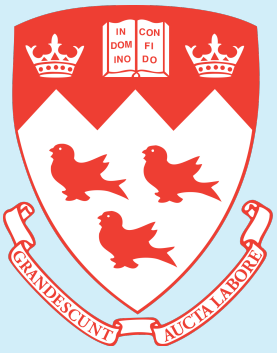


Applied Machine Learning

Assignment 3 Report
COMP 551

Authors: *Ling Fei Zhang, Brandon Ma, Giordano Di Marzio*

Instructor:	*Yue Li*
Department:	Computer Science
Date:	December 5, 2022

Contents

1 Abstract

In this assignment, we investigated the performance of multilayer perceptron on the Fashion-MNIST benchmark dataset consisting of 28 x 28 pixels and a 10-class classification. We created MLP models with different parameters. The following modified models were all trained over 15 epochs and a greedy approach was used to minimize the loss function. Zero to five hidden layers were tested, with one layer giving the best results. Adding layers slowed down the convergence of the model. Our default activation function, ReLU, proved to be the optimal one. Other functions that were considered but did not bring any significant improvement to the model were Leaky-ReLU and tanh. L2-regularization was another adjustment that was tested but did not increase the accuracy of the model. Lastly, Convolutional Neural Network (CNN) was applied and enhanced the model notably. The best parameters with CNN were a kernel size of 2x2, stride length of 1x1, max pooling window size of 2x2 and a dropout rate of 0.25 on the dense layers. After careful consideration, the best model for this dataset was the default one-layer MLP using ReLU activation functions. Using this, we were able to achieve an accuracy of approximately 88%, which is still relatively far away from the accuracy of CNN (90%). Overall, all models modified with the different parameters mentioned above performed well, scoring at least 85% in each case after 15 epochs.

2 Introduction

The main task of the assignment was to implement a multilayer perceptron from scratch in order to classify images with the highest accuracy. Additionally, we would have the opportunity to experiment with convolutional neural networks (CNNs). Unlike the previous assignment, the dataset did not require significant preprocessing given we only needed to normalize both training and testing datasets where the vectorized pixels of the images would act as the inputs of our classifier. In task 3.1, we implemented three types of MLPs : 0, 1 and 2 hidden layers where we also used a greedy approach to minimize the fluctuation of cross entropy. We saw that the 0 hidden layer yielded the smallest validation accuracy with approximately 84%, whereas the 1 hidden layer had the highest accuracy with 86%, and the 2 hidden layers had 85%. This highlighted the fact that nonlinearity allowed for more complex decisions to be made in the data, which coincided with the increase of accuracy between no hidden layers and 1-2 hidden layers. In regards to the increase of depth, there is no guarantee that an increase in the number of hidden layers would also increase the accuracy, given this would depend on the necessity of the model to make complex decisions. Thus, a small decrease in accuracy is reasonable. In task 3.2, we compared 2 hidden MLPs with different activation functions: Leaky- ReLU vs tanh. We observed that the accuracy of the Leaky-ReLU model was slightly higher than that of the tanh model (84.99% vs 84.65%). However, ReLU was the slightly more accurate activation model with 85.08%. Given the fact that Leaky-ReLU is mostly used to counter the vanishing gradient problem, and tanh is mostly effective in RNNS, this coincided with the popularity of ReLU [1]. In task 3.3, we were asked to implement L2 regularization where it obtained a testing accuracy of 85.7% after fitting the learning rate, which was slightly better than the unweighted model with 85.08%. In task 3.4 [2], we used the original 2 hidden layer MLP on unnormalized data and observed that the accuracy was slightly better, with an accuracy of 86.76% vs 85.08%. In task 3.5, we implemented convolutional neural networks and fit multiple parameters such as kernel size, stride length and max pooling window size. We observed that the best accuracies were recorded when the kernel size was of (2,2) dimensions, the stride length of (1,1) dimensions and the max pooling window size of (2,2) dimensions. After fitting these parameters, we decided to see whether or not dropout would increase the accuracy of our CNN. Three dropout models were compared to the original CNN model (with fitted parameters). The first [3] only applied a very small dropout to only the convolutional layers, the second [4] applied a more significant dropout only on the dense layers and the third combined both the 1st and 2nd model, applying partial dropout to the convolutional layers and

heavier dropout to the dense layers. The best model appeared after only applying dropout to the dense layers with a dropout rate of 0.25, which partially differed from the common 0.5 rates. Note that the original CNN model was already significantly outperforming the MLP models above with an accuracy of approximately 89% after only 15 epochs. However, after fitting parameters and adding dropout, the accuracy increased even more to 90% with a rather small difference between training and validation accuracy compared to the MLP. Finally, in task 3.6, we attempted to adjust the number of hidden layers of the MLP in order to try and create an even more powerful model, but it still fell short compared to the best-fitted CNN, where the accuracies were 87.55% and 90.56% respectively.

