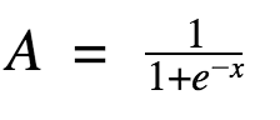
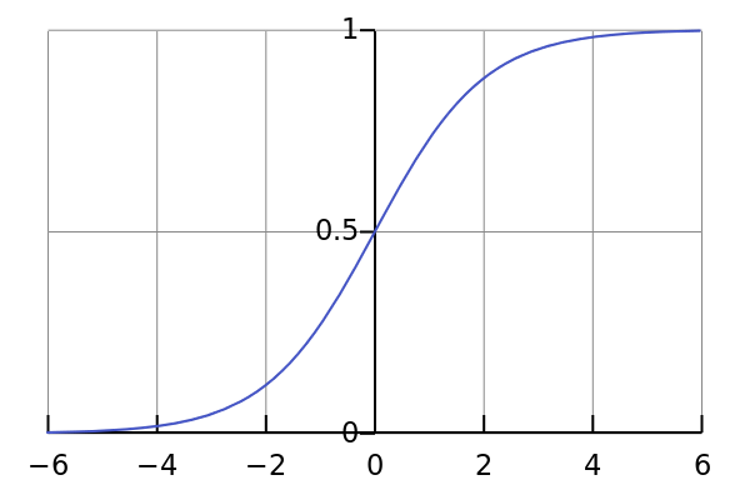
s

**Computer Vision – Homework 5**

Q1.1

**Sigmoid Function:**

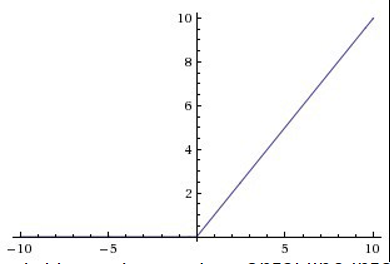




In sigmoid function whenever the activations reach near the “near horizontal” part of the curve, it gives rise to the problem of vanishing gradient. As a result of the network refuses to learn further or is drastically slow. Also, there is no sparsity of the activation, that means, all most all the neurons get fired up by the activation function.

**ReLU:**

A(x) = max(0,x)



In ReLU, there is no problem of vanishing gradients and there is sparsity of activation, which means that fewer neurons are fired, which makes the network light and helps in reducing computation expenses.

So, for the above-mentioned reasons, ReLU is preferred over Sigmoid.

Q1.1.2

Q1.1.3

If the network is initialized with all zeros, then all the gradients will be the same, which means the changes from the gradients will be propagated back to the network equally on all layers and therefore, the system will not be able to learn.

In case of initializing the network with a constant value, it may cause the problem of symmetry in the network which essentially means that all the neurons will be activated/deactivated by a similar method. This also increases the chance of the network to get stuck in the local optima. So, to prevent these, the network is initialized with random weights.

Q 2.1.3

I initialized the weights randomly with a normal(Gaussian )distribution having a 0 mean and a variance of 0.01. I did this because the gaussian distribution tends to help in breaking the symmetry.

|  |  |
| --- | --- |
| **Stochastic Gradient** | **Batch Gradient** |
| Computes gradient in mini batches of the randomly shuffled dataset. | Computers gradient using the whole batch. |
| In terms of convergence it is faster, as it updates the weight more frequently. | It is slower, as it must process the entire dataset before updating. |
| Noise in each of the update is more | Noise in each of the update is less |
| It can escape shallow local minima | Cannot escape shallow local minima |

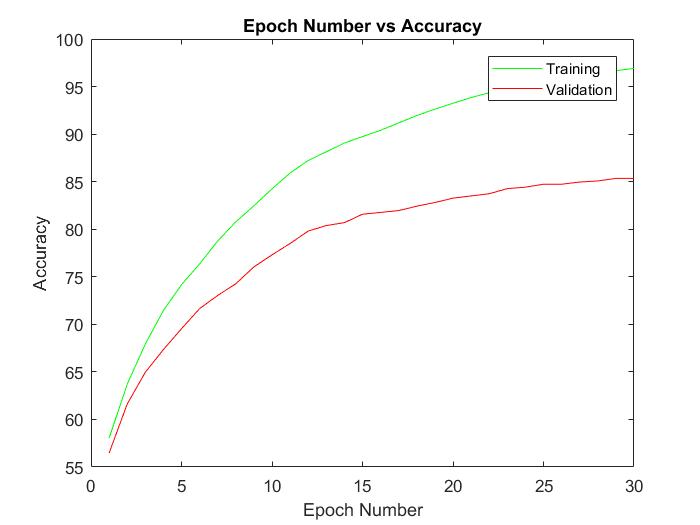
Q 2.1.4

Batch gradient is faster to train in terms of number of epochs as the whole dataset is processed to complete one epoch. But on the other side, the SGD is faster in terms of number of iterations, as the network is continuously updated in mini batches whose size is less than that of the entire batch.

