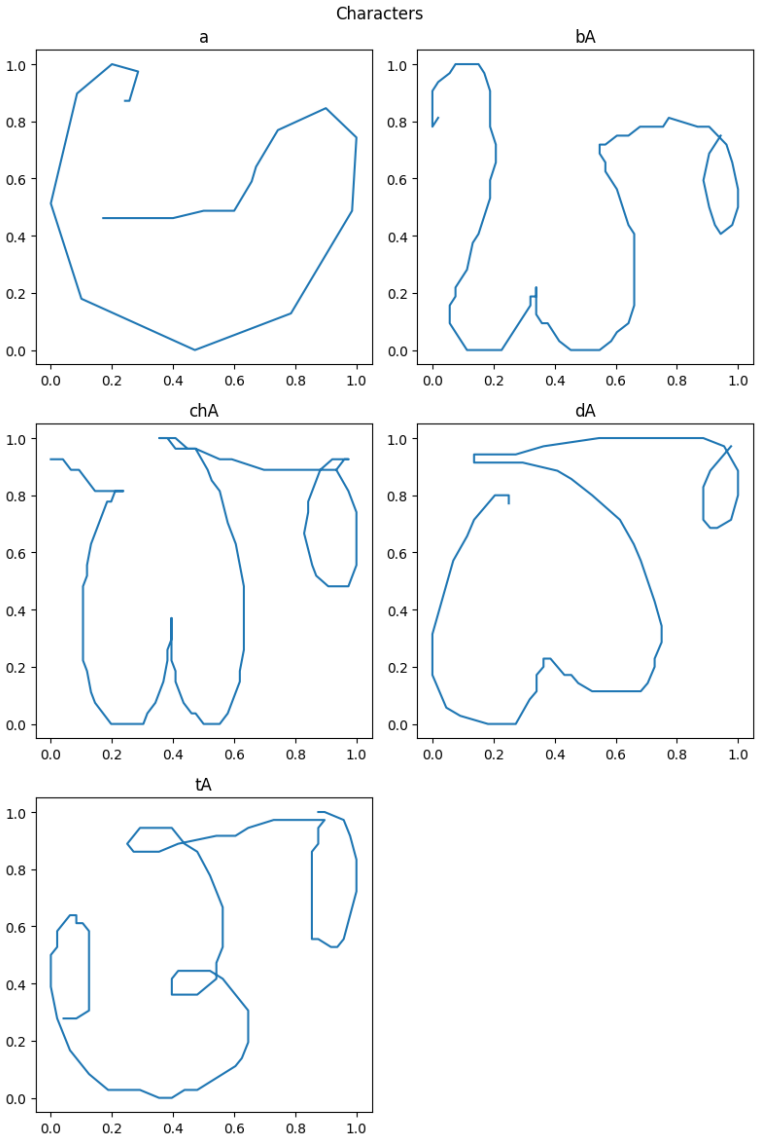
CS671 Assignment-4

GROUP - 8

Akash Karnatak (B19147)
Laishram Ponghtangamba (B19011)
Abhishek Mishra (B19231)

Handwritten character dataset:

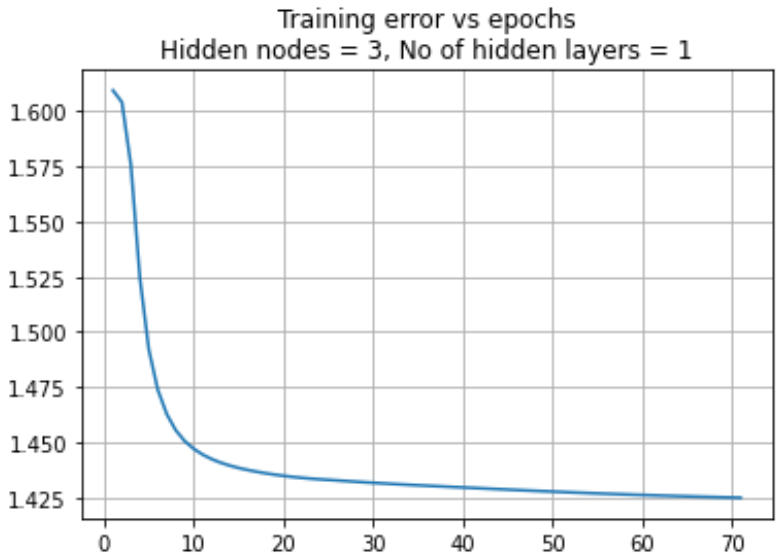
Below is a plot of 5 characters included in our training and testing data.



