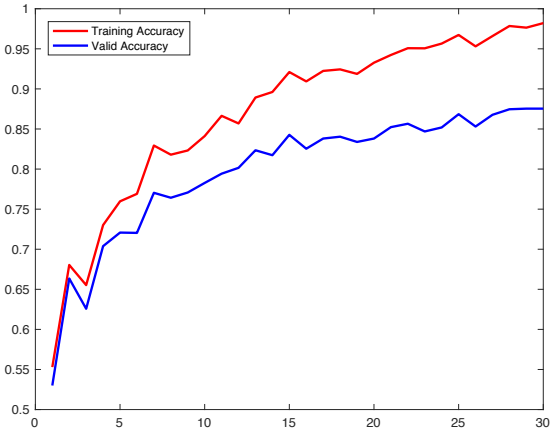
Using uniform distribution to initialize weights. In order to make the variance of the inputs equal to the variance of the outputs, $n \times Var(W_i) = 1$, then $Var(W_i) = \frac{1}{n} = 1/n_{in}$. During backward propagation, $Var(W_i) = \frac{1}{n} = 1/n_{out}$, in order to generally satisfy both, $Var(W_i) = \frac{2}{n_{in}+n_{out}}$. And the weights should be initialized close to 0 but not equal to 0. Then we set the distribution of initial weights observe uniform distribution, the mean is 0 and the variance is $\frac{2}{n_{in}+n_{out}}$. $W_i \sim U\left[-\frac{\sqrt{6}}{\sqrt{n_{in}+n_{out}}}, \frac{\sqrt{6}}{\sqrt{n_{in}+n_{out}}}\right]$. And all the bias should be set to 0.

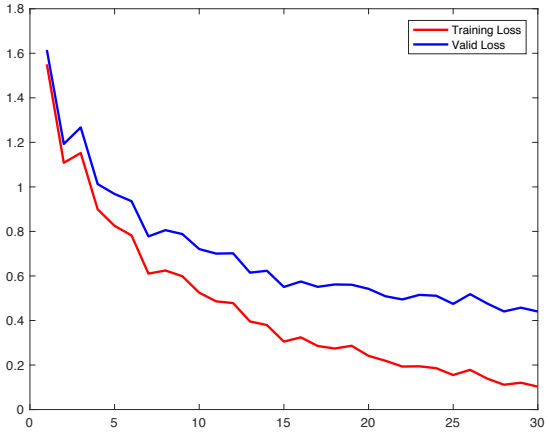
Q2.4.1

In Batch Gradient Descent(BGD) method, we compute the gradient for all the examples in the training set during each calculation, in which condition it would be really time consuming if the example is very big. But BGD would get the best optimal solution over the whole dataset. The Stochastic Gradient Descent(SGD) computes the gradient with one sample and updates the weights during each iteration, in which condition would improve the time-consuming condition of the BGD. However, one problem is that the accuracy would decrease compare to the BGD method.

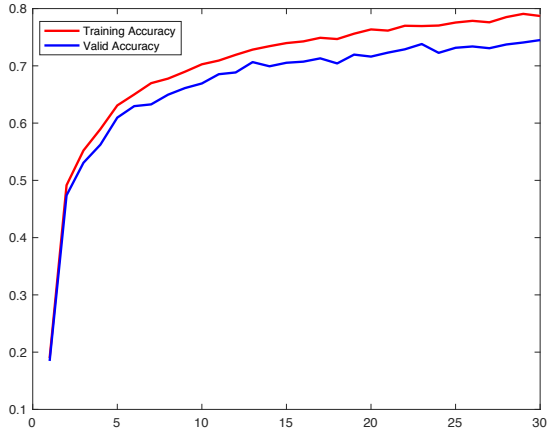
Q3.1.2

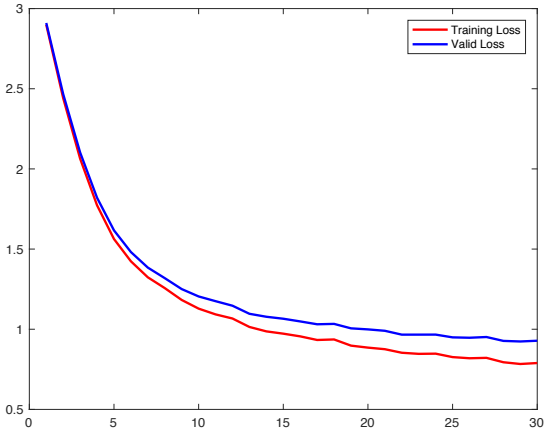
Learning rate = 0.01:





Learning rate = 0.001

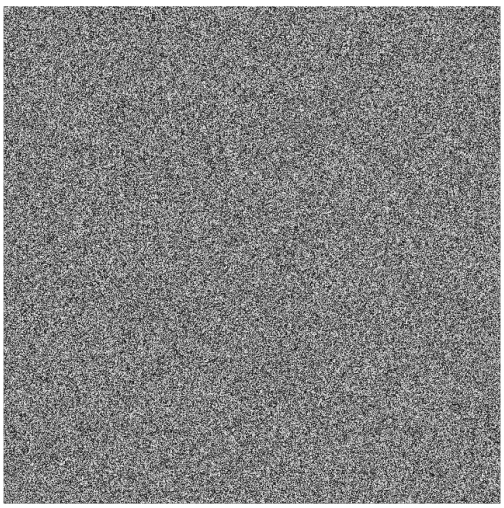


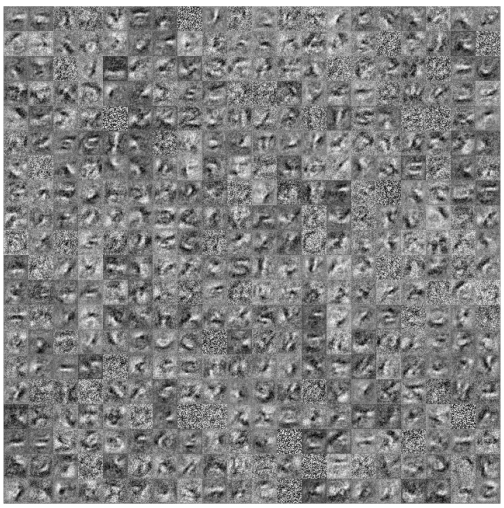


I used Stochastic Gradient Descent here. From the plots above, we can see that smaller learning rate would yield smoother curve since the step during each iteration is smaller. But we can see that 30 epochs is not enough for 0.001 learning rate because the curve has not saturated yet at the 30 epochs. The best accuracy I got is 0.86462 at 0.01 learning rate.

Q3.1.3

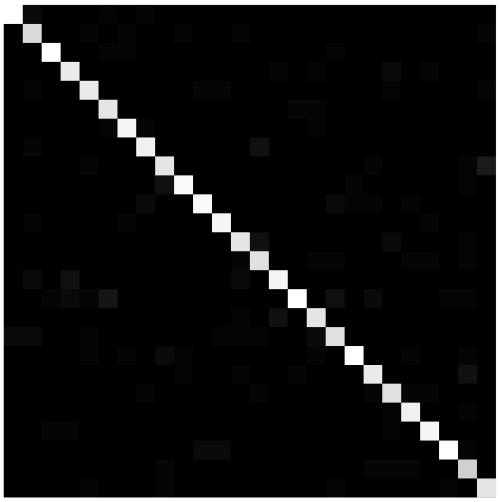
The accuracy is 0.86462 and the average cross-entropy loss is 0.44996.





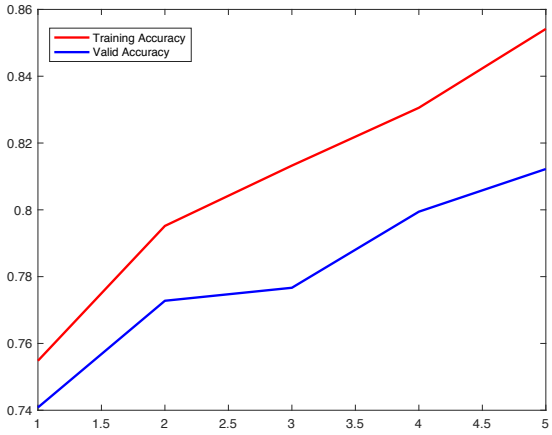
The above two pictures show the visualization of the initial first layer weights(left) and the visualization of the network learned first layer weights(right). The left image is just totally in a mess and the right image shows some similar patterns.

