Artificial Intelligence II - Homework 2

Ioanna Oikonomou

December 2022

Contents

1 Introduction

In this report I am going to explain my solution to the second assignment which requires creating a feed forward neural network using Pytorch.

2 Assignment

2.1 Data preprocessing

The first step is the data preprocessing which is the same as in the first assignment. You can take a look at the equivalent report for further explanation.

2.2 Glove Embeddings

The glove vectors are extracted the same way that was shown at the lectures. Then, for every word of every preprocessed review, the equivalent word vector is assigned if it exists. Otherwise, the glove vector of "0" is assigned. Finally, the glove vectores are summed and divided by the number of words.

After experimenting with the 4 different dimensions of glove vectors, the dimension of 300 gave the best results so I continued with this one.

2.3 DataLoaders

Training and Validation loaders are made out of the embedded torches.

2.4 Neural Network

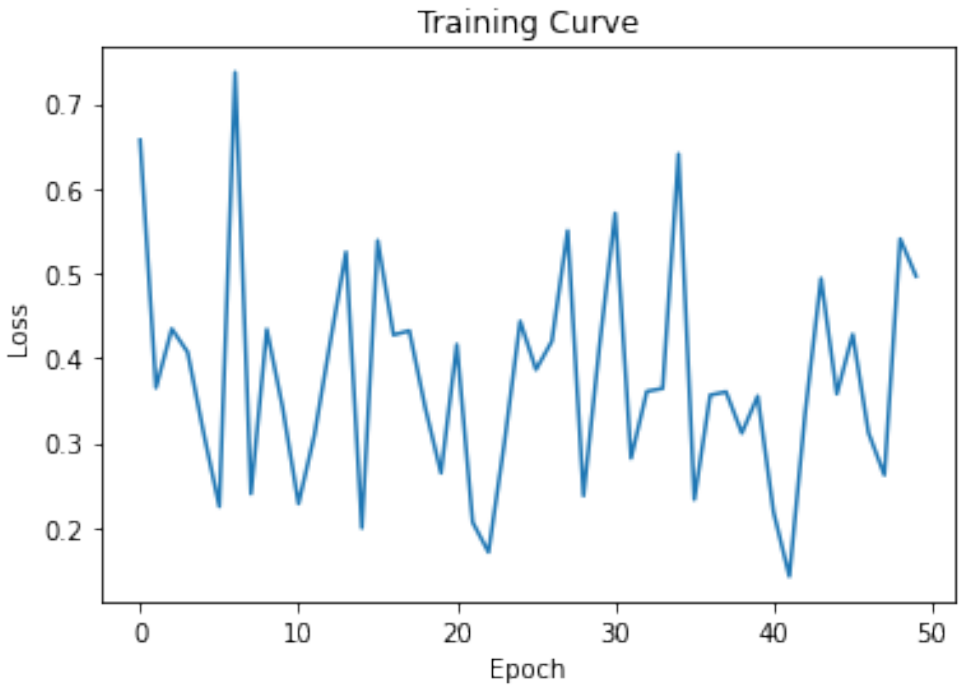
The final network tat I use in my implementation consists of 3 linear layers. The final layer is a sigmoid to determine a binary result.

2.5 Hyperparameters

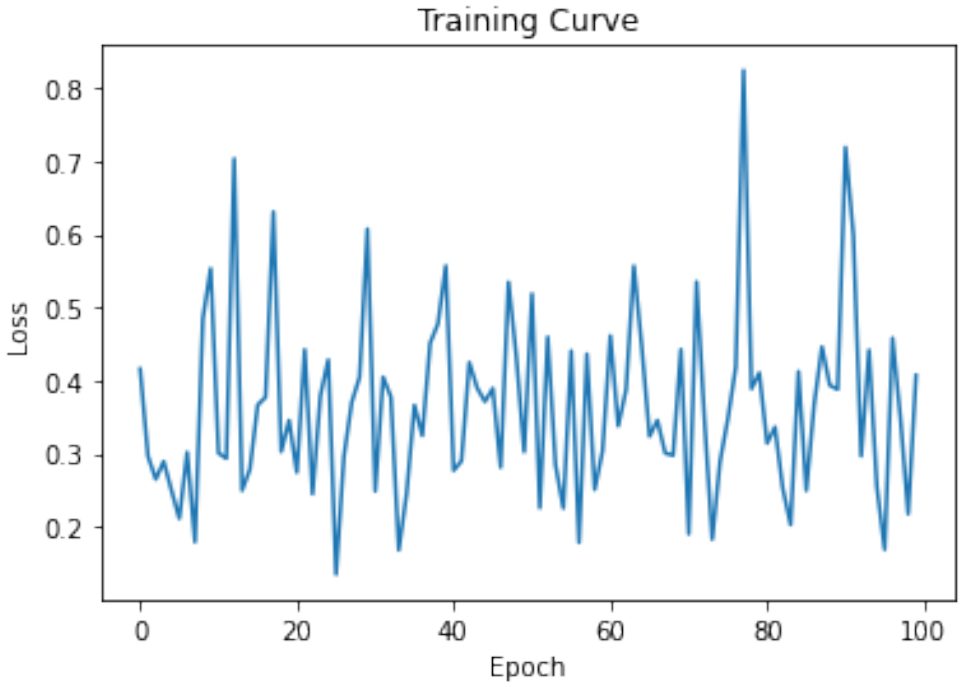
Below I'll try to explain the selection of every hyperparameter that I used. Before starting, I'd like to stress that I also tried an architecture with 2 linear layers instead of 3 but the loss was a bit bigger so I sticked to the other one.