RNN

Data was normalized to the range 0 to 1 and models with several architectures were trained. The model has `num_layers` stacked RNN layers with `hidden_size` neurons in each layer and an output layer with `softmax` activation function. Below are the different architectures with their training and test accuracy.

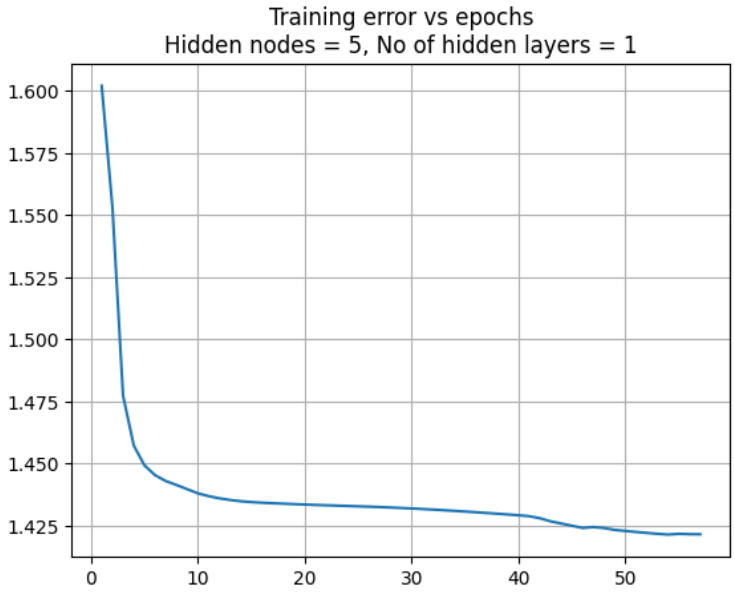
- **`num_layers` = 1 and `hidden_size` = 3**



Training accuracy = 44.47

Test accuracy = 46.0

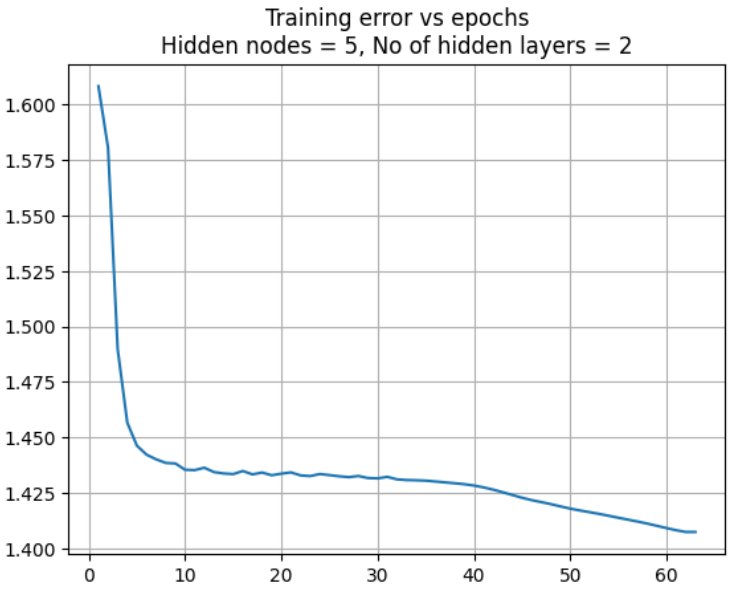
- **`num_layers` = 1 and `hidden_size` = 5**



Training accuracy = 48.25

Test accuracy = 46.0

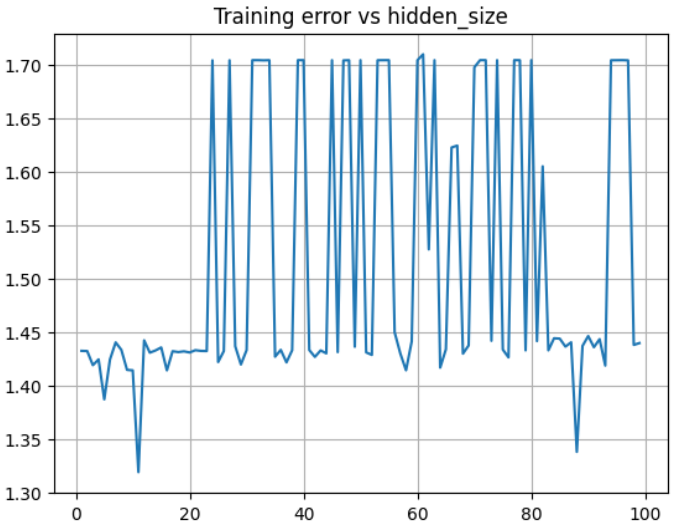
- **`num_layers` = 2 and `hidden_size` = 5**



