UNIVERSITY OF ATHENS
DEPARTMENT OF INFORMATICS AND TELECOMMUNICATIONS

Deep Learning for NLP

Student name: <ΑΓΓΕΛΟΣ ΚΟΝΤΟΣ>
sdi: <*sdi2000089*>

Course: *Artificial Intelligence II (M138, M226, M262, M325)*
Semester: *Fall Semester 2023*

Contents

1. Abstract

the purpose of the work is to make a neural network that will pass a word2vec made of Tweets and find if they are positive, negative or neutral I solved this problem by initially making the Tweets in word2vec, then, I use optuna to do parameter experiments, I also played with how I will pass the inputs with average, addition, all together, etc., and of course playing with how many levels the neural network will have

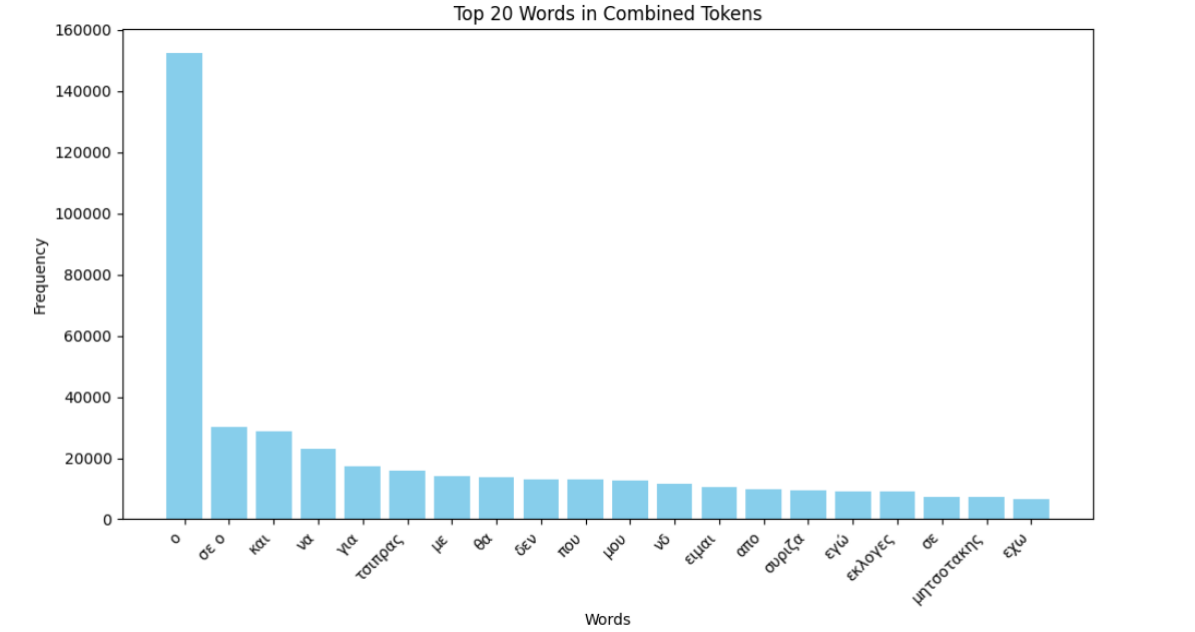
2. Data processing and analysis

2.1. Pre-processing

For pre process I use what I did in the previous work removing English words hastags, links, accents, I did lemmetazions and tokkenazations and finally I did padding so that they have the same number of words and average time and addition so that they have the same input size

2.2. Analysis

I have these for word frequency and word count



2.3. Data partitioning for train, test and validation

o dataloader when I used it with the optuna tests, it came out 36, not better than normal, but they still passed randomly, I never had the same result twice, that is, if you put it once, it came out 35, then 33, then 34, and then 36, but it was never stable with the same parameters I wish I had (save_version number later too put)

2.4. Vectorization

the vectorization used is the one recommended by word2vec and what specifically I

put with 100 vector acurrace initially to see which layers are better and they had almost the same strength neural hidden layers but then I will play with increasing the single sums because padding with the same value passes much more time but I will test it if I find that increasing the vector_size makes a real difference to the sum (because the all_word values and padding tests sometimes take hours)

3. Algorithms and Experiments

3.1. Experiments

at first my things were the hidden layers on average, sum and all of values with the help of padding (specifically what I do is pass a big list with each word and add empty words after so that they have the same number of input size elements) I saw how the average value(1) gives a general 33-34% and 36% the sum(2) and all values(3), so the average value was a bad choice probably because they share a lot of sentences, then I put a batch size but I saw that the experiments through optuna did not come out the same if I ran them again with the same values because the batches are random every time(more on dataloader).then I added the parties and tried the sum, it came out 36 to 37, and then I tried because I know the neuron works better with values from 0 to 1 to convert them by dividing by the largest party with the number 6 I have them from (1 to 6) and at the end I knew if I had a bigger epoch I would have a better result but I didn't

(In english)
(1) values
(2) athrisma=total
(3) meso oro=Average

i only write the diffences in the tittles

hidden layers 1 without dataloaders epoch 1-5k
(1) values
(2) athrisma
(3) meso oro

hidden layers 2 without dataloaders epoch 1-5k
(4) values
(5) athrisma
(6) meso oro

hidden layers 3 without dataloaders epoch 1-5k
(7) values
(8) athrisma
(9) meso oro

hidden layers 3 with dataloaders epoch 1-5k
(10) values
(11) athrisma

hidden layers 3 with partys and without dataloader epoch 1-5k
(12) athrisma

hidden layers 3 with partys but the input_number[0,1] and without dataloader epoch 1-5k
(13) athrisma

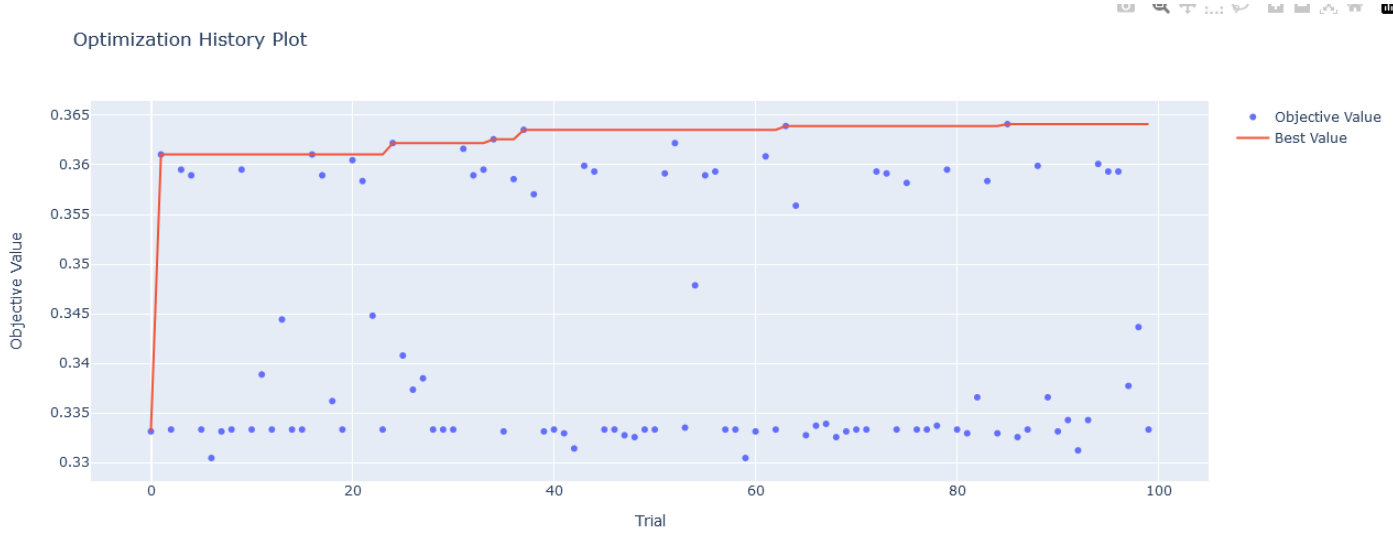
hidden layers 3 and with partys but the input_number[0,1] without dataloader epoch 2K-7k
(14) athrisma
(15) values