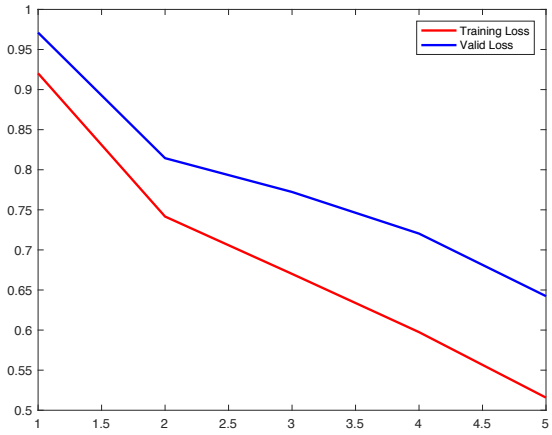
Q3.1.4



From the convolution matrix, we can see that the top two pairs that are most usually confused are “I” with “Z” and “P” with “F”. This is reasonable since the characters in these two pairs look similar with each other.

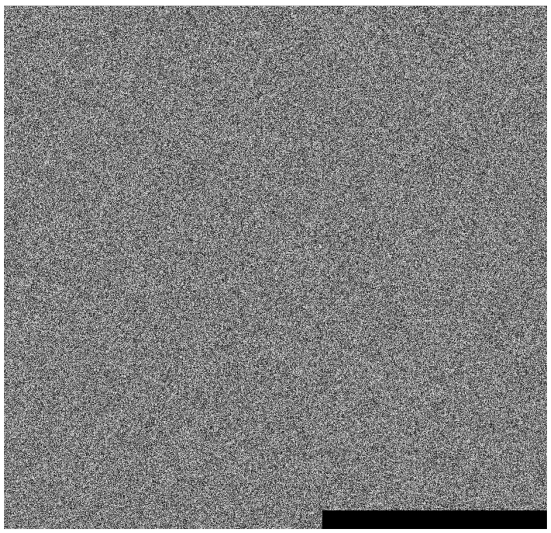
Q3.2.1

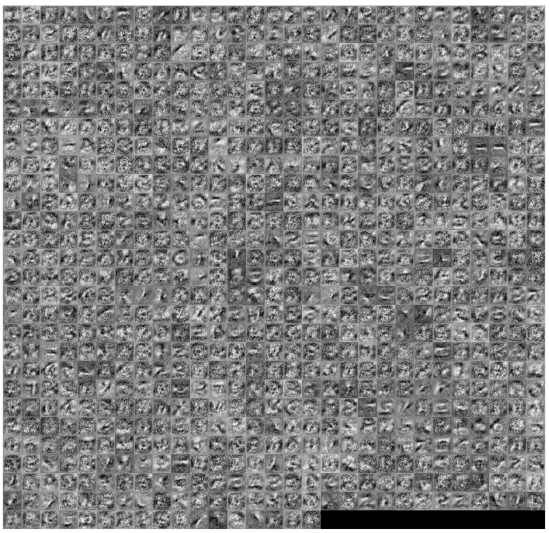




The left is the plot of accuracy and the right is the plot of loss.

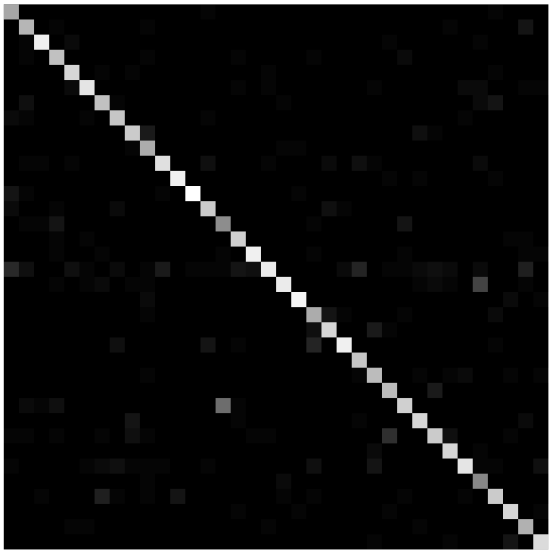
Q3.2.2





The accuracy is 0.8122 and the loss is 0.64232.

