Yue Zhou
29 March 2020

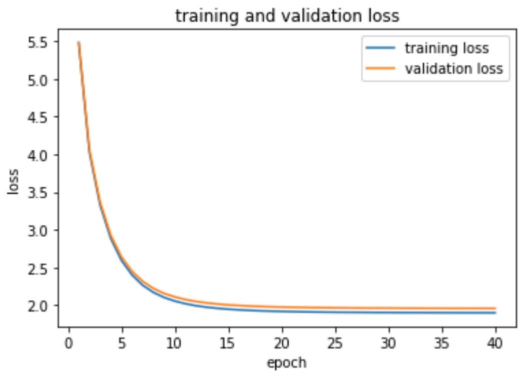
DD2424 Report assignment 1-bonus

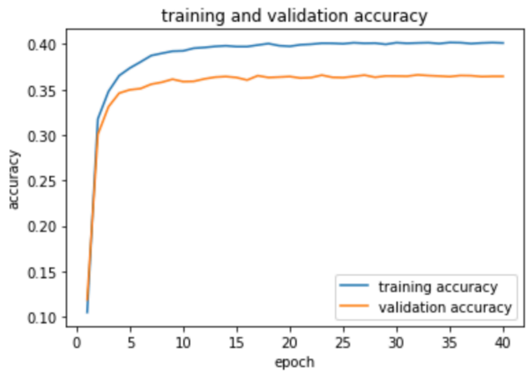
Yue Zhou

1. Optimize the performance of the network

1.1 Learning rate decay

Setting a decay rate to 0.9, the learning rate is smaller after each epoch. Comparing the performance between -lambda=1, n_epochs=40, n_batch=100, eta=0.001, decay_factor=1 and -lambda=1, n_epochs=40, n_batch=100, eta=0.001, decay_factor=0.95.



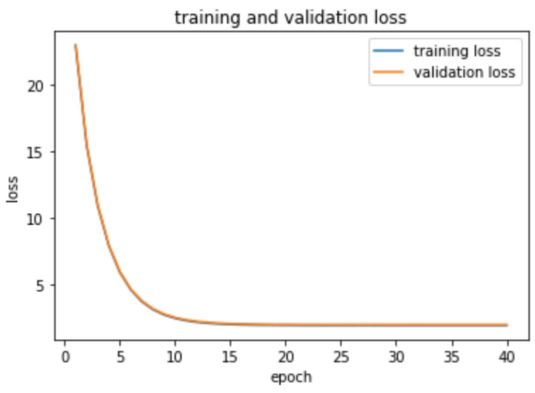


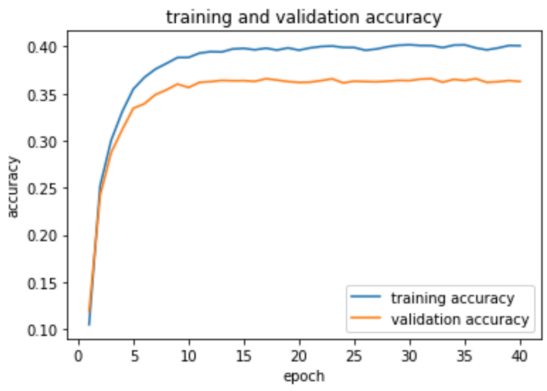
Decay factor	Test cost	Test accuracy	Validation cost	Validation accuracy	Train cost	Train accuracy
1	1.94	37.84%	1.96	35.28%	1.90	40 %
0.95	1.94	37.61%	1.96	36.45%	1.94	40.12%

1.2 Xavier initialisation

The aim of weight initialisation is to prevent layer activation outputs from exploding or vanishing during the course of a forward pass through a deep neural network. If either occurs, loss gradients will either be too large or too small to flow backwards beneficially, and the network will take longer to converge, if it is even able to do so at all.

We define the std of w and b: std=$\sqrt{\frac{2}{n_1}}$, in this place, n_1 denotes the input number, which is dimension of X.