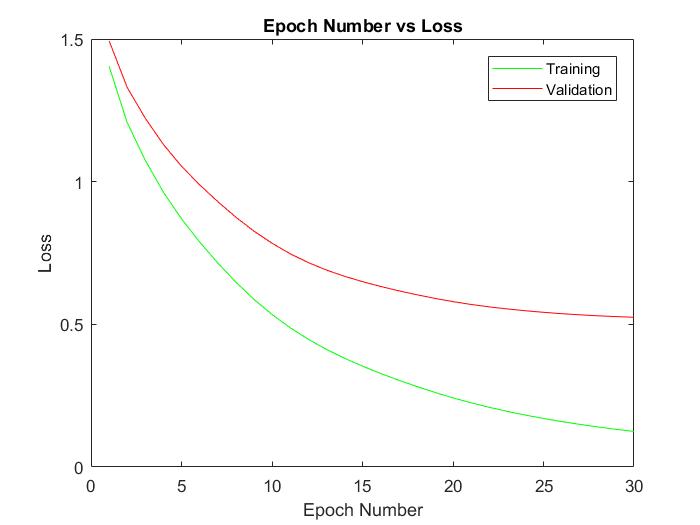
Q 3.1.2

Learning Rate: 0.01

a)



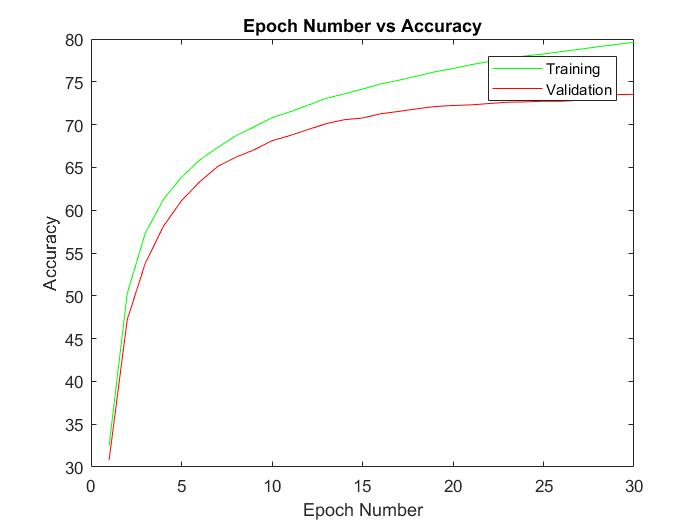
b)



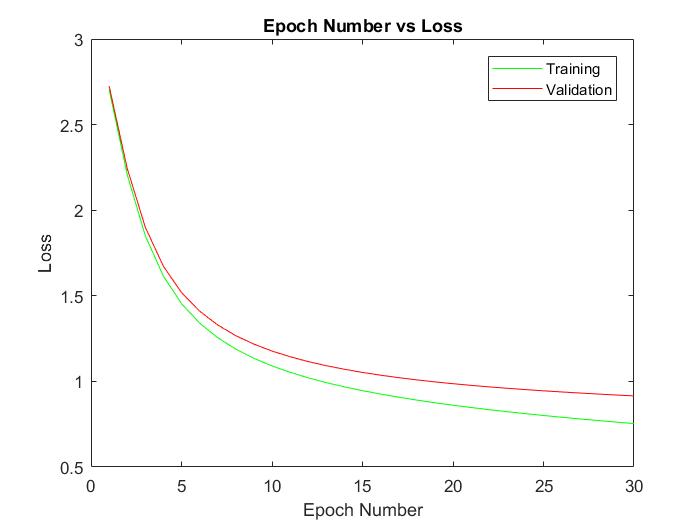
Learning Rate: 0.001

Classes=26

a)



b)



1. Too High learning rate means, that the gradient will overshoot the minimum and therefore the system will not converge.
2. Too low learning rate means, that the system will take a lot of time to converge as it will take lots of miniscule steps downhill.

Usually, a learning rate is selected such that, the system is faster in terms of convergence and will not overshoot the minimum.

