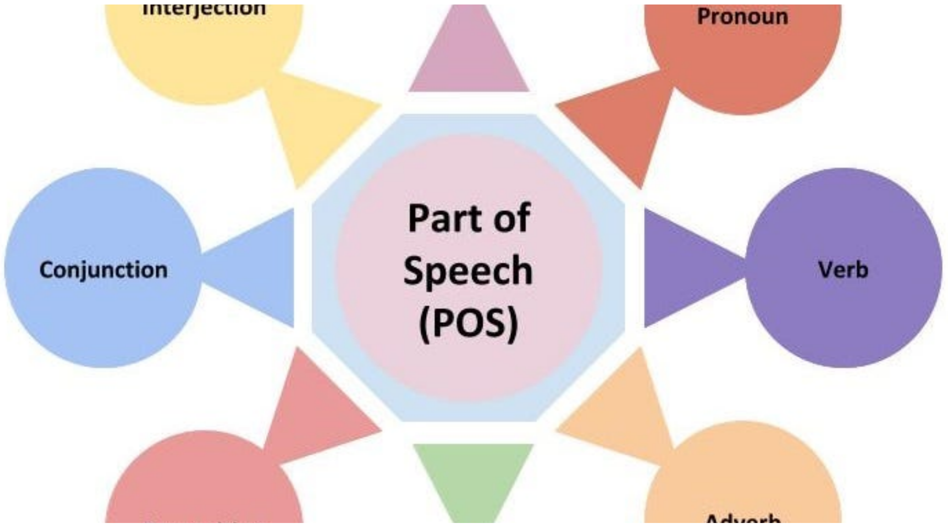
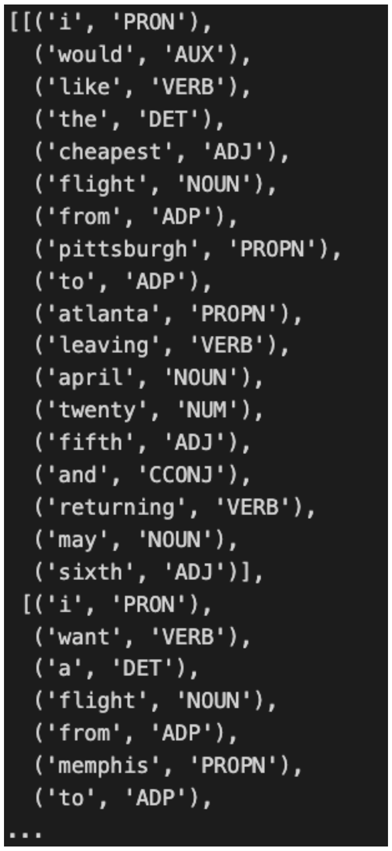


POS Tagger Report:

Input Format :

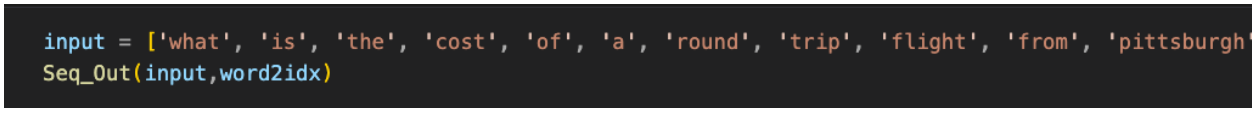
→ List of List of tuples (word and its tag)



```
[[('i', 'PRON'),
  ('would', 'AUX'),
  ('like', 'VERB'),
  ('the', 'DET'),
  ('cheapest', 'ADJ'),
  ('flight', 'NOUN'),
  ('from', 'ADP'),
  ('pittsburgh', 'PROPN'),
  ('to', 'ADP'),
  ('atlanta', 'PROPN'),
  ('leaving', 'VERB'),
  ('april', 'NOUN'),
  ('twenty', 'NUM'),
  ('fifth', 'ADJ'),
  ('and', 'CCONJ'),
  ('returning', 'VERB'),
  ('may', 'NOUN'),
  ('sixth', 'ADJ')],
 [('i', 'PRON'),
  ('want', 'VERB'),
  ('a', 'DET'),
  ('flight', 'NOUN'),
  ('from', 'ADP'),
  ('memphis', 'PROPN'),
  ('to', 'ADP'),
...
```