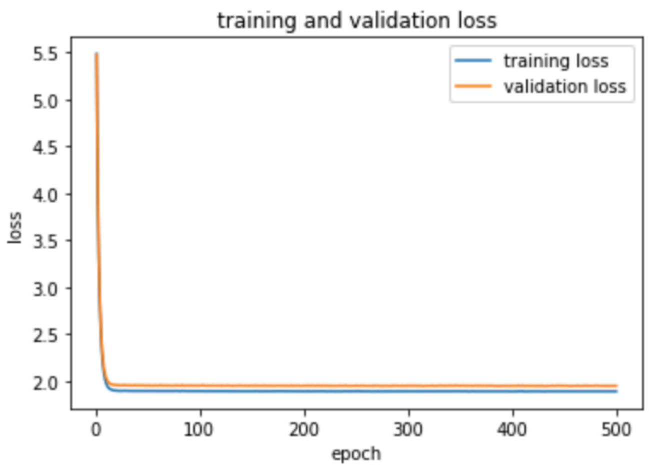


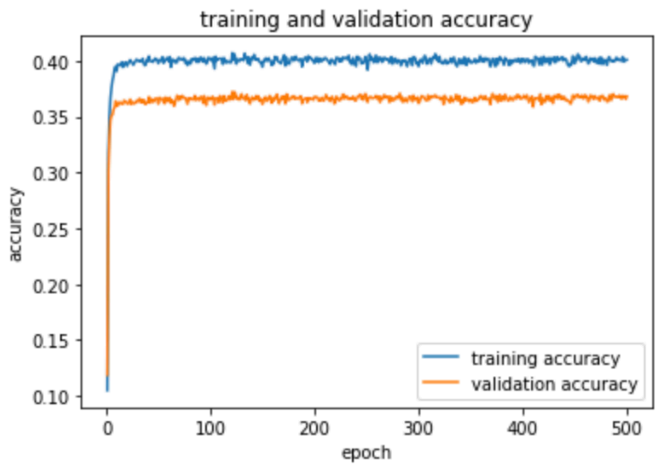


W var	Test cost	Test accuracy	Validation cost	Validation accuracy	Train cost	Train accuracy
0.01	1.94	37.84%	1.96	35.28%	1.90	40 %
$\sqrt{\frac{2}{ni}}$	1.94	37.84%	1.96	36.29%	1.90	40.08%

1.3 Train for a longer time

Train for a longer time and keep a record for the best model to make sure that I Don't overfit.



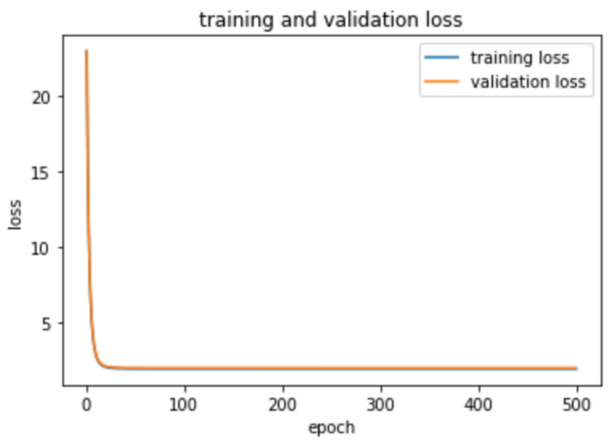


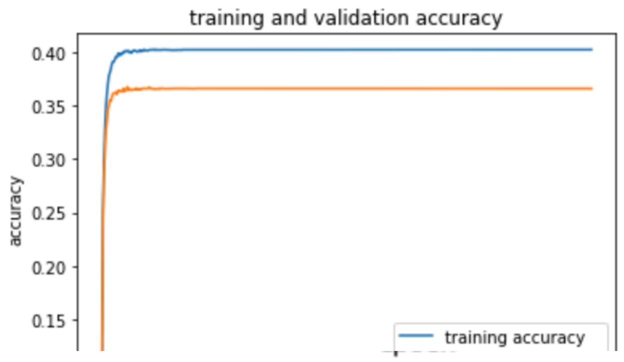
e_poch	**Test cost**	**Test accuracy**	**Validation cost**	**Validation accuracy**	**Train cost**	**Train accuracy**
40	1.94	37.84%	1.96	35.28%	1.90	40 %
500	1.93	37.95%	1.95	36.83%	1.89	40.1%

