NEWS-TEXT CLASSIFICATION BASED ON A WEIGHTED RNN

Aviral Sethi – 2016B3A70532P

Manav Kaushik – 2016B3A30472P

Neelabh Sinha – 2016B5A80600P

OBJECTIVE

- To implement a model which classifies news-text using a **weighted RNN**
- To do a comparative study of its performance with the performance of LSTM and Bidirectional LSTM
- To show classification results of 15 such documents

Source:

W-RNN: News text classification based on a weighted RNN

(Wang, Gong, Song, Oct 2019)

Need for text classification

On the account of certainty and comprehensibility of its expression, text has become a popular way of storing information. Thus, text classification is an important research direction.

Applications of text classification –

- **News classification**
- Emotional analysis
- Answering question system
- Classify blog/tweet of people into various categories

Current Challenges in Text Classification -

- Simplifying text into bag of words(BOW) ignores the relationship between semantic units.
- General dimension of document representation is high resulting in semantic sparseness.
- Problem of vanishing gradients and long term dependencies.

Contributions of this model

- Replacing the BOW technique by Word2Vec, thus, reducing the dimensions effectively and solving the semantic sparseness problem.
- Obtaining the intermediate word vectors through units of LSTM and weighing them individually to obtain a document vector.
- Introduce the WRNN classification process in detail and classify the above document representation using a neural network.
- Compare the effectiveness of WRNN against traditional techniques.

Experimental Setup and Pre-processing

- The data set is obtained from qwone website and is split into 90% and 10%, which is used as training and testing data respectively

- It contains news articles across 20 labelled categories

- Using the *text_to_word_sequence* as available in keras, all the documents were tokenized and thus :-
  - Punctuations were removed
  - Entire text was converted to lowercase
  - Delimiter= ‘ ’

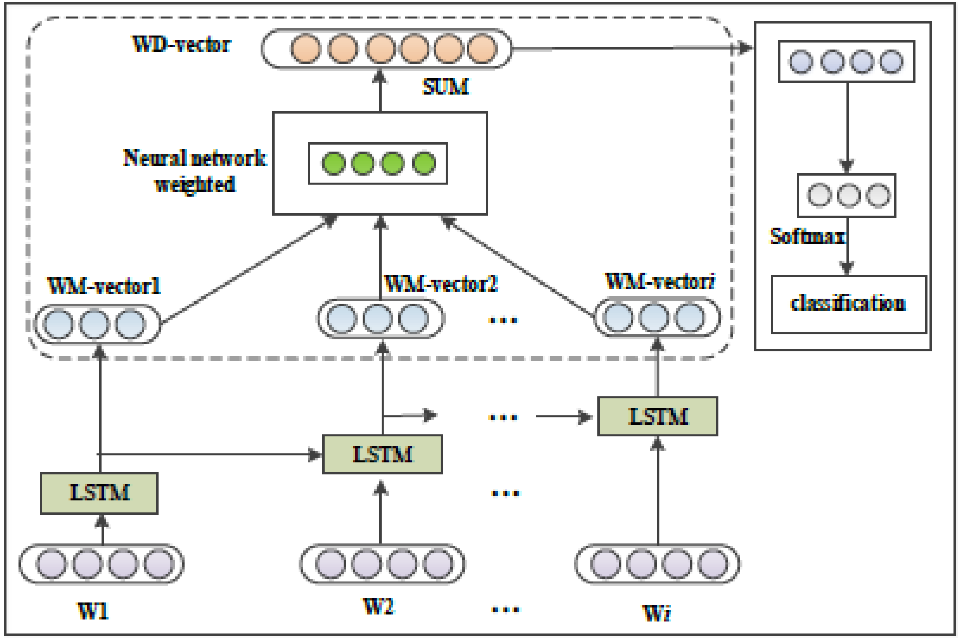
- The above created list of lists of documents in the dataset was further fed to the word2vec model