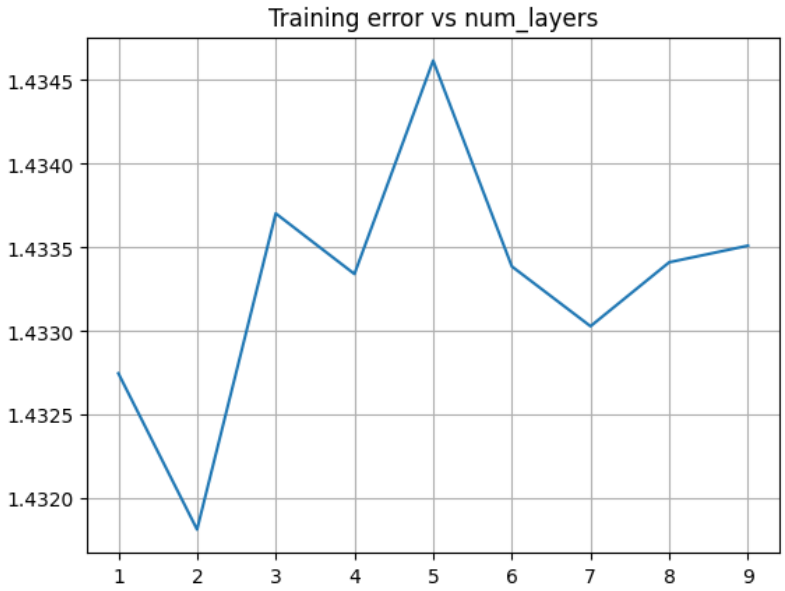
Training accuracy = 49.42

Test accuracy = 48.0

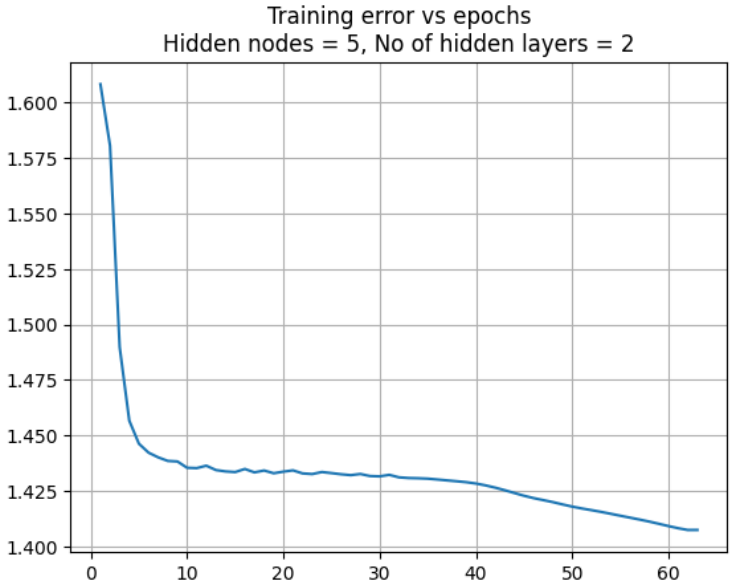
`hidden_size` was varied from 1 to 99 keeping the `num_layers` fixed to 1. Following is the plot of `training error vs hidden_size`



Out of all, `hidden_size` = 5 seems to perform better. So keeping `hidden_size` fixed to 5, `num_layers` was varied from 1 to 9. Following is the plot of `training error vs num_layers`



The best architecture was chosen to be the model with num_layers = 2 and hidden_size = 5.



Training accuracy = 49.42

Test accuracy = 48.0

Confusion matrix:-

		Predicted class				
		1	**3**	**4**	**8**	**9**
Actual class	**1**	69	0	0	0	0
	3	0	3	52	1	11
	4	0	1	49	0	20
	8	0	0	52	0	17
	9	0	1	34	0	34

LSTM

The model has `num_layers` stacked LSTM layers with `hidden_size` neurons in each layer and an output layer with `softmax` activation function. Below are the different architectures with their training and test accuracy.