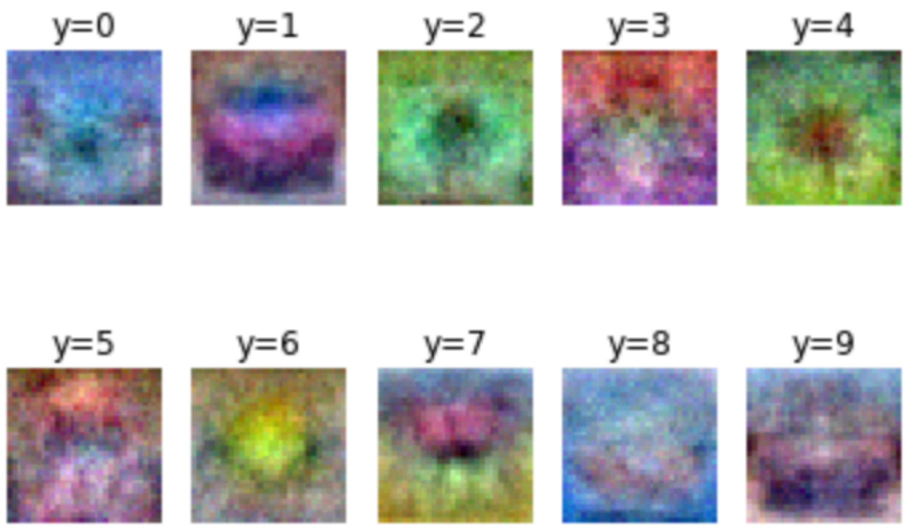
2.4 Conclusion

In my optimisation, training for a longer time can give the best improvement, while decay factor and Xavier can only give a slight improvement.The following is a network combined all the possible improvement.

-lambda=1, n_epochs=500, n_batch=100, eta=0.001, decay_factor=0.95 with Xavier initialisation.