3.1.1. Table of trials. (χωρις αυτο δεν με αφηνει να κανω newline)

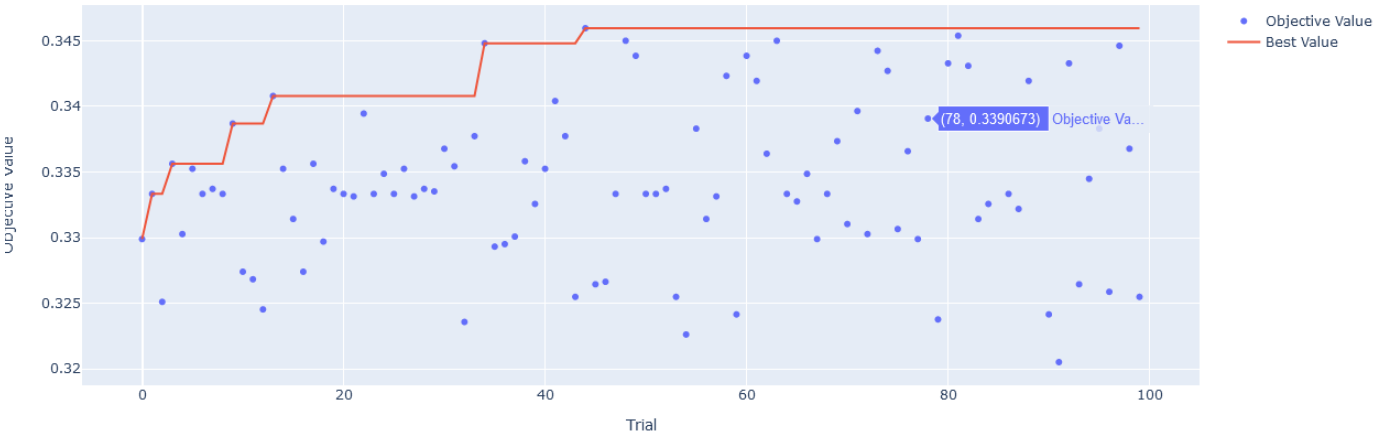
Trial	NEGATIVE	NEUTRAL	POSITIVE	Score
1	39%	41%	28%	36%
2	40%	45%	06%	36%
3	37%	11%	39%	33%
4	36%	43%	28%	36%
5	39%	46%	06%	36%
6	40%	00%	39%	33%
7	39%	43%	20%	36%
8	40%	44%	15%	36%
9	00%	50%	00%	33%
10	00%	00%	50%	33%
11	00%	06%	50%	33%
12	44%	33%	31%	37%
13	43%	33%	37%	38%
14	40%	36%	38%	38%
15	42%	38%	29%	37%

I put some of the loss functions and optuna tests as examples apart from the f1 score ,1 reason to make that the dataloader is not good in how i made the dataset

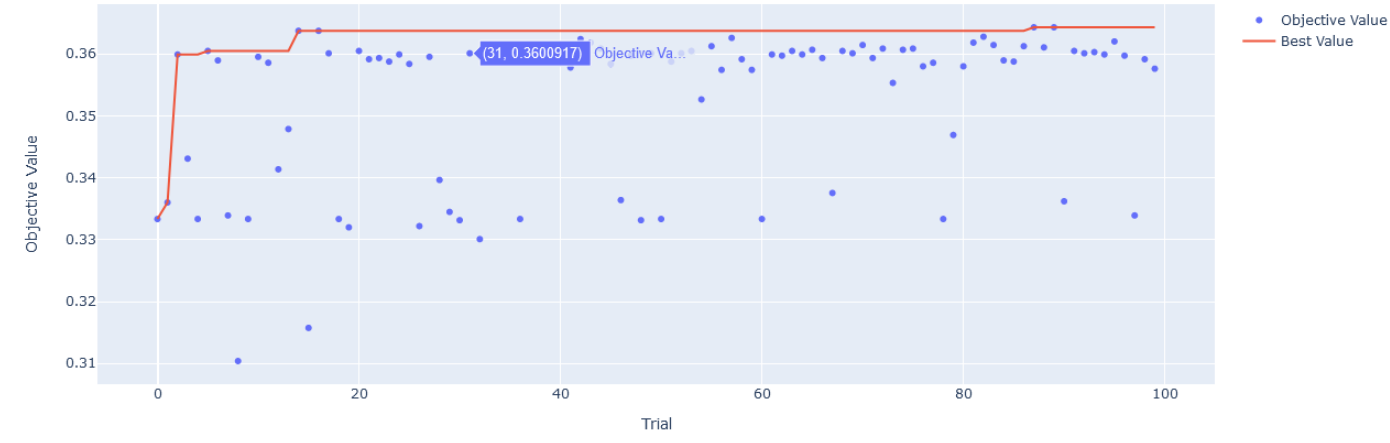
3.1.2. Optuna Tests. (newline)
(2)


Optimization History Plot
Objective Value
Best Value
Objective Value
Trial
0.365
0.36
0.355
0.35
0.345
0.34
0.335
0.33
0
20
40
60
80
100

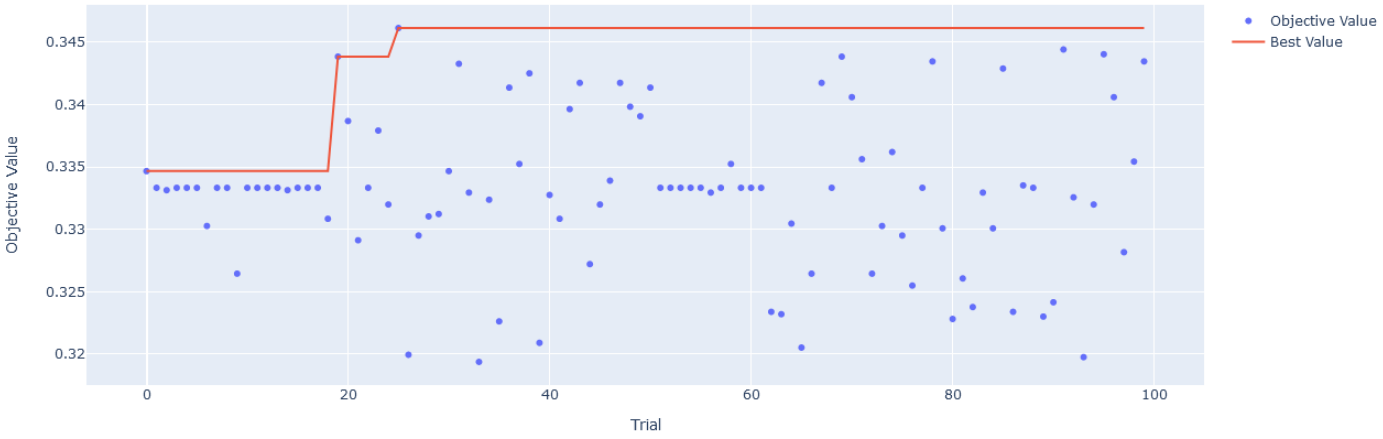
(3)


Objective Value
Best Value
(78, 0.3390673) Objective Va...
Trial
0.345
0.34
0.335
0.33
0.325
0.32
0
20
40
60
80
100