2.5.1 Loss function

I used BCE as loss function because the final layer of my network consists of one class and the result is binary. As a result, this loss function was best fitting.



The learning curve of losses for a range of 50 epochs is shown above.



The learning curve of losses for a range of 100 epochs is shown above.

2.5.2 Learning rate

The learning rate of 1e-4 gave the best results. 1e-5 and 1e-3 were tried.

1e-5 gave a bigger loss but also bigger difference between epochs.

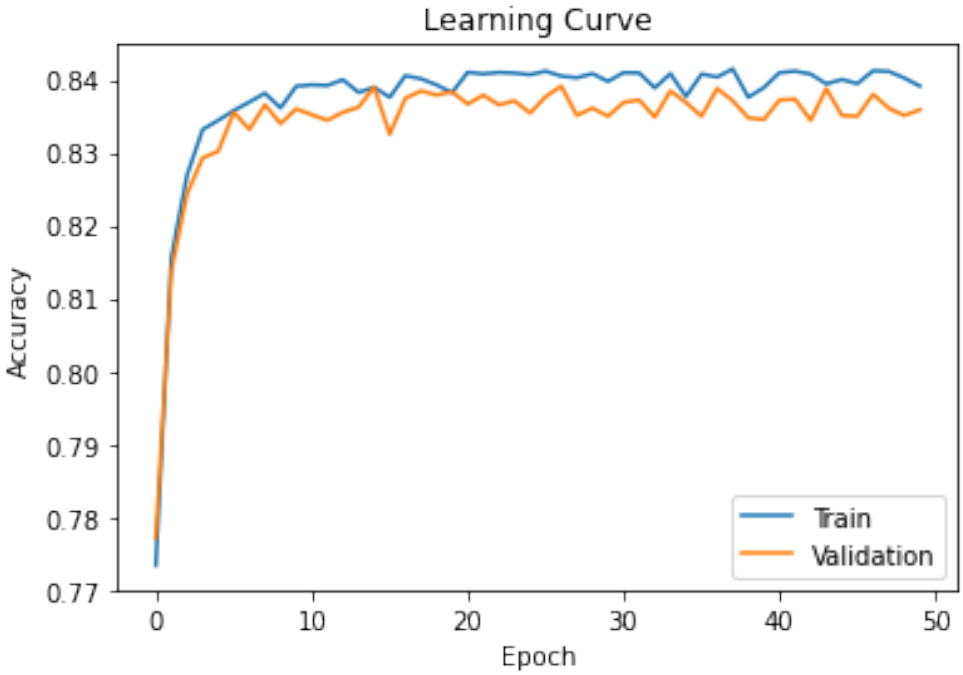
1e-3 also gave a bit bigger loss, so 1e-4 was kept as learning rate.

2.5.3 Optimizer

Adam optimizer is used in my model because SGD optimizer that was shown in class gave much bigger loss and much less accuracy.

