Name : Sait Akturk

Student No : 21501734

QUESTION 1

- **Maximum Posteriori a Estimation**

$$\hat{W}_{MAP} = \underset{W}{argmax}\ \mathrm{P}\ (\mathrm{W} \mid \mathrm{X}) = \underset{W}{argmax} \frac{\mathrm{P}\ (\mathrm{W})\ \mathrm{P}(\mathrm{X} \mid \mathrm{W})}{P(X)}$$

$$\hat{W}_{MAP} \propto \underset{W}{argmax}\ \mathrm{P}\ (\mathrm{W})\ \mathrm{P}(\mathrm{X} \mid \mathrm{W})$$

$$\hat{W}_{MAP} = \underset{W}{argmax}\ \log \mathrm{P}\ (\mathrm{W})\ \mathrm{P}(\mathrm{X} \mid \mathrm{W})$$

$$\hat{W}_{MAP} = \underset{W}{argmax}\ \{\ \mathrm{logP}\ (\mathrm{W}) + \log \mathrm{P}(\mathrm{X} \mid \mathrm{W})\}$$

$$\hat{W}_{MAP} = \underset{W}{argmin}\{-\log P(W) - \sum_{i=1}^{N} \log P(x_i|w_i)\}$$

- **Probability**

Assume :

$y^n = h(x^n, W) + \epsilon$

Assume noise ϵ has Gaussian distribution property :

$\epsilon \sim \mathcal{N}(0, \sigma^2)$

Then each response y^nbecomes a draw from Gaussian:

$y \sim \mathcal{N}(h(x^n, W), \sigma^2)$

$$P(y^n|x^n, W) = \mathcal{N}(y^n|h(x^n, W), \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp[-\frac{(y^n - h(x^n, W))^2}{2\sigma^2}]$$

- **Log Likelihood**

$$\log \mathcal{L}(W) = \log P(Y|X,W) = \log \prod_{n=1}^{N} P(y^n|x^n,W)$$
$$= \sum_{n=1}^{N} \log P(y^n|x^n,W)$$
$$= \sum_{n=1}^{N} \log\{\frac{1}{\sqrt{2\pi\sigma^2}} \exp[-\frac{(y^n - h(x^n,W))^2}{2\sigma^2}]\}$$
$$= \sum_{n=1}^{N} \{-\frac{1}{2}\log(2\pi\sigma^2) - \frac{(y^n - h(x^n,W))^2}{2\sigma^2}\}$$

- **Maximum Log Likelihood**

$$\log \mathcal{L}(W) = \underset{W}{argmax} -\frac{1}{2\sigma^2} \sum_{n=1}^{N} (y^n - h(x^n,W))^2$$
$$= \underset{W}{argmin} \frac{1}{2\sigma^2} \sum_{n=1}^{N} (y^n - h(x^n,W))^2$$

Assume $\sigma^2 = 0.5$:

$$= \underset{W}{argmin} \sum_{n=1}^{N} (y^n - h(x^n,W))^2$$

- **Inference**

$$\hat{W}_{MAP} = \underset{W}{argmin} \sum_{n=1}^{N} (y^n - h(x^n,W))^2 + \beta \sum_{i=1}^{M} w_i^2$$
$$= \underset{W}{argmin} \{\log \mathcal{L}(W) - logP(W)\}$$
$$= \underset{W}{argmin} \sum_{n=1}^{N} (y^n - h(x^n,W))^2 - logP(W)$$
$$\Rightarrow \underset{W}{argmin} - logP(W) = \beta \sum_{i=1}^{M} w_i^2$$

$$\Rightarrow \underset{W}{argmax} \log P(W) = \beta \sum_{i=1}^{M} w_i^2$$

$$\Rightarrow \underset{W}{argmax} P(w) = \exp(\beta \sum_{i=1}^{M} w_i^2)$$

At this point we can observe that:

$$P(W) = \mathcal{N}(0, \frac{1}{-2\beta}) = \sum_{i=1}^{M} (\frac{\sqrt{-\beta}}{\sqrt{\pi}} \exp(\beta(w_i - 0)^2))$$

Then:

$$\Rightarrow \log P(W) = \log \sqrt{-\frac{\beta}{\pi}} + \beta \sum_{i=1}^{M} w_i^2$$

$$\Rightarrow \underset{W}{argmax} \log P(W) = \beta \sum_{i=1}^{M} w_i^2$$

QUESTION 2

Since we know that the strength of connections
between neurons are independent of the spiking activity :

$$\sum_{i=1}^{N} E[W_{ki} s_i(t)] = \sum_{i=1}^{N} E[W_{ki}] E[s_i(t)]$$

And also we know that spiking activity of neurons in the network are uncorrelated :

$E[s_a(t) s_b(t)] = E[s_a(t)] E[s_b(t)]$ for a $\neq b$

Then we could find μ_k as :

$$\mu_k = E[\sum_{i=1}^{N} W_{ki} s_i(t)] = \sum_{i=1}^{N} E[W_{ki} s_i(t)]$$

$$= \sum_{i=1}^{N} E[W_{ki}] E[s_i(t)], \text{ we know that } E[s_i(t)] = m$$

$$= m \sum_{i=1}^{N} E[W_{ki}] \Rightarrow E[W_{ki}] = \epsilon J\beta - \epsilon J(1-\beta) + (1-\epsilon)0 = \epsilon J(2\beta - 1)$$

We find result that independent from i :

$$= m \sum_{i=1}^{N} \epsilon J(2\beta - 1)$$
$$= mN\epsilon J(2\beta - 1)$$

$$\sigma_{kl}^2 = E[(x_k(t) - \mu_k)(x_l(t) - \mu_l)(k \neq l)$$

We know that any μ_x has result independent from x, we could write:

$$\sigma_{kl}^2 = E[(x_k(t) - \mu)(x_l(t) - \mu)(k \neq l)$$
$$\sigma_{kl}^2 = E[x_k(t)x_l(t) - \mu x_k(t) - \mu x_l(t) + \mu^2]$$
$$\sigma_{kl}^2 = E[x_k(t)x_l(t)] - \mu E[x_k(t)] - \mu E[x_l(t)] + \mu^2$$

We know that for any a $\Rightarrow E[x_a(t)] = \mu$,thus :

$$\sigma_{kl}^2 = E[x_k(t)x_l(t)] - \mu^2 - \mu^2 + \mu^2$$
$$\sigma_{kl}^2 = E[x_k(t)x_l(t)] - \mu^2$$
$$\sigma_{kl}^2 = E[(\sum_{a=1}^{N} W_{ka} s_a(t))(\sum_{b=1}^{N} W_{kb} s_b(t))] - \mu^2$$
$$\sigma_{kl}^2 = E[(W_{k1}s_1(t) + W_{k2}s_2(t) \cdots + W_{kn}s_n(t))((W_{l1}s_1(t) + W_{l2}s_2(t) \cdots + W_{ln}s_n(t))] - \mu^2$$