3 Datasets

The Fashion-MNIST dataset consisted of 60 000 training images and 10 000 testing images, where each example is a 28x28 grayscale image. The dataset is also equally divided into 10 classes, each belonging to a clothing category. To preprocess our dataset, we first flattened each image to a one-dimensional vector of size 784. Next, we normalized the images from both the training set and the testing set in order to transform them to a common scale, without distorting differences in the range of values. Furthermore, One-Hot Encoding was used to transform the categorical class labels into numerical class labels. Finally, we decided to split the training dataset (60 000 images) into 80% training and 20% validation and leave the testing dataset (10 000 images) as is.

4 Results

All following experiments were run over 15 epochs and all accuracies were obtained using a test set.

4.1 Task 3.1

Our zero, one and two-layer MLP models all have close accuracies. On most test runs, the zero layer gave 86.09% accuracy, one layer gave 87.34% and two layers gave 86.53%. Figure 1 describes the accuracies and CE loss as a function of the epochs of the training set compared with a validation set.

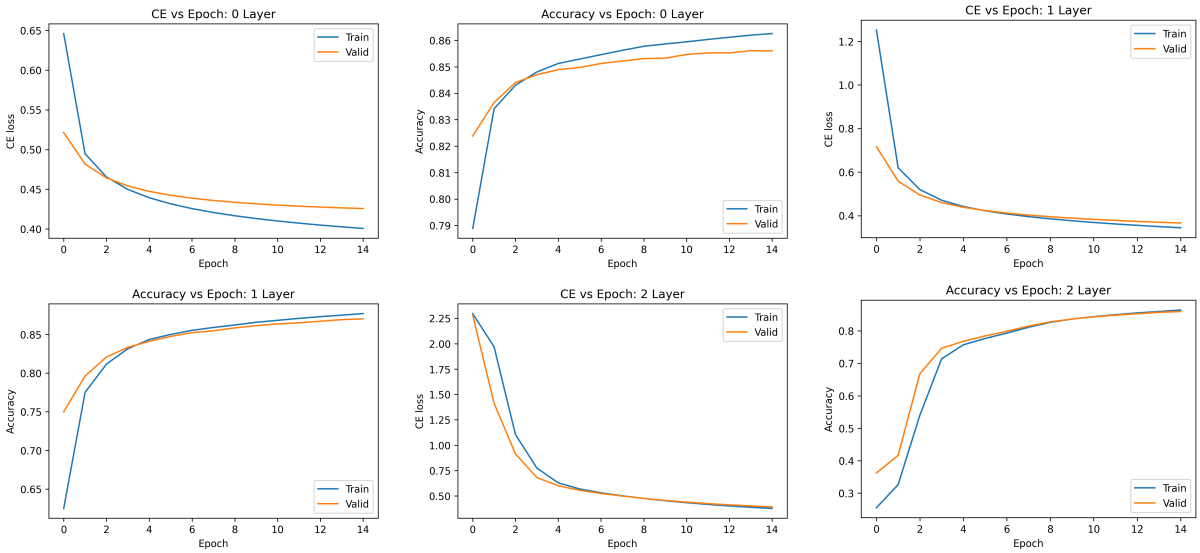


Figure 1: CE and Accuracy vs Epoch For Zero, One and Two-Layer MLP

4.2 Task 3.2

Figure 2 shows that the performance of the models using Leaky-ReLU and tanh did not improve yielding 86.53% and 86.37% respectively.