- **`num_layers` = 1 and `hidden_size` = 13**

Training accuracy = 59.30

Test accuracy = 61.0

- **`num_layers` = 1 and `hidden_size` = 26**

Training accuracy = 68.60

Test accuracy = 75.0

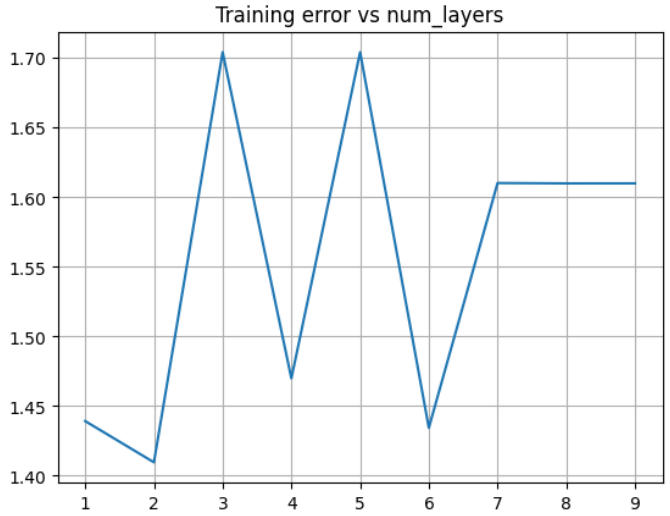
- **`num_layers` = 2 and `hidden_size` = 26**

Training accuracy = 80.81

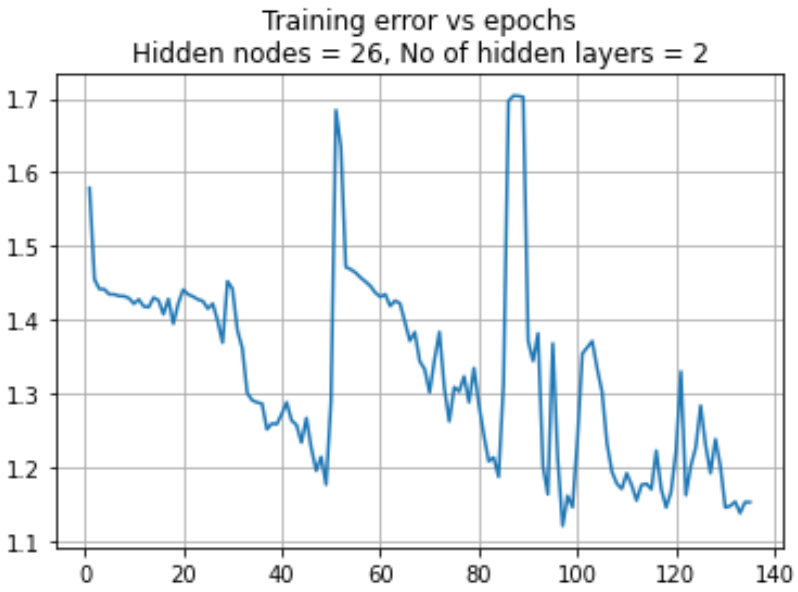
Test accuracy = 83.0

`hidden_size` was varied from 1 to 99 keeping the `num_layers` fixed to 1. Following is the plot of `training error vs hidden_size`

Out of all, `hidden_size` = 26 seems to perform better. So keeping `hidden_size` fixed to 26, `num_layers` was varied from 1 to 9. Following is the plot of `training error vs num_layers`



The best architecture was chosen to be the model with num_layers = 2 and hidden_size = 26.



Training accuracy = 80.81

Test accuracy = 83.0

Confusion matrix:-

		Predicted class				
		1	**3**	**4**	**8**	**9**
Actual class	**1**	68	1	0	0	0
	3	0	64	3	0	0
	4	0	3	67	0	0
	8	0	0	0	69	0
	9	0	24	16	19	10

Consonant Vowel (CV) segment dataset:

RNN :

a. Average Error Vs Epochs

We tried various RNN units ranging from 1 to 46 . In each case input size to each of the RNN cells was 39 (39 dimensional data), output was taken from a 5 dimensional softmax layer and 1 hidden layer consisting of RNN units.
The average error vs epochs plot for various models we tried is as follows :

