Name:-Shubham Kathiriya

Roll No:- 2023201050

Subject:-Intro to NLP

Assignment 2:- POS Tagging

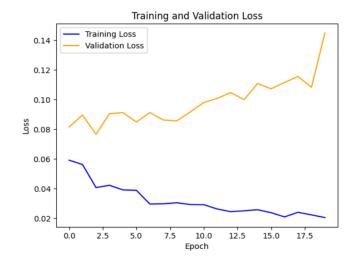
Hyperparameter tunning result:-

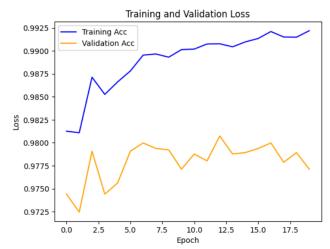
1. FFNN Result

• Configuration 1:-

```
p = 3
s = 2
BATCH_SIZE = 64
EPOCHS = 10
embedding_dim = 100
HIDDEN_LAYER_DIMENSION = 200

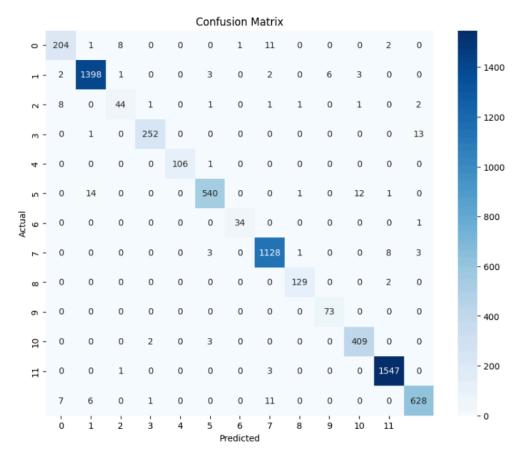
def forward(self, x):
    x = x.to(self.11.weight.dtype)
    out = self.11(x)
    out = self.relu(out)
    out = self.relu(out)
    out = self.relu(out)
    out = self.relu(out)
    out = self.lout)
    out = self.lout)
    out = self.lout)
```





For Validation:-

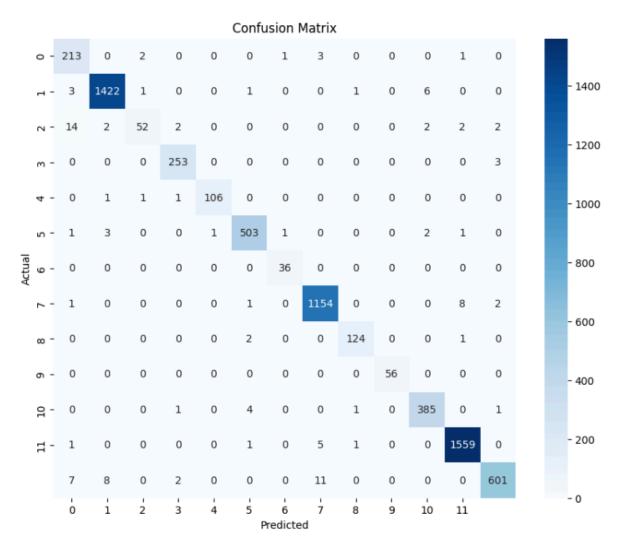
Validation Loss: 0.1448 Validation Accuracy: 0.9771 Validation Precision: 0.9993 Validation Recall: 0.9882 Validation F1 Score: 0.9937



For Test:-

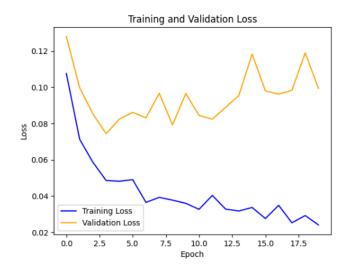
----- Test Data -----

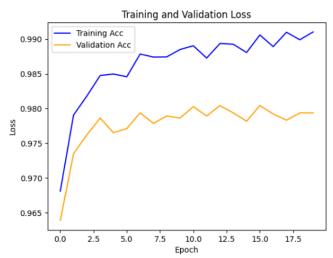
Test Loss: 0.1043
Test Accuracy: 0.9824
Test Precision: 1.0000
Test Recall: 0.9815
Test F1 Score: 0.9907



