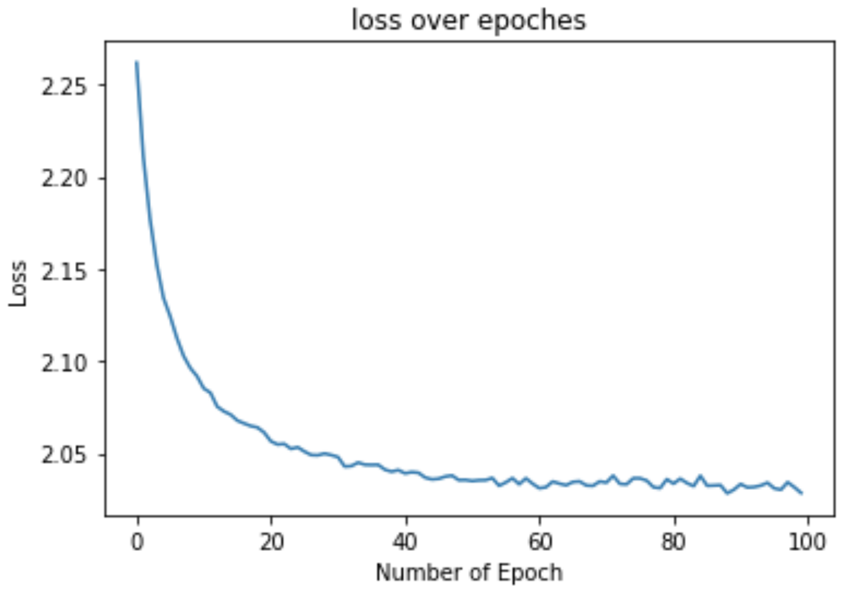
CIS 519: Homework 3

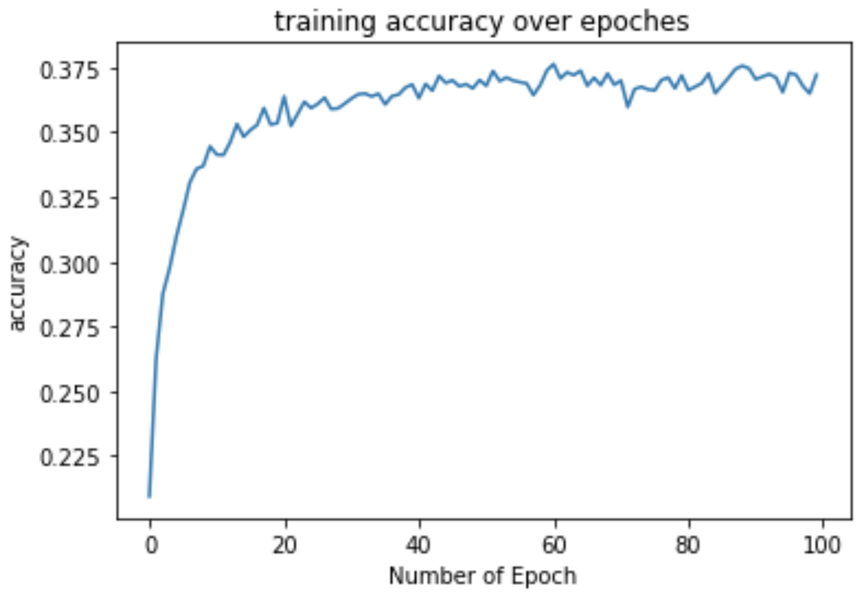
Nikhil Jamdade jnikhil@seas.upenn.edu Penn ID: 56849791