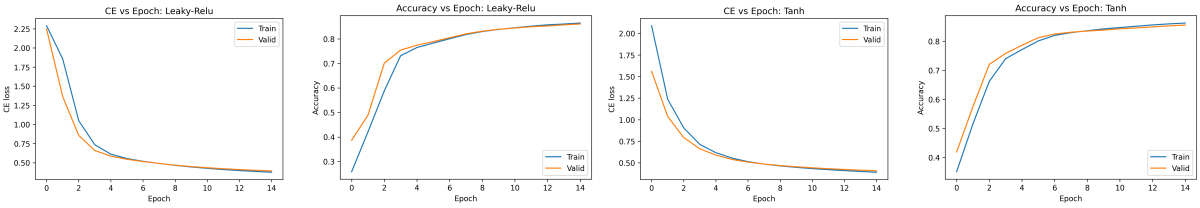


Figure 2: CE and Accuracy vs Epoch with Leaky-ReLU and Tanh Activation Functions

4.3 Task 3.3

L2-regularization also did not significantly improve our classification accuracy, obtaining at best 86.58% accuracy with $\lambda = 0.01$. Changing the value of λ barely changes the accuracy either as shown in Figure 3.

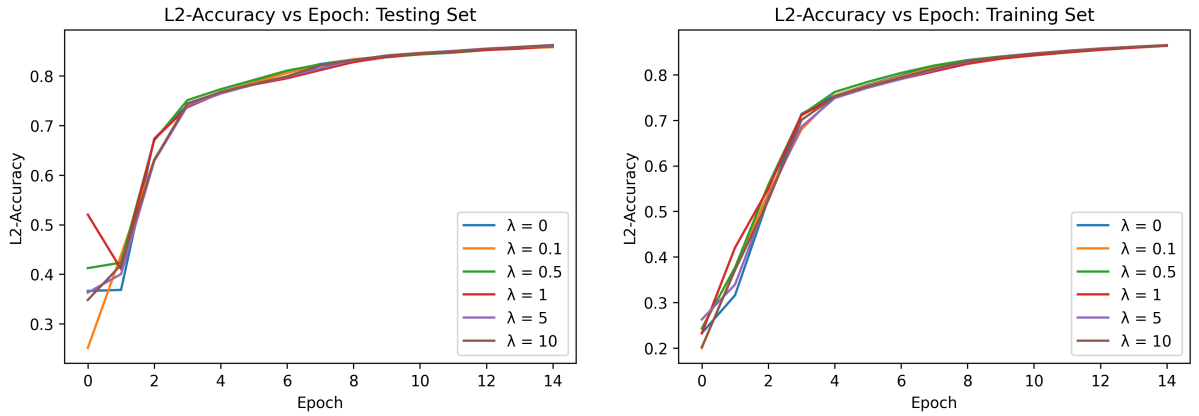


Figure 3: CE and Accuracy vs Epoch with L2-Regularization

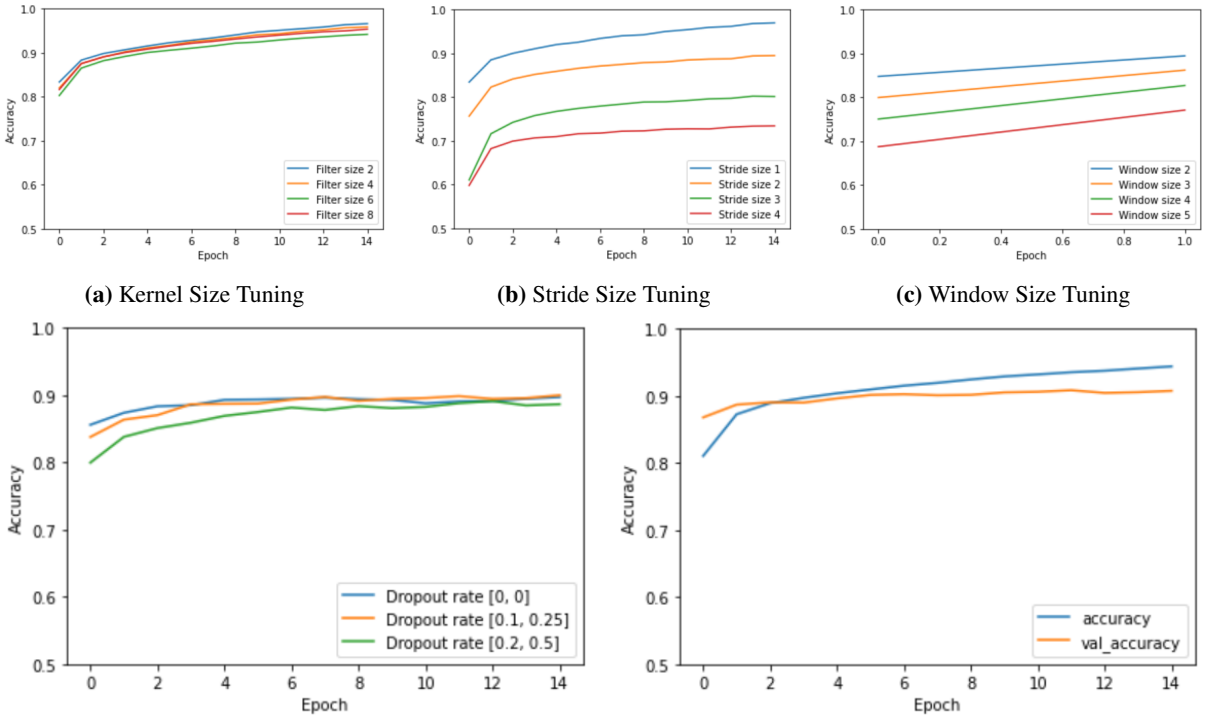
4.4 Task 3.4

We also tried avoiding the normalization of the data. Surprisingly, this gave us slightly better accuracy, hovering at about 86.77%.

4.5 Task 3.5

After incorporating a base CNN model, we observed from Figure 4a a testing accuracy of 89.05% after 15 epochs from which we wanted to improve upon. Thus, for the same model, we decided to fit different parameters such as the kernel size, stride length, and the max pooling window size. We saw that kernel dimensions of (2,2) brought forward the best testing accuracies across the 15 epochs (ranging between 89-90.16% in the final epochs). In regards to the stride size/length, it was made fairly obvious from Figure 4b that a smaller length was preferred with an accuracy of 90.51% on the 15th epoch while increasing the stride length led to an accuracy of 72.43%. Lastly, we observed from Figure 4c that a smaller max pooling window size increased the accuracy from 82.79% to 90.03%.