**BEST NETWORK:**

Accuraccy:0.9716

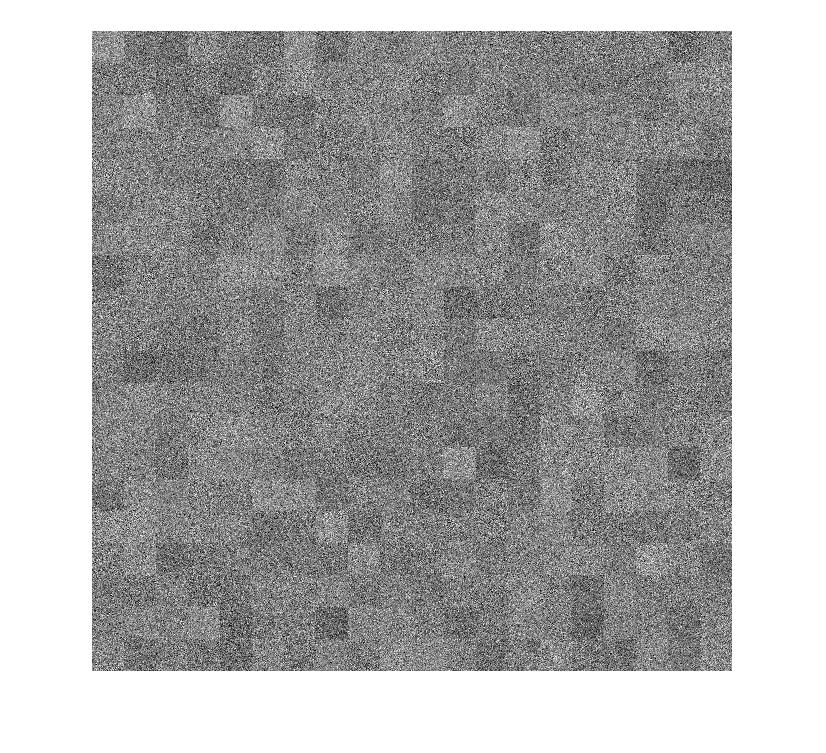
Learning rate:0.01

Q 3.1.3

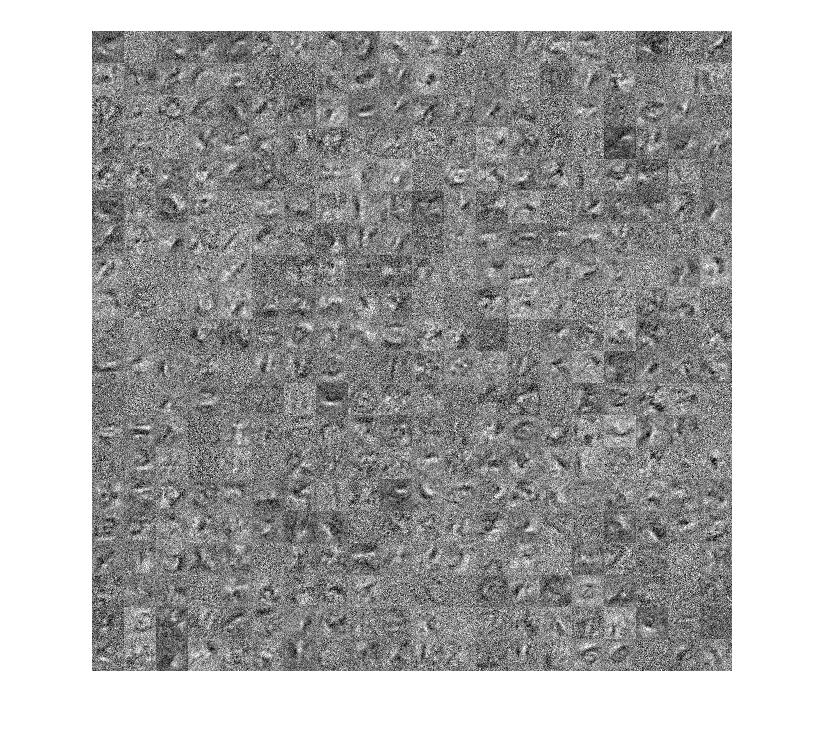
Best Network Accuracy on Test Set: 84.26

Best Network Cross Entropy Loss on Test Set: 0.567

1. Initial Weights



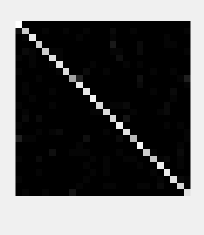
1. Final Weights after training



By comparing the initial weights and the learned weights it can be seen that the learned weights have picked up some off the features like horizontal lines, vertical lines and curves.