Experiment 1: Feed-forward Neural Network

Training Accuracy: 37.22 %

Test Accuracy: 35 %

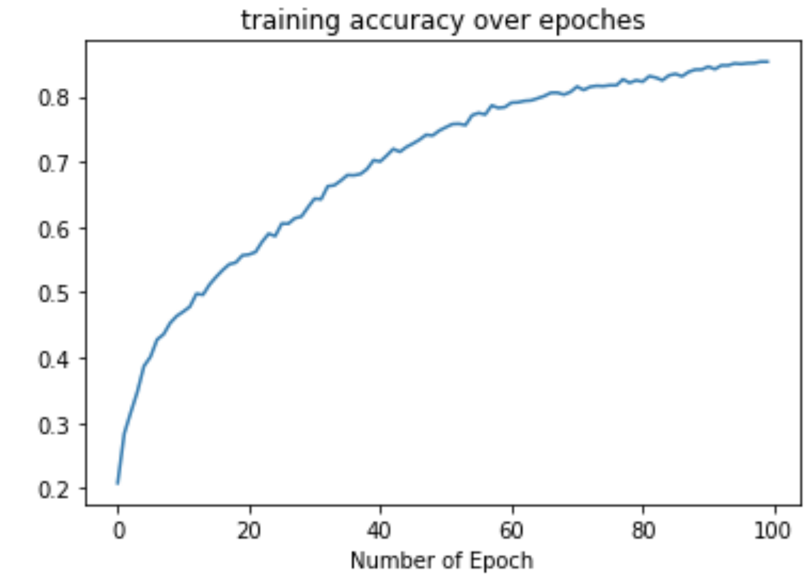


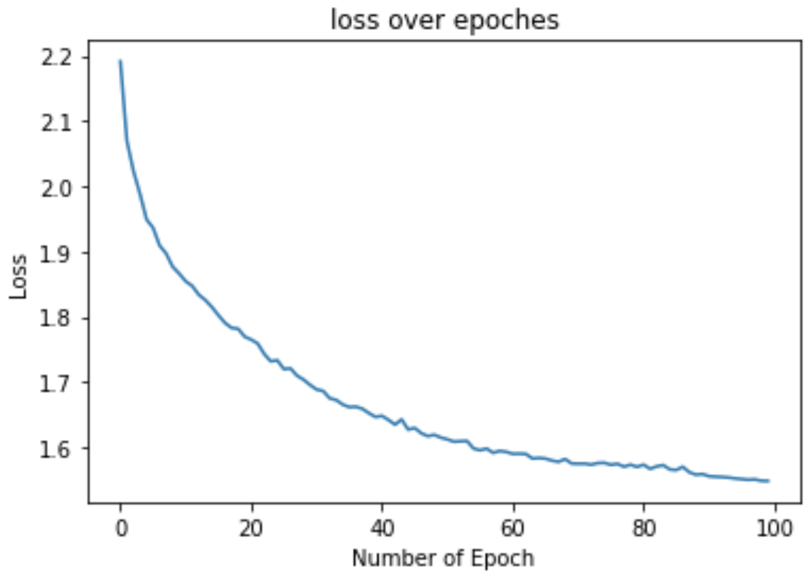


Experiment 2 : Convolutional Neural Network

Training Accuracy: 85.33 %

Test Accuracy: 47.9 %

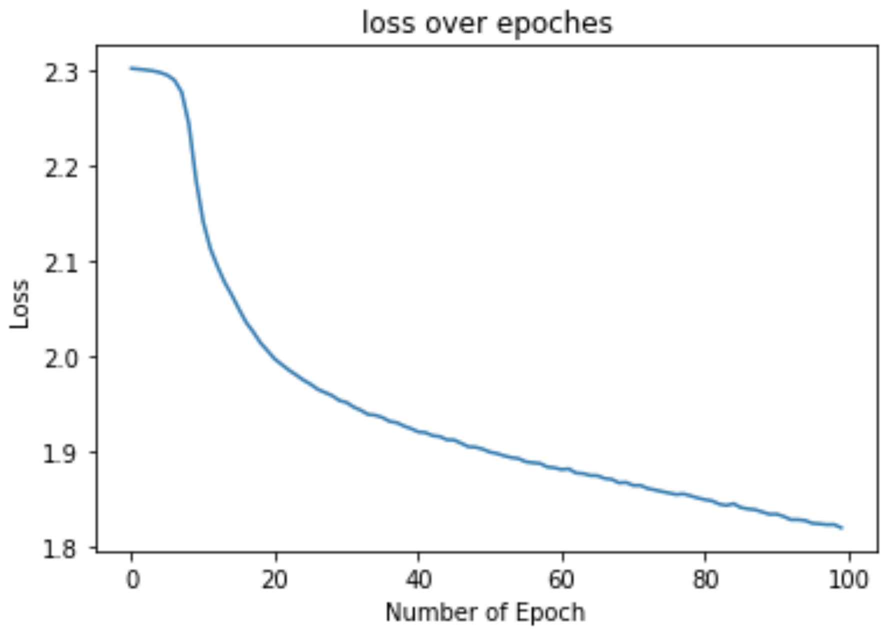


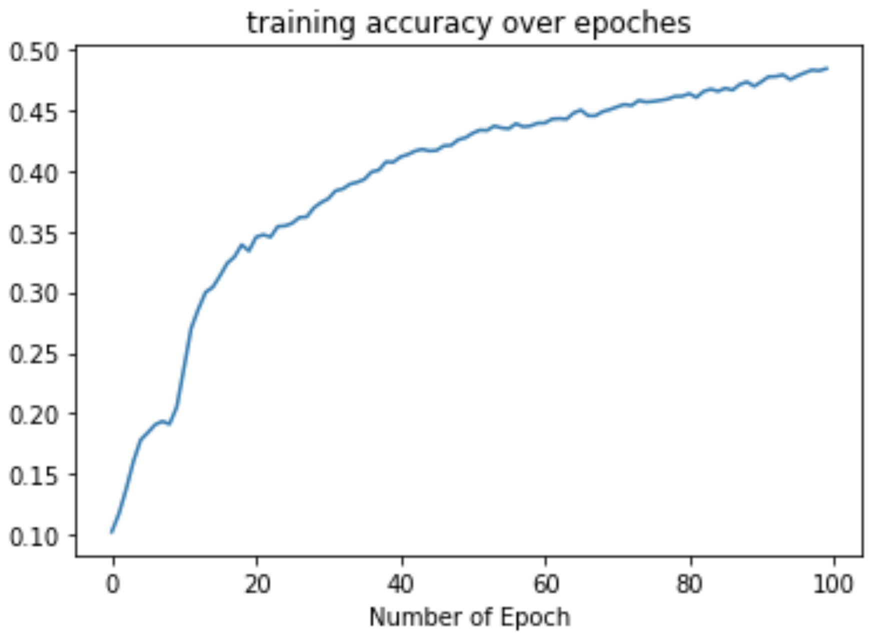


Experiment 3

Training Accuracy: 48.43 %

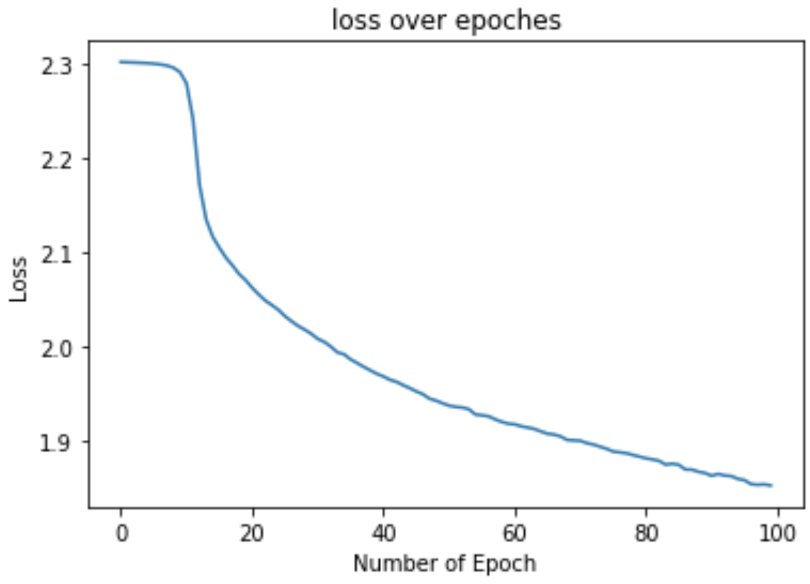
Test Accuracy: 46.15 %

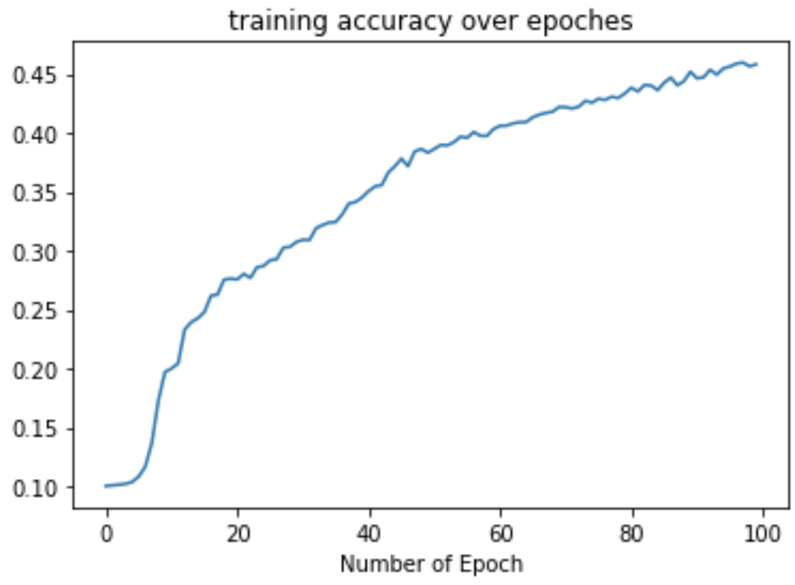




Experiment 4:

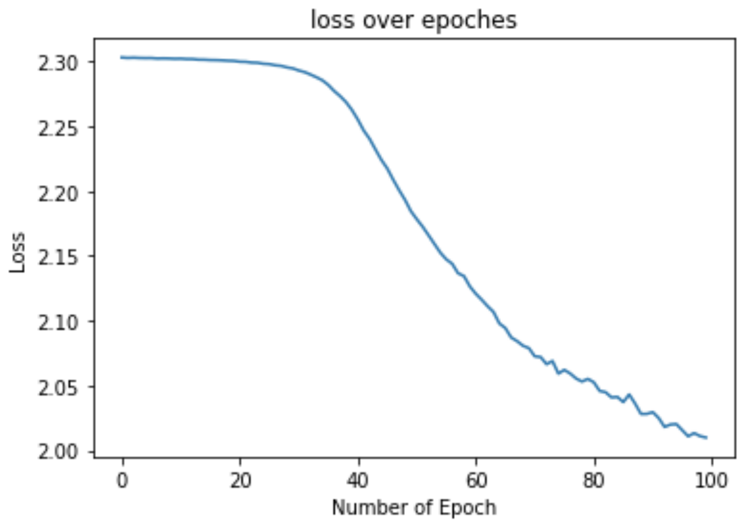
1. Batch Size: 64
   training accuracy: 45.81
   testing accuracy: 43.9

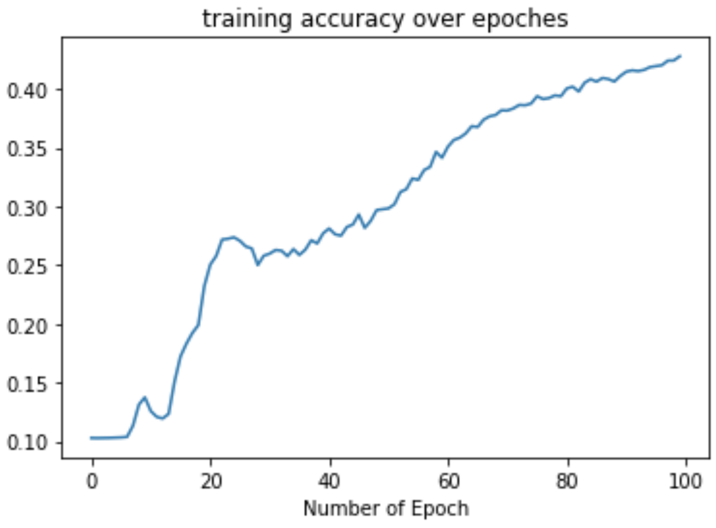




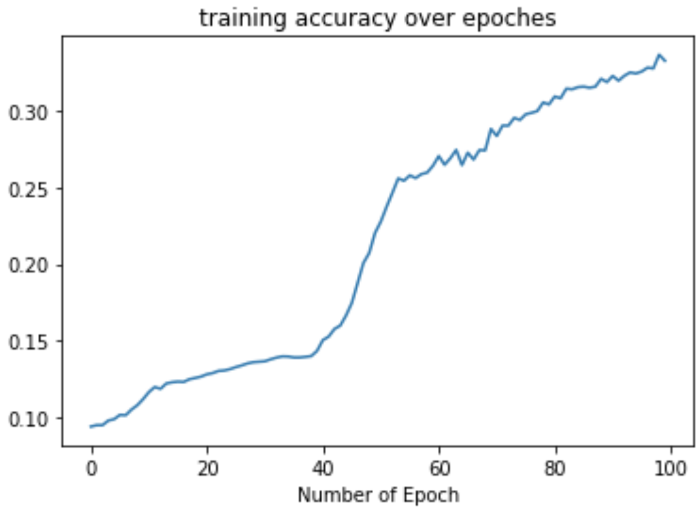
2. Batch Size: 128

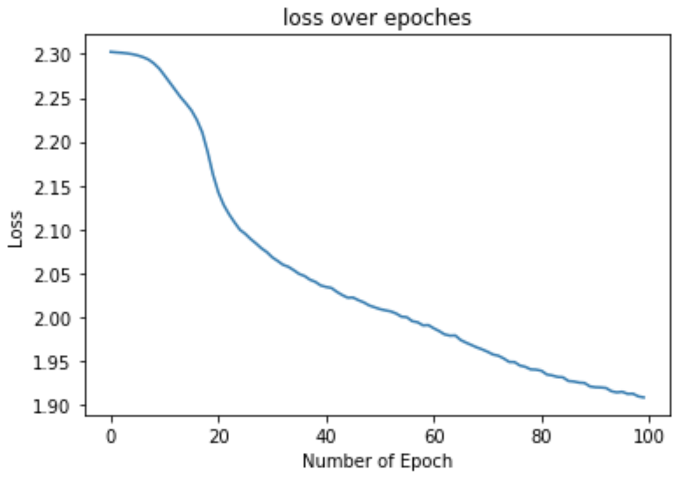
   training accuracy: 42.78 %
   testing accuracy: 41.15. %





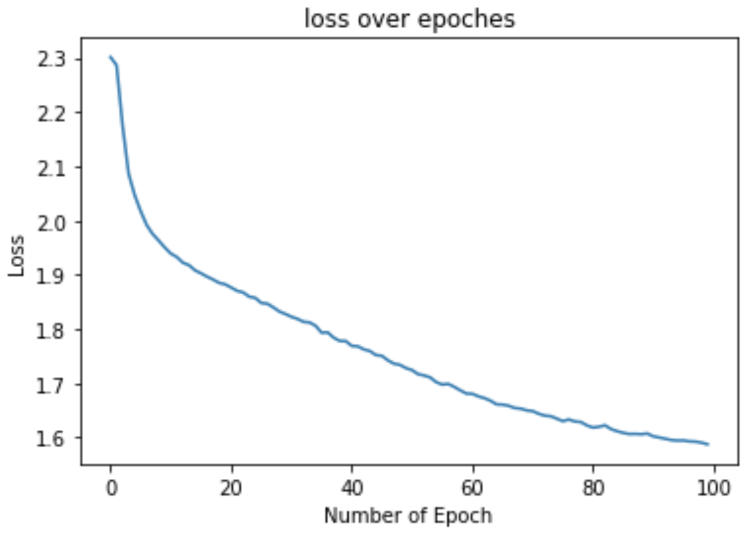
Batch Size: 256
training accuracy: 33.28 %
testing accuracy: 33.45 %