We have two different results that $a \neq b$ and $a = b$

$$\sigma_{kl}^2 = E[(\sum_{a=b}^{N} W_{ka} W_{la} s_a^2(t)) + (\sum_{a \neq b}^{N^2-N} W_{ka} W_{lb} s_a(t) s_b(t))] - \mu^2$$

$$\sigma_{kl}^2 = E[(\sum_{a=b}^{N} W_{ka} W_{la} s_a^2(t))] + E[(\sum_{a \neq b}^{N^2-N} W_{ka} W_{lb} s_a(t) s_b(t))] - \mu^2$$

$$\sigma_{kl}^2 = \sum_{a=b}^{N} E[W_{ka} W_{la} s_a^2(t)] + \sum_{a \neq b}^{N^2-N} E[W_{ka} W_{lb} s_a(t) s_b(t)] - \mu^2$$

$$\sigma_{kl}^2 = \sum_{a=b}^{N} E[W_{ka}W_{la}s_a^2(t)] + \sum_{a\neq b}^{N^2-N} E[W_{ka}W_{lb}s_a(t)s_b(t)] - \mu^2$$

Since spiking activities and the strength of connections independent :

$$\sigma_{kl}^2 = \sum_{a=b}^{N} E[W_{ka}W_{la}]E[s_a^2(t)] + \sum_{a\neq b}^{N^2-N} E[W_{ka}W_{lb}]E[s_a(t)s_b(t)] - \mu^2$$

$$\Rightarrow E[(s_i(t)-m)^2] = v = E[s_i^2(t)] - (E[s_i(t)])^2 = E[s_i^2(t)] - m^2$$

$$\Rightarrow E[s_i^2(t)] = v + m^2$$

We know that spiking activities are uncorrelated :

$$\Rightarrow E[s_a(t)s_b(t)] = E[s_a(t)]E[s_b(t)] = m^2$$

$$\sigma_{kl}^2 = (v+m^2)\sum_{a=b}^{N} E[W_{ka}W_{la}] + m^2 \sum_{a\neq b}^{N^2-N} E[W_{ka}W_{lb}] - \mu^2$$

$$\sigma_{kl}^2 = (v+m^2)\sum_{a=b}^{N} E[W_{ka}W_{la}] + m^2 \sum_{a\neq b}^{N^2-N} E[W_{ka}W_{lb}] - \mu^2$$

Using Dale's Law, we know that the neuron makes either entirely excitatory connections, or entirely inhibitory connections to its targets.Therefore,if we know the neuron "a" excitatory or inhibitory connections, we can assure that the neuron "a" will have same value for all targets.

$$\sigma_{kl}^2 = (v+m^2)\sum_{a=b}^{N}((\epsilon\beta J)(\epsilon(1)J) + (\epsilon(1-\beta)J)(\epsilon(1)J)) + m^2 \sum_{a\neq b}^{N^2-N} E[W_{ka}W_{lb}] - \mu^2$$

$$\sigma_{kl}^2 = (v+m^2)\sum_{a=b}^{N}(\epsilon^2 J^2) + m^2 \sum_{a\neq b}^{N^2-N} E[W_{ka}W_{lb}] - \mu^2$$

$$\sigma_{kl}^2 = (v+m^2)\sum_{a=b}^{N}(\epsilon^2 J^2) + m^2 \sum_{a\neq b}^{N^2-N} E[W_{ka}W_{lb}] - \mu^2$$

$$\sigma_{kl}^2 = (v+m^2)\sum_{a=b}^{N}(\epsilon^2 J^2) + m^2 \sum_{a\neq b}^{N^2-N} (\epsilon^2\beta^2 J^2 + 2\epsilon^2 J^2\beta - 2\epsilon^2 J^2\beta^2 + \epsilon^2 J^2 + \epsilon^2\beta^2 J^2 - 2\epsilon^2 J^2\beta) - \mu^2$$

$$\sigma_{kl}^2 = (v+m^2)\sum_{a=b}^{N}(\epsilon^2 J^2) + m^2 \sum_{a\neq b}^{N^2-N}(\epsilon^2 J^2) - \mu^2$$

$$\sigma_{kl}^2 = (v+m^2)(\epsilon^2 J^2)N + m^2(\epsilon^2 J^2)(N^2-N) - \mu^2$$

$$\sigma_{kl}^2 = (v+m^2)(\epsilon^2 J^2)N + m^2(\epsilon^2 J^2)(N^2-N) - \mu^2$$

$$\sigma_{kl}^2 = \epsilon^2 J^2 N(v+m^2 N) - \mu^2$$

$$\Rightarrow \mu^2 = m^2 N^2 \epsilon^2 J^2 (2\beta-1)^2$$

$$\sigma_{kl}^2 = \epsilon^2 J^2 N(v+m^2 N) - m^2 N^2 \epsilon^2 J^2 (2\beta-1)^2$$

$$\sigma_{kl}^2 = \epsilon^2 J^2 N(v+m^2 N) - \epsilon^2 J^2 m^2 N^2 (2\beta-1)^2$$