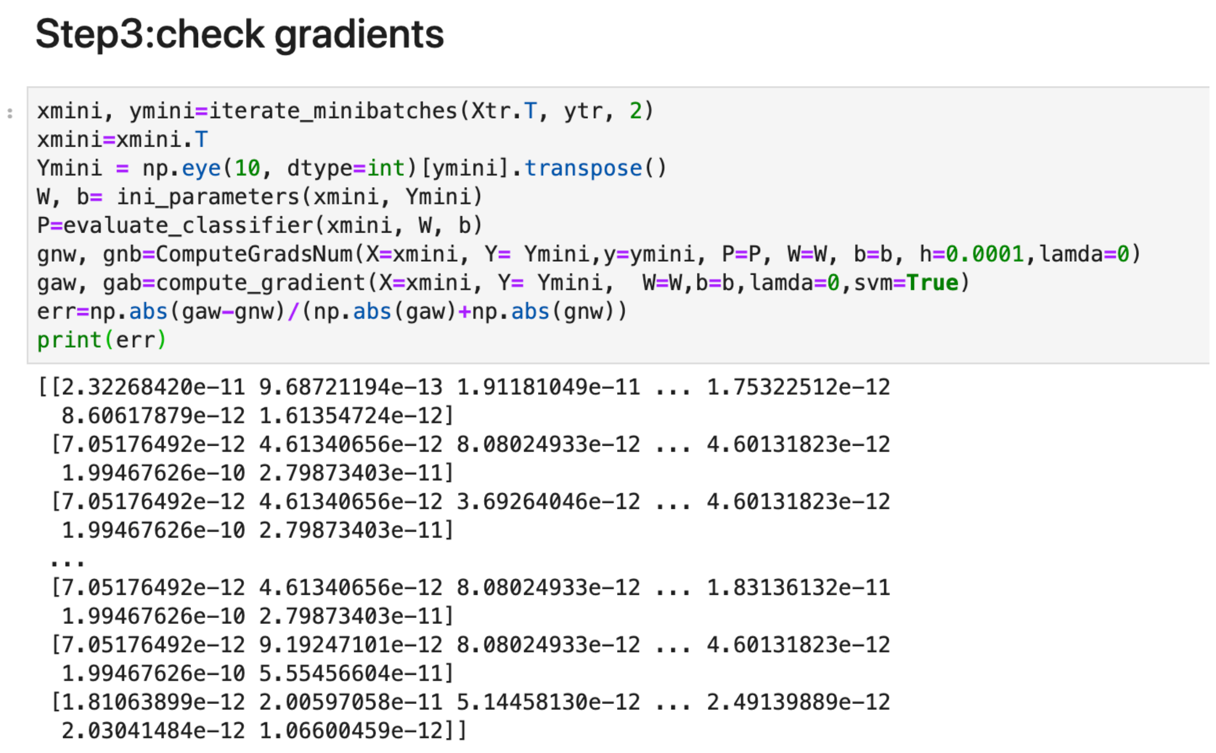




2. SVM multi-class loss

I implemented sum loss function and the result is followed.

The gradient check:

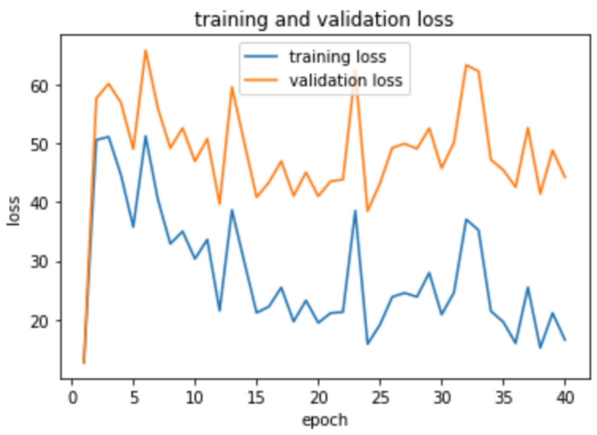


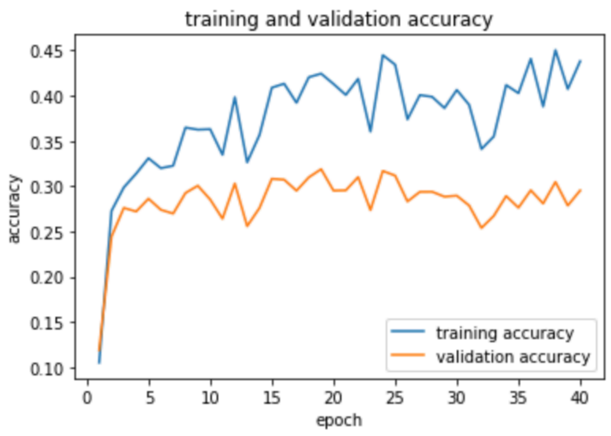
Step3:check gradients

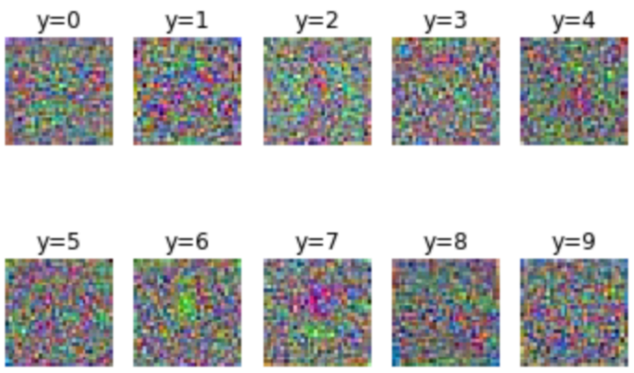
```
xmini, ymini=iterate_minibatches(Xtr.T, ytr, 2)
xmini=xmini.T
Ymini = np.eye(10, dtype=int)[ymini].transpose()
W, b= ini_parameters(xmini, Ymini)
P=evaluate_classifier(xmini, W, b)
gnw, gnb=ComputeGradsNum(X=xmini, Y= Ymini,y=ymini, P=P, W=W, b=b, h=0.0001,lamda=0)
gaw, gab=compute_gradient(X=xmini, Y= Ymini,  W=W,b=b,lamda=0,svm=True)
err=np.abs(gaw-gnw)/(np.abs(gaw)+np.abs(gnw))
print(err)
```

```
[[2.32268420e-11 9.68721194e-13 1.91181049e-11 ... 1.75322512e-12
  8.60617879e-12 1.61354724e-12]
 [7.05176492e-12 4.61340656e-12 8.08024933e-12 ... 4.60131823e-12
  1.99467626e-10 2.79873403e-11]
 [7.05176492e-12 4.61340656e-12 3.69264046e-12 ... 4.60131823e-12
  1.99467626e-10 2.79873403e-11]
 ...
 [7.05176492e-12 4.61340656e-12 8.08024933e-12 ... 1.83136132e-11
  1.99467626e-10 2.79873403e-11]
 [7.05176492e-12 9.19247101e-12 8.08024933e-12 ... 4.60131823e-12
  1.99467626e-10 5.55456604e-11]
 [1.81063899e-12 2.00597058e-11 5.14458130e-12 ... 2.49139889e-12
  2.03041484e-12 1.06600459e-12]]
```