Q3.1.4

Confusion Matrix for 26 classes

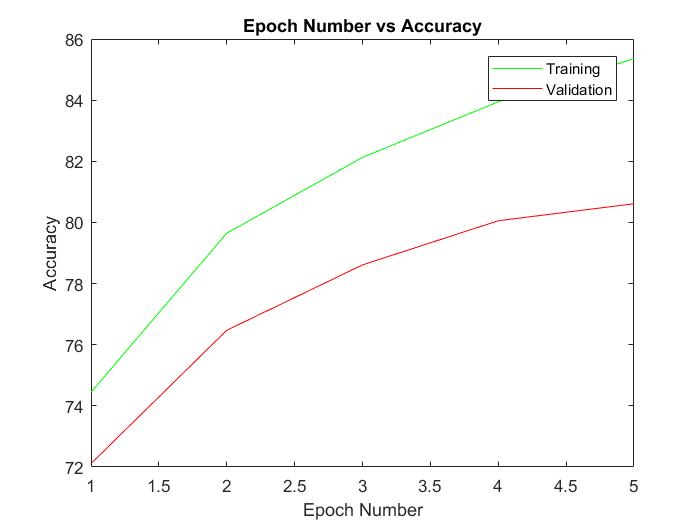


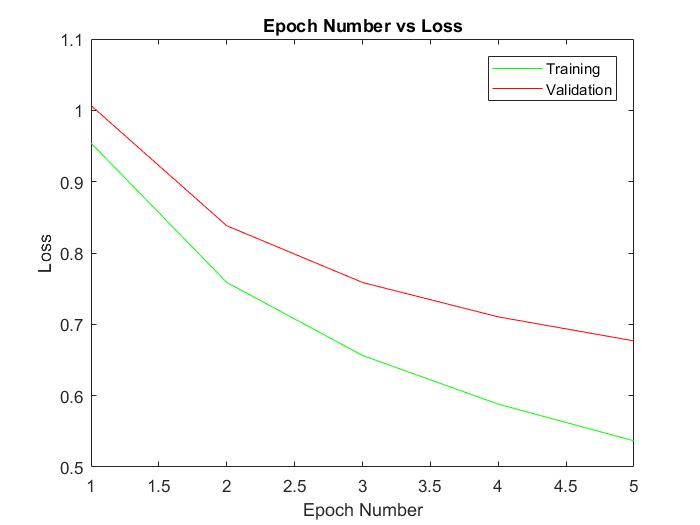
Class 9 and Class 10 is most confused in my network, which corresponds to the letter **‘I’** and **‘J’**.