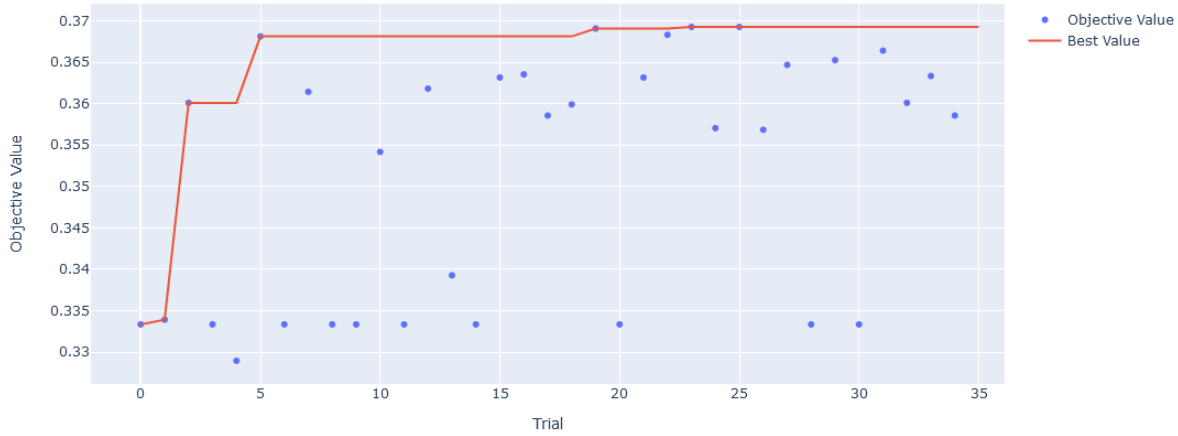
(5)


Objective Value
Best Value
(31, 0.3600917) Objective Va...
Objective Value
Trial
0.36
0.35
0.34
0.33
0.32
0.31
0
20
40
60
80
100

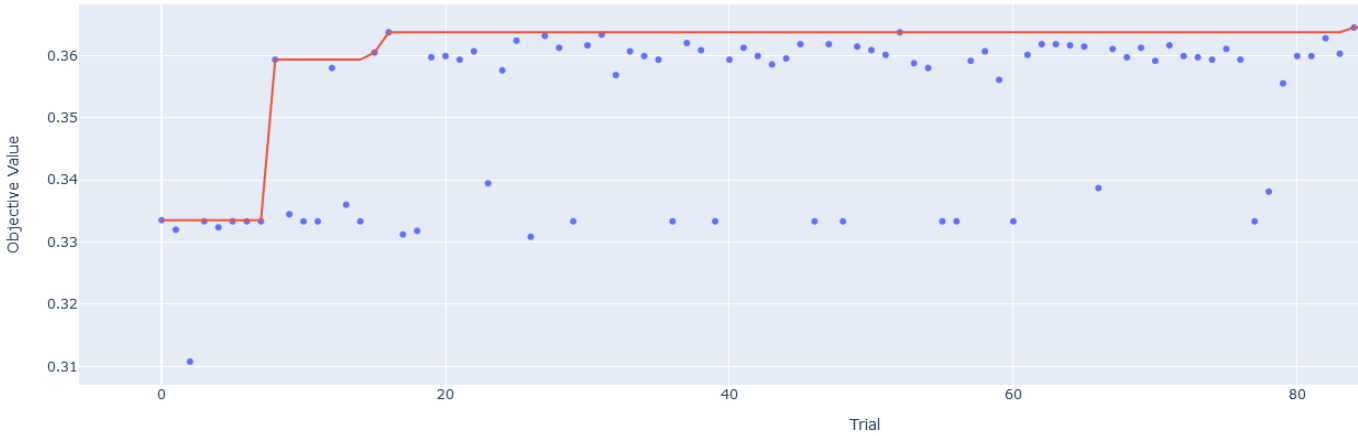
(6)



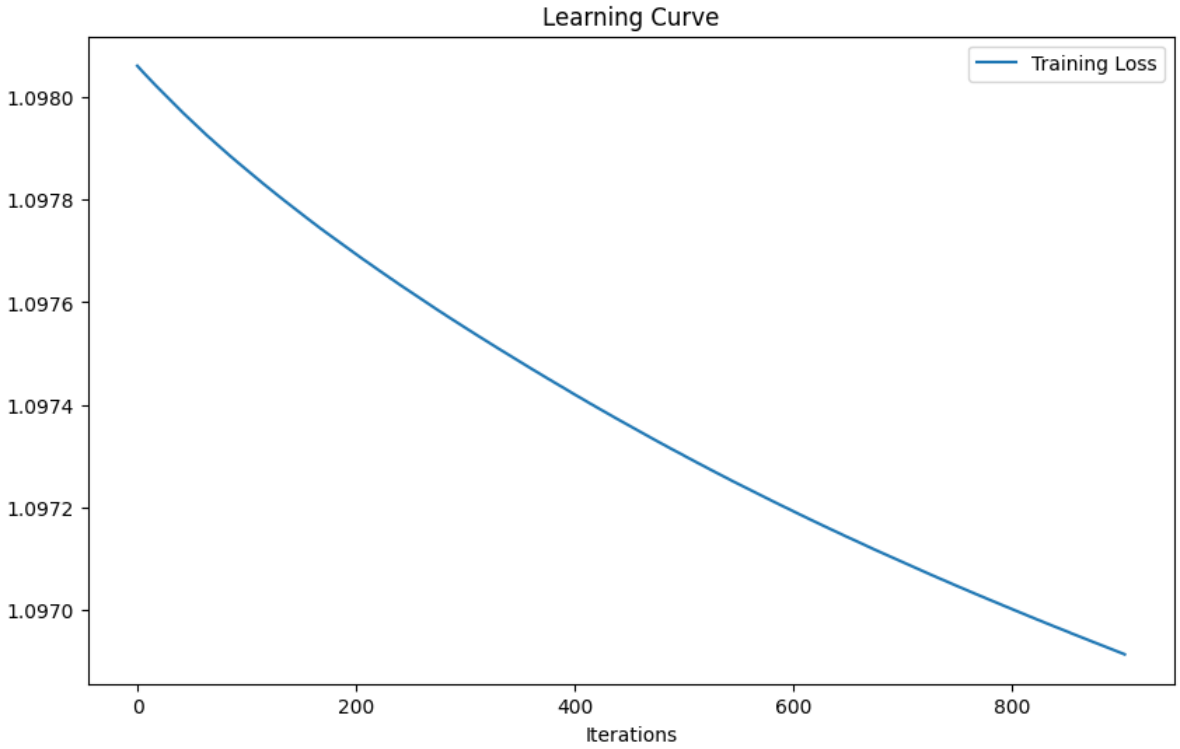
(7)



