

REPORT

Model Architecture:

An LSTM-based recurrent neural network is used for creating POS tagger in this assignment. Model consists of Embedding layer, LSTM, Fully connected layer, and the input to the model is sequence of words and output are there POS tags.

How it works

If we give a sentence, the model will output the tags for every word in sentences.

If sentences include punctuations, code will remove punctuations and only give output for words.

Hyperparameters of the Models:

Optimizer: Adam

Number of epochs : 25

Learning rate-0.01

Hidden Dimension-200

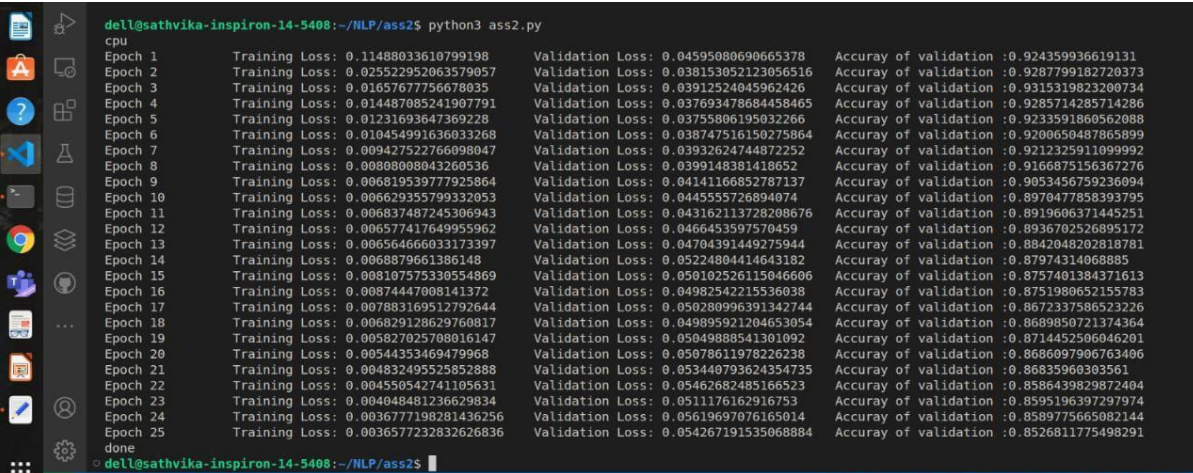
Embedding Dimension-300

Network-Embeddings, LSTM, Linear

Results

For hyperparameters like embedding size(300), hidden size(200), number of layers(3), learning rate(0.01) I trained the model for 25 epochs. Below are the Accuracy for validation set on each epoch.