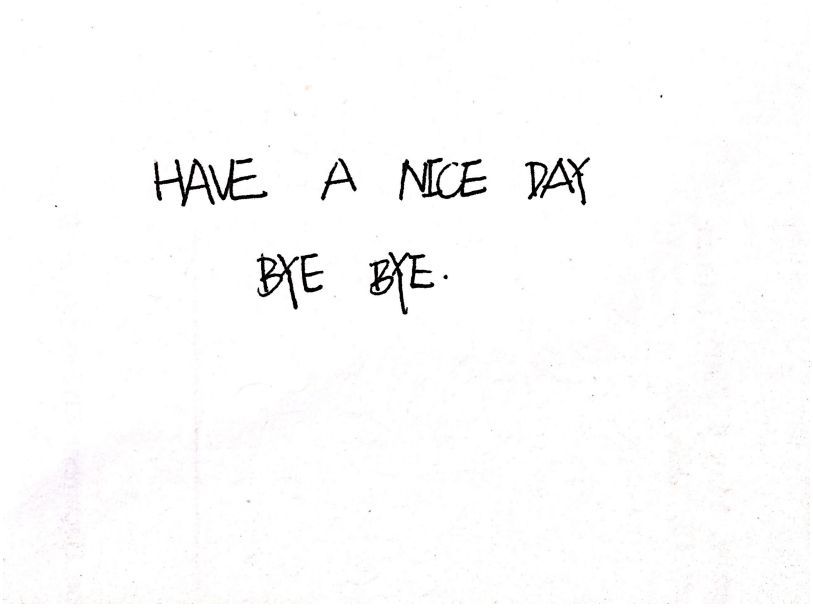
Q3.2.3



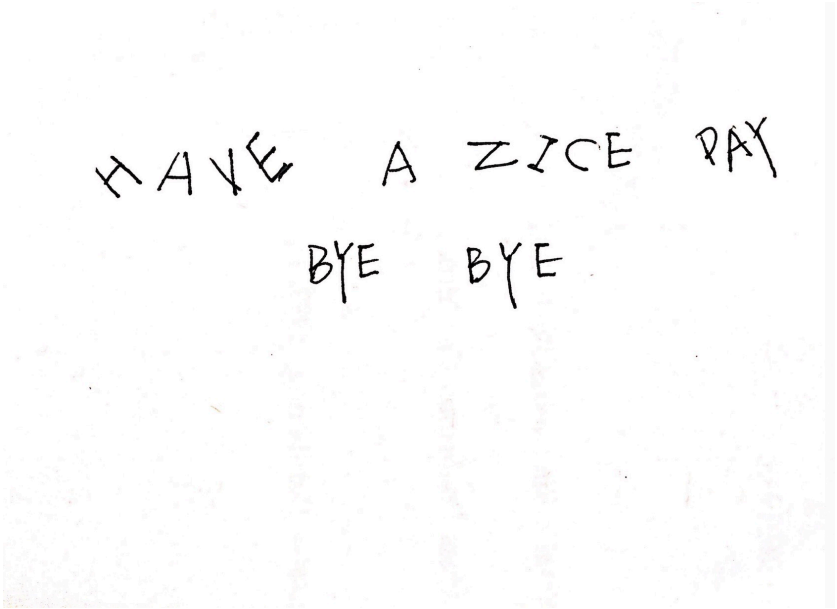
From the convolution matrix, we can see that the top two pairs that are most usually confused are “O” with “0” and “S” with “5”. This is reasonable because “O” is really looks similar with “0”, even people would sometimes cannot figure out the difference if they are written by hand. And the same condition with “S” and “5”.

Q4.1

The first assumption is the two neighbor letters cannot be connected or the distance between them cannot be too small. If not, the method cannot classify them separately, which may lead to the inaccuracy of classification.