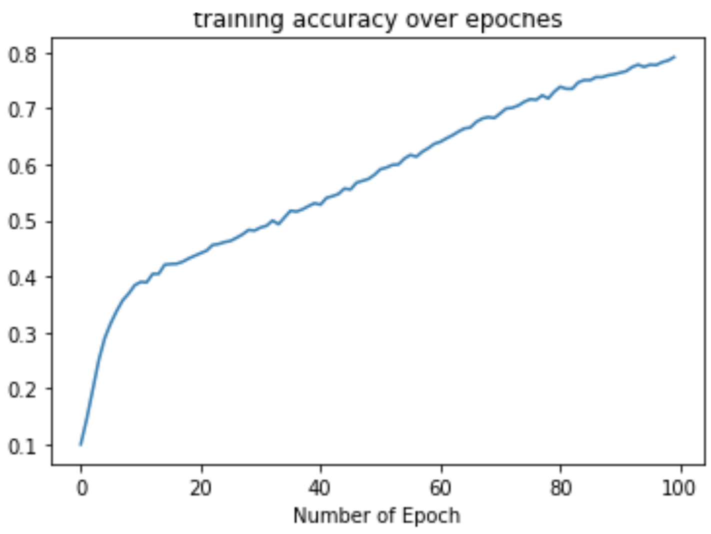


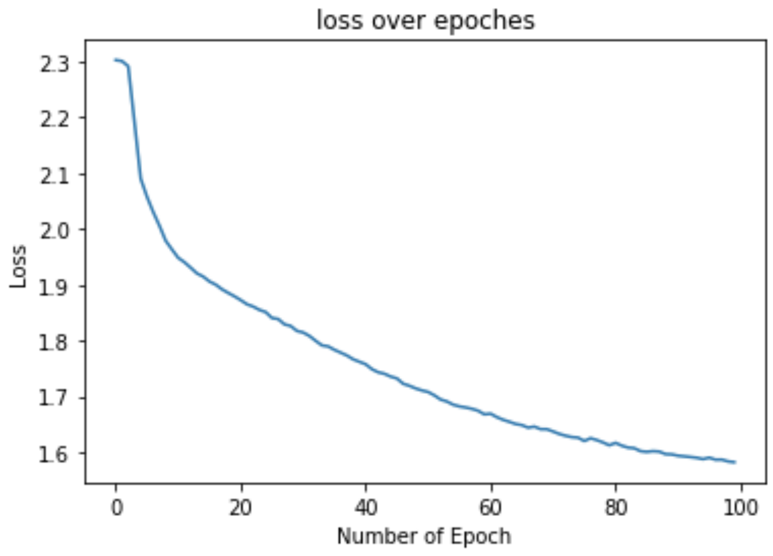


Learning Rate : 0.005

training accuracy: 79.18 5
testing accuracy: 48.25 %

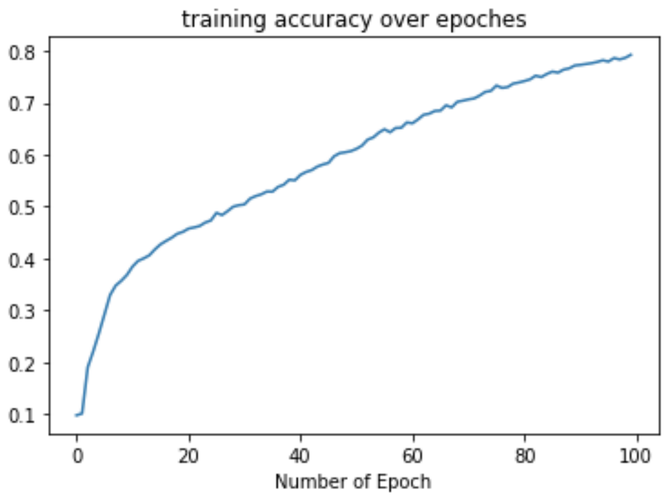






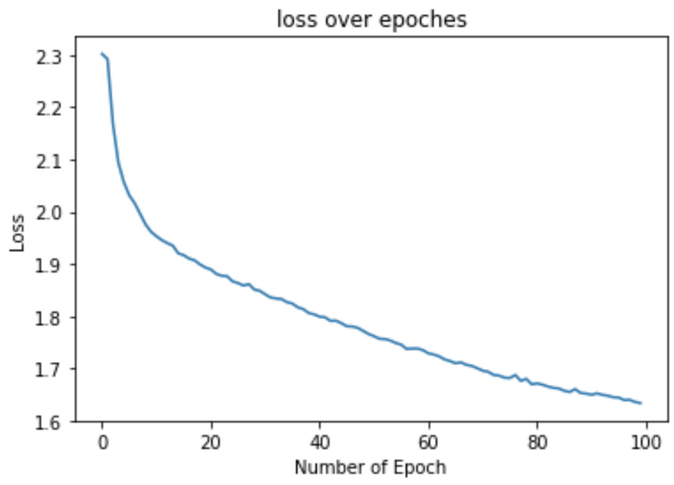
3. **Convolution Kernel: 3*3**

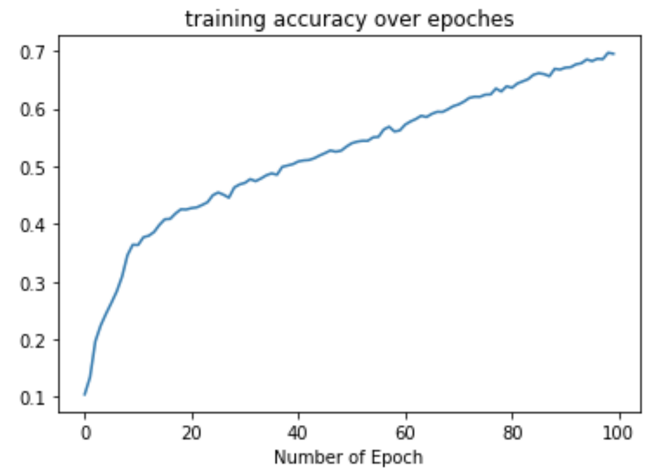
training accuracy: 79.28. %
testing accuracy: 50.15 %