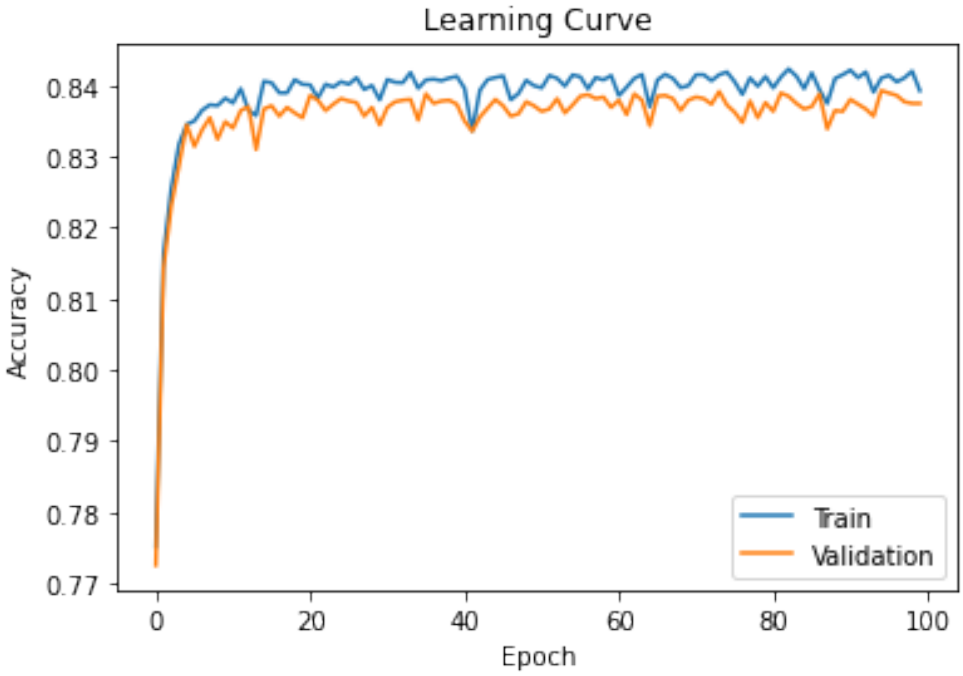
2.6 Accuracy

The accuracy of my model is approximately 0.83-0.85 and the learning curve for the training and validation set is shown below:



We can see that the accuracy in the training and validation sets are close enough and there is no overfitting or underfitting for 50 epochs.

The results were equally good for 100 epochs but the loss didn't change in the last 50 epochs so it was not needed to keep so many epochs in my final implementation. Below, however, is the laerning curve for a 100 epochs:



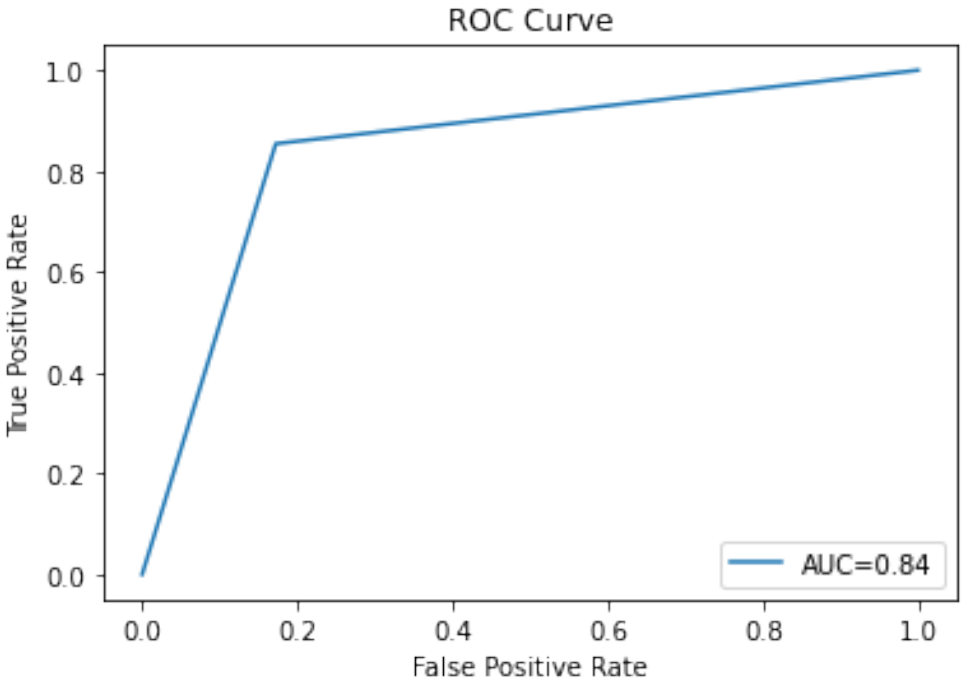
3 Torchmetrics

F1 score, precision and recall are calculated in my implementation.

They are all close to 0.83.

4 ROC Curve

The ROC curve thatresulted from my model is the following:



The auc seems good but the angle confused me a bit. I didn't know what else I could do so I kept this implementation. We can see that the shape of the curve and the auc show that my model does well.

5 Testing

The last part of my code is a section you can use to test with the test set. I'm not sure this is what you needed in order to test it. I hope it helps :)

6 Sources

-Torchmetrics
-Pytorch
-Lecture's slides from eclass
-Stack Overflow