$$\sigma_{kl}^2 = \epsilon^2 J^2 (vN + m^2 N^2 - m^2 N^2 (2\beta-1)^2)$$

$$\sigma_{kl} = \epsilon J \sqrt{(vN + m^2 N^2 - m^2 N^2 (2\beta-1)^2)}$$

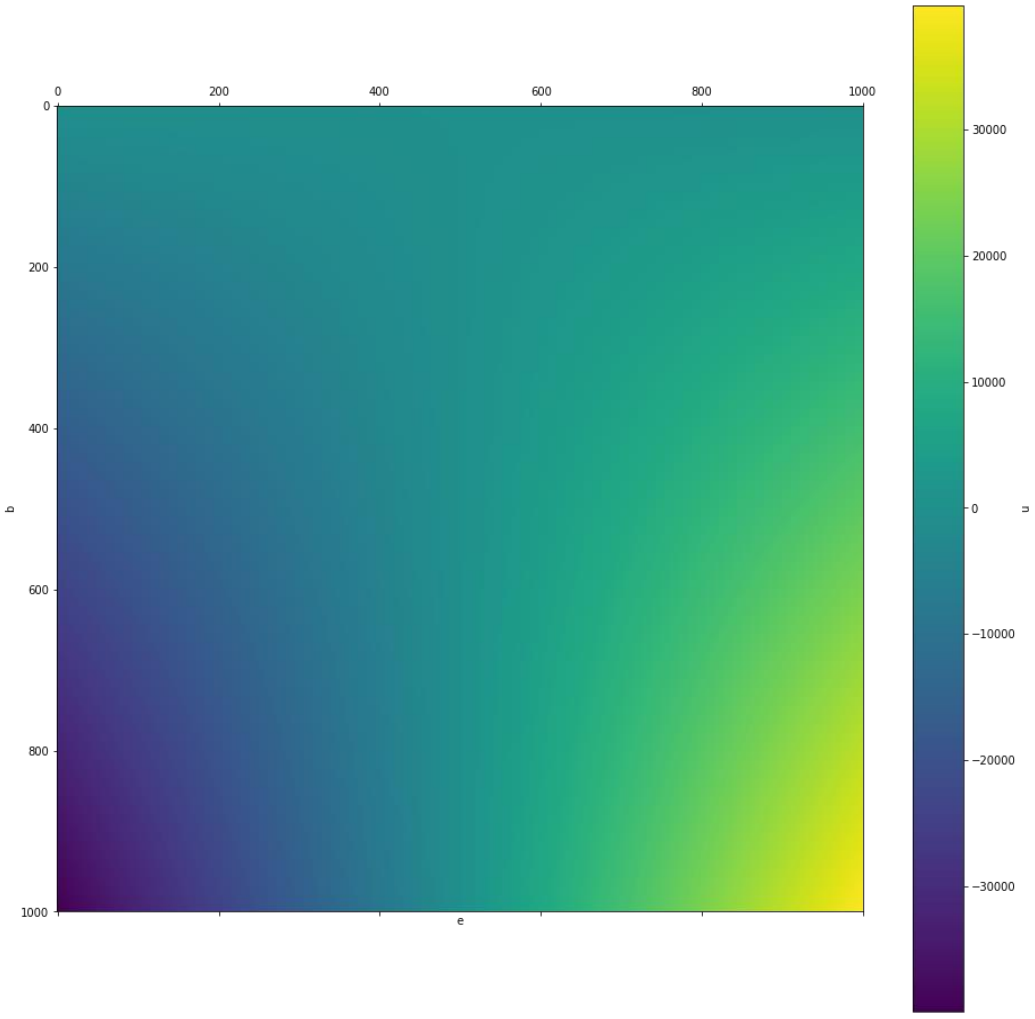


Figure : Question-2 Mean

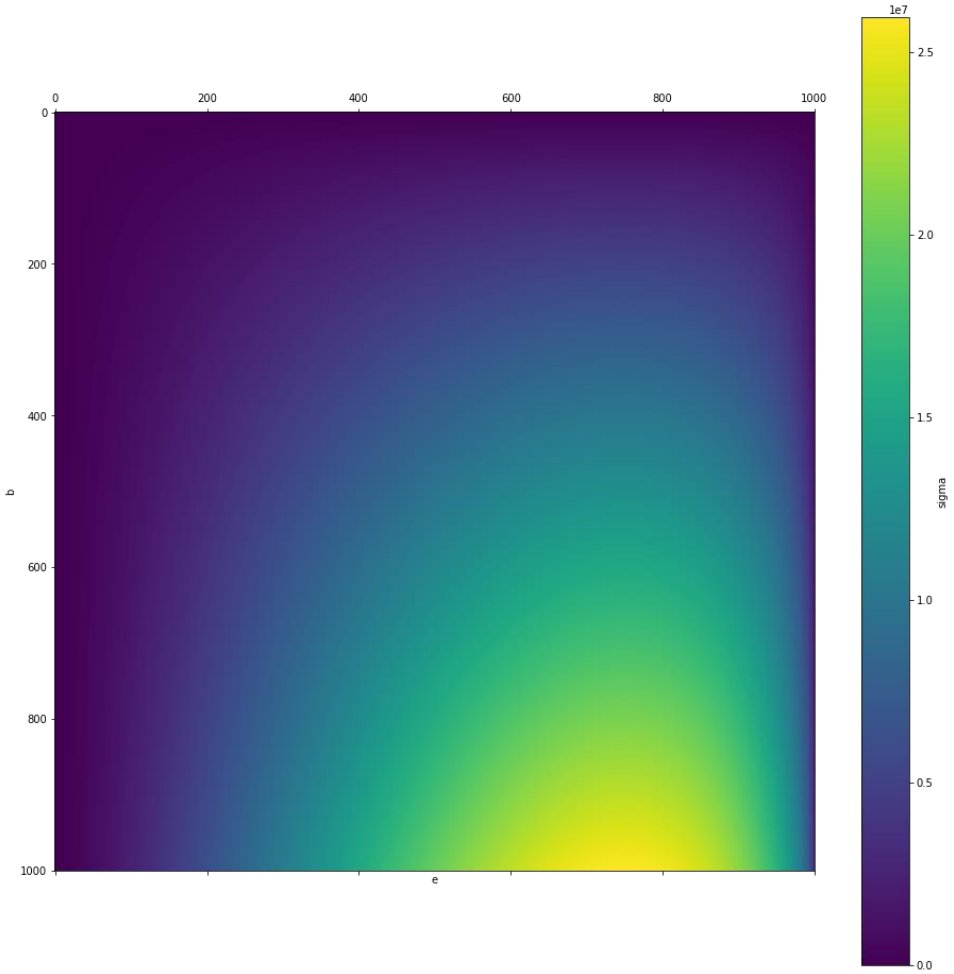


Figure: Question-2 Std

QUESTION 3

3.a

$(X_1 \vee \neg X_2) \oplus (\neg X_3 \vee \neg X_4) = (\neg (X_1 \vee \neg X_2) \wedge (\neg X_3 \vee \neg X_4)) \vee ((X_1 \vee \neg X_2) \wedge \neg(\neg X_3 \vee \neg X_4))$

$(X_1 \vee \neg X_2) \oplus (\neg X_3 \vee \neg X_4) = ((\neg X_2 \wedge X_3 \wedge X_4) \vee (\neg X_1 \wedge X_2 \wedge \neg X_3) \vee (X_1 \wedge X_3 \wedge X_4) \vee (\neg X_1 \wedge X_2 \wedge \neg X_4))$

φ = unipolar activation function(step)

W_0 = bias

D = does not affect result

General output for hidden unit → $\varphi(w_1X_1 + w_2X_2 + w_3X_3 + w_4X_4 + w_0)$

For 1.Hidden Unit Neuron our calculation is "**($\neg X_2 \wedge X_3 \wedge X_4$)**"

W_1 should be zero because we know that X_1 has no importance for above formula.