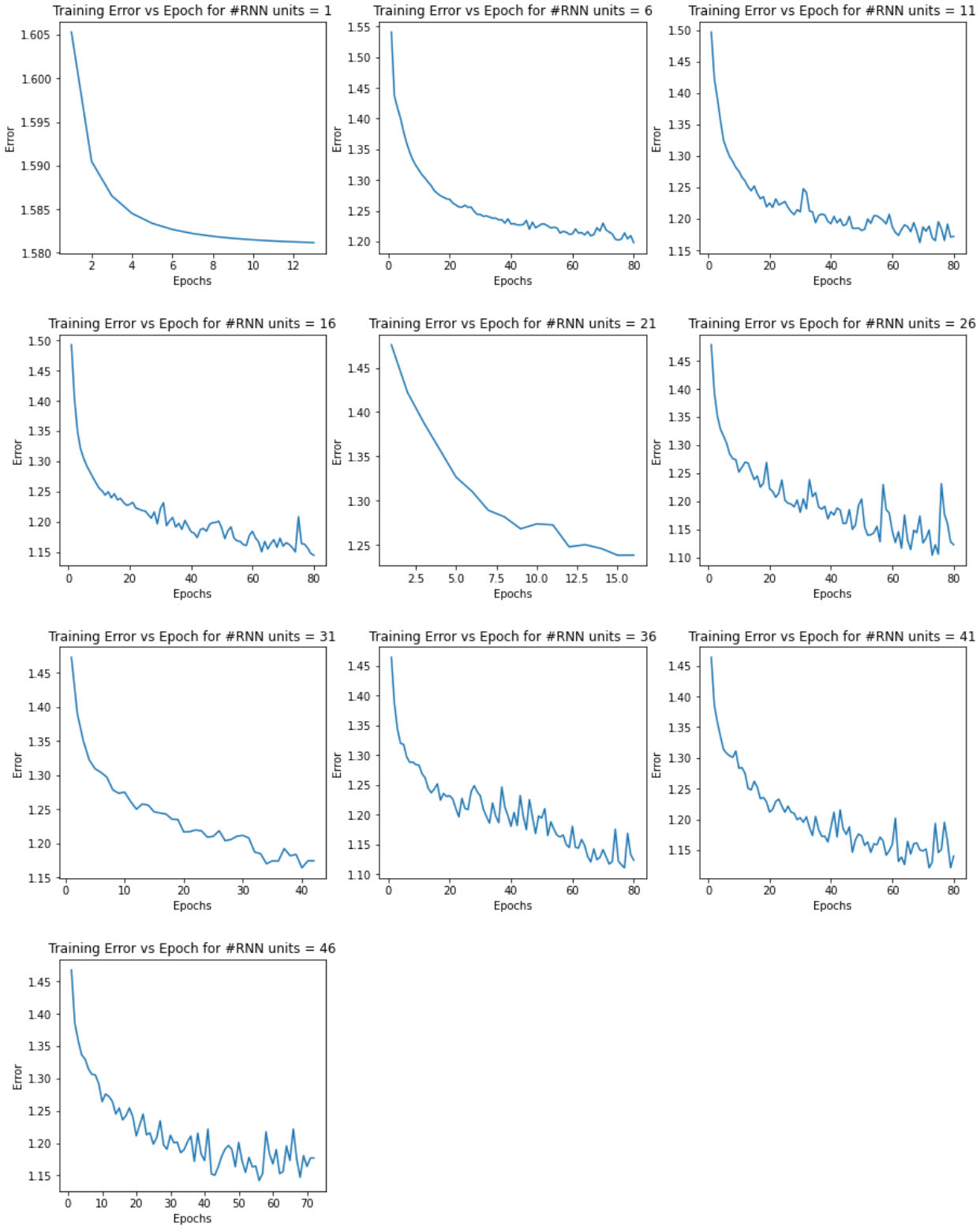


Fig. : Average Error vs Epochs for various tried models

b. Training And Test Accuracies

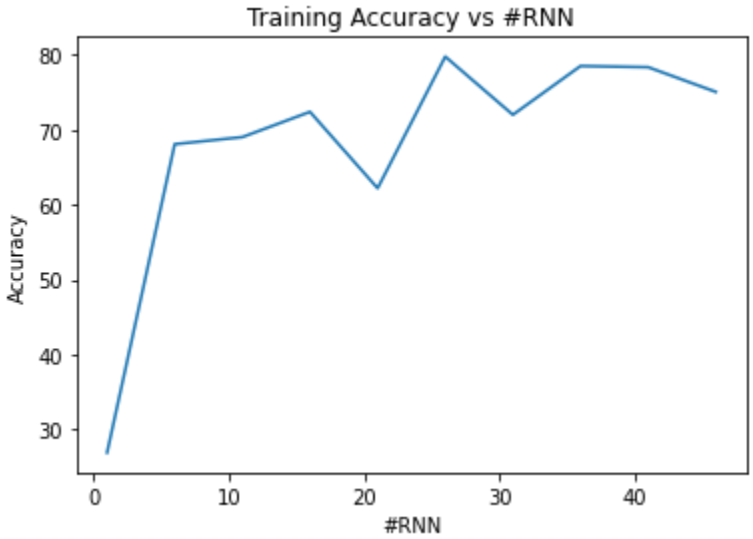


Fig. : Training Accuracies for various tried model

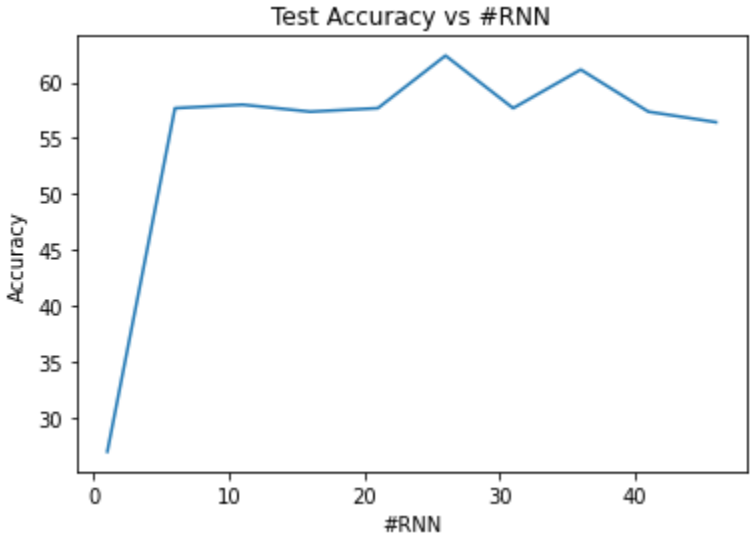


Fig. : Test Accuracies for various tried model

c. Best Chosen Model

Based on the above accuracies graph we concluded that our model was performing best when the number of rnn units were 26 .

The training data accuracy in this case was found to be 77.77% while the test data accuracy was found to be 62.38% .