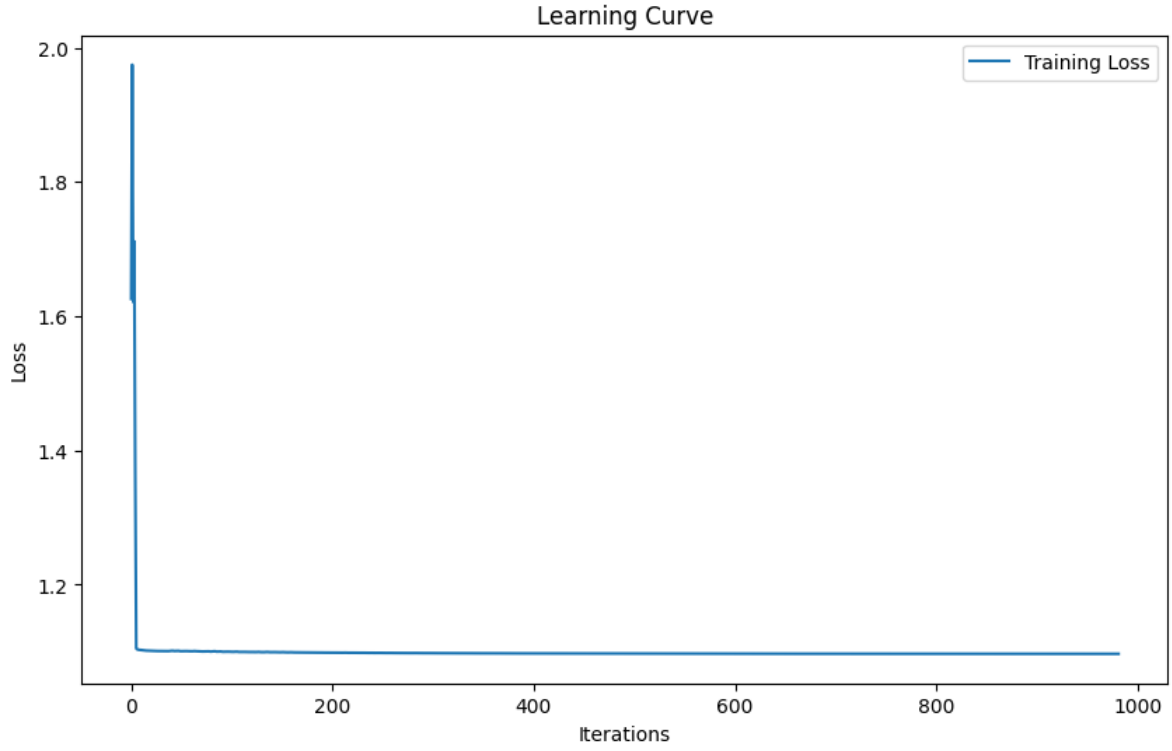
(8)



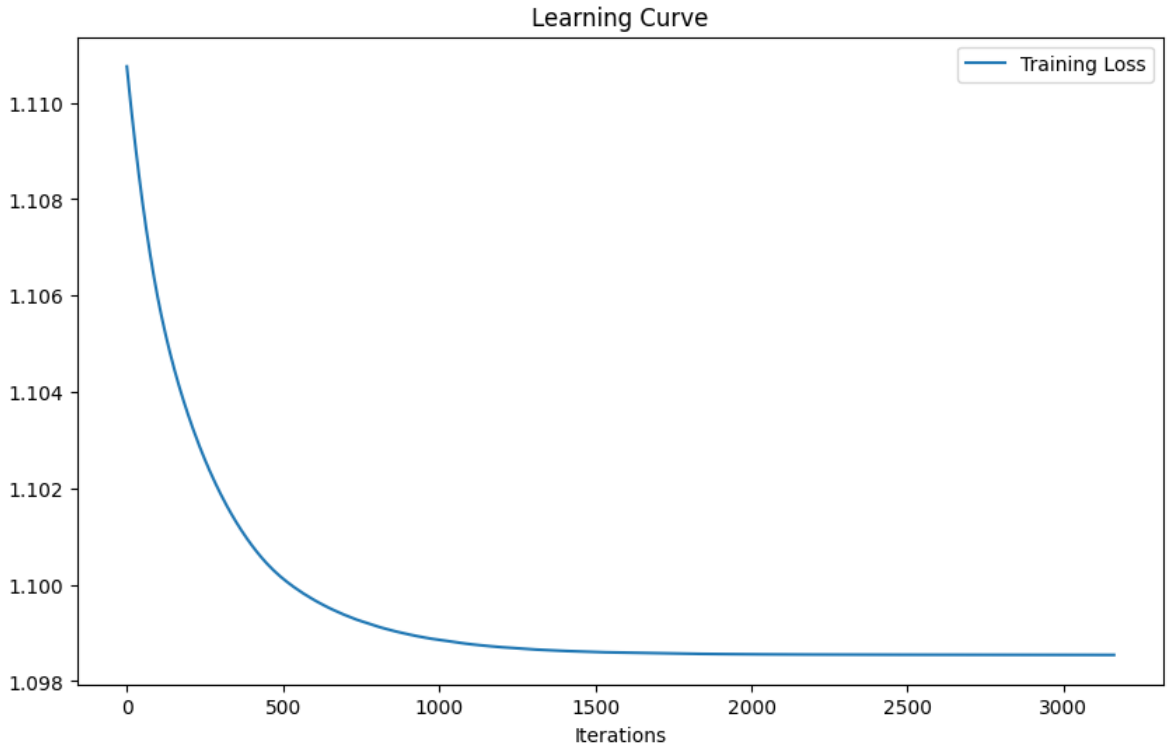
3.1.3. Learning Curve. (newline)
(1)


Learning Curve
Training Loss
1.0980
1.0978
1.0976
1.0974
1.0972
1.0970
0
200
400
600
800
Iterations

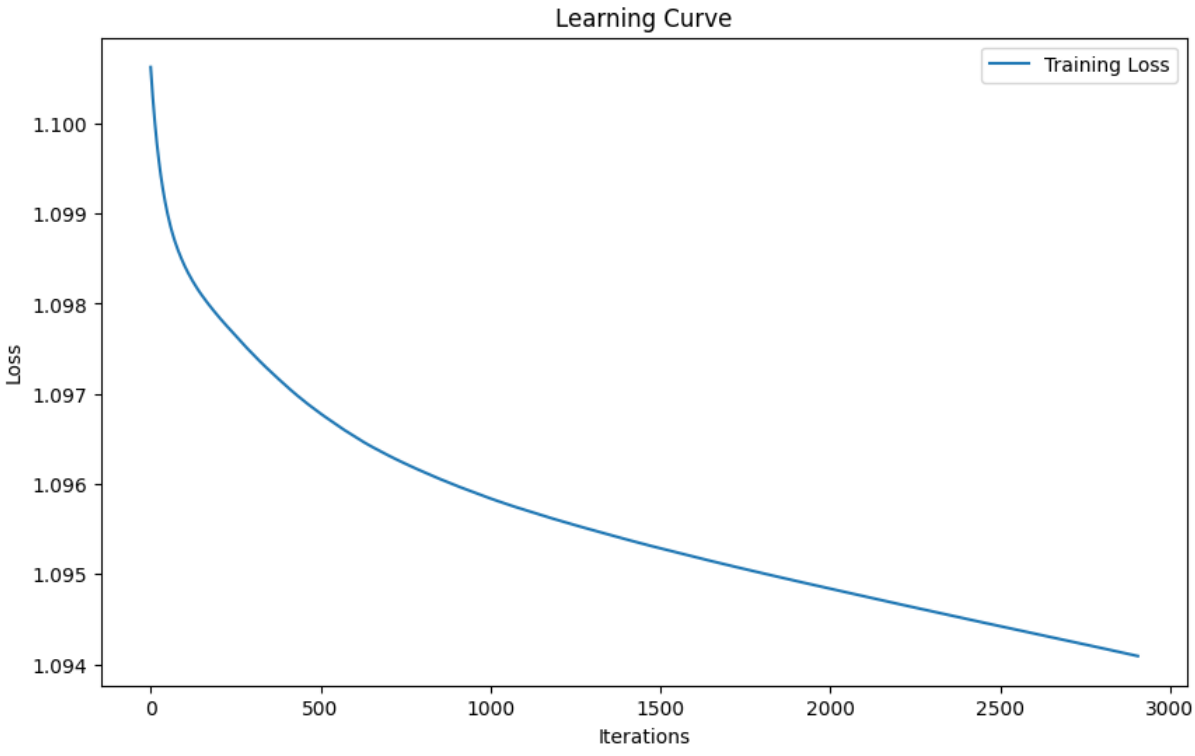
(2)