cpu

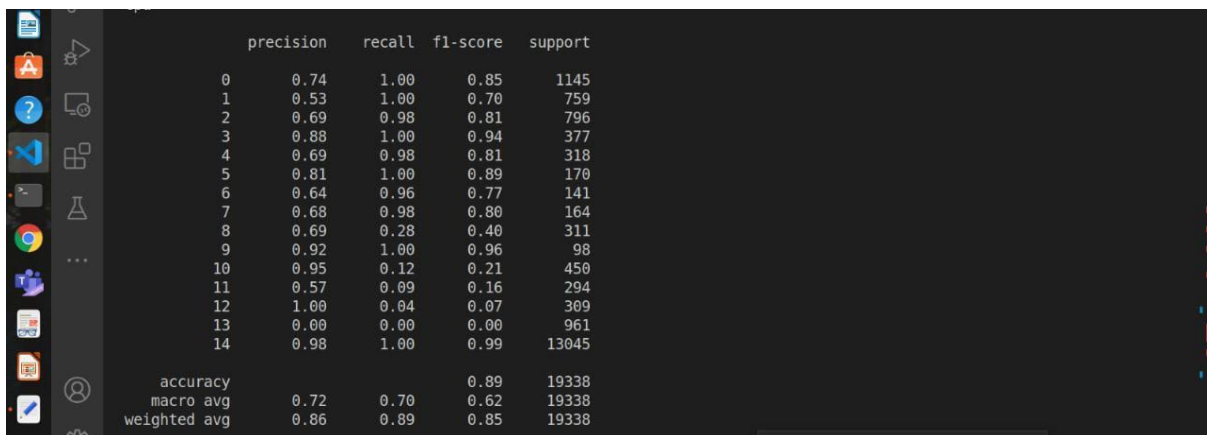


Epoch	Training Loss	Validation Loss	Accuracy of validation
Epoch 1	0.11488033610799198	0.04595080690665378	0.924359936619131
Epoch 2	0.025522952063579057	0.038153052123056516	0.9287799182720373
Epoch 3	0.01657677756678035	0.03912524045962426	0.9315319823200734
Epoch 4	0.014487085241907791	0.037693478684458465	0.9285714285714286
Epoch 5	0.01231693647369228	0.03755806195032266	0.9233591860562088
Epoch 6	0.010454991636033268	0.038747516150275864	0.9200650487865899
Epoch 7	0.009427522766098047	0.03932624744872252	0.9212325911099992
Epoch 8	0.008080808043260536	0.0399148381418652	0.9166875156367276
Epoch 9	0.006819539777925864	0.04141166852787137	0.9053456759236094
Epoch 10	0.006629355799332053	0.0445555726894074	0.8970477858393795
Epoch 11	0.006837487245306943	0.043162113728208676	0.8919606371445251
Epoch 12	0.006577417649955962	0.0466453597570459	0.8936702526895172
Epoch 13	0.006564666033173397	0.04704391449275944	0.8842048202818781
Epoch 14	0.0068879661386148	0.05224804414643182	0.87974314068885
Epoch 15	0.008107575330554869	0.050102526115046606	0.8757401384371613
Epoch 16	0.00874447008141372	0.04982542215536038	0.8751980652155783
Epoch 17	0.007883169512792644	0.050280996391342744	0.8672337586523226
Epoch 18	0.006829128629760817	0.049895921204653054	0.8689850721374364
Epoch 19	0.005827025708016147	0.05048080541301092	0.8714452586046201
Epoch 20	0.00544353460479968	0.05078611070226238	0.868697986763406
Epoch 21	0.004832495525852888	0.053440793624354735	0.86835960303561
Epoch 22	0.004550542741105631	0.05462682485166523	0.8596439829872404
Epoch 23	0.004048481236620834	0.0511176162916753	0.8595196397297074
Epoch 24	0.0036777198281436256	0.05619687076165014	0.858975665082144
Epoch 25	0.0036577232832626836	0.054267191535068884	0.8526011775498291

We can observe that Accuracy of validation goes on increasing to 0.9315319823200734 for epochs 3, and starts decreasing after 3 epochs. So I saved the model after running 3 epochs.

These are the precision, recall, F1-score and accuracy on test data by using the saved model.

ON test

A terminal window showing the output of a script. The output is a table with 5 columns: precision, recall, f1-score, and support. The rows are indexed from 0 to 14. At the bottom, there are summary rows for accuracy, macro avg, and weighted avg.

	precision	recall	f1-score	support
0	0.74	1.00	0.85	1145
1	0.53	1.00	0.70	759
2	0.69	0.98	0.81	796
3	0.88	1.00	0.94	377
4	0.69	0.98	0.81	318
5	0.81	1.00	0.89	170
6	0.64	0.96	0.77	141
7	0.68	0.98	0.80	164
8	0.69	0.28	0.40	311
9	0.92	1.00	0.96	98
10	0.95	0.12	0.21	450
11	0.57	0.09	0.16	294
12	1.00	0.04	0.07	309
13	0.00	0.00	0.00	961
14	0.98	1.00	0.99	13045
accuracy			0.89	19338
macro avg	0.72	0.70	0.62	19338
weighted avg	0.86	0.89	0.85	19338