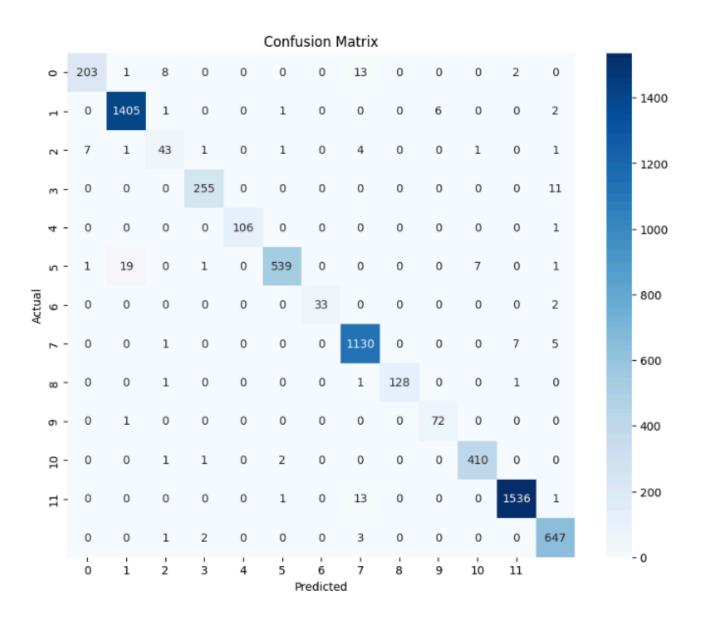
• Configuration 2:-

```
p = 2
s = 3
BATCH SIZE = 64
EPOCHS = 20
embedding dim = 100
HIDDEN LAYER DIMENSION = 100
def forward(self, x):
    x = x.to(self.ll.weight.dtype)
    out = self.11(x)
    out = self.relu(out)
    out = self.12(out)
    out = self.relu(out)
    out = self.13(out)
    out = self.relu(out)
    out = self.14(out)
    out = self.relu(out)
    out = self.15(out)
    return out
```



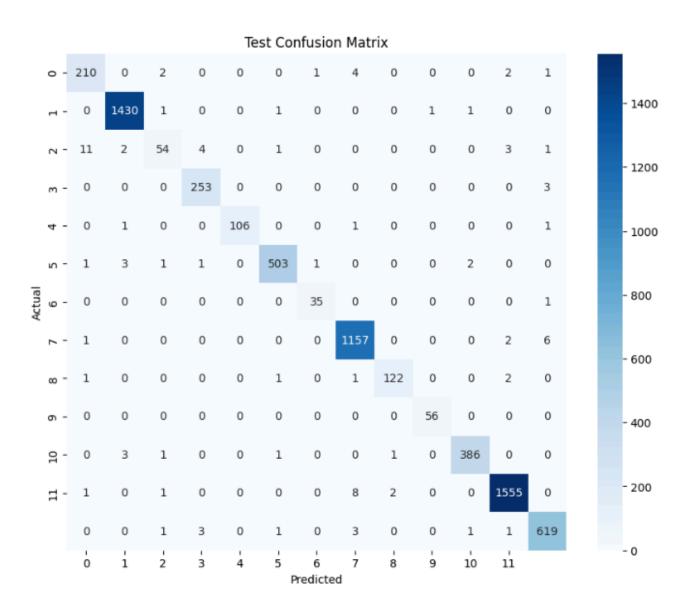


Validation Loss: 0.0993 Validation Accuracy: 0.9794 Validation Precision: 0.9993 Validation Recall: 0.9944 Validation F1 Score: 0.9969