Learning Curve
Training Loss
2.0
1.8
1.6
1.4
1.2
Loss
0
200
400
600
800
1000
Iterations

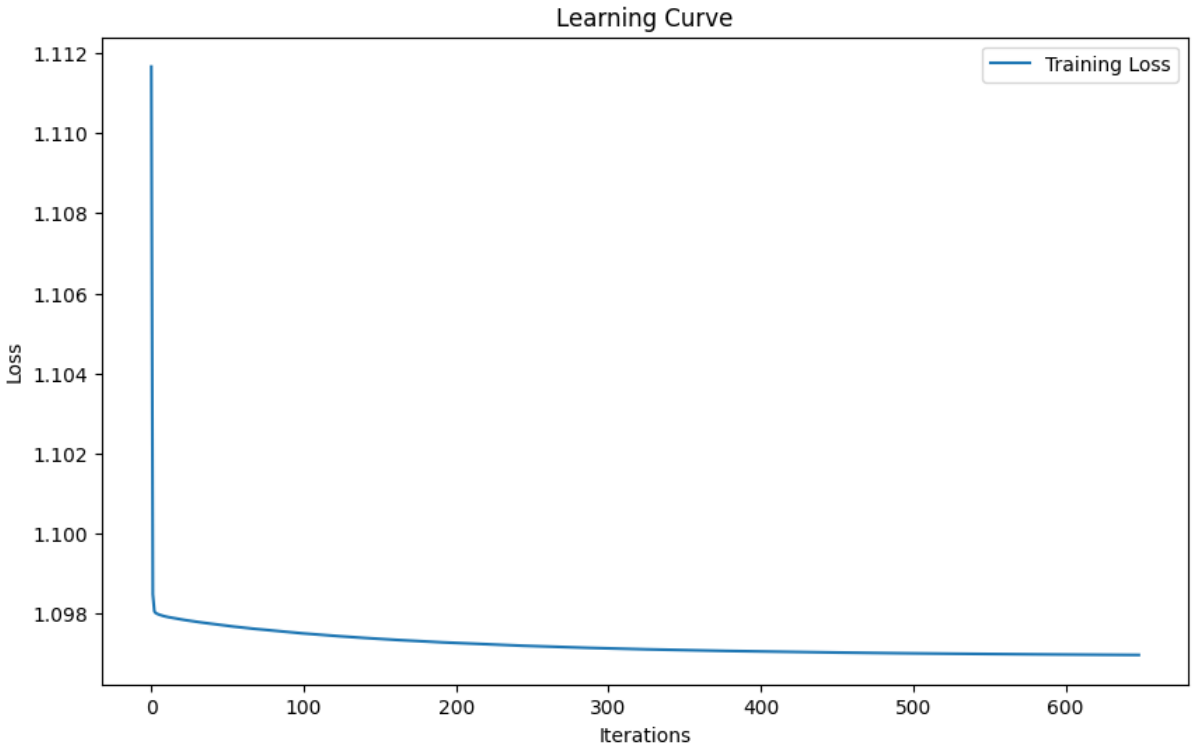
(3)


Learning Curve
Training Loss
1.110
1.108
1.106
1.104
1.102
1.100
1.098
0
500
1000
1500
2000
2500
3000
Iterations

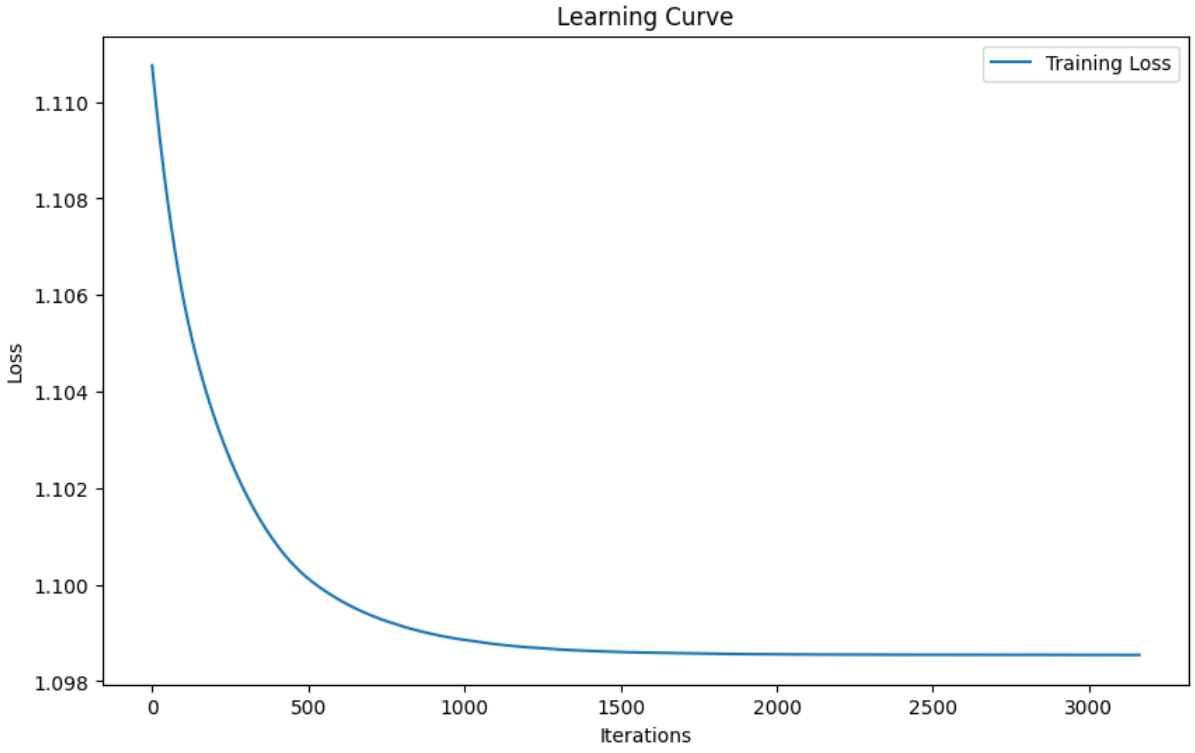
(4)


Learning Curve
Training Loss
1.100
1.099
1.098
1.097
1.096
1.095
1.094
Loss
0
500
1000
1500
2000
2500
3000
Iterations

