Homework 3 - Practical Report

Reza Bayat

December 2021

1 Question 2: Experimenting on the FashionMNIST dataset

1.1

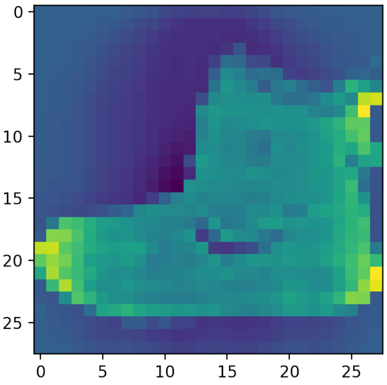


Figure 1: Original Image

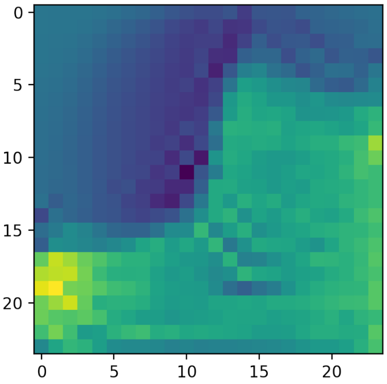


Figure 2: Output Features

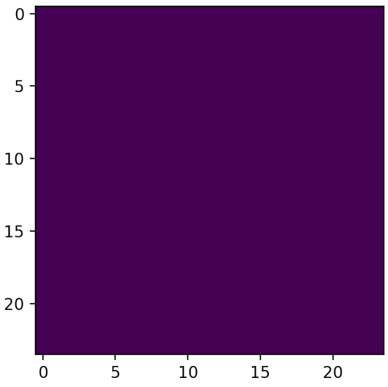


Figure 3: Shifted Absolute Difference

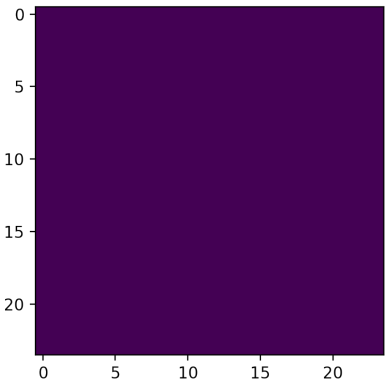


Figure 4: Rotated Absolute Difference

(a) Convolutional layers are equivariant to Shift and Rotation, since both above images are very dark, which means that absolute element-size differences between two images are close to zero.

(b) Based on the application some of equivariance are more important than others, for example, maybe, in self driving car application illumination is more important than rotation, since camera always look at the front but sun light for example changes during a day.

1.2

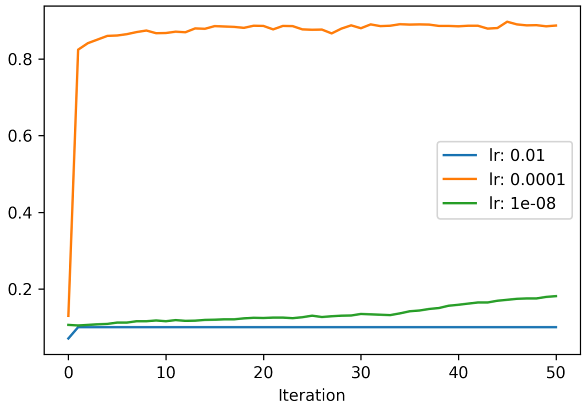


Figure 5: Without Normalization

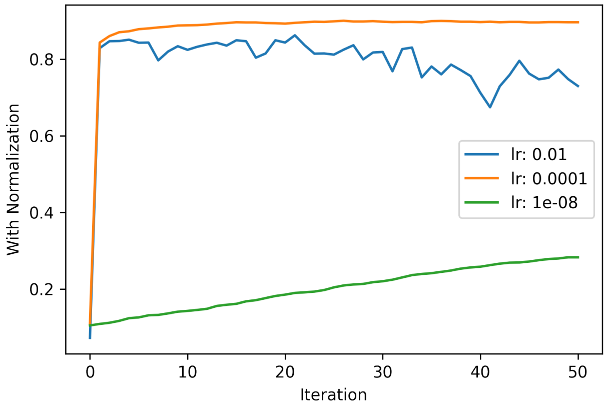


Figure 6: With Normalization

1.2.1 What is the effect of normalization?

The goal of normalization is to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values (this). The normalization step helps the model to converge better that a model which uses raw values.