Indexing Function :

→ ***Input:*** *List of words and word to index dictionary*

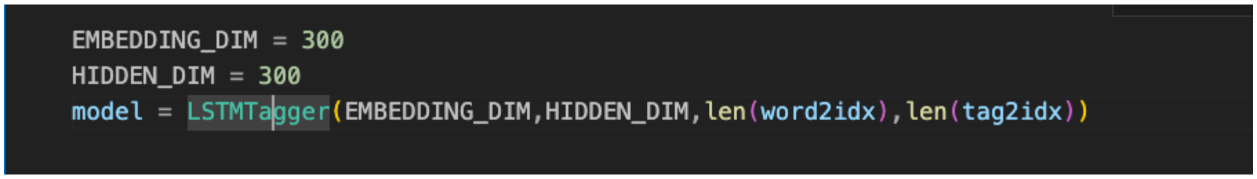


```
input = ['what', 'is', 'the', 'cost', 'of', 'a', 'round', 'trip', 'flight', 'from', 'pittsburgh'
Seq_Out(input,word2idx)
```

Embedding Dimension: 250

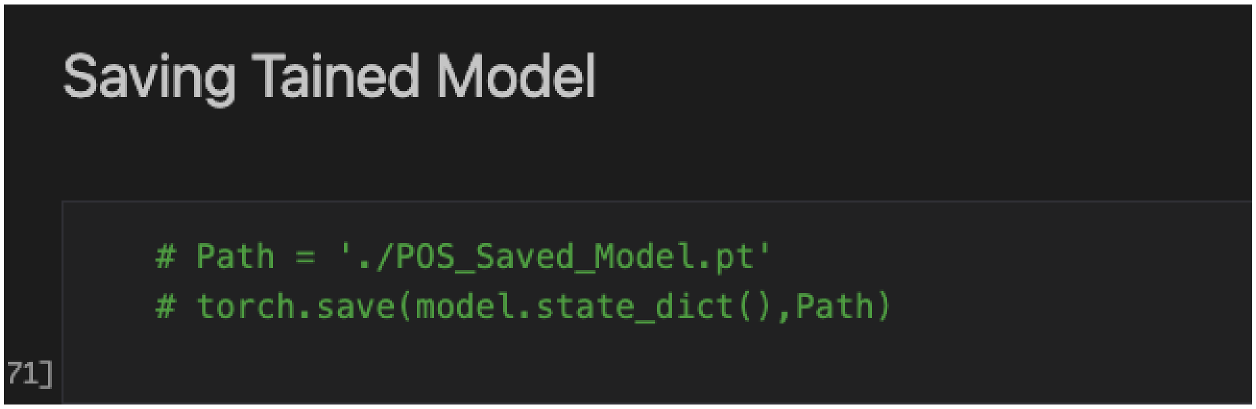
→Method: Manually checked for multiple values and compared output.

Hidden Dimension: 300



```
EMBEDDING_DIM = 300
HIDDEN_DIM = 300
model = LSTMTagger(EMBEDDING_DIM,HIDDEN_DIM,len(word2idx),len(tag2idx))
```

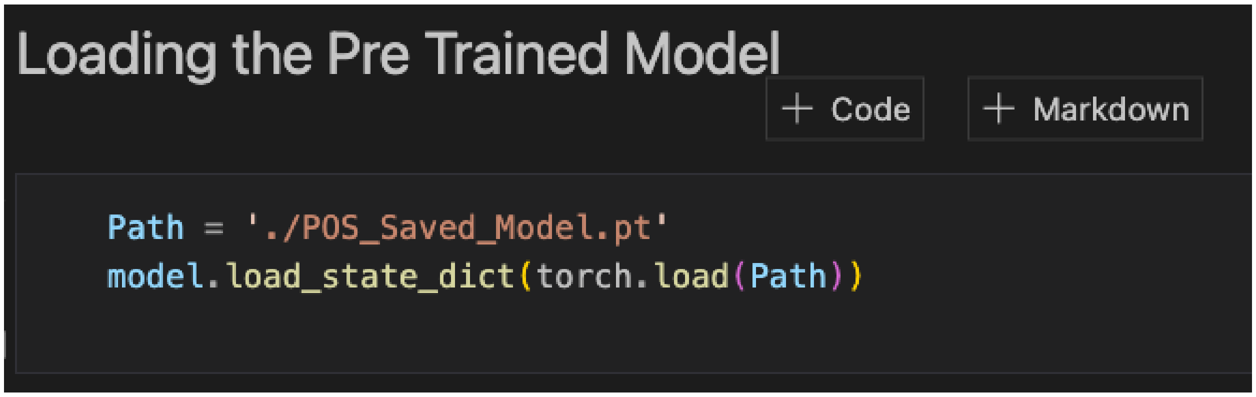
Saving and Loading of Training Data:



Saving Tained Model

```
# Path = './POS_Saved_Model.pt'
# torch.save(model.state_dict(),Path)
```

71]

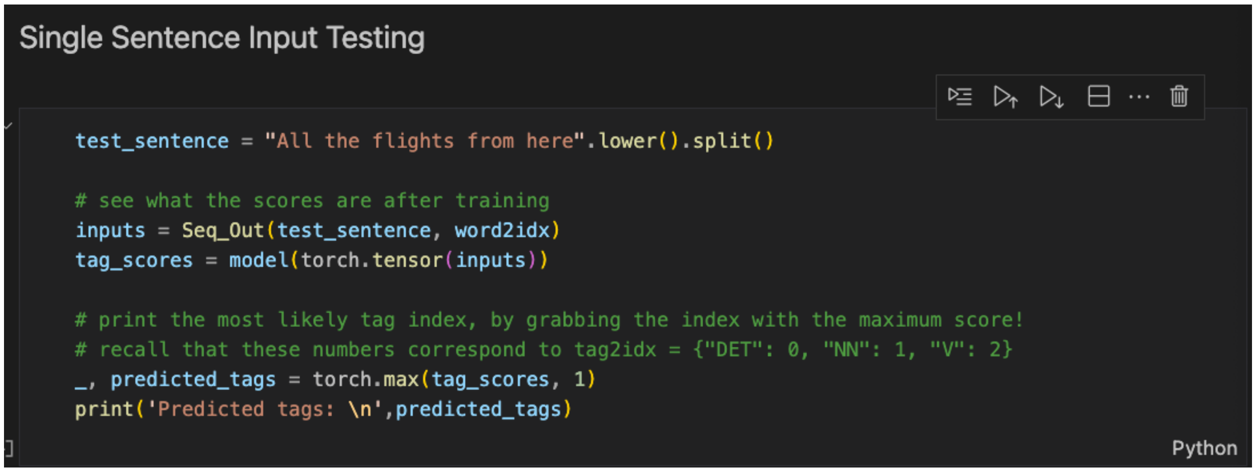


Loading the Pre Trained Model

```
Path = './POS_Saved_Model.pt'
model.load_state_dict(torch.load(Path))
```

Model Inputs:

→ Model will take tensor as input and provide tag_scores.