-lambda=0, n_epochs=40, n_batch=100, eta=0.1, decay_factor=1, svm=True

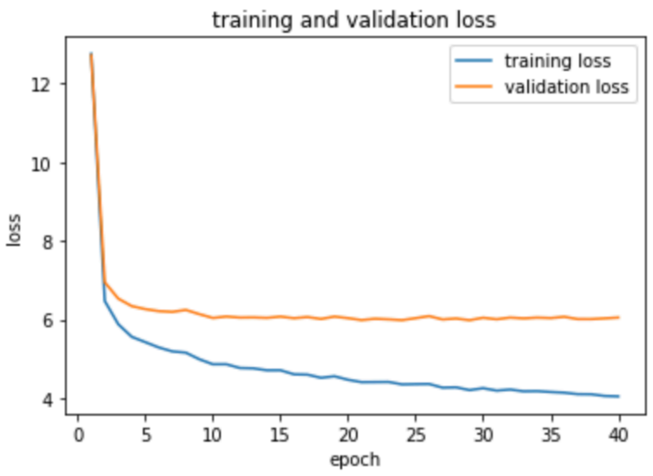


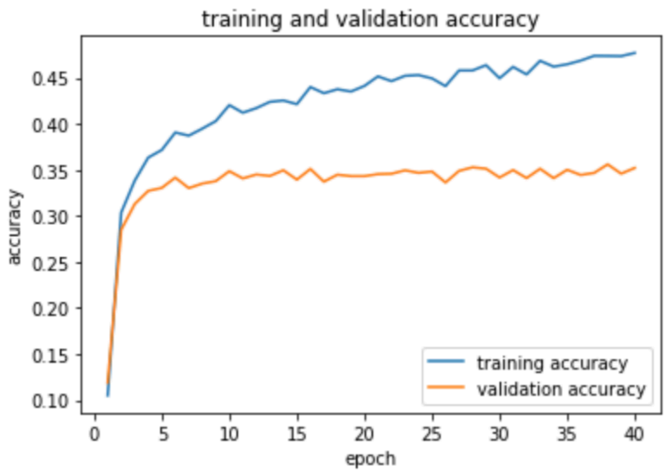


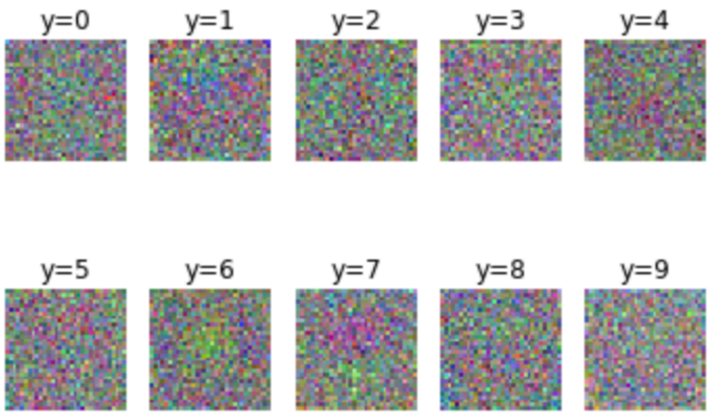


Test cost	Test accuracy	Validation cost	Validation accuracy	Train cost	Train accuracy
1.94	25.93%	63.77	29.528%	16.55	43.81 %

-lambda=0, n_epochs=40, n_batch=100, eta=0.001, decay_factor=1, svm=True

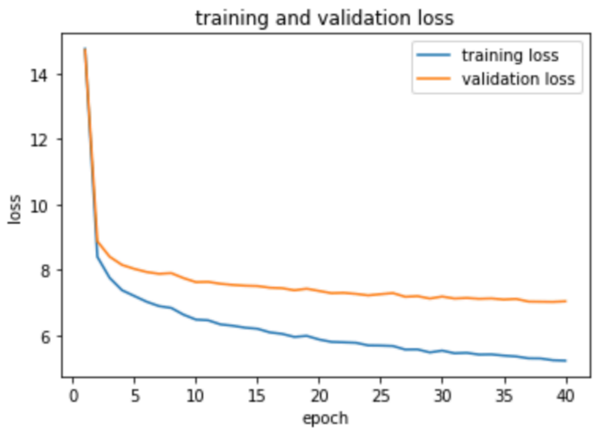


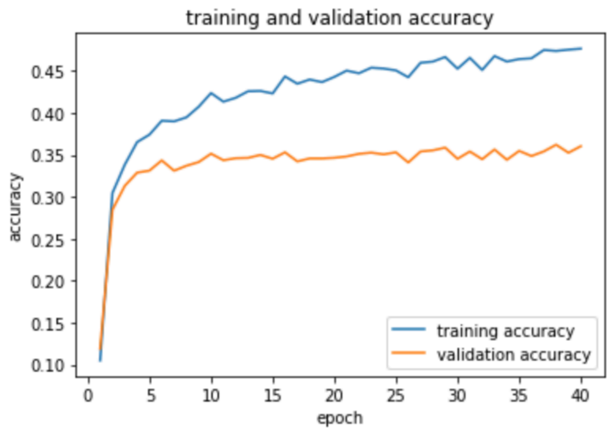


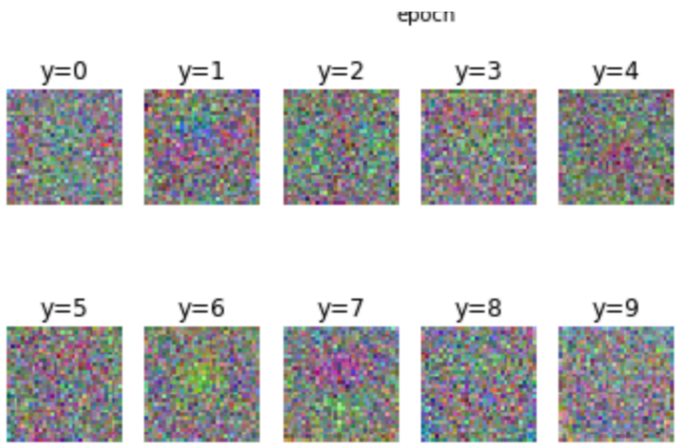


Test cost	Test accuracy	Validation cost	Validation accuracy	Train cost	Train accuracy
6.01	34.9%	6.05	35.22%	4.05	47.41 %