X_1	X_2	X_3	X_4	Expected Result	Inequality
D	0	0	0	0	$w_1 = 0, \varphi(w_0) = 0 \rightarrow w_0 < 0$
D	0	0	1	0	$w_1 = 0, \varphi(w_4 + w_0) = 0 \rightarrow w_4 + w_0 < 0$
D	0	1	0	0	$w_1 = 0, \varphi(w_3 + w_0) = 0 \rightarrow w_3 + w_0 < 0$
D	0	1	1	1	$w_1 = 0, \varphi(w_3 + w_4 + w_0) = 1 \rightarrow w_3 + w_4 + w_0 > 0$
D	1	0	0	0	$w_1 = 0, \varphi(w_2 + w_0) = 0 \rightarrow w_2 + w_0 < 0$
D	1	0	1	0	$w_1 = 0, \varphi(w_2 + w_4 + w_0) = 0 \rightarrow w_2 + w_4 + w_0 < 0$
D	1	1	0	0	$w_1 = 0, \varphi(w_2 + w_3 + w_0) = 0 \rightarrow w_2 + w_3 + w_0 < 0$
D	1	1	1	0	$w_1 = 0, \varphi(w_2 + w_3 + w_4 + w_0) = 0 \rightarrow w_2 + w_3 + w_4 + w_0 < 0$

For 2.Hidden Unit Neuron our calculation is "**($\neg X_1 \wedge X_2 \wedge \neg X_3$)**"

w_4 should be zero because we know that X_4 has no importance for above formula.

X_1	X_2	X_3	X_4	Expected Result	Inequality
0	0	0	D	0	$w_4 = 0, \varphi(w_0) = 0 \rightarrow w_0 < 0$
0	0	1	D	0	$w_4 = 0, \varphi(w_3 + w_0) = 0 \rightarrow w_3 + w_0 < 0$
0	1	0	D	1	$w_4 = 0, \varphi(w_2 + w_0) = 1 \rightarrow w_2 + w_0 > 0$
0	1	1	D	0	$w_4 = 0, \varphi(w_2 + w_3 + w_0) = 0 \rightarrow w_2 + w_3 + w_0 < 0$
1	0	0	D	0	$w_4 = 0, \varphi(w_1 + w_0) = 0 \rightarrow w_1 + w_0 < 0$
1	0	1	D	0	$w_4 = 0, \varphi(w_1 + w_3 + w_0) = 0 \rightarrow w_1 + w_3 + w_0 < 0$
1	1	0	D	0	$w_4 = 0, \varphi(w_1 + w_2 + w_0) = 0 \rightarrow w_1 + w_2 + w_0 < 0$
1	1	1	D	0	$w_4 = 0, \varphi(w_1 + w_2 + w_3 + w_0) = 0 \rightarrow w_1 + w_2 + w_3 + w_0 < 0$

For 3.Hidden Unit Neuron our calculation is "**($X_1 \wedge X_3 \wedge X_4$)**"

w_2 should be zero because we know that X_2 has no importance for above formula.

X_1	X_2	X_3	X_4	Expected Result	Inequality
0	D	0	0	0	$w_2 = 0, \varphi(w_0) = 0 \rightarrow w_0 < 0$
0	D	0	1	0	$w_2 = 0, \varphi(w_4 + w_0) = 0 \rightarrow w_4 + w_0 < 0$
0	D	1	0	0	$w_2 = 0, \varphi(w_3 + w_0) = 0 \rightarrow w_3 + w_0 < 0$
0	D	1	1	0	$w_2 = 0, \varphi(w_4 + w_3 + w_0) = 0 \rightarrow w_4 + w_3 + w_0 < 0$
1	D	0	0	0	$w_2 = 0, \varphi(w_1 + w_0) = 0 \rightarrow w_1 + w_0 < 0$
1	D	0	1	0	$w_2 = 0, \varphi(w_1 + w_4 + w_0) = 0 \rightarrow w_1 + w_4 + w_0 < 0$
1	D	1	0	0	$w_2 = 0, \varphi(w_1 + w_3 + w_0) = 0 \rightarrow w_1 + w_3 + w_0 < 0$
1	D	1	1	1	$w_2 = 0, \varphi(w_1 + w_3 + w_4 + w_0) = 1 \rightarrow w_1 + w_3 + w_4 + w_0 > 0$

For 4.Hidden Unit Neuron our calculation is "**$(\neg X_1 \wedge X_2 \wedge \neg X_4)$**"

w_3 should be zero because we know that X_3 has no importance for above formula.

X_1	X_2	X_3	X_4	Expected Result	Inequality
0	0	D	0	0	$w_3 = 0, \varphi(w_0) = 0 \rightarrow w_0 < 0$
0	0	D	1	0	$w_3 = 0, \varphi(w_4 + w_0) = 0 \rightarrow w_4 + w_0 < 0$
0	1	D	0	1	$w_3 = 0, \varphi(w_2 + w_0) = 1 \rightarrow w_2 + w_0 > 0$
0	1	D	1	0	$w_3 = 0, \varphi(w_2 + w_4 + w_0) = 0 \rightarrow w_2 + w_4 + w_0 < 0$
1	0	D	0	0	$w_3 = 0, \varphi(w_1 + w_0) = 0 \rightarrow w_1 + w_0 < 0$
1	0	D	1	0	$w_3 = 0, \varphi(w_1 + w_4 + w_0) = 0 \rightarrow w_1 + w_4 + w_0 < 0$
1	1	D	0	0	$w_3 = 0, \varphi(w_1 + w_2 + w_0) = 0 \rightarrow w_1 + w_2 + w_0 < 0$
1	1	D	1	0	$w_3 = 0, \varphi(w_1 + w_2 + w_4 + w_0) = 1 \rightarrow w_1 + w_2 + w_4 + w_0 < 0$