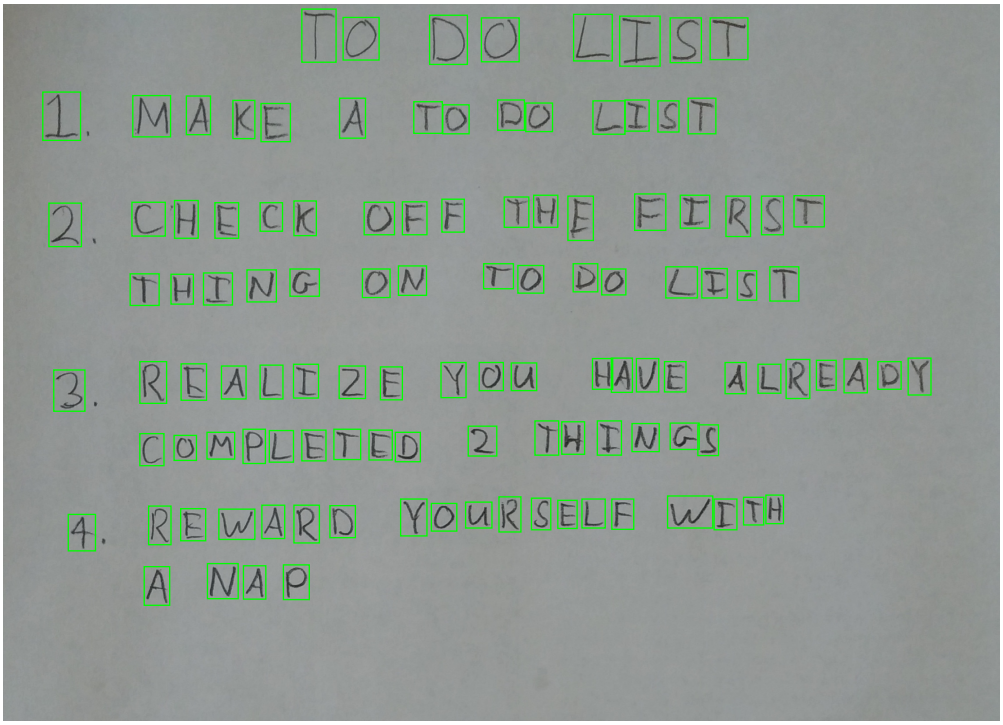


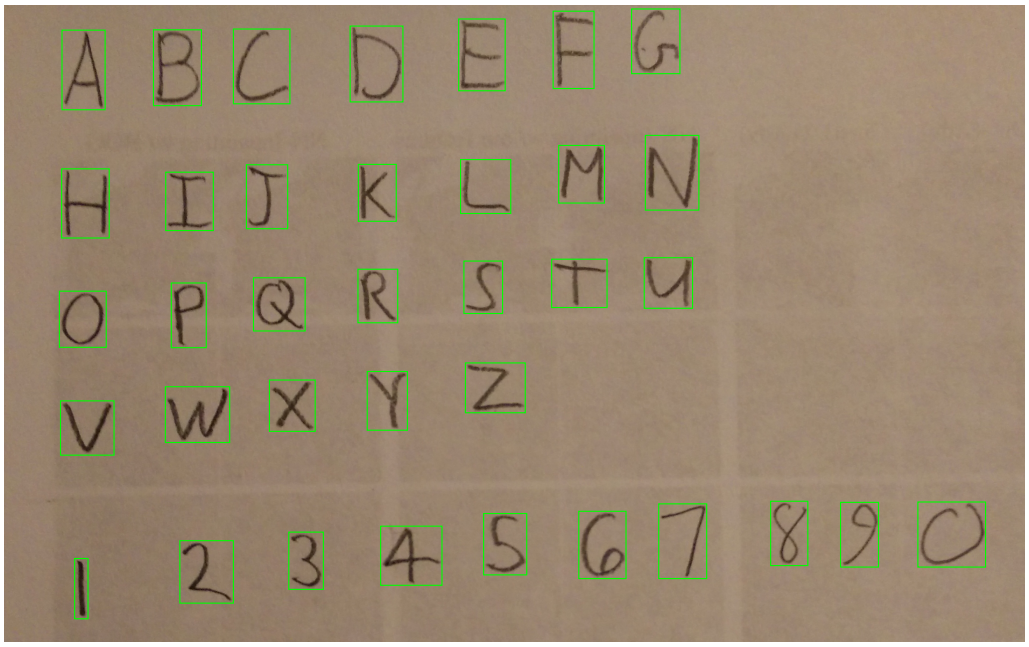
The second assumption is the characters must be written horizontally and written in lines, because this method cannot recognize rotated characters.