Additionally, we wanted to see the effects of dropout on the model and implemented dropout on layers based on exterior research [4]. We fit the dropout rates on these models and observed, from Figure 4d, that applying a dropout rate of 0.1 on the convolutional layers and a dropout rate of 0.25 on the dense layers yielded the greatest accuracy (90.20%) and severely decreased the difference between training and test accuracy. This was observed over 15 epochs. Finally, we combined all the fitted elements and the corrected dropout rate and model into one final CNN, where Figure 4e showed that the testing accuracy was 90.56% and the training accuracy was 94.33% in 15 epochs.



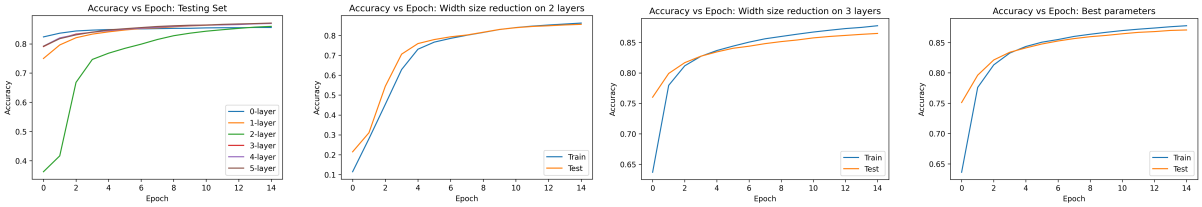
(a) Kernel Size Tuning **(b)** Stride Size Tuning **(c)** Window Size Tuning

(d) Dropout Rate Tuning **(e)** Best CNN Model

Figure 4: CNN Hyperparameters Tuning

4.6 Task 3.6

We have found that the best parameters were the following: one hidden layer using ReLU activation function, normalized data and without using L2-regularization. The choice of the number of hidden layers was made by testing zero to five hidden layers and gave the following accuracies respectively: 86.09%, 87.34%, 86.53%, 86.73%, 86.73%, 86.82%, which can be observed from Figure 5a. We also tried reducing the width as we went deeper into the neural network. This parameter was tuned on the two and three-layer MLPs. Results from Figure 5b and 5c indicated that this slightly decreased the accuracy by around 0.3% in each case. After fine-tuning the parameters and training the model over 15 epochs, Figure 5d shows that we obtain an accuracy of 87.55%. When comparing the CNN model with our best model, CNN achieves better accuracy in less training epochs.