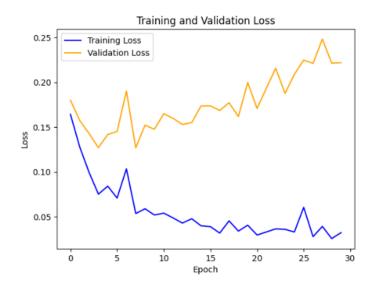
For Test Data:-

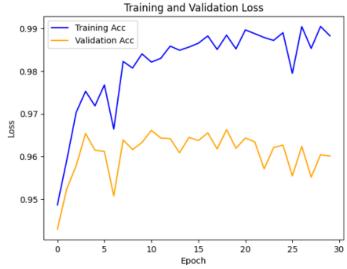
Test Loss: 0.0858 Test Accuracy: 0.9857 Test Precision: 1.0000 Test Recall: 0.9897 Test F1 Score: 0.9948



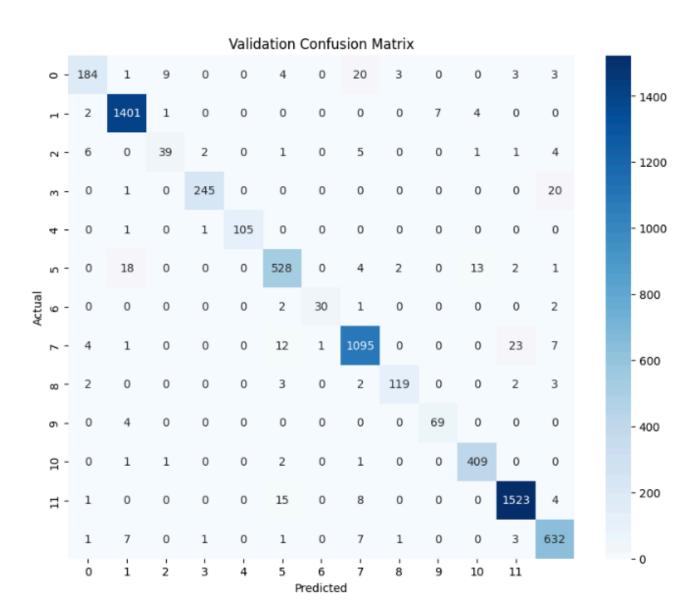
• Configuration 3:-

```
s = 5
BATCH SIZE = 64
round precision = 5
EPOCHS = 30
embedding_dim = 100
HIDDEN LAYER DIMENSION = 200
def forward(self, x):
        x = x.to(self.l1.weight.dtype)
        out = self.11(x)
        out = self.leaky_relu(out)
        out = self.12(out)
        out = self.leaky_relu(out)
        out = self.13(out)
        out = self.leaky_relu(out)
        out = self.14(out)
        return out
```





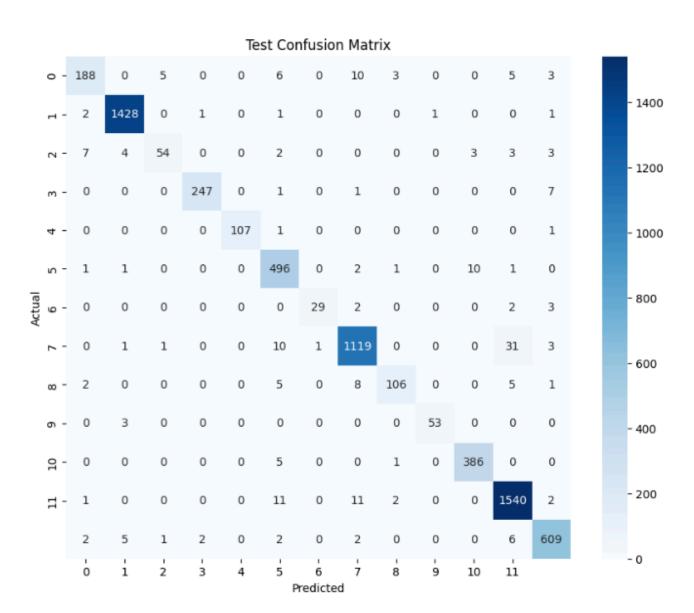
Validation Loss: 0.2217 Validation Accuracy: 0.9601 Validation Precision: 0.9993 Validation Recall: 0.9890 Validation F1 Score: 0.9941