Q3.2.1

Learning rate: 0.01

Classes=36



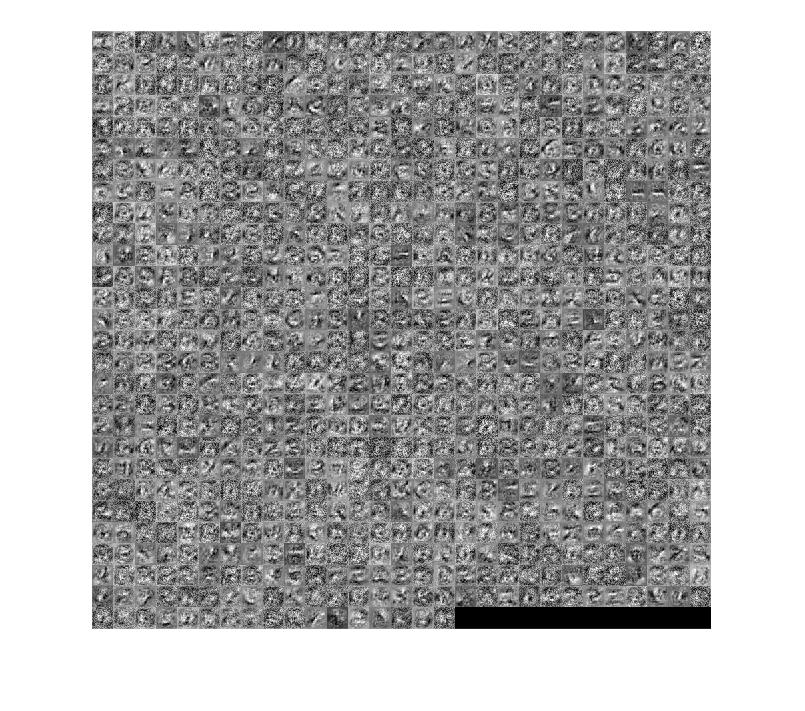


Q3.2.2

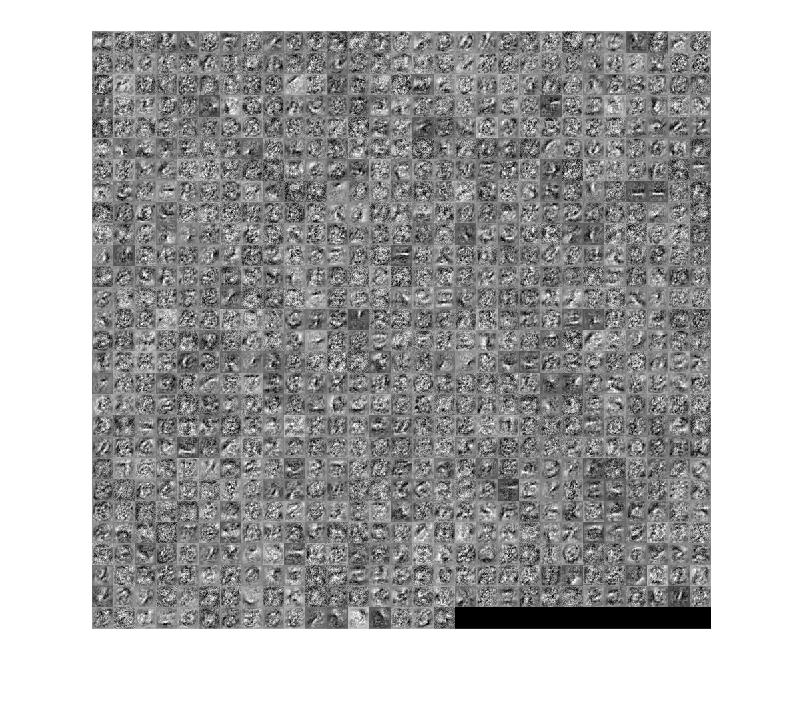
Network Accuracy on Test Set: 81.67

Network Cross Entropy Loss on Test Set: 0.5528

Pre-trained weights



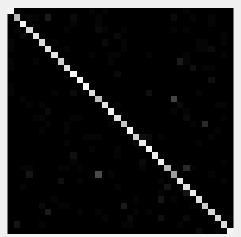
Final Weights



On the fine-tuned NIST 36 it is can be seen that the difference between the pre trained and the trained one is that it the features such as curves, lines ,circles are more sharper and visible.

Q3.3.3

Confusion Matrix

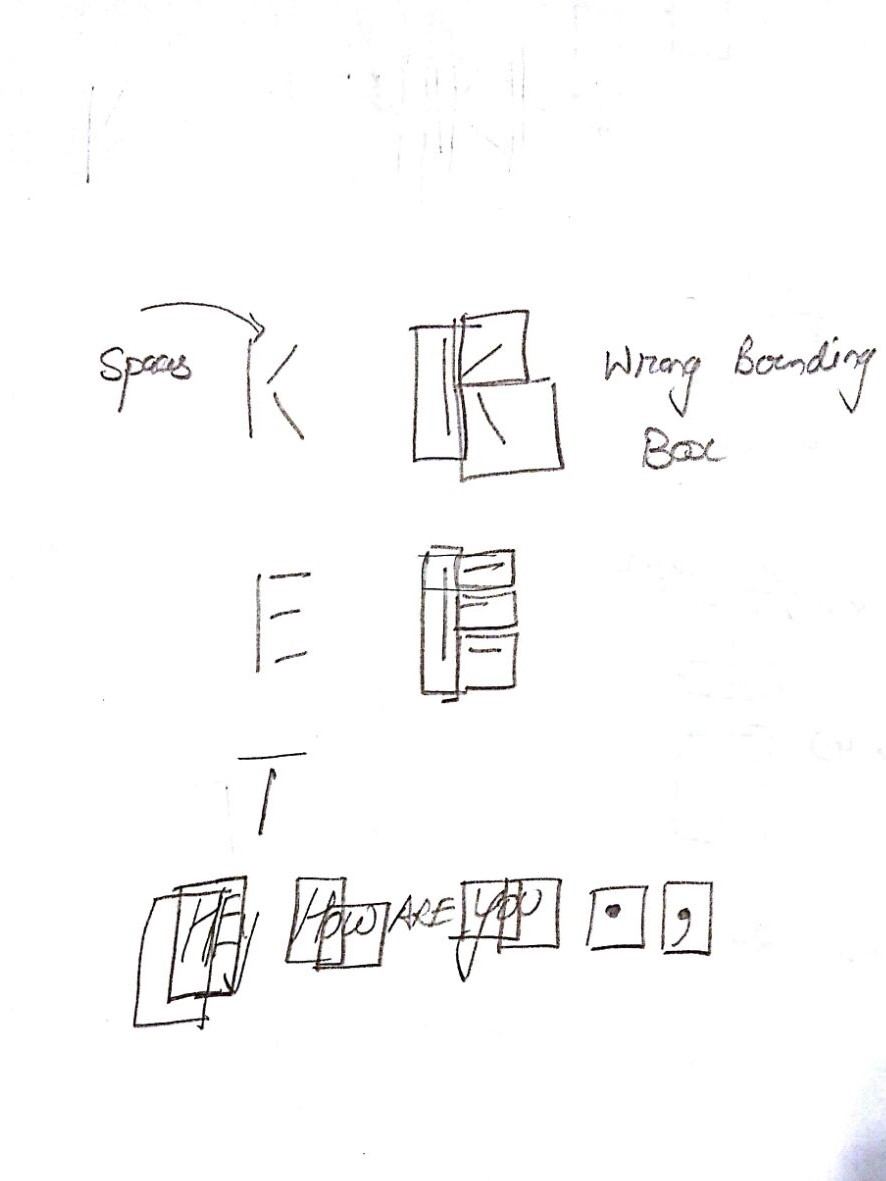


Class 15 and 27 are mostly confused with each other, which corresponds to the letter ‘O’ and number ‘0’. This is very common as almost all the features for the letter O and 0 are same. Introducing more classes has introduced the confusion between the characters as seen above and therefore reduced the accuracy of the system.