Kernel Size: 7x7

training accuracy: 69.48
testing accuracy: 47.65

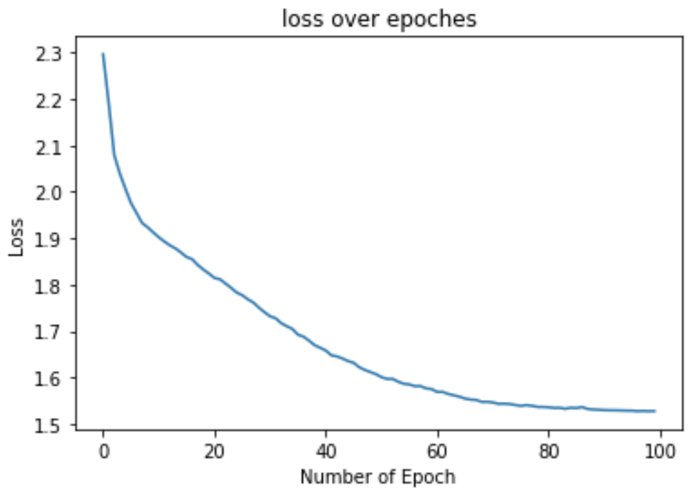


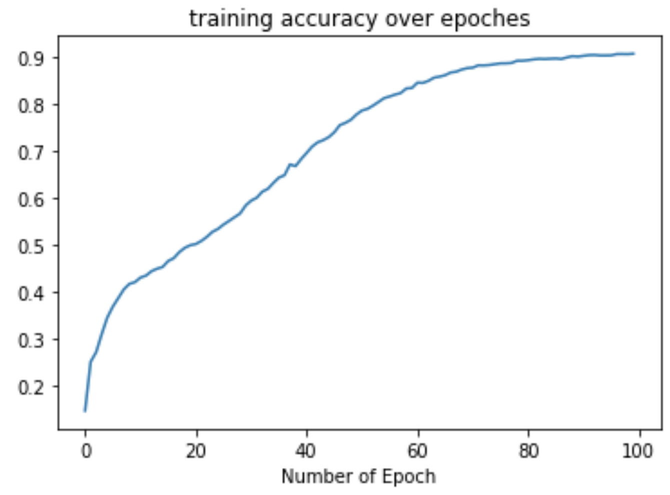


4 **Number of Neurons**

Fully connected layer1 output: 4096
Fully Connected layer2 output:1024

training accuracy: 90.77%
testing accuracy: 51.15%

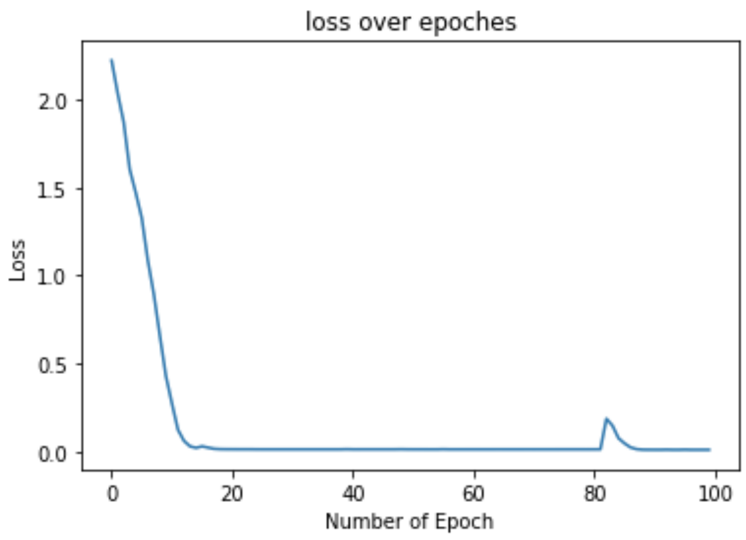


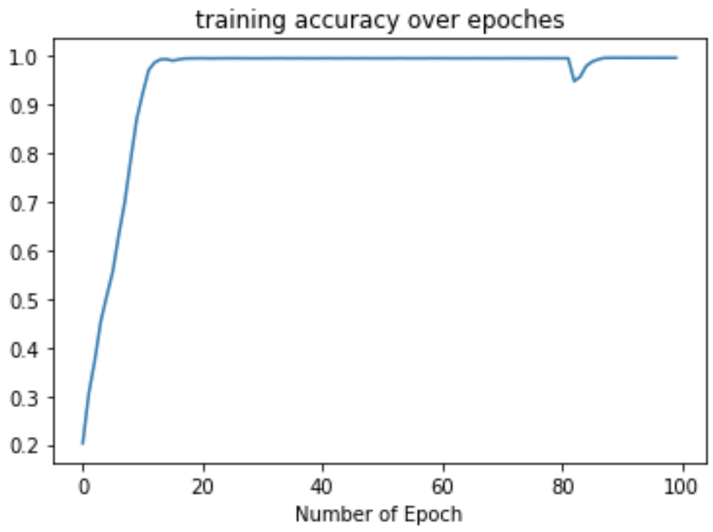


5 **Number of Fully connected layers - 2**

Number of Neurons Fully connected layer1: 4096

final training accuracy: 99%
final testing accuracy: 54%





Therefore, best test accuracy and the best hyperparameters:
training accuracy: 99%
test accuracy: 54%

Convolution Kernel Size: 3x3
Batch Size: 64
Learning Rate: 0.005

Number of neurons in the fully connected layers: 4096
Number of Layers- 2

Layer	Hyperparameters
Convolution1	Kernel size = (3x3x6), stride = 1, padding = 0. Followed by ReLU
Pool1	MaxPool operation. Kernel size = (2x2)
Convolution2	Kernel size = (3x3x16), stride = 1, padding = 0. Followed by ReLU
Fully Connected1	Output channel = 4096. Followed by ReLU
Fully Connected2	Output channel = 10. Followed by ReLU

Experiment	Training Accuracy in (%)	Test Accuracy in (%)
Experiment 1:Feed-forward Neural Network	37.22	35
Experiment 2: Convolutional Neural Network	85.33	47.9
Experiment 3: Image Processing	48.43	46.15
Experiment 4: Hyper-parameterization with 3 fully connected layer	90.77	51.15
Experiment 4: Hyper-parameterization with 2 fully connected layer	99	54

In case, Experiment 4 Hyper parameterization, test accuracy improves with changing Kernel size to 3x3, Learning rate to 0.005, changing neurons and number of fully connected layers.

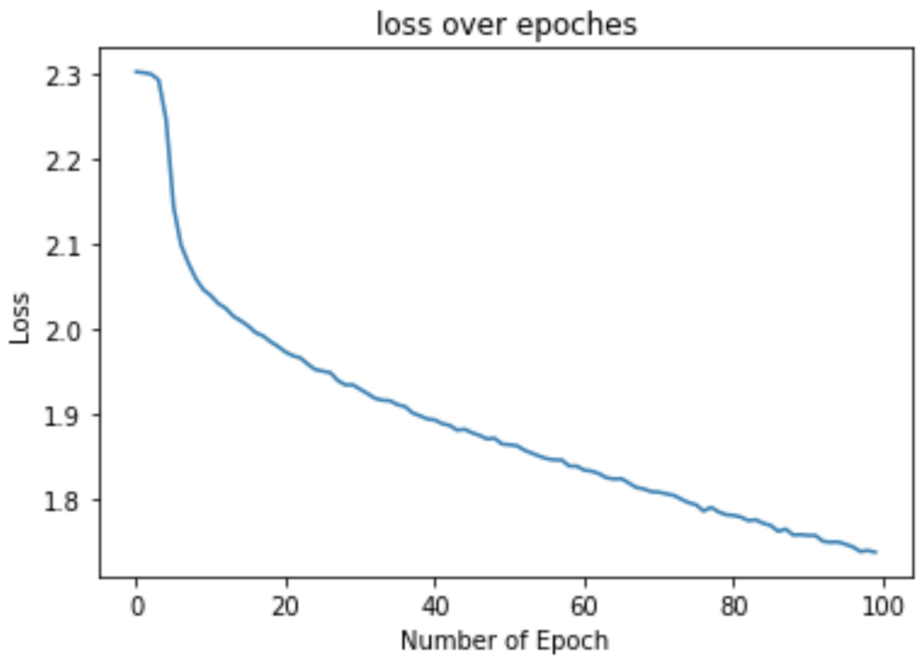
Accuracy increases in experiment two because the architecture implements 3 FC layers with two convolution layers including filtering and Introducing Non Linearity (ReLU). This can handle more variations. In experiment one because of less features, the overfitting is very likely. Moreover weight-sharing between nodes of the network in convolutional layers does help a bit since it limits the class of models a bit. It helps to eliminate non-maximal values and to provide a form an invariant translation. The output layers ensures the classification of the input character. In these layers all neurons are fully connected and have a unique set of weights so they can detect complex features and perform classification.