(a) Different Layers **(b)** Width Reduction 2 Layers **(c)** Width Reduction 3 Layers **(d)** Best MLP Model

Figure 5: MLP Hyperparameters Tuning

5 Discussion and Conclusion

One can quickly observe that all of our models, regardless of the finetuning, gave very good classification accuracies on a test set. This is due to our greedy method of training. At every epoch, the accuracy can only increase. Since we were able to obtain over 85% accuracy, this suggests that the data does not follow a very complex model. This premise can also be further backed by how a one-layer MLP model outperforms the two, three or four layers model. Hence, to compare our models, we will consider small changes in accuracy, i.e. 1% change, to be significant.

Different activation functions did not provide greater accuracy. We did not encounter a vanishing gradient problem with our low amount of layers, hence Leaky-ReLU did not provide any advantages and the tanh function did not improve the results either. L2-regularization did not prove to be necessary as we already avoid overfitting by comparing the training set along with a validation set when calculating the cross entropy. Un-normalizing the data provided marginally better results. Normalization does not provide any advantage for non-linear results. Hence, similar results are to be expected when we avoid it. We found that using Convolutional Neural Networks boosted the accuracy to 90% with the best parameters. This is likely because of CNN's ability to detect important pixels for each image.

Other parameters that were tuned were the width of the hidden layers, as well as adding a 3rd, 4th and 5th hidden layer. The extra hidden layers did not prove to be effective at all, giving less accuracy than the one-layer MLP. Reducing the width also did not increase the accuracy of the model. Hence, our best model was the default parameters for the one-layer MLP using ReLU activation functions.

6 Subject Extension/Originality

In the first assignment, we were asked to apply KNN and Decision Trees in order to determine a patient's likelihood of having diabetic retinopathy. However, knowing the power of Convolutional neural networks on image classification, it would be extremely interesting to see whether or not we could classify the risk of a patient having diabetic retinopathy by utilizing CNNs [5].This prevents the intermediate use of exterior specialists to identify various features that are necessary in order to apply KNN and takes advantage of the increasing accuracy of medical imaging [6].

7 Statement of Contribution

Work was equally distributed among the three team members.

References

[1] R. Pramoditha, "How to choose the right activation function for neural networks," 2022. [Online]. Available: https://towardsdatascience.com/how-to-choose-the-right-activation-function-for-neural-networks-3941ff0e6f9c#:~:text=In%20MLP%20and%20CNN%20neural,activation%20function%20is%20considered%20linear

[2] Amir, "Is it a good practice to always scale/normalize data for machine learning?" 2016. [Online]. Available: https://stats.stackexchange.com/questions/189652/is-it-a-good-practice-to-always-scale-normalize-data-for-machine-learning

[3] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," 2012. [Online]. Available: https://www.google.com/url?q=https://arxiv.org/pdf/1207.0580.pdf&sa=D&source=docs&ust=1670275787405513&usg=AOvVaw34NwSKB7CMaPnWIjdhr5JV

[4] S. Park and N. Kwak, "Analysis of the dropout effect in convolutional neural networks." [Online]. Available: https://www.google.com/url?q=http://mipal.snu.ac.kr/images/1/16/Dropout_ACCV2016.pdf&sa=D&source=docs&ust=1670274455160699&usg=AOvVaw18I91AdxOzHdJXOd1Ajzi8

[5] H. Pratt, F. Coenen, D. M. Broadbent, S. P. Hardin, and Y. Zheng, *Convolutional Neural Networks for Diabetic Retinopathy*. Elsevier, 2016, vol. 90, online; accessed 05-Dec-2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050916311929

[6] A. Shanthini, G. Manogaran, and G. Vadivu, *Deep Convolutional Neural Network for The Prognosis of Diabetic Retinopathy*. Springer, 2022, online; accessed 05-Dec-2022. [Online]. Available: https://link-springer-com.proxy3.library.mcgill.ca/chapter/10.1007/978-981-19-3877-1_1