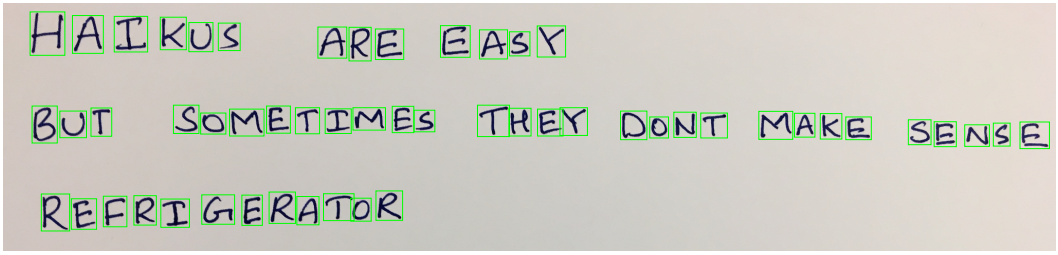


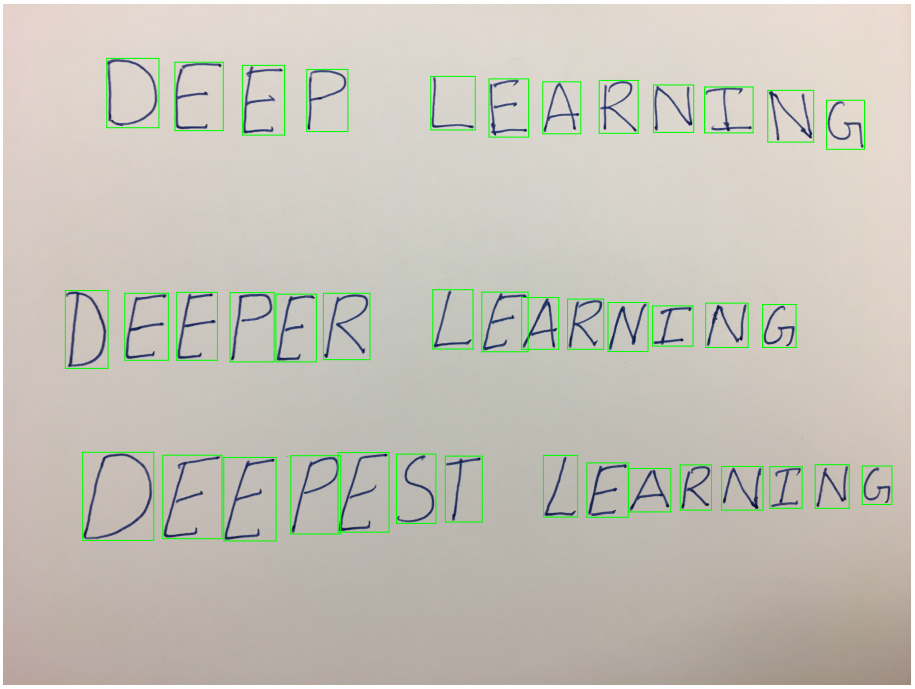
Q4.3

The result images with boundaries are shown below:




A B C D E F G
H I J K L M N
O P Q R S T U
V W X Y Z
1 2 3 4 5 6 7 8 9 0


HAIKUS ARE EASY
BUT SOMETIMES THEY DONT MAKE SENSE
REFRIGERATOR


DEEP LEARNING
DEEPER LEARNING
DEEPEST LEARNING

Q4.5

The first image: ‘01_list.jpg’

Original text:

TODOLIST

1MAKEATODOLIST

2CHECKOFFTHEFIRST

THINGONTODOLIST

3REALIZEYOUHAVEALREADY

COMPLETED2THINGS

4REWARDYOURSELFWITHANAP

Result:

TQDJLIST

LNAKEAT0DQLIST

2CHKCKDFETHEFIRST

TMINGQNTQDQLIST

3RKAKIZEYQUNAUEALR6ADY

C0MPLETLDZTHINGS

4RKWARDY0URSELFWITH

ANAP

The second image: ‘02_letters.jpg’

Original text:

ABCDEFG

HIJKLMN

OPQRSTU

VWXYZ

1234567890

Result:

AKCDGFG

MIIKLMW

QVQKSTW

VWXYZ

BZ3GSG7K7S

The third image: ‘03_haiku.jpg’

Original text:

HAIKUSAREEASY

BUTSOMETIMESTHEYDONTMAKE

REFRIGERATOR

Result:

HAIWWSAREGASY

BWTSQMETIMESTHEYDQWTMAKESENGE

RBFRIGERATQR

The fourth image: ‘04_deep.jpg’

DEEPLEARNING

DEEPERLEARNING

DEEPESTLEARNING

Result:

CCFFCCAKYINW

DFFTEKLEAKNIMG

CFCFF5T1EARNIMG

Q5.1

In conv2D function, the number of filters were set wrong and the number of channels were set wrong, too. The number of channels should equal to the number of output layers from the last convolution.

conv2D(4,1,8,1,2) should be changed to conv2D(4,1,4,1,2).

conv2D(4,8,16,1,2) should be changed to conv2D(4,4,8,1,2).

conv2D(4,16,64,0,1) should be changed to conv2D(4,8,64,0,1).

transpConv2D(4,16,4,1,2) should be changed to conv2D(4,8,4,1,2)

transpConv2D(4,8,1,1,2) should be changed to conv2D(4,4,1,1,2)

Q5.2.2

The initial value of learning rate is too big, I changed the initial learning rate to 1e-5. One adjustment I set is increasing the value of learning rate. By increasing this value to 1e-3, the loss would be significantly improved. The final mini-batch loss at the epoch 3 is 8.8242, and the mini-batch RMSE is 4.20.