1.2.2 Does it help convergence? Why / why not?

The normalization helps because it ensures (a) that there are both positive and negative values used as inputs for the next layer which makes learning more flexible and (b) that the network's learning regards all input features to a similar extent(this). It helps model to converge better because all pixels' values are bounded using the normalization method.

1.2.3 What are the effects of learning rates that are very small or very large?

By using a very large learning rage, model may not convergence to optimal minimum because of having a large step at each gradient updates. And using a very small learning rate model may converge very late or even can cause the process to get stuck.

1.3

1.3.1 a

Number of parameters is equal to sum of all parameters in all layers, so:
First parameters set: $28 * 28 * 256 = 200704$
Second parameters set: $256 * 256 = 65536$
Third parameters set: $256 * 10 = 2560$
All parameters = $200704 + 65536 + 2560 = 268800$

1.3.2 b

First layer's parameters: $28 * 28 * 64 = 50176$
N Middle layers' parameters = $n * 64 * 64$
Last layer's parameters: $64 * 10 = 610$
All layers' parameters: $50176 + 610 + n * 64 * 64$
This quantity should be equal to the previous one, so:

$$268800 = 50176 + 610 + n * 64 * 64$$

$$n \approx 54$$

We also could confirm it using empirical results by training multiple models.

i) The model is totaly overfit, which means it has a very high accuracy on the dataset but very low accuracy on the validation set.

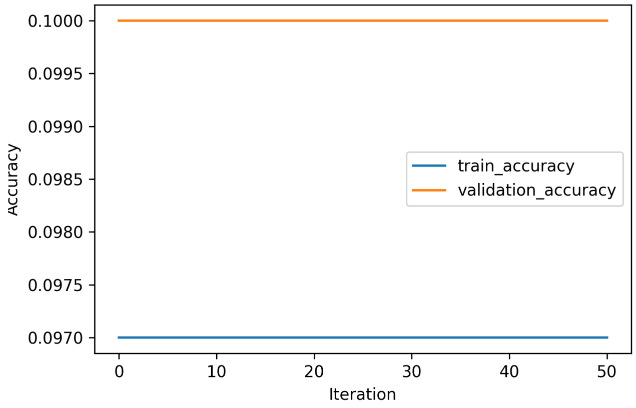


Figure 7: Train and Validation Accuracy

ii) First mode is the previous mode with 54 layers, and the second model is a simple MLP with two layers, (256 neurons).

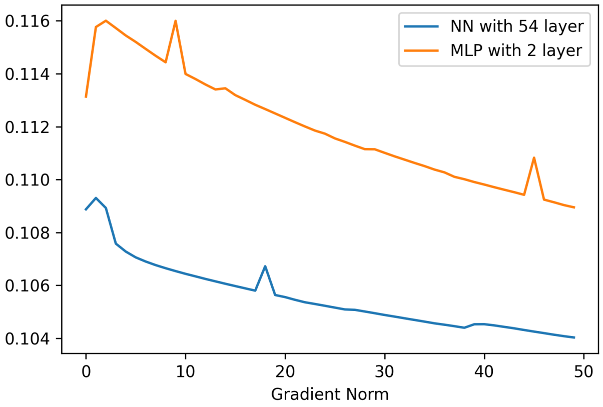


Figure 8: Gradient Norms

iii) Does the behavior of the gradient norm explain the performance?

Look at two following plots of accuracies and gradient norms of a NN model with different learning rates; When the gradient norm is stable and changes a little bit around zero, we can see that the model also has smooth curve validation accuracy, without significant changes from one epoch to the next epoch.