The confusion matrix that was obtained is as follows :

```
{'nii': 0, 'pa': 1, 'rI': 2, 're': 3, 'sa': 4}
```

Training Data :

```
array([[112,   2,  62,  12,   6],
       [  6, 167,  11,  12,  16],
       [ 25,   4, 264,  17,   8],
       [ 21,   8,  78,  87,  14],
       [  6,  30,  19,   8, 280]])
```

Fig. : Train Data Confusion Matrix

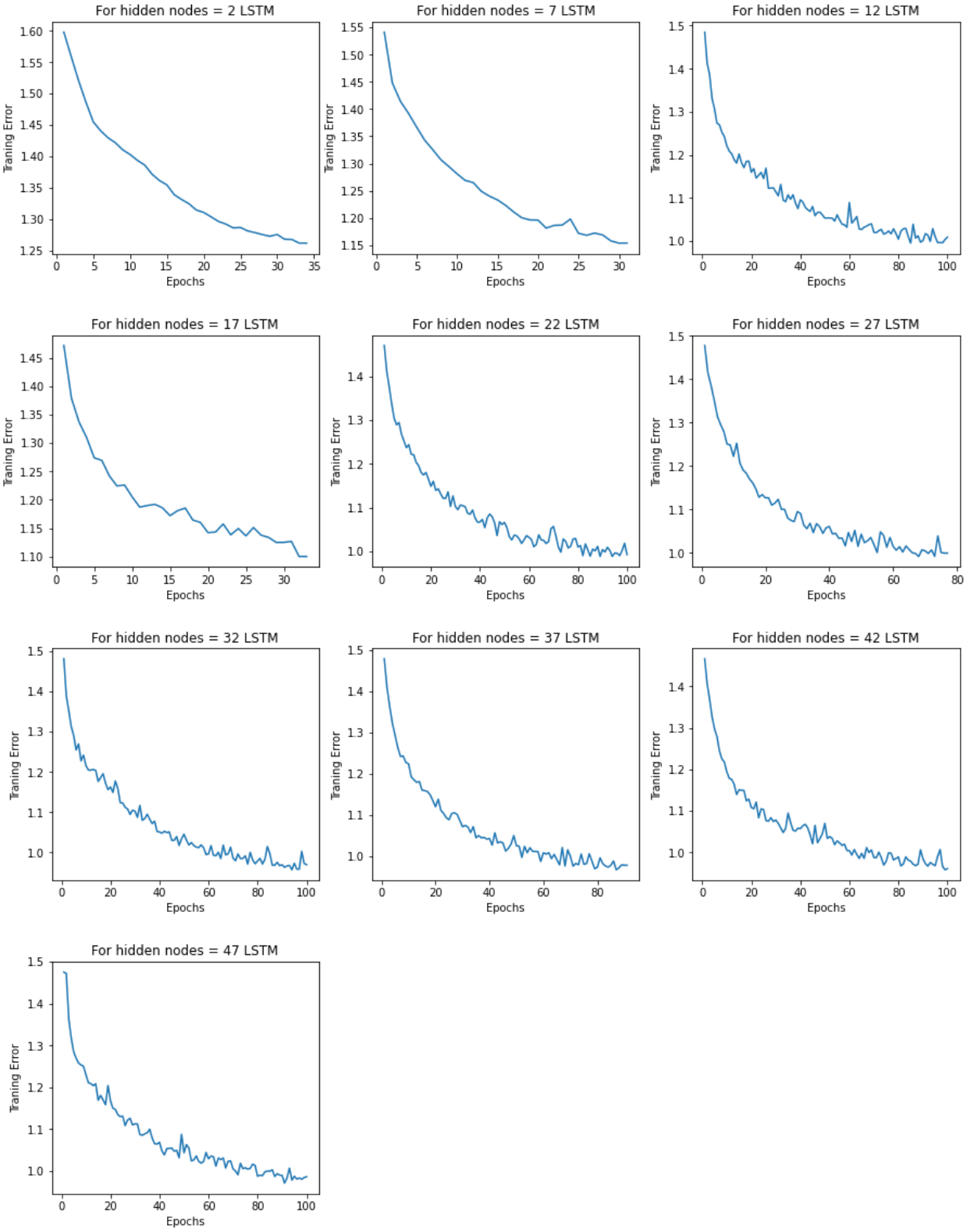
Test Data :

```
array([[23,  3, 18,  2,  2],
       [ 1, 34,  4,  1, 13],
       [10,  0, 65,  3,  2],
       [11,  3, 25, 10,  3],
       [ 0, 10,  3,  0, 73]])
```