For Output Neuron our calculation is **"$H_1 + H_2 + H_3 + H_4$"**

H_x = Hidden layer Outputs

H_1	H_2	H_3	H_4	Expected Result	Inequality
0	0	0	0	0	$\varphi(w_0) = 0 \rightarrow w_0 < 0$
0	0	0	1	1	$\varphi(w_4 + w_0) = 1 \rightarrow w_4 + w_0 > 0$
0	0	1	0	1	$\varphi(w_3 + w_0) = 1 \rightarrow w_3 + w_0 > 0$
0	0	1	1	1	$\varphi(w_3 + w_4 + w_0) = 1 \rightarrow w_3 + w_4 + w_0 > 0$
0	1	0	0	1	$\varphi(w_2 + w_0) = 1 \rightarrow w_2 + w_0 > 0$
0	1	0	1	1	$\varphi(w_2 + w_4 + w_0) = 1 \rightarrow w_2 + w_4 + w_0 > 0$
0	1	1	0	1	$\varphi(w_2 + w_3 + w_0) = 1 \rightarrow w_3 + w_2 + w_0 > 0$
0	1	1	1	1	$\varphi(w_2 + w_3 + w_4 + w_0) = 1 \rightarrow w_2 + w_3 + w_4 + w_0 > 0$
1	0	0	0	1	$\varphi(w_1 + w_0) = 1 \rightarrow w_1 + w_0 > 0$
1	0	0	1	1	$\varphi(w_4 + w_1 + w_0) = 1 \rightarrow w_4 + w_1 + w_0 > 0$
1	0	1	0	1	$\varphi(w_1 + w_3 + w_0) = 1 \rightarrow w_1 + w_3 + w_0 > 0$
1	0	1	1	1	$\varphi(w_3 + w_4 + w_0) = 1 \rightarrow w_3 + w_4 + w_0 > 0$
1	1	0	0	1	$\varphi(w_2 + w_1 + w_0) = 1 \rightarrow w_2 + w_1 + w_0 > 0$
1	1	0	1	1	$\varphi(w_1 + w_2 + w_4 + w_0) = 1 \rightarrow w_1 + w_2 + w_4 + w_0 > 0$
1	1	1	0	1	$\varphi(w_1 + w_2 + w_3 + w_0) = 1 \rightarrow w_1 + w_2 + w_3 + w_0 > 0$
1	1	1	1	1	$\varphi(w_1 + w_2 + w_3 + w_4 + w_0) = 1 \rightarrow w_1 + w_2 + w_3 + w_4 + w_0 > 0$

3.b

According to set of inequalities that we can obtain weight vectors with biases that could achieve %100 accuracy

1.Hidden Unit Neuron weight vector with biases → $[w_1, w_2, w_3, w_4, w_0] \rightarrow [0, -1, 1, 1, -1.5]$

2.Hidden Unit Neuron weight vector with biases → [w_1, w_2, w_3, w_4, w_0] → [-1, 1, -1, 0, -0.5]

3.Hidden Unit Neuron weight vector with biases → [w_1, w_2, w_3, w_4, w_0] → [1, 0 , 1, 1, -2.5]

4.Hidden Unit Neuron weight vector with biases → [w_1, w_2, w_3, w_4, w_0] → [-1, 1 , 0, -1, -0.5]

Output Neuron weight vector with biases → [w_1, w_2, w_3, w_4, w_0] → [1, 1 , 1, 1, -0.5]

3.c

According to weight vectors of 3.b, small fluctuations like like std 0.1 has 100% accuracy. However, the model that we created will not work robustly(work with ~90%) under some extreme noise situations.

Updated :

1.Hidden Unit Neuron weight vector with biases → [w_1, w_2, w_3, w_4, w_0] → [-0.0157257 , -1.01299569, 0.98026133, 0.98344734, -1.53866706]

2.Hidden Unit Neuron weight vector with biases → [w_1, w_2, w_3, w_4, w_0] → [-1.01352933, 0.9832329 , -1.01528407, -0.01811416, -0.53777454]

3.Hidden Unit Neuron weight vector with biases → [w_1, w_2, w_3, w_4, w_0] → [0.9810651 , -0.0169859 , 0.98095781, 0.98318884, -2.53547764]

4.Hidden Unit Neuron weight vector with biases → [w_1, w_2, w_3, w_4, w_0] →[-1.01377487, 0.98346286, -0.02154296, -1.01240249, -0.53902452]

Output Neuron weight vector with biases → [w_1, w_2, w_3, w_4, w_0] → [0.86936658, 0.86936658, 0.86936658, 0.86936658, -0.71394658]

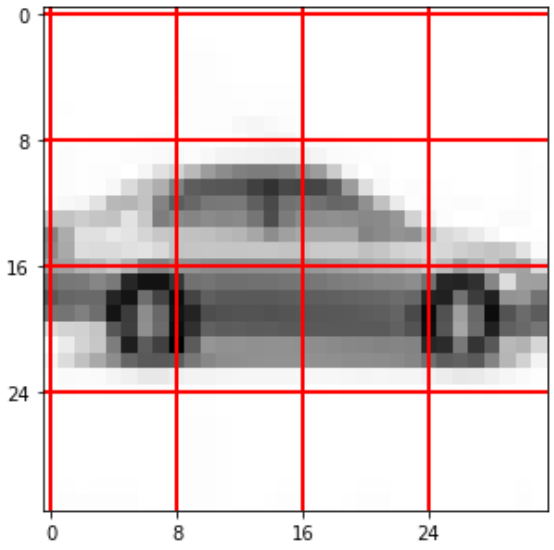
However, updated version of model(got with few epoch training of 3.d conditions over initial weights, you can see my implementation in .py file under “q3_network()” function) can achieve at least 1-5% accuracy improvement over noisy conditions like indicate in question 3.d.

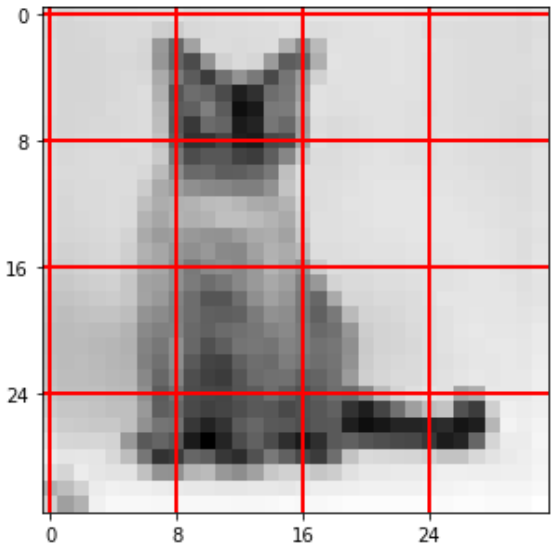
3.d

Updated version of the model can achieve 100% accuracy over data given 3.a and 3.c, also makes improvement 1-5% accuracy on data 3.d . However, I cannot guarantee that this network will always work in more noisy condition in this accuracy.