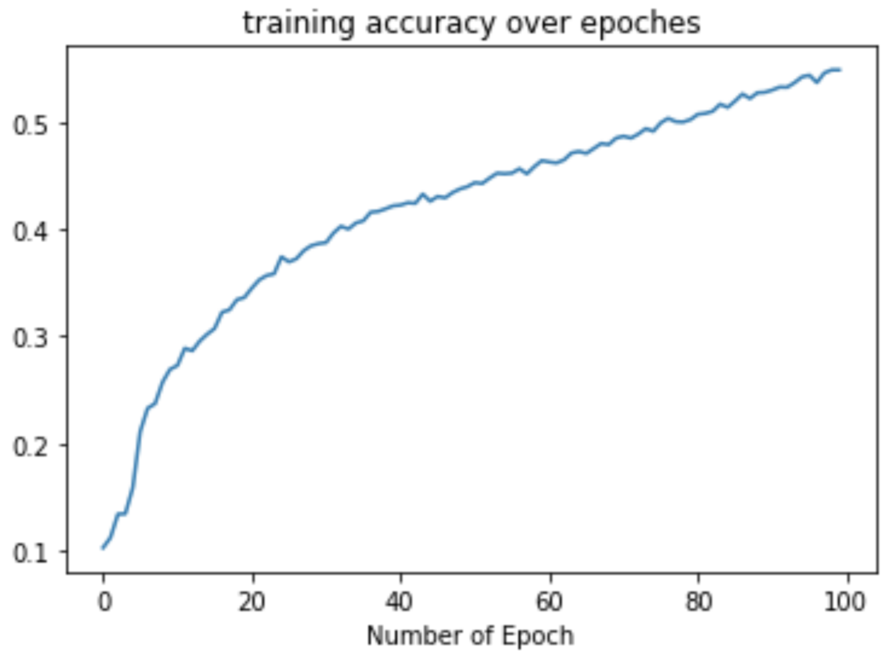
In experiment 3, we normalize image to cater to variations issues in images and to get features irrespective of variations. The reason we do normalization is because it helps to get each feature to have a similar range so that our gradients don't go out of control, In the process of training our network, we add and multiply weights in order to cause activations that we then back propogate with the gradients to train the model. This process ideally improves the classification and hence accuracy (ideally).

The learning rate multiplicative effects on the training dynamics because if affected computed gradient Moreover higher learning rates will decay the loss faster.
Small batch size helps in time efficiency of training and reduce noisiness of the gradient estimate. Similarly, the number of neurons affects the dynamics of NN, because it directly involves number of image features and proportions of details from one layer to another. She times more features helps to improve accuracy but it increases complexity.

Additional Experiment:

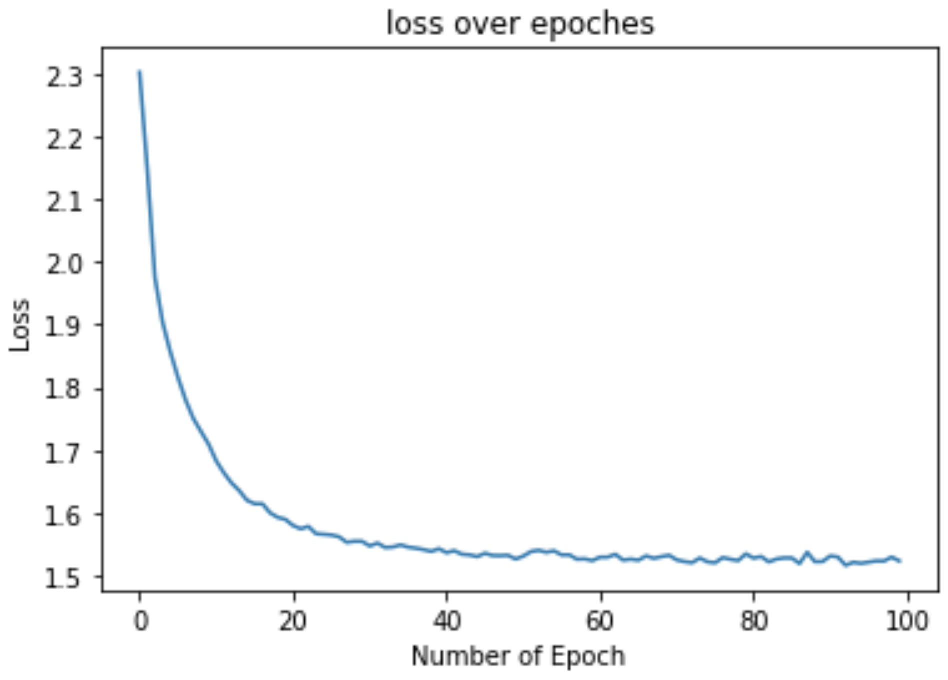
Data Augmentation: Horizontal Flip, Vertical Flip and crop
training accuracy: 54.88%
testing accuracy: 42.8 %