Fig. : Test Data Confusion Matrix

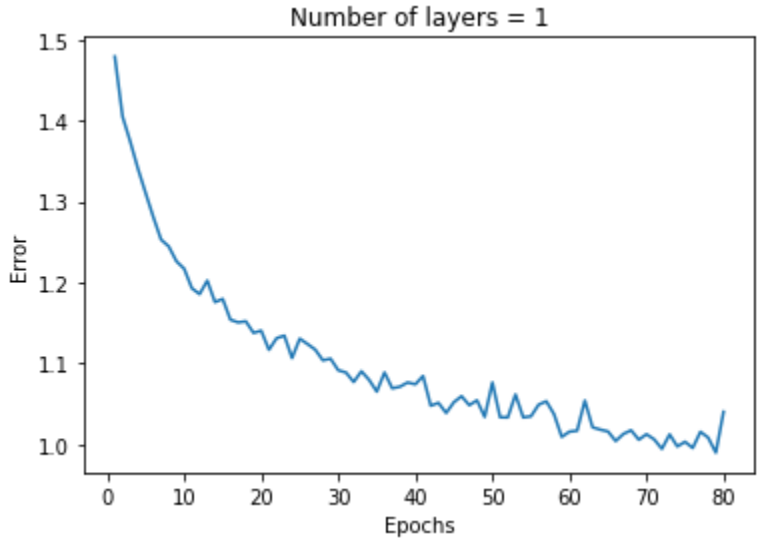
LSTM:
Keeping the number of hidden layers as 1 , the LSTM model is trained for different numbers of hidden nodes ranging from 2 to 50 with a gap of 5.

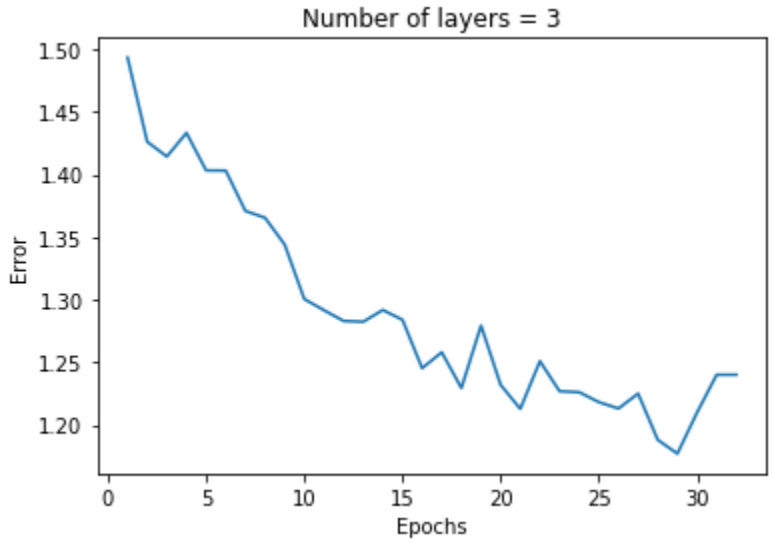
Training Error vs number of epochs for different set of hyperparameters:
Number of hidden layer = 1