Q4.1

The following are taken as assumptions

1. The first big assumption is that the text has sufficient spacing in between them both in horizontal and vertical direction, which might not be the case in the real world.
2. Second assumption is that, all the letters themselves are assumed to be connected. For example, in the case of E, the vertical and horizontal lines are assumed to be connected. This is not true in the real word. Refer the below figure for better understanding.
3. The third assumption is that, the text is assumed to be void of all the non-letter elements such as, “ “, . In real scenario, this may add to the failure of the system.

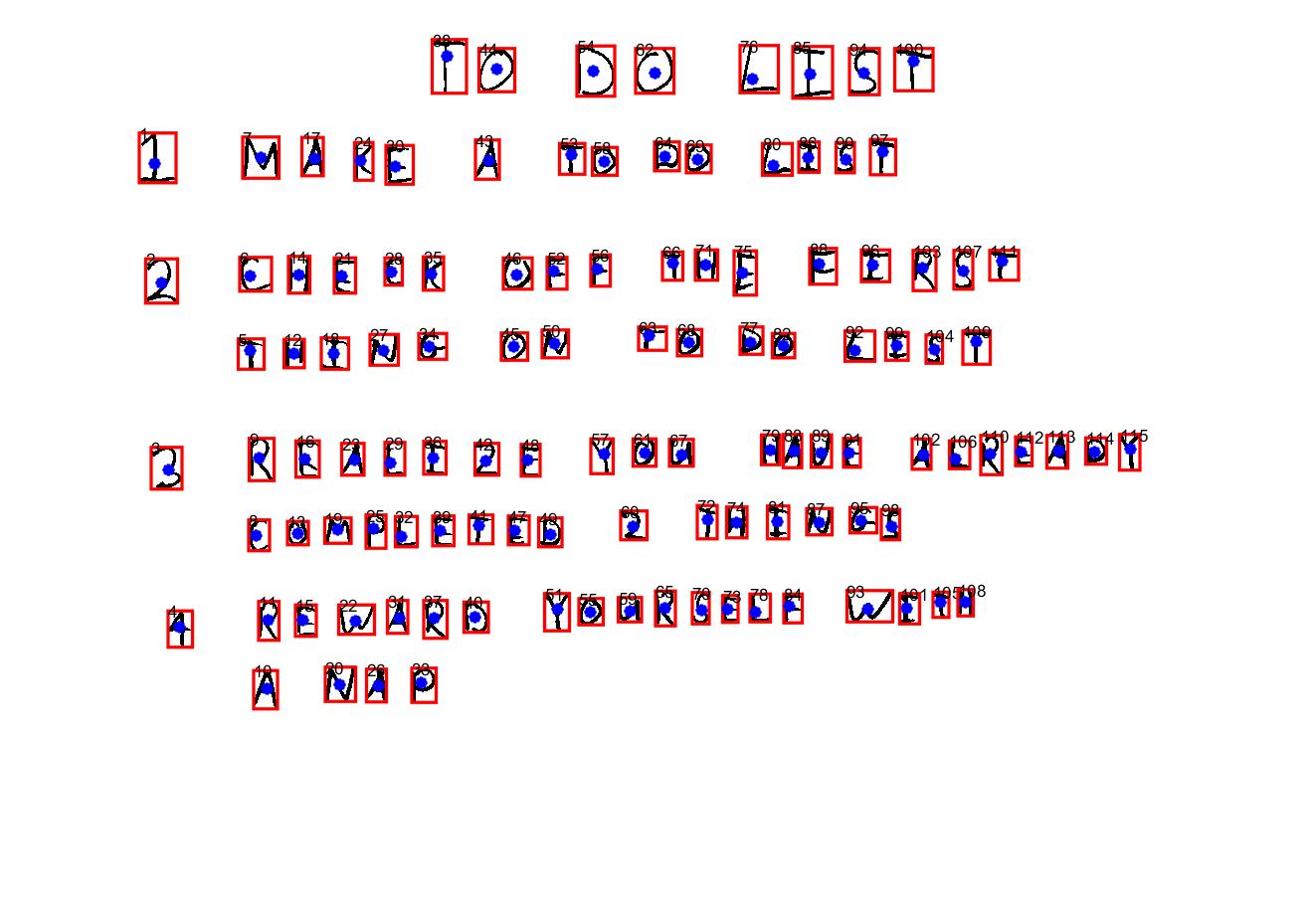


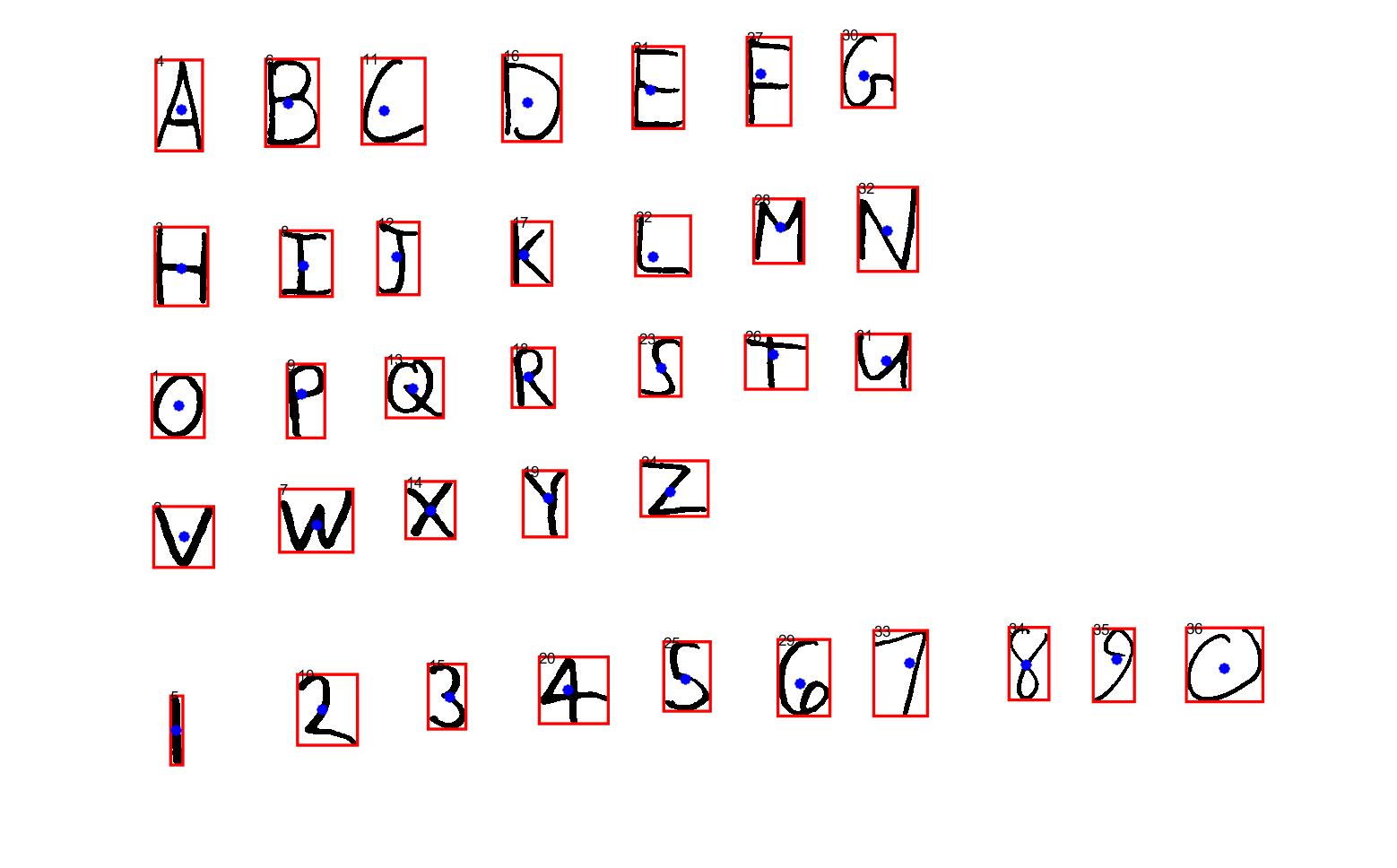
As seen from the above, the character detection can fail,

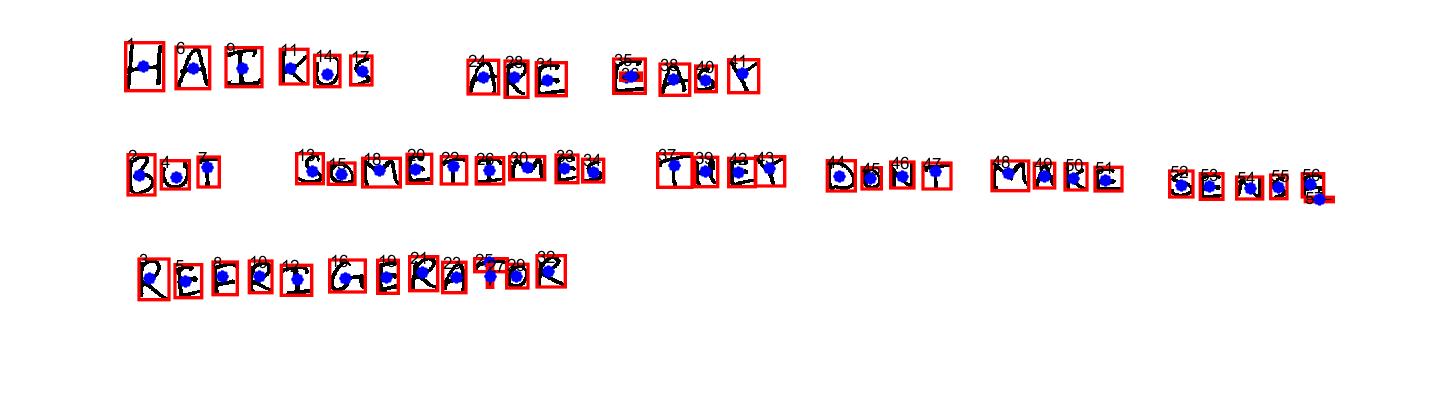
1. when the letters are closely spaced and contains elements like , and .
2. when the letters are not connected.