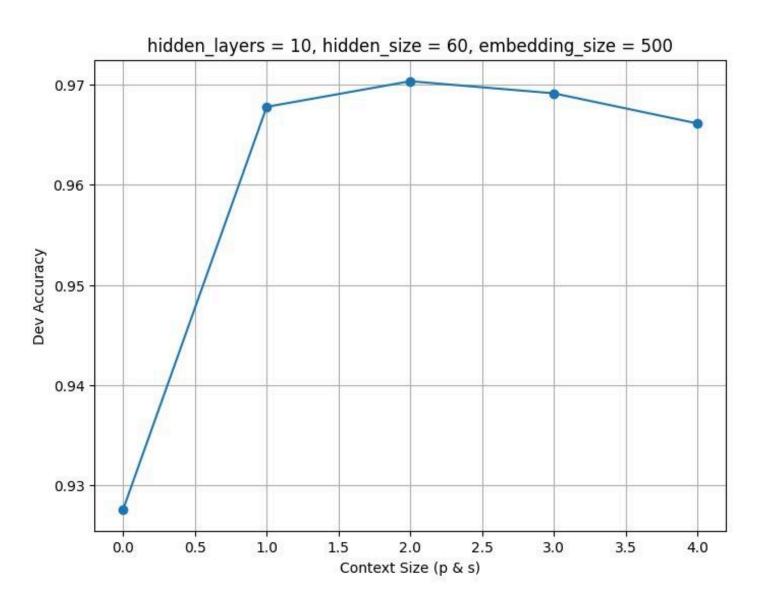
For test data:-

----- Test Data -----

Test Loss: 0.2062 Test Accuracy: 0.9669 Test Precision: 1.0000 Test Recall: 0.9897 Test F1 Score: 0.9948



For Various P and S for configuration:-



2. <u>LSTM</u>

• Confriguation 1:-

```
BATCH_SIZE = 64

round_precision = 5

EPOCHS = 50

embedding_dim = 100

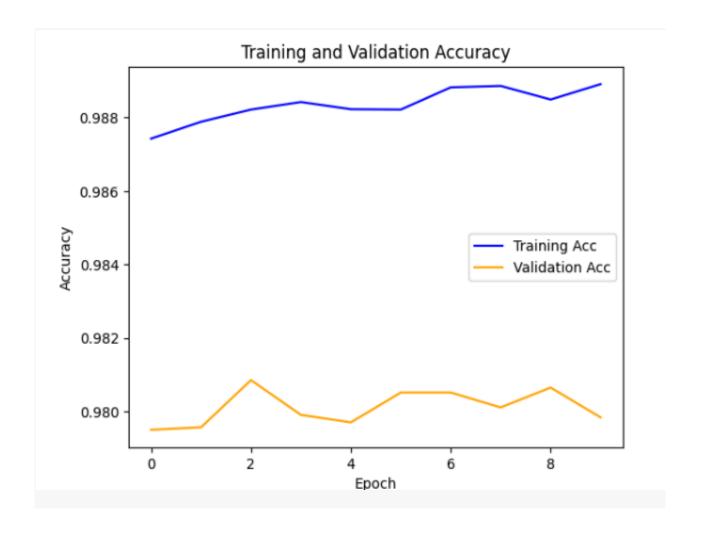
hidden_layer_dim = 200

learning_rate = 1e-3

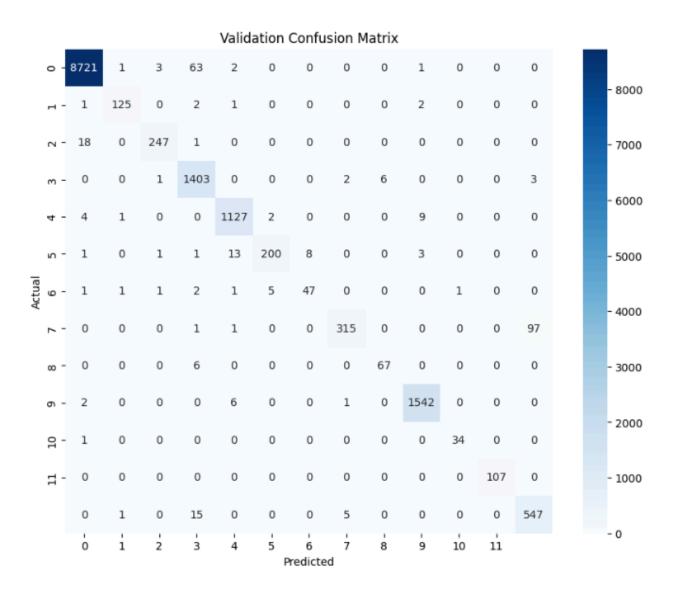
epochs = 10

Bidirectional = False

Num_layer = 1
```