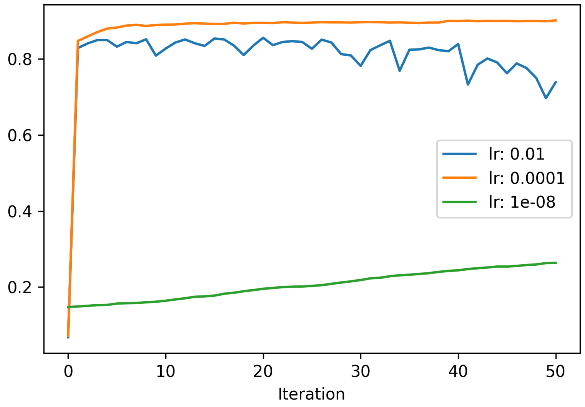


Figure 9: CNN Validation Accuracy

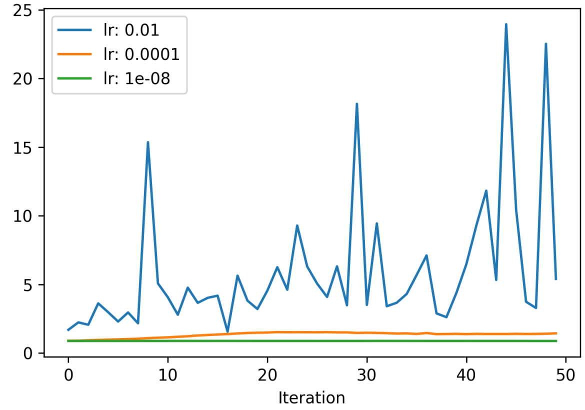
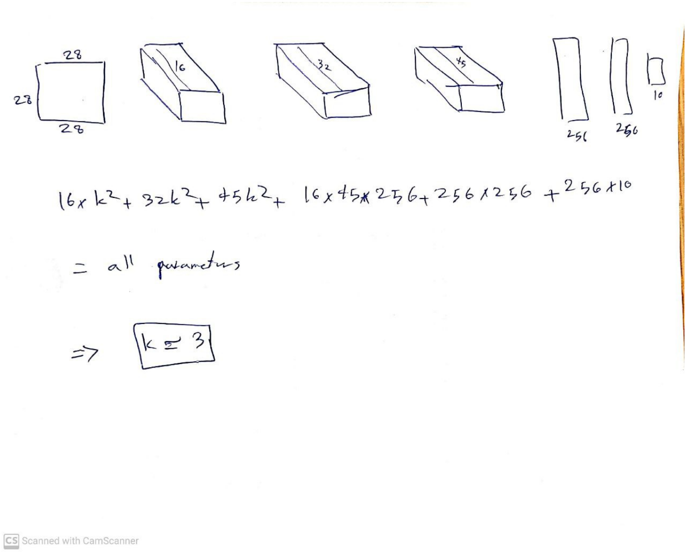


Figure 10: CNN Gradient Norms

1.4

From above, all parameters = 268800, and from empirical results on implementations of models and the following approach, k is 3:



$16 \times k^2 + 32k^2 + 45k^2 + 16 \times 45 \times 256 + 256 \times 256 + 256 \times 10$

$= \text{all parameters}$

$\Rightarrow \boxed{k \simeq 3}$

Figure 11: K of CNN

(a) CNN vs MLP Validation Accuracy

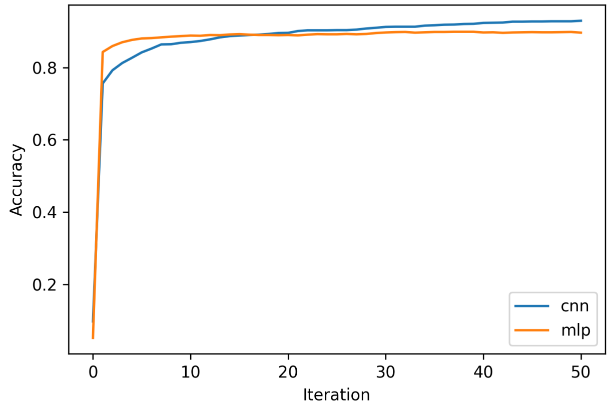
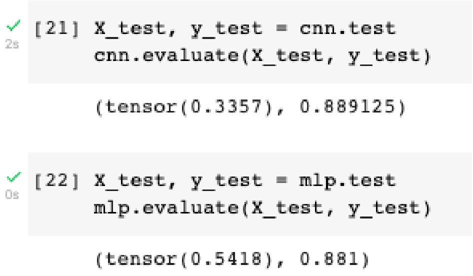


Figure 12: CNN vs MLP (Validation dataset)

(b) Which one does perform better? What can you say about generalization capabilities of the two architectures?