-lambda=0.1, n_epochs=40, n_batch=100, eta=0.001, decay_factor=1, svm=True

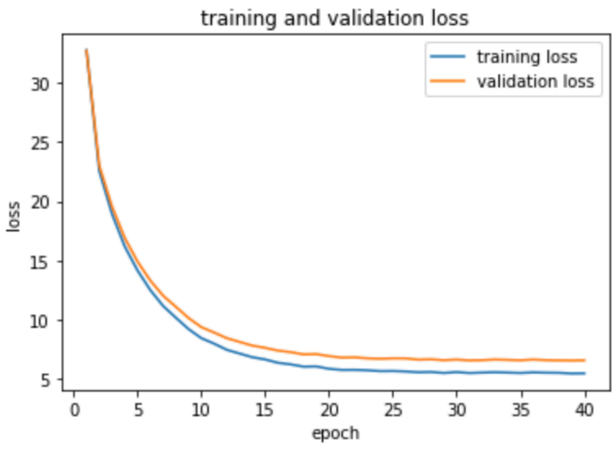


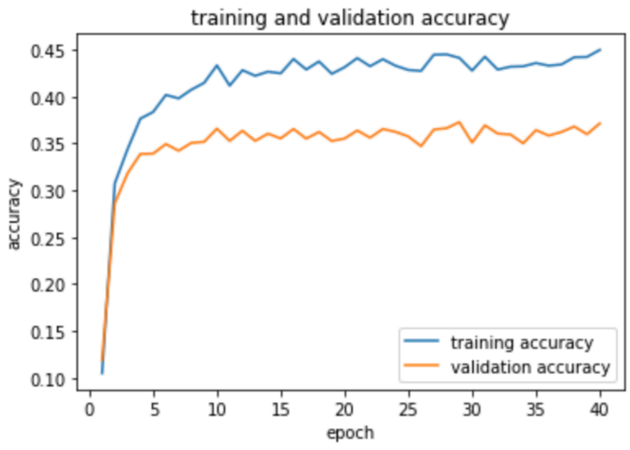


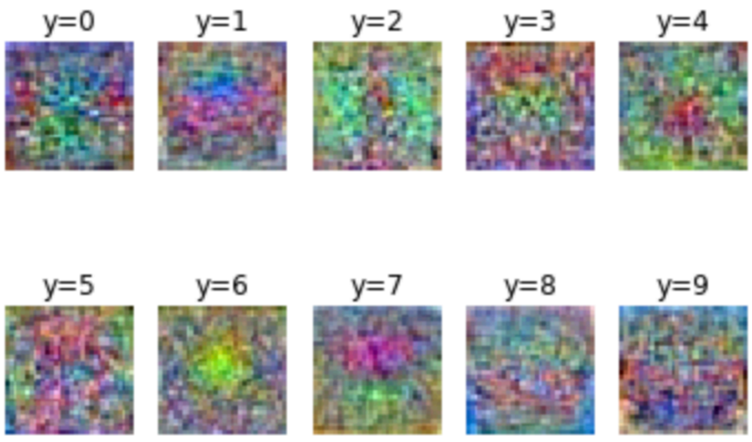


Test cost	Test accuracy	Validation cost	Validation accuracy	Train cost	Train accuracy
7.01	35.19%	7.03	36.06%	5.22	47.67 %

-lambda=0.1, n_epochs=40, n_batch=100, eta=0.001, decay_factor=1, svm=True







Test cost	**Test accuracy**	**Validation cost**	**Validation accuracy**	**Train cost**	**Train accuracy**
6.55	36.48%	6.56	37.13%	5.46	44.97 %

In general, Svm performes better than cross-entropy.