Validation Accuracy: 0.9798 Validation Precision: 0.9921 Validation Recall: 0.9921 Validation F1 Score: 0.9921



For Test Data:-

Test Accuracy: 0.9832 Test Precision: 0.9918 Test Recall: 0.9918 Test F1 Score: 0.9918

Test Confusion Matrix																	
	0	- 8	3714	1	2	53	5	3	1	0	0	0	2	0	0		
	П	-	1	121	1	1	2	0	0	0	0	1	0	0	0		- 8000
	2	-	10	0	246	0	0	0	0	0	0	0	0	0	0		- 7000
	m	-	1	0	0	1425	0	0	1	1	1	0	0	0	5		
	4	-	4	0	0	1	1155	0	0	0	0	6	0	0	0		- 6000
	2	-	2	0	0	0	3	209	3	0	0	2	1	0	0		- 5000
Actual	9	-	0	0	1	0	4	8	60	0	0	3	0	0	0		
4	7	-	0	1	0	1	0	0	0	290	0	0	0	0	100		- 4000
	œ	-	0	0	0	1	0	0	0	0	55	0	0	0	0		- 3000
	6	-	1	2	0	1	2	2	0	0	0	1559	0	0	0		
	10	-	2	0	0	0	0	0	0	0	0	0	34	0	0		- 2000
	==	-	0	0	0	0	0	0	0	0	0	0	0	109	0		- 1000
			0	0	0	2	1	0	0	0	0	1	0	0	508		
			0	i	2	3	4	5 P	6 redicted	, 1	8	9	10	11			- 0

• Confriguation 2:-

```
BATCH_SIZE = 64

round_precision = 5

EPOCHS = 50

embedding_dim = 100

hidden_layer_dim = 200

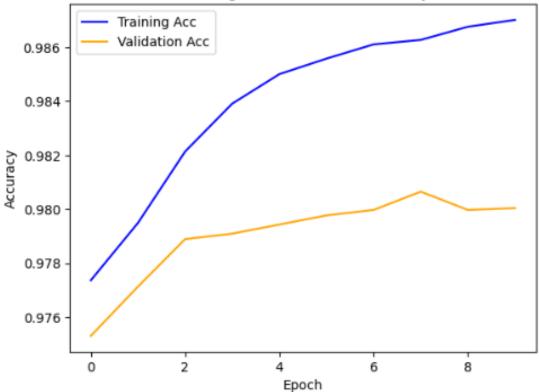
learning_rate = 1e-3

epochs = 10

bidirectional = True

num_layer = 2
```