QUESTION 4

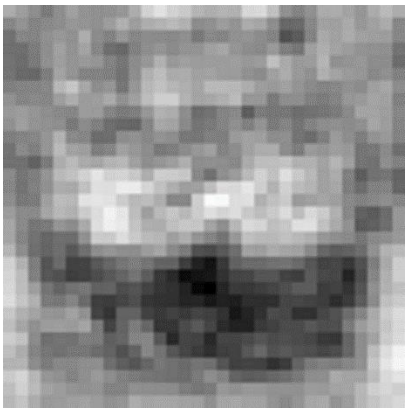
4.a





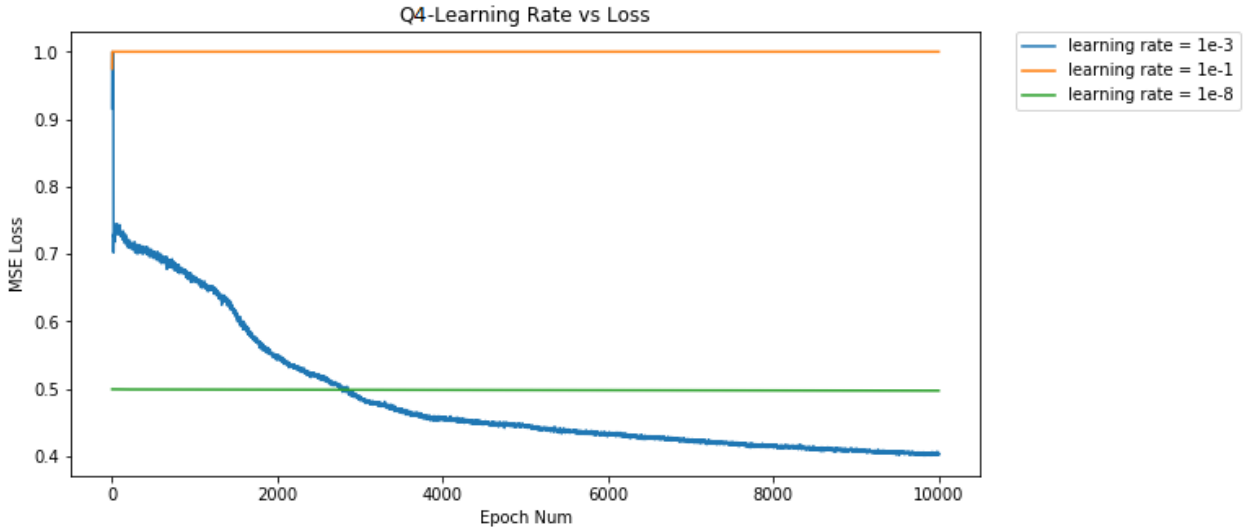
As we can see car and cat images have discriminative visual features like car's tires , cat's ears, cat's tail, car's chassis that have different borders and regions. Since two classes are relatively uncorrelated due to its visual features. Simple one layer network could classify correctly with great percentage.

4.b



It is a weight vector image(has accuracy ~77%) looks like cat with has ears, eyes. In middle of the image has more higher values for weight, below center, the image has more lower values for weights.

4.c



The optimal learning rate for my network was “0.001” that as we can see achieve the lowest MSE Loss over training data. However, the bigger learning rate like “0.1”, the model oscillate between values and could not find local minima due to bigger step, thus oscillations occurs are MSE Loss “1.0”. In smaller learning rate like “1e-8” again stuck around MSE Loss “0.5” because learning rate was so small that disable the network learn anything or gradient step was so small that never get closer to the local minima. Additionally, I train my model with 100 batch size to accelerate the process and generalize better using average of results.

4.d

Precision: for all instances classified positive, what percent was correct.

Recall : for all instances that were actually positive, what percent was classified correctly.

F1-score : is a weighted harmonic mean of precision and recall.

Support : the number of data for test.

I believe that the best tool to measure model performance looking “f1-score” that actually good metric that give mean of two metric

	precision	recall	f1-score	support
cat	0.00	0.00	0.00	500
car	0.50	1.00	0.67	500
avg / total	0.25	0.50	0.33	1000

Figure: Model Performance for learning rate 1e-1 →0.33

	precision	recall	f1-score	support
cat	0.76	0.79	0.77	500
car	0.78	0.75	0.76	500
avg / total	0.77	0.77	0.77	1000

Figure: Model Performance for learning rate 1e-3 → 0.77

	precision	recall	f1-score	support
cat	0.42	0.41	0.42	500
car	0.43	0.45	0.44	500
avg / total	0.43	0.43	0.43	1000

Figure: Model Performance for learning rate 1e-3 → 0.43

As an observation, the random selection has 50% accuracy on binary classes. The lower and higher learning rate model performance is the less than random selection. However, our optimal learning rate performance has higher performance over random selection so that we can say that the model learn some discriminative features of two classes.