Single Sentence Input Testing

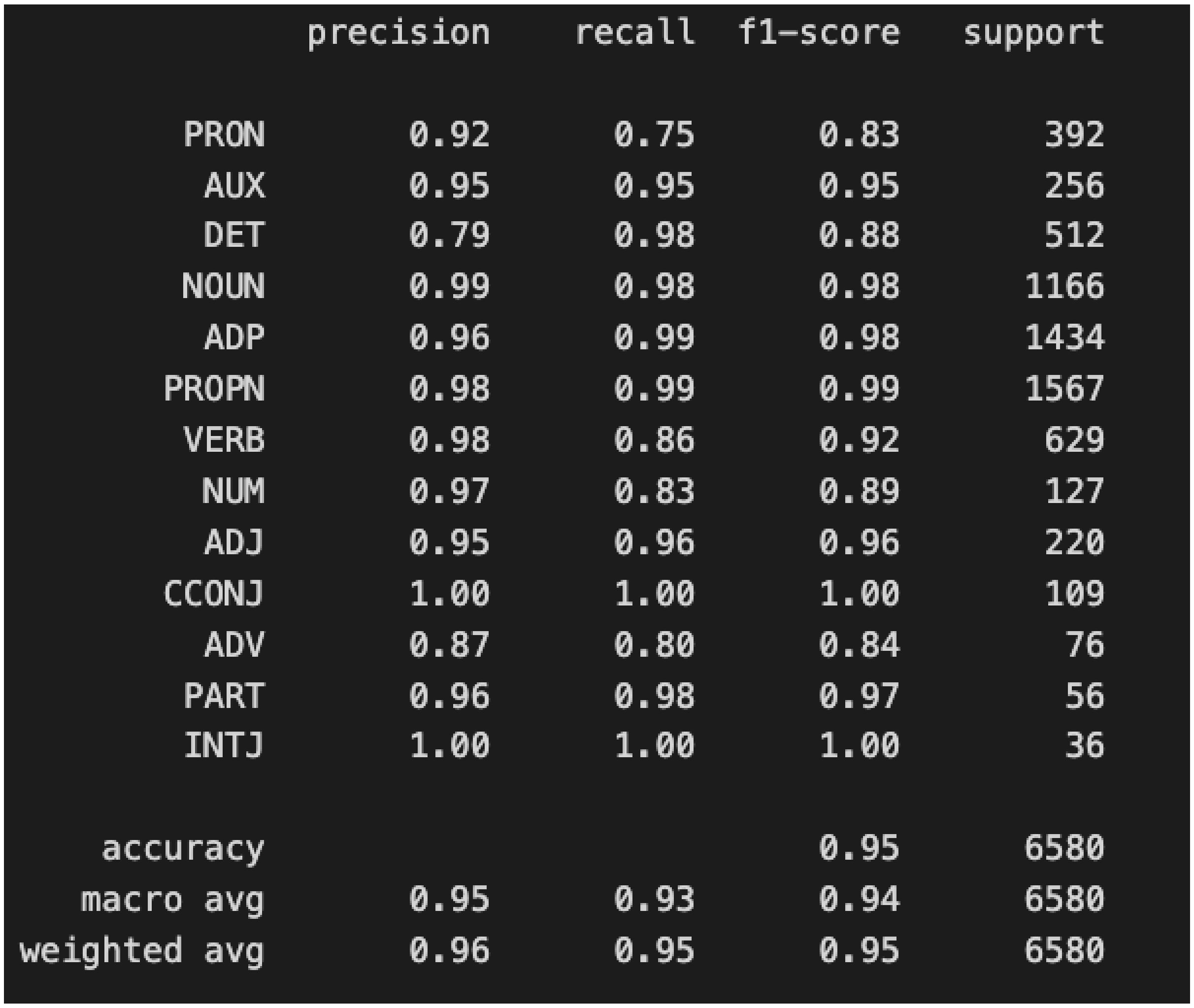
```
test_sentence = "All the flights from here".lower().split()

# see what the scores are after training
inputs = Seq_Out(test_sentence, word2idx)
tag_scores = model(torch.tensor(inputs))

# print the most likely tag index, by grabbing the index with the maximum score!
# recall that these numbers correspond to tag2idx = {"DET": 0, "NN": 1, "V": 2}
_, predicted_tags = torch.max(tag_scores, 1)
print('Predicted tags: \n',predicted_tags)
```

Observations From the Testing Data :

Precession / Recall / f1-score / Support



```
              precision    recall  f1-score   support

        PRON       0.92      0.75      0.83       392
         AUX       0.95      0.95      0.95       256
         DET       0.79      0.98      0.88       512
        NOUN       0.99      0.98      0.98      1166
         ADP       0.96      0.99      0.98      1434
       PROPN       0.98      0.99      0.99      1567
        VERB       0.98      0.86      0.92       629
         NUM       0.97      0.83      0.89       127
         ADJ       0.95      0.96      0.96       220
       CCONJ       1.00      1.00      1.00       109
         ADV       0.87      0.80      0.84        76
        PART       0.96      0.98      0.97        56
        INTJ       1.00      1.00      1.00        36

    accuracy                           0.95      6580
   macro avg       0.95      0.93      0.94      6580
weighted avg       0.96      0.95      0.95      6580
```

General Observations:

→Found out that Embedded and Hidden dimensions are most effective in range of 250 to 300

→Epoch is most effective at value 6-8 (very close) and as the value of epoch increase test accuracy decrease.

→ The point of the influx of dev accuracy tell about overfitting hence the max value point will give the most optimised value.