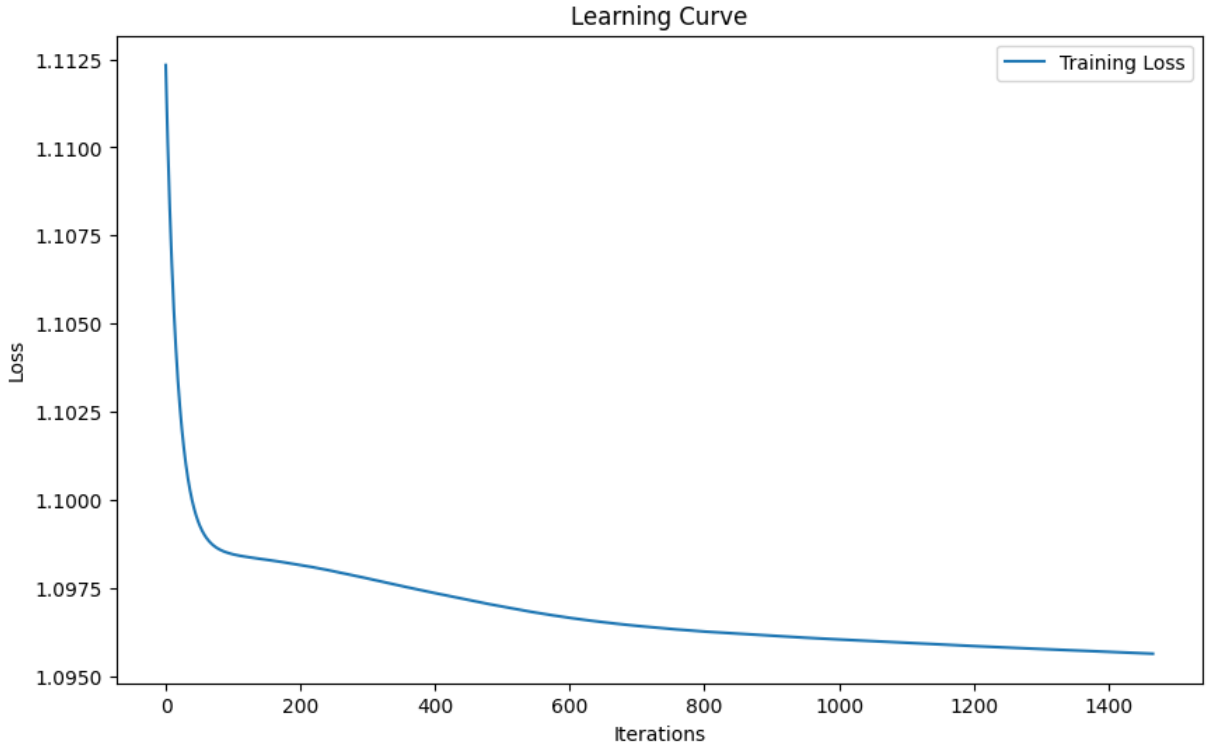
(5)


Learning Curve
Training Loss
Loss
Iterations
1.112
1.110
1.108
1.106
1.104
1.102
1.100
1.098
0
100
200
300
400
500
600

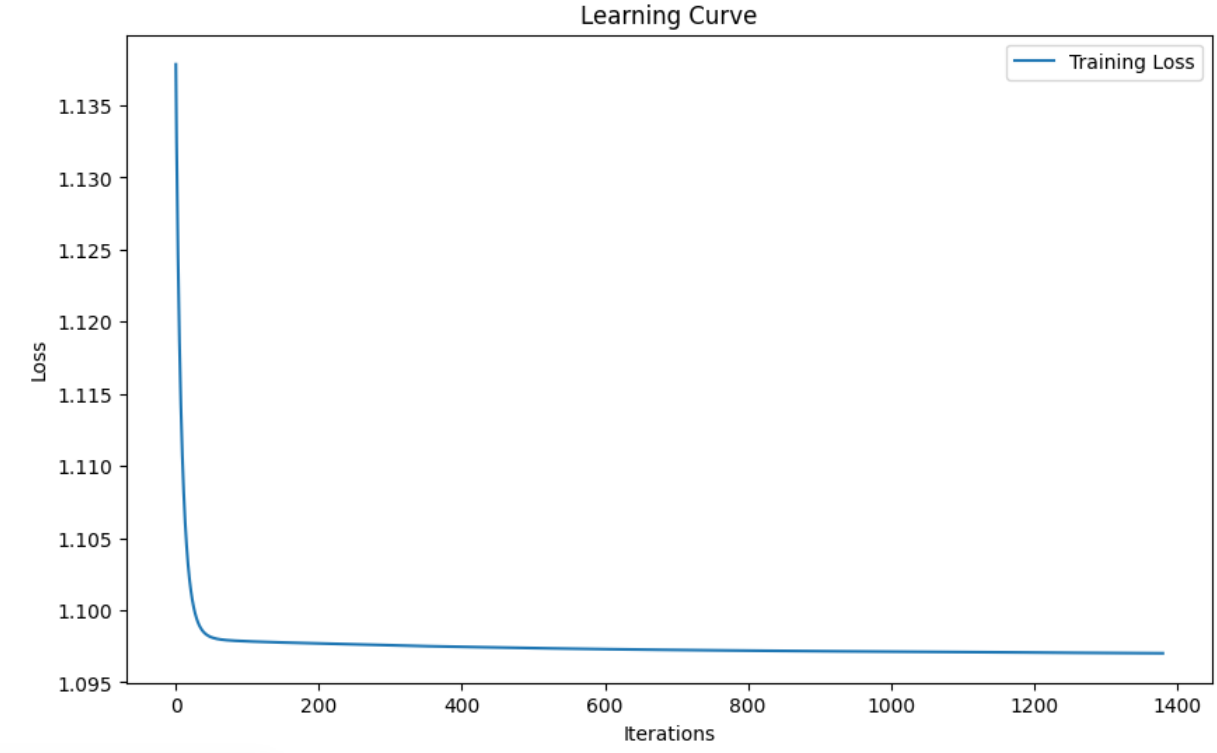
(6)


Learning Curve
Training Loss
Loss
Iterations
1.110
1.108
1.106
1.104
1.102
1.100
1.098
0
500
1000
1500
2000
2500
3000

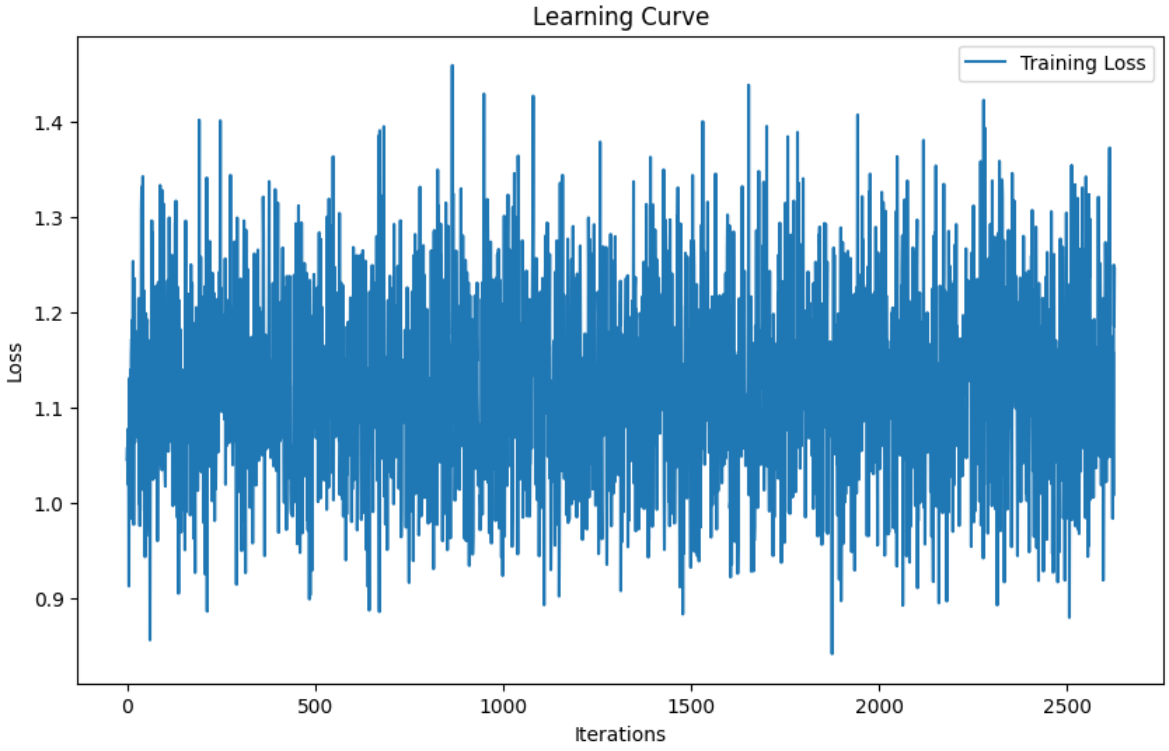
(7)