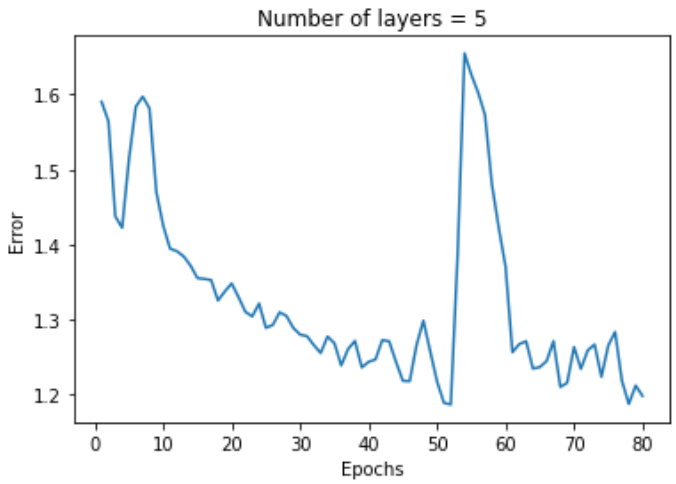


The number of hidden nodes which gives the highest accuracy of 72% is 32.

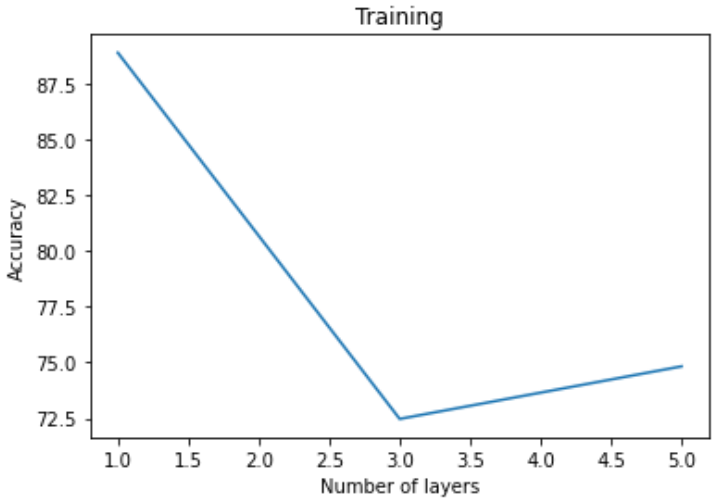
Again ,keeping the number of hidden nodes as 32, the model is again trained for different numbers of hidden layers ranging from 1 to 7 with a gap of 2. The error vs epochs of training data for each number of hidden layers is recorded and plotted as a graph.



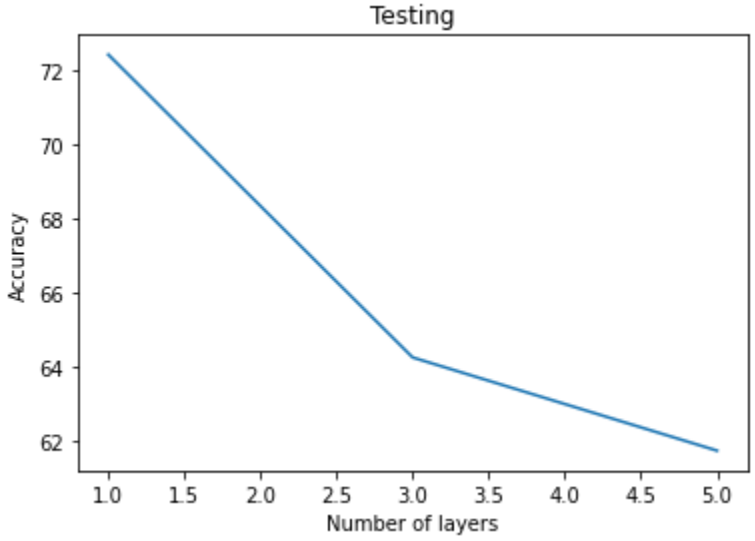




The accuracy of training data for different numbers of hidden layers.



The accuracy of test data for different numbers of hidden layers.



So, the best architecture is

Number of hidden layers = 1
Number of hidden nodes = 32
For this architecture, the classification layer is kept with only one layer .

Test accuracy for the best architecture is 72.41 %
Training accuracy for the best architecture is 89.88 %

```
{'nii': 0, 'pa': 1, 'rI': 2, 're': 3, 'sa': 4}
```

The confusion matrix of test data:

```
array([[35,  3,  7,  3,  0],
       [ 0, 42,  1,  0, 10],
       [ 7,  1, 62,  7,  3],
       [ 2,  3, 27, 18,  2],
       [ 0,  6,  3,  3, 74]])
```