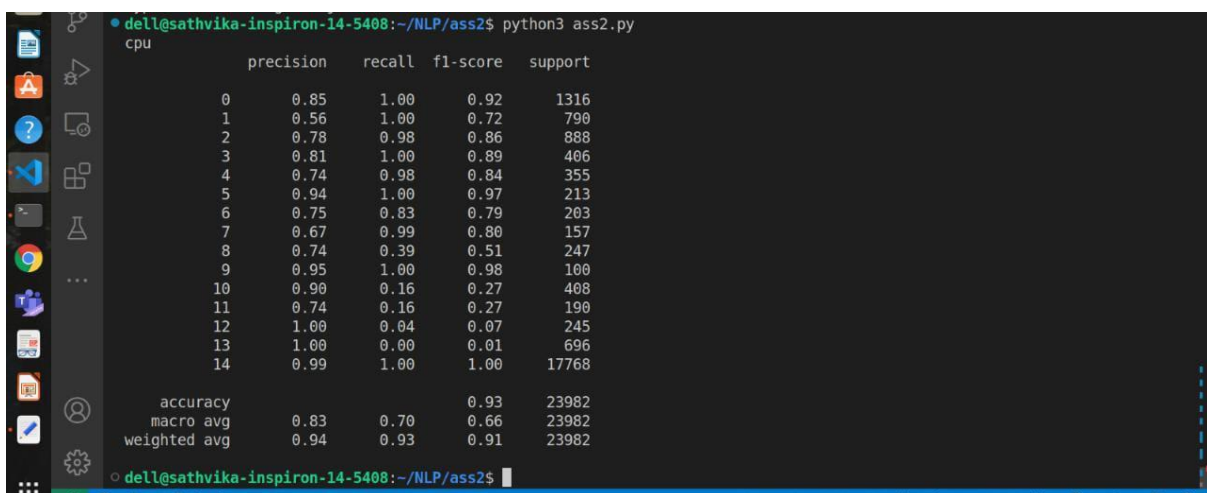
Accuracy on Test data : 0.89

Precision: 0.86

Recall: 0.89

F1-score: 0.85

On validation

A terminal window showing the output of a script. The output is a table with 5 columns: precision, recall, f1-score, and support. The rows are indexed from 0 to 14. At the bottom, there are summary rows for accuracy, macro avg, and weighted avg.

	precision	recall	f1-score	support
0	0.85	1.00	0.92	1316
1	0.56	1.00	0.72	790
2	0.78	0.98	0.86	888
3	0.81	1.00	0.89	406
4	0.74	0.98	0.84	355
5	0.94	1.00	0.97	213
6	0.75	0.83	0.79	203
7	0.67	0.99	0.80	157
8	0.74	0.39	0.51	247
9	0.95	1.00	0.98	100
10	0.90	0.16	0.27	408
11	0.74	0.16	0.27	190
12	1.00	0.04	0.07	245
13	1.00	0.00	0.01	696
14	0.99	1.00	1.00	17768
accuracy			0.93	23982
macro avg	0.83	0.70	0.66	23982
weighted avg	0.94	0.93	0.91	23982

Accuracy on validation set: 0.93

Precision: 0.94

Recall: 0.93

F1-score: 0.91

Analysis

I trained the POS tagger model for 25 epochs and I got good accuracy at 3 epochs for the validation set.

WHY best Accuracy at 3 epochs

After 3 epochs the accuracy started decreasing ,this is because of overfitting of the train data.As we increase the number of epochs in a neural network model can lead to decreasing accuracy due to a phenomenon called overfitting. Overfitting occurs when the model learns to fit the training data too well and becomes too specialised to that data, losing its ability to generalise to new, unseen data.When a model is trained for too many epochs, it can start to memorise the training data instead of learning the underlying patterns and relationships between the features and labels. This can lead to the model becoming too complex and sensitive to noise in the data, which can result in poor performance on new data.so when the new data comes it can't perform well ,as it is memorising completely without learning.Also,as we increase the epochs model cannot learn parameters effectively.

Also,we can observe that the model is performing well on the validation set as compared to test data,but not much difference.After 3 epochs we can see decrease in accuracy and there are some cases where we can see ups and downs in values.

Model is giving correct tags for sentences which are related to context of data,if we give sentences which are less related to context of train data ,model is not performing well,this is because the train data size is less .so model is not efficiently learning the morphological features

Also we can observe that in both test and validation sets,model performed well only on some parts-of -speech tags like noun,verb etc and performed less on conjunctions by seeing the precision and recall values.This is also because the data set is small and model able to tune parameters effectively.

Methods to increase the performance of model

If we train this model with a large data set ,we can get good results and will get good results for all parts-of-speech tags.Also we can improve the performance by including the architectures like encoders,decoders,WordToVec.

Experiments on Hyperparameters:

Model gave better results using Adam optimizer than SGD ,because of its adaptive learning rate and robustness.As we increase the epochs model is overfitting the training data i.e mimicking the data and not learning .so for my model 3 epochs gave the best results.

Increasing the embedding size and hidden size model is learning complex relations between input and output because of increased capability of model,its ability to capture information is increasing.Model perform also depending upon the learning rate as we keep more learning rate ,executing the model is very fast and model doesn't learn parameters effectively ,so we have to choose a learning rate which gives better results.