Learning Curve
Training Loss
Loss
Iterations
1.1125
1.1100
1.1075
1.1050
1.1025
1.1000
1.0975
1.0950
0
200
400
600
800
1000
1200
1400

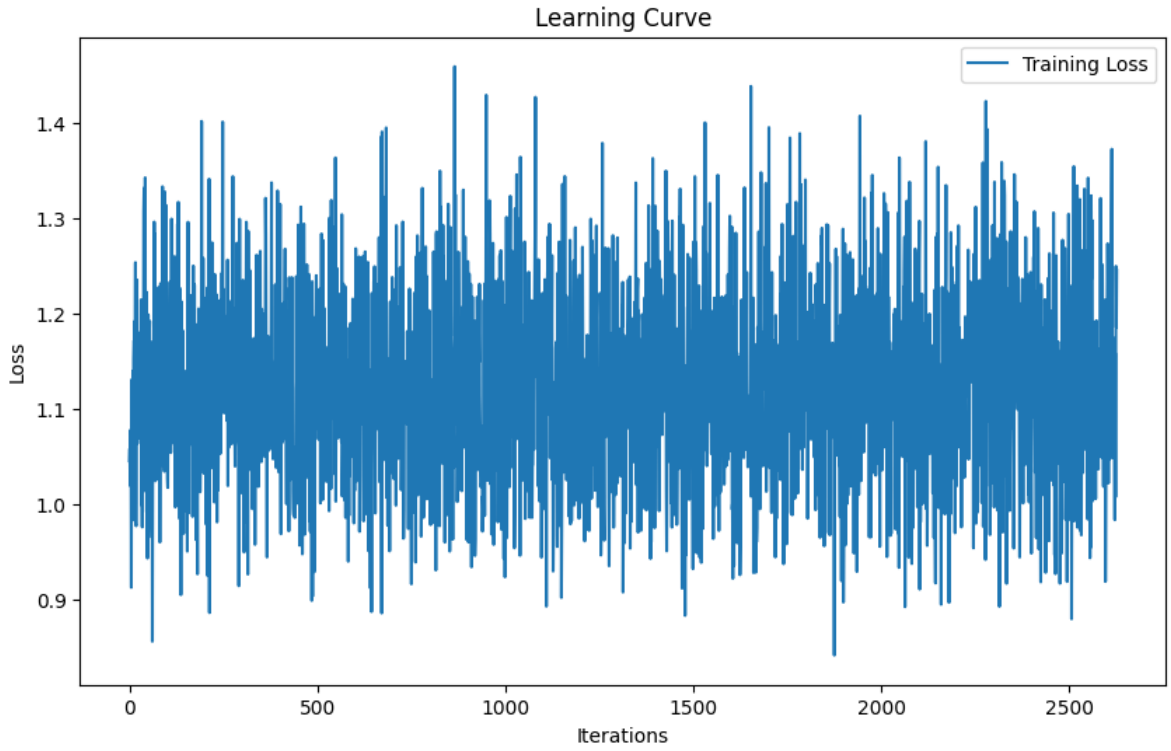
(8)


Learning Curve
Training Loss
Loss
Iterations
1.135
1.130
1.125
1.120
1.115
1.110
1.105
1.100
1.095
0
200
400
600
800
1000
1200
1400

(10)



(11)



3.2. Hyper-parameter tuning

I use optuna to play with hyper parameters, each result had all 100 trials except for level 1,2 for the values because they spend a lot of time and did not increase the f1 score much below are the hyperparameters for my tests specifically, I played with what value the loss parameter should have, of course, as I mentioned in the tests, how

many hidden layers will I have, also with how many epochs, with how many digits will each hidden layer give, how many when it finds worse loss not to continue searching with which function loss to run (adams etc), explain also with batch size

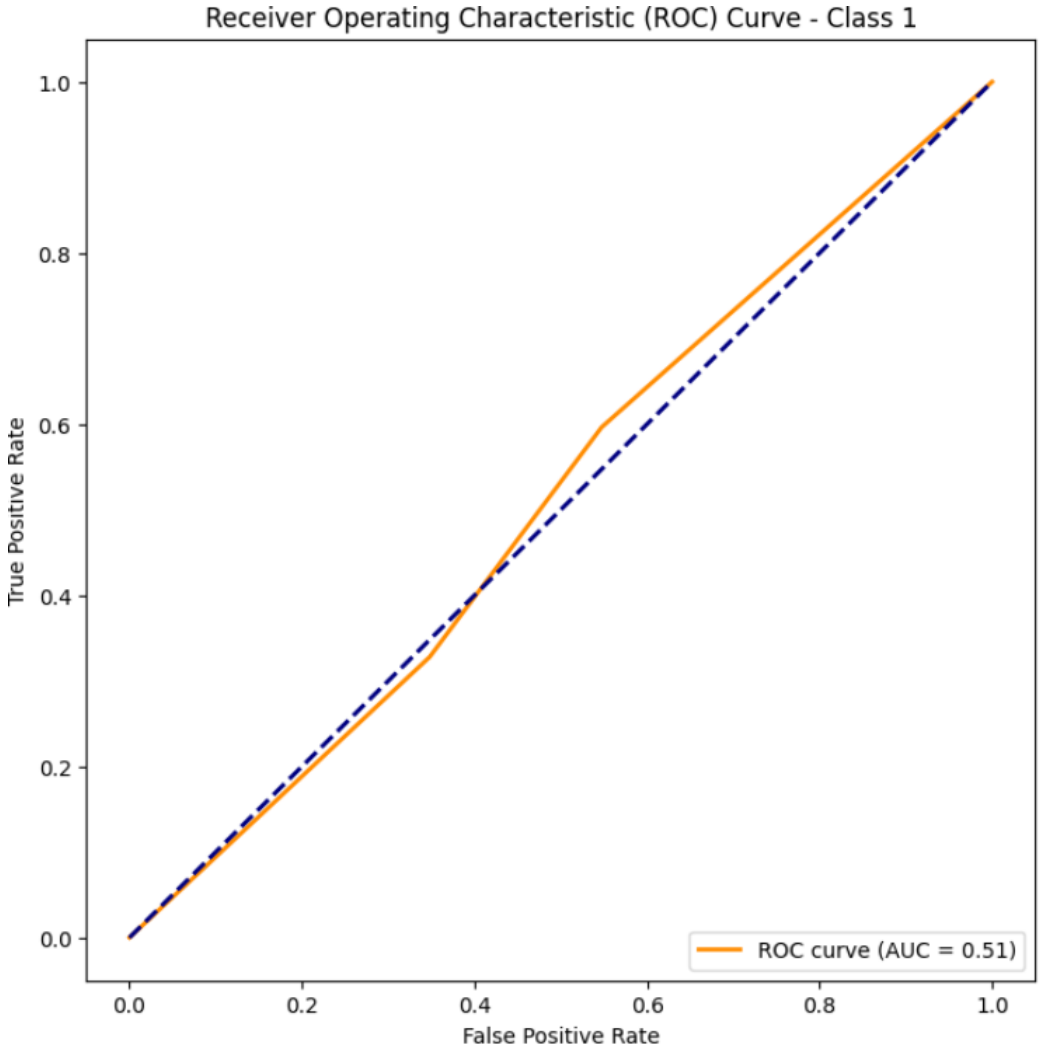
3.3. Optimization techniques

I used optuna where I put these parameters in these connections - values
loss parameter=[0.001,0.1]
number of trials =[1 ,5000]
loss_toll=[1, 3] (how many loss that are worst the best i+1 better then 0 to i)
optimizer_name=[Adam,RMSprop, SGD]
hidden layer 1=[3,100]
hidden layer 2 =[3,80]
hidden layer 3=[3,50]
batch_size =[3,10]