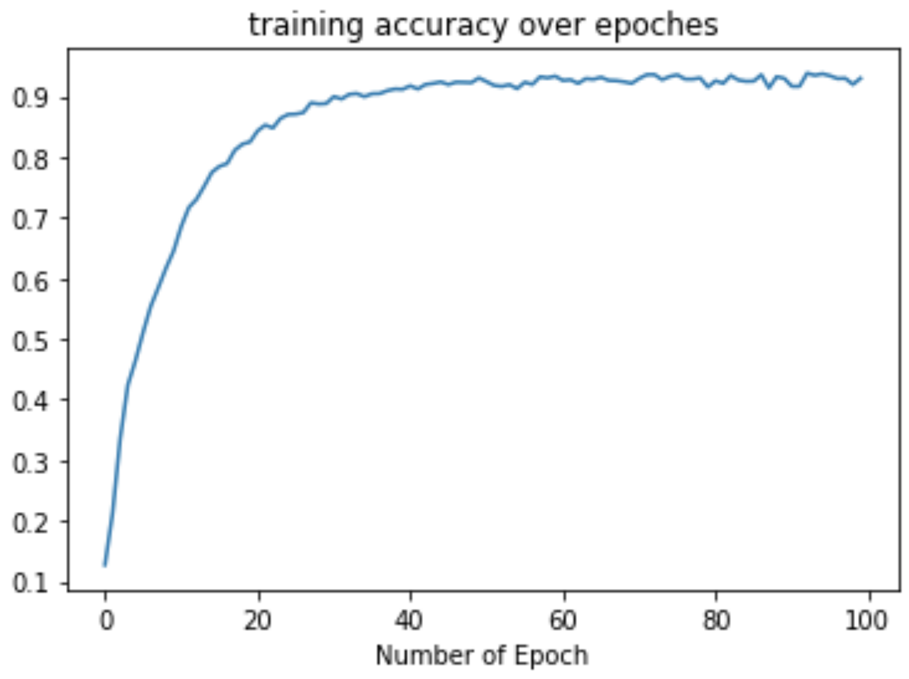




Using Adam optimizer

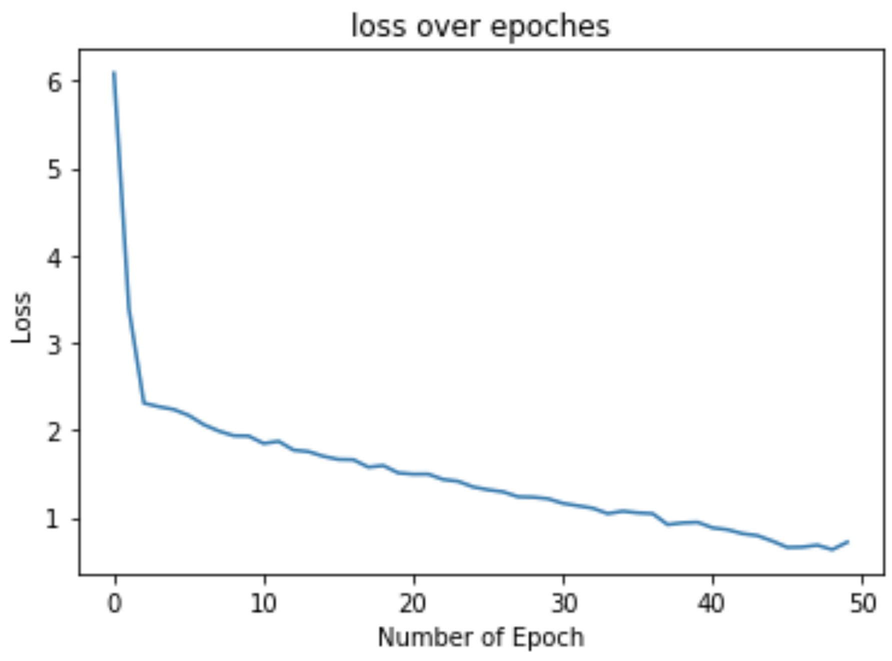
training accuracy: 93.01%
testing accuracy: 44.2%

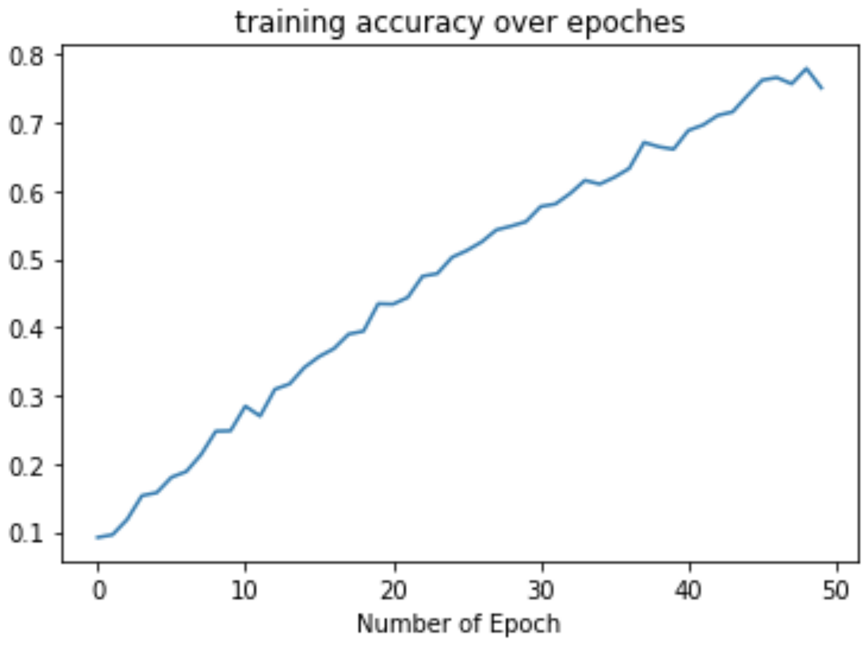




Use of Alexnet:

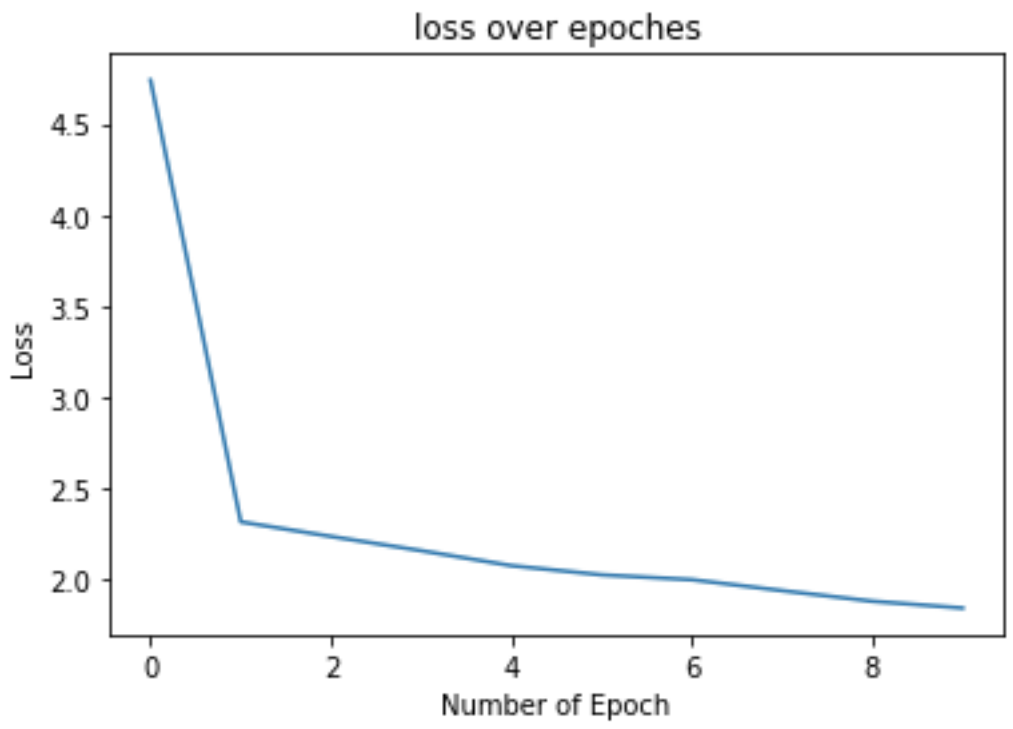
Number of epoch: 50
training accuracy: 75.16%
testing accuracy: 44.15%