QUESTION 5

5.a

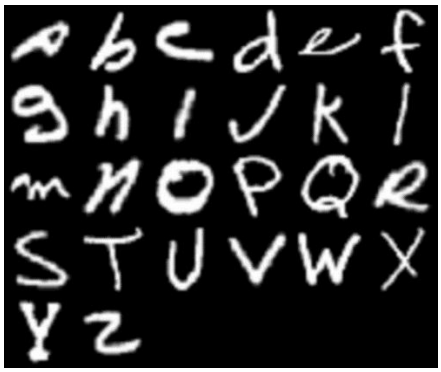


Figure : Sample images from each class

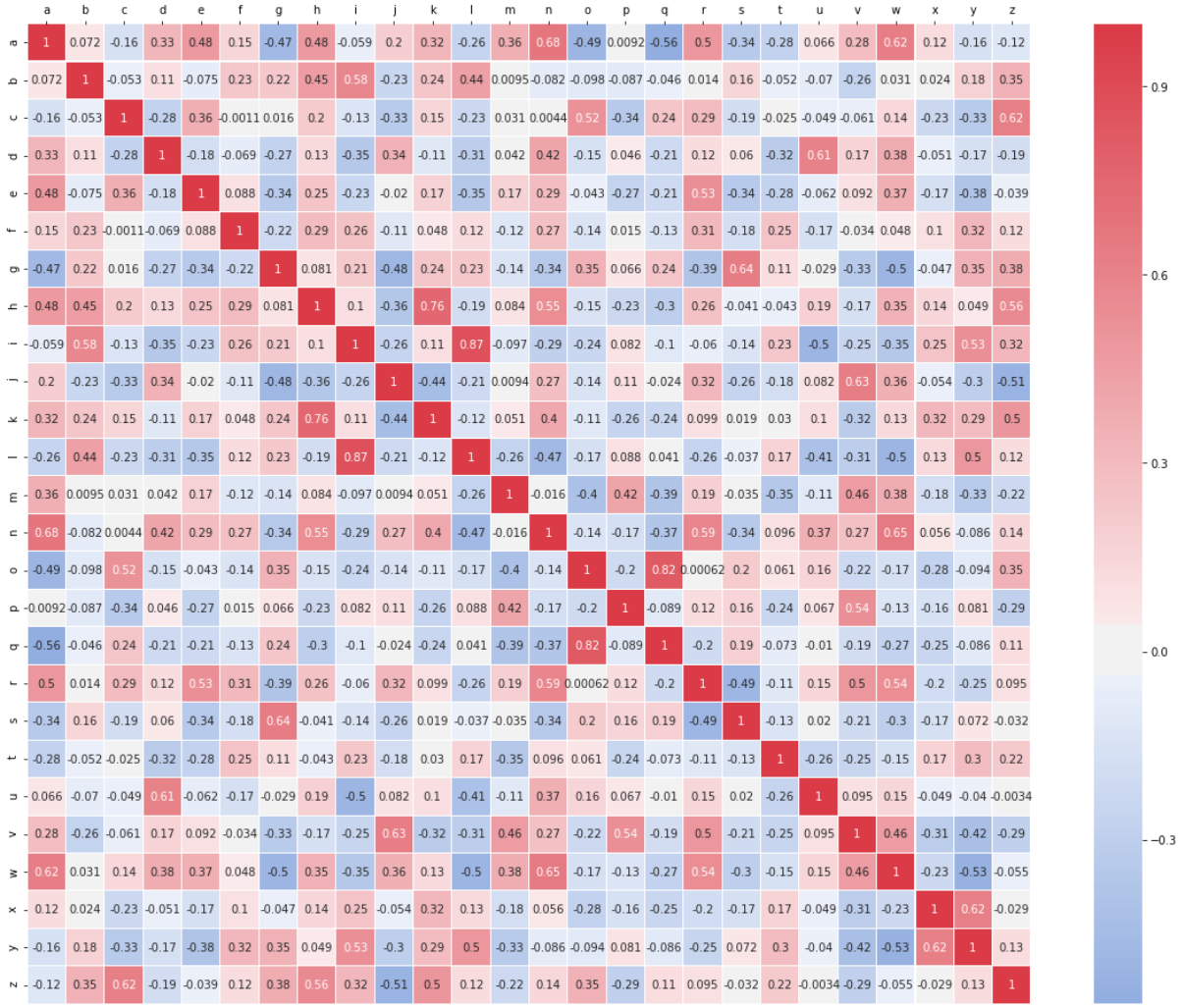
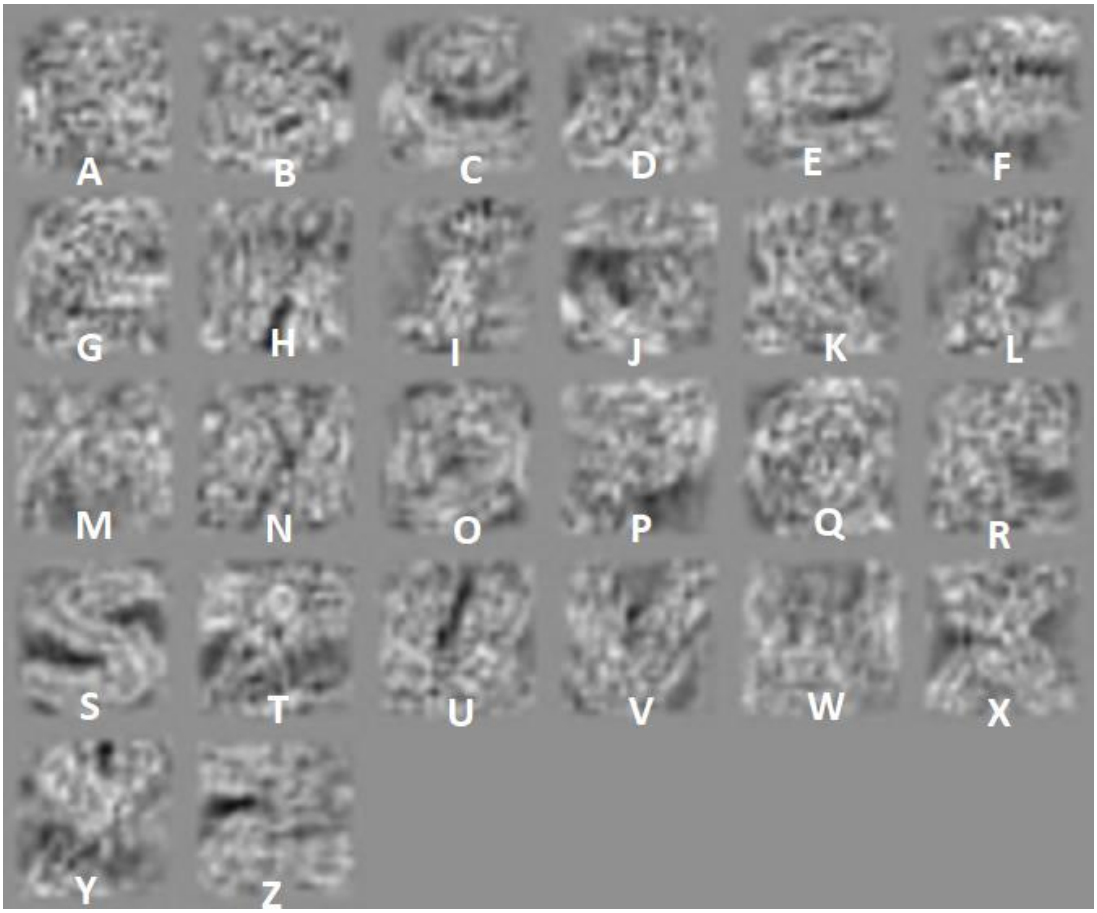