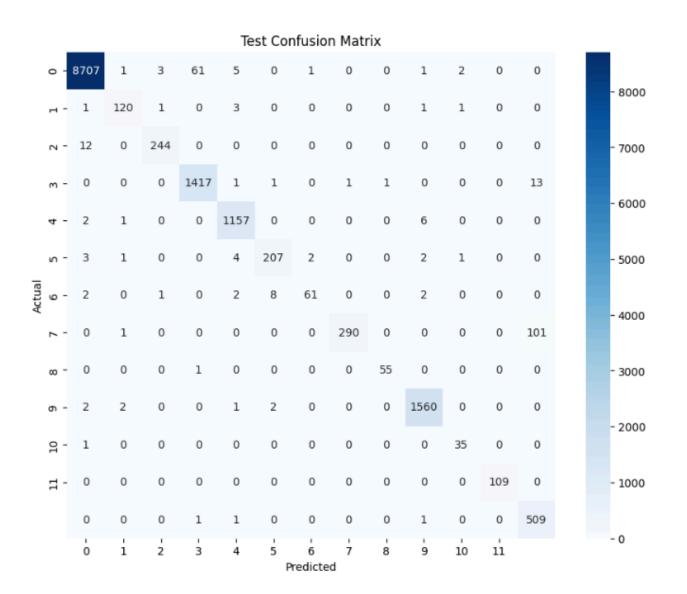
Validation Accuracy: 0.9800 Validation Precision: 0.9922 Validation Recall: 1.0000 Validation F1 Score: 0.9961

· ------

Validation Confusion Matrix																
	0 -	8717	1	3	65	4	0	0	0	0	1	0	0	0		
	н -	0	127	0	0	1	0	0	0	0	3	0	0	0	- 80	00
	۲ -	18	0	248	0	0	0	0	0	0	0	0	0	0	- 700	00
	m -	0	0	0	1398	0	0	1	0	8	0	0	0	8		
	4 -	5	1	0	0	1126	1	0	0	0	10	0	0	0	- 60	00
	w -	1	0	0	0	14	198	10	0	0	3	1	0	0	- 500	00
Actual	φ-	3	1	0	0	3	3	49	0	0	0	0	0	0	40.	00
	7 -	0	0	0	0	1	0	0	315	0	0	0	0	98	- 400	00
	∞ -	0	0	0	4	0	0	0	0	69	0	0	0	0	- 300	00
	ი -	3	0	0	0	5	1	0	1	0	1541	0	0	0	- 20	00
	영 -	1	0	0	0	0	0	0	0	0	0	34	0	0	- 20	00
	:: -	0	0	0	0	0	0	0	0	0	0	0	107	0	- 100	00
		0	1	0	7	0	0	0	4	0	0	0	0	556		
		ó	i	2	3	4	5	6 Predicted	7	8	9	10	ıΊ		- 0	

For Test Data:-

Test Accuracy: 0.9823 Test Precision: 0.9917 Test Recall: 0.9917 Test F1 Score: 0.9917



• Confriguation 3:-

```
BATCH_SIZE = 64

round_precision = 5

EPOCHS = 10

embedding_dim = 300

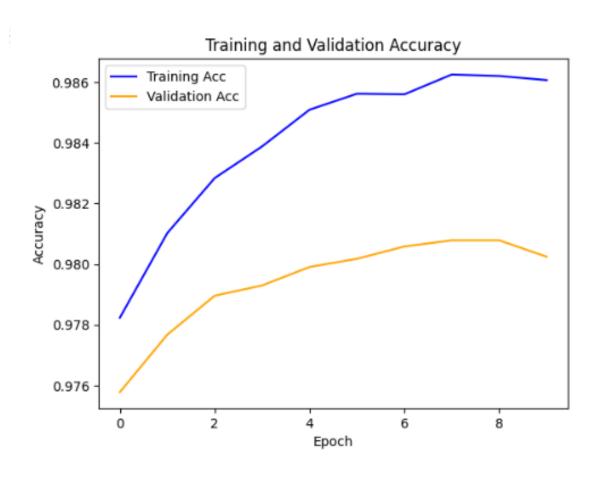
hidden_layer_dim = 400

learning_rate = 1e-3

epochs = 10

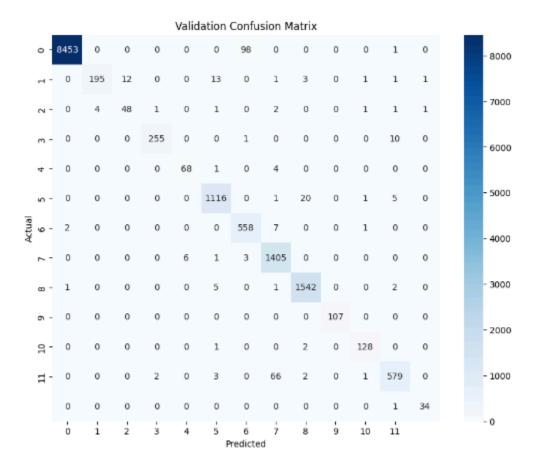
bidirectional = True

num_layer = 4
```