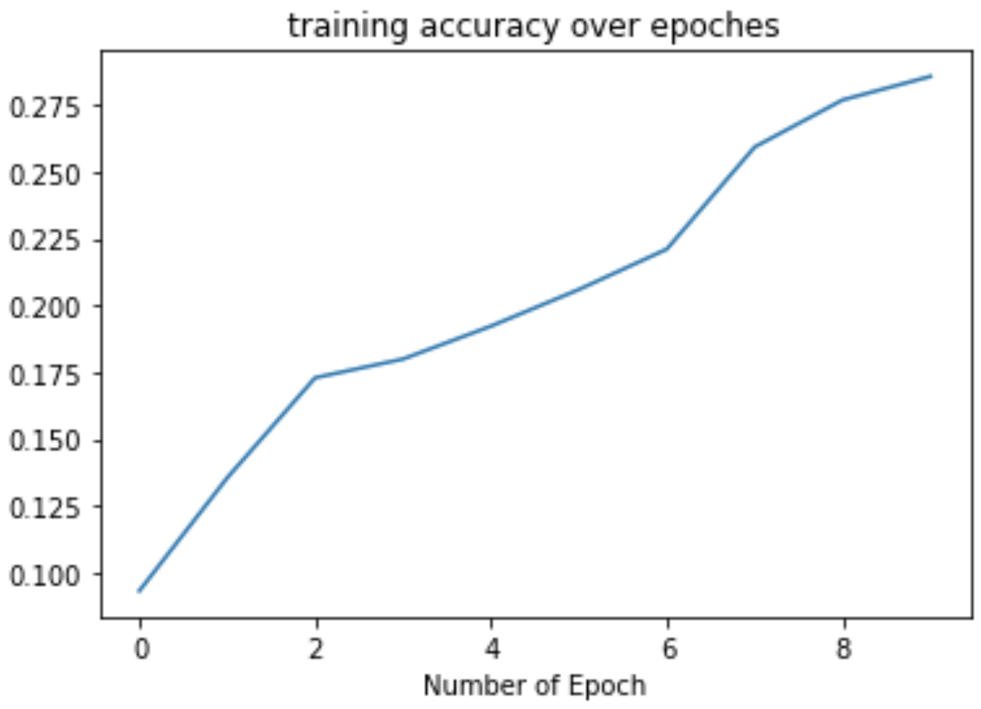




Number of epoch:10

training accuracy: 28.56 %
testing accuracy: 30.2 %





		Hypothesis 1				Hypothesis 2			
i	Label	D_0	$f_1 \equiv$ [x >1]	$f_2 \equiv$ [y >5]	$h_1 \equiv$ [x >1]	D_1	$f_1 \equiv$ [x >2]	$f_2 \equiv$ [y >11]	$h_2 \equiv$ [y >11]
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
1	$-$	0.1	-	+	-	0.0625	-	-	-
2	$-$	0.1	-	-	-	0.0625	-	-	-
3	$+$	0.1	+	+	+	0.0625	+	-	-
4	$-$	0.1	-	-	-	0.0625	-	-	-
5	$-$	0.1	-	+	-	0.0625	-	+	+
6	$-$	0.1	+	+	+	0.25	+	-	-
7	$+$	0.1	+	+	+	0.0625	+	-	-
8	$-$	0.1	-	-	-	0.0625	-	-	-
9	$+$	0.1	-	+	-	0.25	-	+	+
10	$+$	0.1	+	+	+	0.0625	+	-	-

Table 1: Table for Boosting results

1 Boosting

() (1)
D0:Start with uniform distribution: 1/10=0.1

Using graph and given data: Lets consider f1=f(x>1) and f2=f(y>5)//
The labels are shown in the table columns

(2) The number of mistakes in f1 < number of mistakes in f2 so there less error using f1. Selecting f1 as hypothesis. The labels are shown in table

(3)

$$\epsilon = 2/10 = 0.2$$

$$\alpha = 1/2 * \ln((1-\epsilon)/\epsilon)$$

.

$$\alpha_t = 1/2 * \ln((0.8)/0.2) = 1/2 * \ln(4)$$

$$\alpha_t = 0.693$$

$$z_t = \sum D_0(e^{-\alpha_0 * y_i * h_0(x_i)})$$

$$z_t = 0.1 * (0.8 * e^{-0.693}) + 0.1 * (2 * e^{0.693})$$

For y(i)=$h(x(i))$:

$$D_(t+1) = (D_t(i)/z_t) * (e^{-\alpha_t})$$

For not mistake:

$$D_1 = 0.0625$$

For y(i)!=$h(x(i))$:

$$D_(t+1) = (D_t(i)/z_t) * (e^{\alpha_t})$$

For mistake:

$$D_1 = 0.25$$

(4) H final(x)=$sign(\sum(\alpha_t * h_t(x)))$

H final(x)=$sign(0.693 * h(f(x > 1)) + (0.550 * h(f(y > 11)))$

f(x)=$\{1(+)\ \}\ if x > 1\ else\ -1\ (-)$
f(y)=$\{1(+)\ \}\ if y > 11\ else\ -1\ (-)$

1 SVM

(a) (1)
Easy Solution (w,θ) that can separate the positive and negative examples

$$w = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$$
$$\theta = 0$$

(2)
SVM Optimization Solution (w,θ) using the Hard SVM formulation

$$w = \begin{bmatrix} -1/2 \\ -1/2 \end{bmatrix}$$
$$\theta = 0$$

(3)
SVM Solution understanding:
Although several classifiers separate the data, the distance with which the separation is achieved is different. There we find the find the closest point. After that the W is selected such that the one the gives the maximal margin value across all w's of size 1. Among all w's that separate the data with margin 1, we choose the one with minimal size. The normalization of w0 corresponds to the largest margin separating hyperplane.

Therefore, The closest point to the classifier is selected such it has maximum margin. The line x+y=0 i.e hyperplane having equal distance from two points with margin atleast greater than 1. This simple hypothesis separate the data with maximum margin.

(b)(1)
The set of indices of Support Vectors from the six example given

$$I = (2, 0)$$

$$(-2, 0)$$

These are the indices which lie on support vectors

(2)
From the given equation, the dual representation of w* is given as..

$$\alpha_t * y_1 * x_1 + \alpha_2 * y_2 * x_2 = w*$$

$$\alpha_t * (-1) * \begin{bmatrix} 2 \\ 0 \end{bmatrix} + \alpha_2 * (+1) * \begin{bmatrix} -2 \\ 0 \end{bmatrix} = w*$$

$$\begin{bmatrix} -2 * \alpha_1 \\ -2 * \alpha_2 \end{bmatrix} = \begin{bmatrix} -1/2 \\ -1/2 \end{bmatrix}$$
$$\alpha_1 = 1/4$$
$$\alpha_2 = 1/4$$

(2)
The Objective Function Value of Hard SVM..

$$= 1/2 * ||w||^2 = 1/2 * ((-1/2)^2 + (-1/2)^2)$$

$$ObjectiveFunctionValue = 1/4$$

(3)
A large value of C means that misclassifications are bad – we focus on a small training error (at the expense of margin). A small C results in more training error, but hopefully better true error.

With the given equation for Soft SVM..there two terms involved first being regularization term and second one empirical loss term

For C=0,(ideal) the solution to given optimization problem yields the hyperplane found in (a)2

If C=∞, then generalization term in negligible and empirical loss term is taken into account

We can use general Form of a learning algorithm i.e minimize empirical loss and Regularize it to avoid over fitting. However we need to avoid under fitting, hence we can make the balance between variance and bias

Lager value of C gives high variance and low bias which increases model complexity and over-fitting

Smaller value of C gives low variance and low bias which provides simple model but it under-fits the data