Figure: Correlation coefficient matrix over sample images

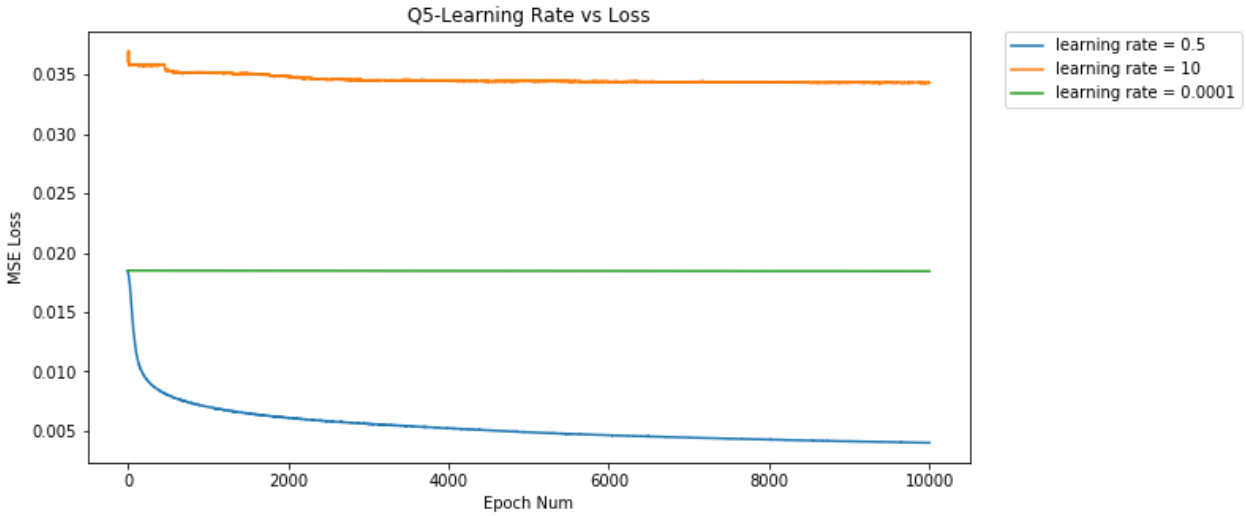
About within class variability, the data has lowercase and uppercase letters and we could observe that “a, b, d, e, f,g,h,l,n,q,r,t” has different shapes for lowercase and uppercase form, thus we could say that within class variability will be more than other letters. Therefore, we could forecast that the model will classify these images more wrongly. About across-class variability, we could observe from correlation coefficient matrix that “q and o”, “l and I”, “k and h” ,”n and a” , “y and x”, “v and u” has some similar structures for letters, thus we could say that these pairs will classify more wrongly from other pairs of letters, since correlation between classes of these pairs much higher than others.

5.b



We can see that letters like “A,B, D, G, R,K,N” has more intra-class variability therefore the weight images does not represent only one structure, therefore, as a human, I cannot say that these images belongs to the the letters I write above. However, other weight images could be understand as letters that they represents and this is because these letters has lower intra-class variability.

5.c



The optimal learning rate for my network was “0.5” that as we can see achieve the lowest MSE Loss over training data. However, the bigger learning rate like “10”, the model oscillate between values and could not find local minima due to bigger step, thus oscillations occurs are MSE Loss “0.035”. In smaller learning rate like “1e-4” again stuck around MSE Loss “0.020” because learning rate was so small that disable the network learn anything or gradient step was so small that never get closer to the local minima. Additionally, I train my model with 100 batch size to accelerate the process and generalize better using average of results.

5.d

	precision	recall	f1-score	support
a	0.07	0.06	0.07	50
b	0.19	0.08	0.11	50
c	0.25	0.40	0.31	50
d	0.03	0.02	0.02	50
e	0.02	0.02	0.02	50
f	0.20	0.20	0.20	50
g	0.07	0.06	0.07	50
h	0.06	0.08	0.07	50
i	0.00	0.00	0.00	50
j	0.03	0.06	0.04	50
k	0.17	0.20	0.19	50
l	0.00	0.00	0.00	50
m	0.25	0.28	0.26	50
n	0.08	0.12	0.09	50
o	0.05	0.04	0.04	50
p	0.05	0.06	0.06	50
q	0.04	0.04	0.04	50
r	0.18	0.22	0.20	50
s	0.02	0.02	0.02	50
t	0.00	0.00	0.00	50
u	0.05	0.06	0.06	50
v	0.06	0.04	0.05	50
w	0.22	0.26	0.24	50
x	0.00	0.00	0.00	50
y	0.20	0.18	0.19	50
z	0.21	0.24	0.22	50
avg / total	0.10	0.11	0.10	1300

Figure: Model Performance for learning rate 1e-4 →0.10

	precision	recall	f1-score	support
a	0.00	0.00	0.00	50
b	0.00	0.00	0.00	50
c	0.00	0.00	0.00	50
d	0.00	0.00	0.00	50
e	0.00	0.00	0.00	50
f	0.00	0.00	0.00	50
g	0.00	0.00	0.00	50
h	0.00	0.00	0.00	50
i	0.00	0.00	0.00	50
j	0.00	0.00	0.00	50
k	0.00	0.00	0.00	50
l	0.00	0.00	0.00	50
m	0.00	0.00	0.00	50
n	0.00	0.00	0.00	50
o	0.11	0.96	0.19	50
p	0.00	0.00	0.00	50
q	0.00	0.00	0.00	50
r	0.00	0.00	0.00	50
s	0.00	0.00	0.00	50
t	0.00	0.00	0.00	50
u	0.00	0.00	0.00	50
v	0.10	0.92	0.19	50
w	0.00	0.00	0.00	50
x	0.10	0.80	0.18	50
y	0.00	0.00	0.00	50
z	0.00	0.00	0.00	50
avg / total	0.01	0.10	0.02	1300

Figure: Model Performance for learning rate 10 →0.02

	precision	recall	f1-score	support
a	0.49	0.54	0.51	50
b	0.67	0.74	0.70	50
c	0.78	0.72	0.75	50
d	0.52	0.48	0.50	50
e	0.74	0.52	0.61	50
f	0.75	0.72	0.73	50
g	0.37	0.38	0.38	50
h	0.62	0.48	0.54	50
i	0.65	0.52	0.58	50
j	0.64	0.76	0.70	50
k	0.59	0.68	0.63	50
l	0.44	0.56	0.50	50
m	0.85	0.80	0.82	50
n	0.56	0.56	0.56	50
o	0.84	0.92	0.88	50
p	0.75	0.86	0.80	50
q	0.57	0.62	0.60	50
r	0.68	0.72	0.70	50
s	0.85	0.66	0.74	50
t	0.57	0.72	0.64	50
u	0.67	0.76	0.71	50
v	0.74	0.74	0.74	50
w	0.84	0.84	0.84	50
x	0.76	0.64	0.70	50
y	0.77	0.68	0.72	50
z	0.86	0.76	0.81	50
avg / total	0.68	0.67	0.67	1300

Figure: Model Performance for learning rate 0.5 →0.67