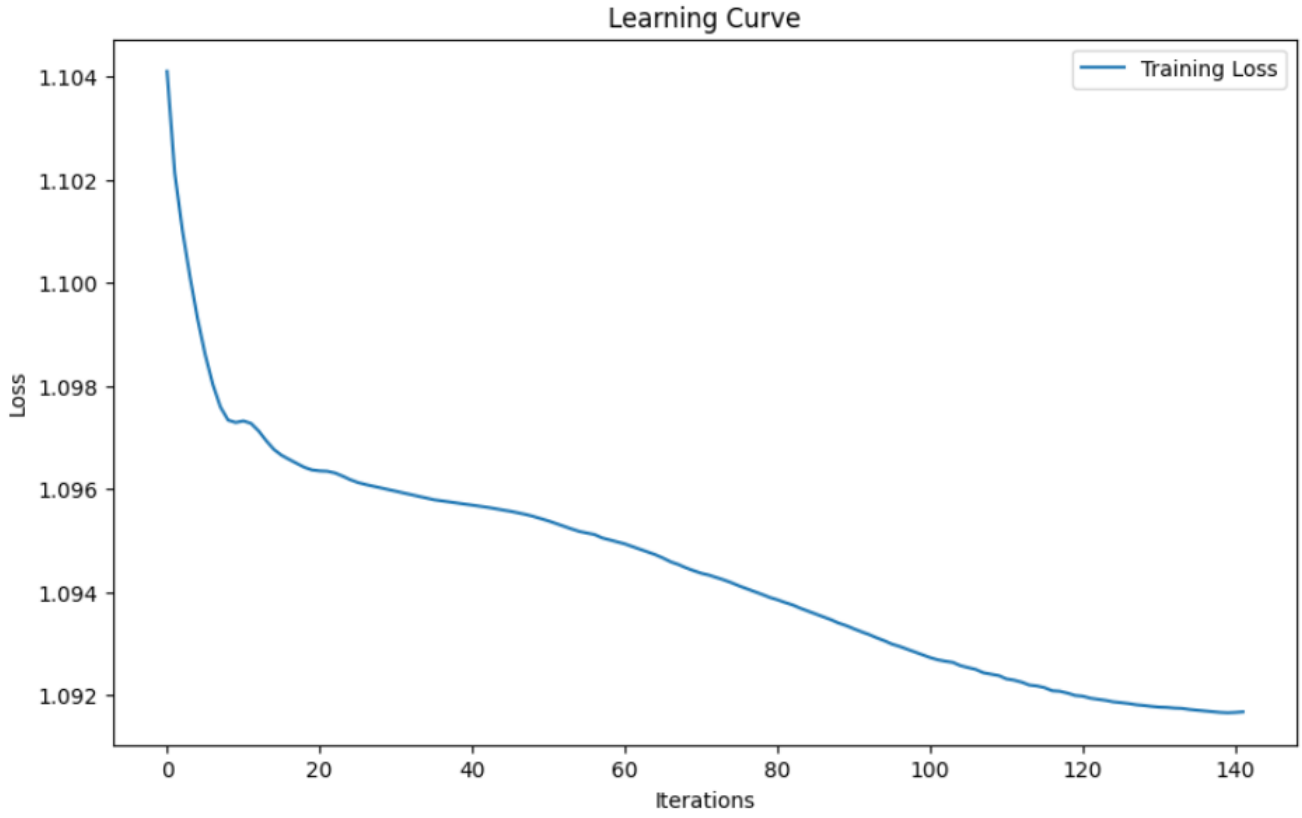
3.4. Evaluation

the predictions had a fairly stable sleep from 33-34 when they didn't have lemetazation (I didn't put them as f1_scores because I didn't do many experiments with them) also 34-36 with lemetazation and when I put the parties at 36-38 it brought out optuna

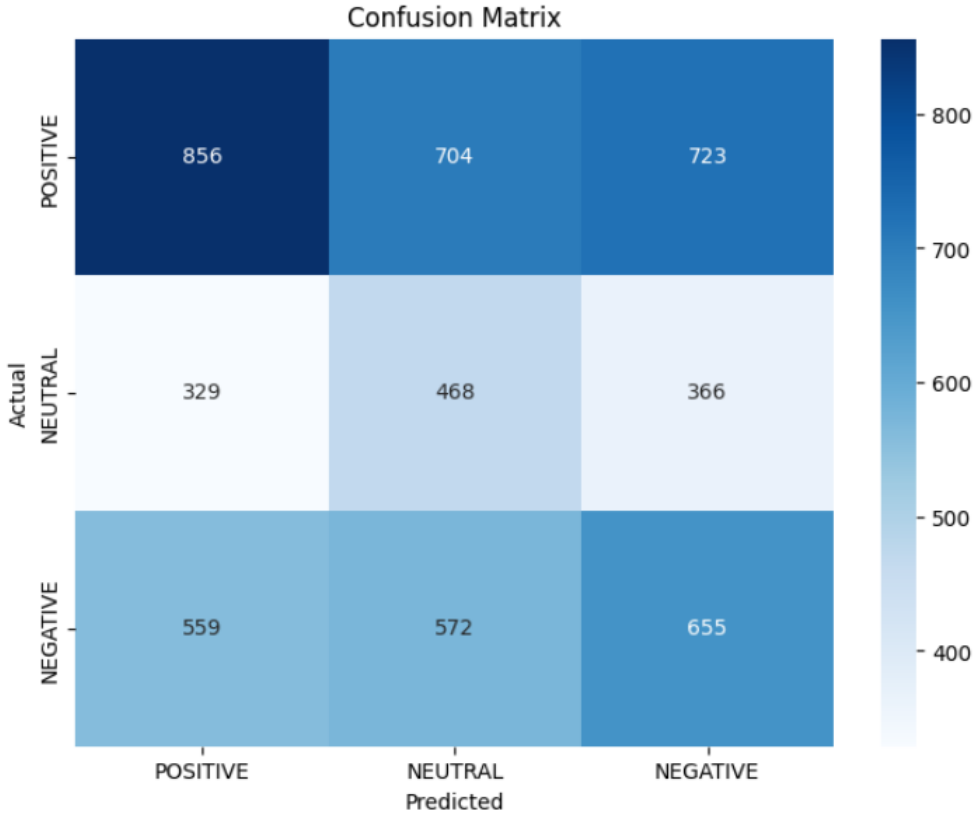
3.4.1. ROC curve.



3.4.2. Learning Curve.



3.4.3. Confusion matrix.



4. Results and Overall Analysis

4.1. Results Analysis

the neural network is easier to add and combine methods like with the political parties, also because of the type it has a bigger archive when searching for the specific dataloader doesn't help so much but in general before the parties I had 25-36 and after 37-38 theoretically if I played better, what words did I have in the dataset, maybe it went up to 39-40, but it couldn't go any further

4.1.1. Best trial.

4.2. Comparison with the first project

general had a greater safety than the first job, I had worse results but that's to blame because the dataset is not good, I also knew that it was easier to put the parties in and have general positive results, and of course there is much more customazation because we have many more parameters also that I see a very big improvement in the loss curve since I had the dataloader, but in general the experimentation was much more interesting and useful because it gave quite a few differences and did not have such a big luck factor

4.3. Comparison with the second project

<Use only for projects 3,4>
<Comment the results. Why the results are better/worse/the same?>

4.4. Comparison with the third project

<Use only for project 4>
<Comment the results. Why the results are better/worse/the same?>

5. Bibliography

References

piazza , optuna tutorial