For Validation Accuracy:-

Validation Accuracy: 0.9802 Validation Precision: 1.0000 Validation Recall: 1.0000 Validation F1 Score: 1.0000



For Test Data :-

----- Test Data -----

Test Accuracy: 0.9811 Test Precision: 1.0000 Test Recall: 1.0000 Test F1 Score: 1.0000

	_	Test Confusion Matrix															
	0 -	8439	0	0	0	0	0	104	0	0	0	1	0	0			8000
	٦ -	0	207	3	0	0	4	0	0	3	0	0	2	1			
	۲ -	0	7	62	1	0	1	0	0	3	0	0	2	0			7000
	m -	0	0	0	244	0	0	0	0	0	0	0	12	0			6000
	4 -	0	0	0	0	55	0	0	1	0	0	0	0	0			
	ω -	0	0	0	0	0	1148	0	1	14	0	1	2	0		-	5000
Actual	g -	0	0	0	0	0	0	509	1	1	0	0	1	0			4000
_	7 -	0	0	0	0	2	0	9	1422	0	0	0	1	0			4000
	_∞ -	0	1	0	0	0	0	0	0	1560	0	2	4	0		-	3000
	ი -	0	0	0	0	0	0	0	0	0	109	0	0	0			2000
	10	0	0	0	1	0	2	1	0	2	0	120	1	0			2000
	Π-	0	1	1	10	0	7	1	62	2	0	1	542	2		-	1000
		0	0	0	0	0	0	0	0	0	0	0	0	36			
		Ó	i	2	3	4	5 Pi	6 redicte	7 d	8	9	10	11				0

Analysis:-

Analytical Result of Feed Forward Neural Network POS Tagger:

- <u>Advancing Model Complexity:</u> By increasing parameters such as embedding dimension and hidden layer sizes, there's a slight uptick in development accuracy, both in macro and weighted averages.
- <u>Test Accuracy Enhancement:</u> Augmenting model complexity also leads to a noticeable boost in test accuracy.
- <u>Influence of Activation Functions:</u> Switching from ReLU to Tanh activation function causes a slight dip in accuracy, favoring ReLU for better performance.
- <u>Context Window Impact:</u>
 - The context window size significantly affects model accuracy.
 - The lowest accuracy is observed when both preceding and succeeding window sizes are 0, while the highest accuracy is achieved when both are 1.
 - Further increase in window size results in a minor decrease in accuracy.
- <u>Activation function:</u> ReLu gives the best performance and leaky ReLu gives slightly less good performance than ReLu and softmax gives even more noticeable lesser accuracy

• Analytical Result of LSTM (RNN) POS Tagger:

- <u>Effect of Layer Count:</u> Increasing the number of layers extends training time without notable accuracy improvement.
- <u>Bidirectionality Effect:</u> Activating bidirectionality notably boosts accuracy, showcasing its importance in capturing context effectively.

- <u>Epoch Count Impact</u>: Despite increasing epochs, accuracy doesn't improve significantly due to bidirectionality's substantial performance enhancement.
- <u>Activation Functions:</u> Tanh function is utilized internally within LSTM architecture, while SoftMax is applied at the output layer as the final activation function.