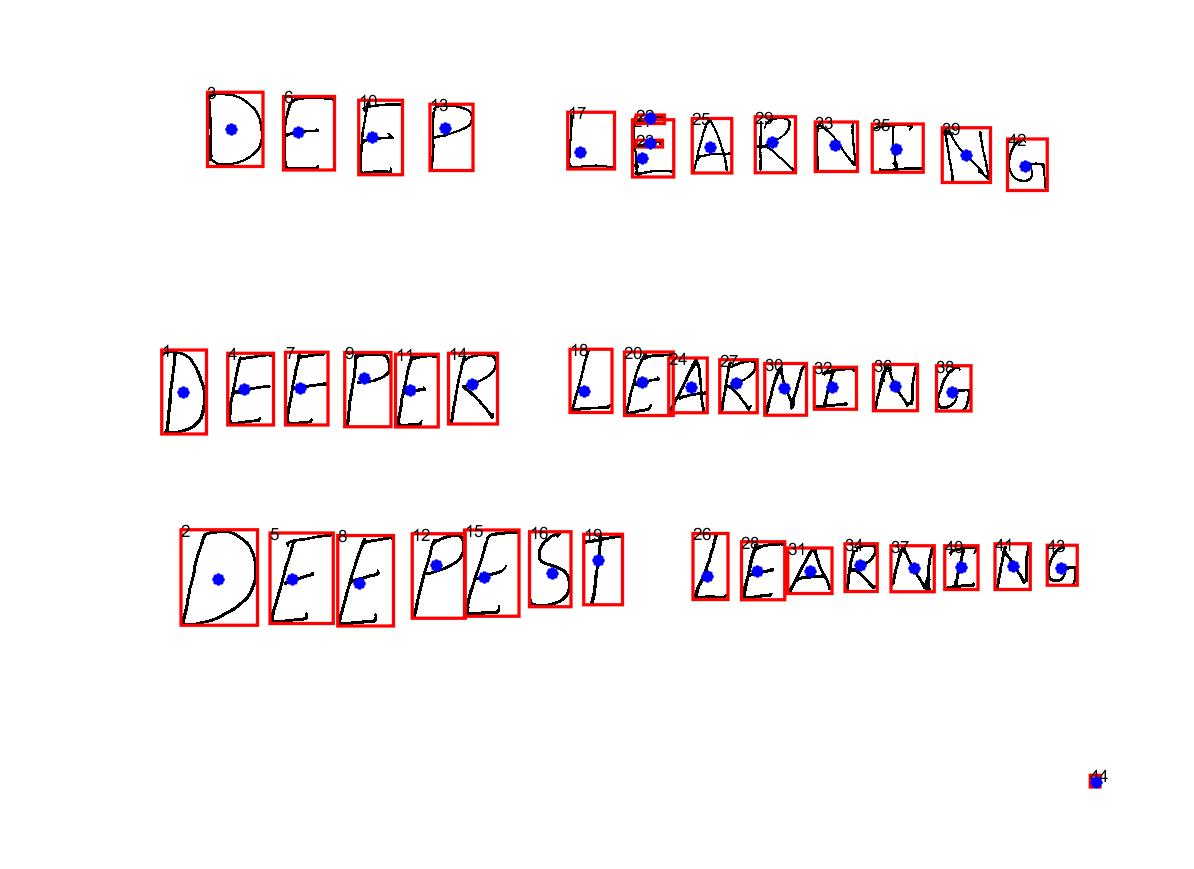
Q4.3

The Letters look bold , because the image has been blurred with a sigma of 2.5.









Q 4.5

Output 1:

'T0DOLIST LMAKEAT0D0LIST 2CHECK0FETHEFIRST THING0NT0D0LIST 3RC0EMAPLLIETZEEDY20UTHNIANVEGSALREADY 4RFWARDY0URSELFWITH PNAA '

Output 2:

'ABCDGFG HIJKLMW QFQRSTV VWXYZ 978JSGG328 '

Output 3:

'HAIKGSAREGMASX GQTSQMETIMESTHEXDDNTMAKESFMANGFM FRR3QFQEARGIE '

Output 4:

'CCOCFCKFMAKULNG DEEYCKLEAKNING TFKNIGNFASLC8CCC '