The CNN model has better performance on validation set, so generally this model is better than the simple MLP, however we should also look at its performance on the test set.

Use the following code we can look at the performance of the CNN and MLP on the test dataset, which the accuracy of the CNN is better that MLP, so the generalization of the CNN model is better than MLP.



```
[21] X_test, y_test = cnn.test
     cnn.evaluate(X_test, y_test)

     (tensor(0.3357), 0.889125)
```

```
[22] X_test, y_test = mlp.test
     mlp.evaluate(X_test, y_test)

     (tensor(0.5418), 0.881)
```

Figure 13: CNN vs MLP (Test dataset)

1.5

a) CNN with kernel size 1 is equal to a fully fully-connected layers since all element in each kernel is connected to all element in the previous layer, which means hidden units are equal to the number of filters because because k = 1.
Note: number of all parameters are 333578.

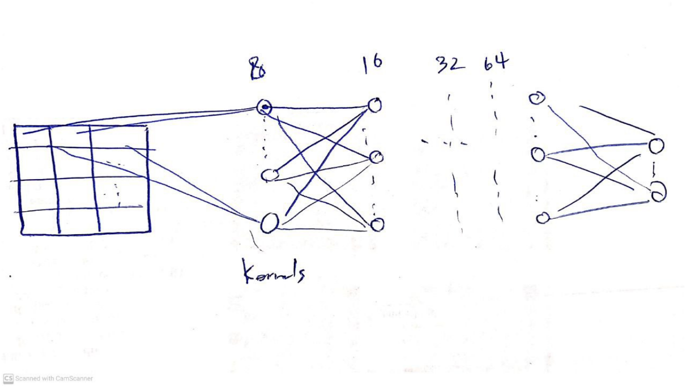


Figure 14: CNN with k equal 1

b) Number of params should be same so, it using empirical results $h \approx 52$

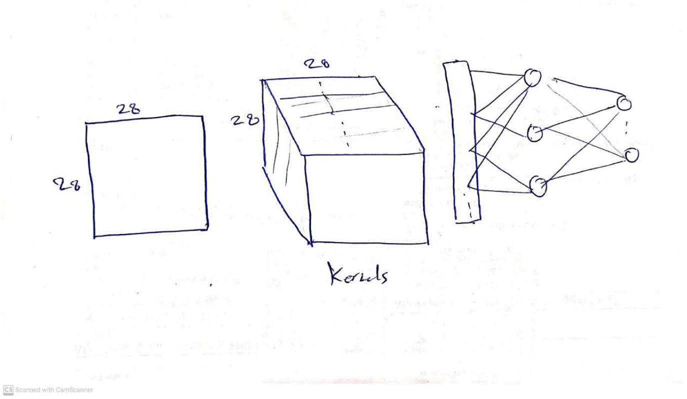


Figure 15: CNN with k equal 28

c)

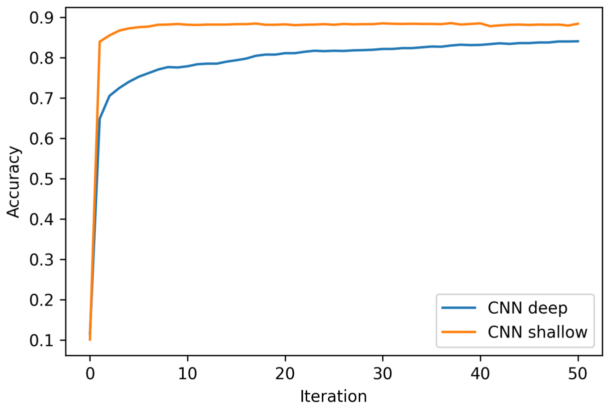


Figure 16: Validation Accuracy of Deep and Shallow CNN