Developing Word2vec Model

- Word2vec as available from *Gensim* library was used to build the vocabulary for the model
- Following were the parameters used for training the Word2vec model :-
  - vector_size = 200
  - Min_count=5 (words with frequency less than this were ignored)
  - Window_size = default

Vocabulary size as per our training = 43,494
Vocabulary size as mentioned in the paper = 40,439

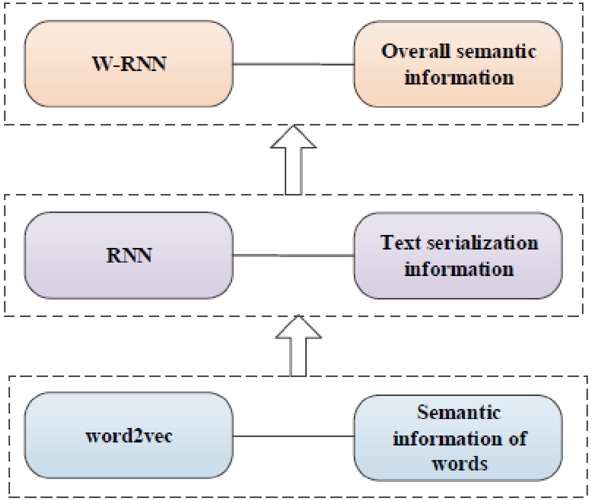
Model Architecture



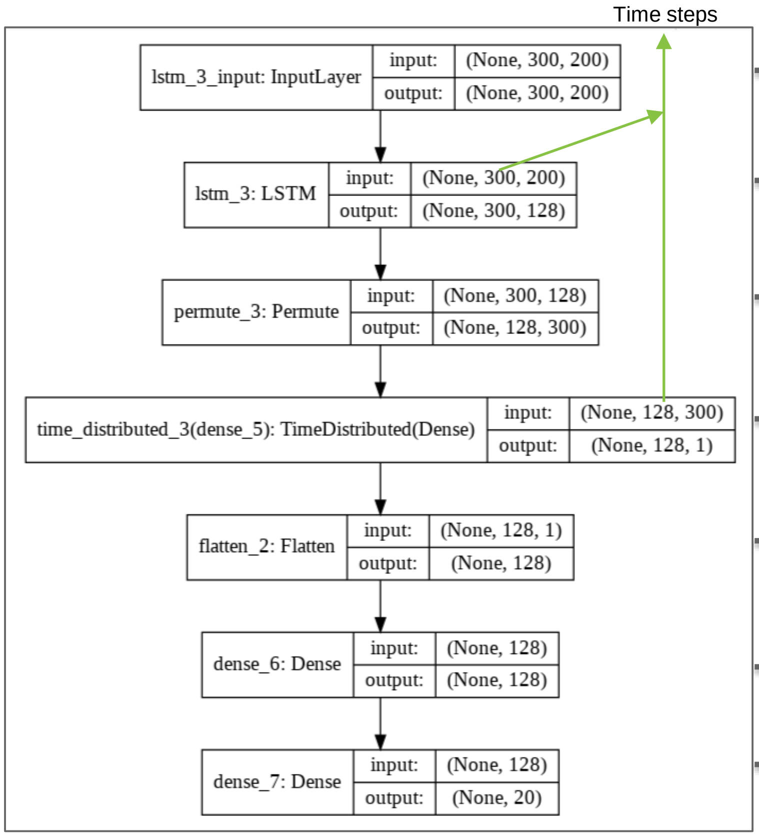
$$WD\text{-}vector = \sum_{i=1}^{seq-length} w_i * WM\text{-}vector_i$$

How is it better than standard LSTM network?

- Although LSTM can better learn textual information and selectively record semantic relationship, it still loses some of the valid information.
- W-RNN pays more attention to important vocabulary information that has positive effects on classification, and reduces attention to unimportant words during iterative training
- Weighing WM-vectors can capture central semantics of the document, and further extract information from the complete paragraph



Our Implementation



Input from Word2Vec (S_L =300)

Used to read the documents by keeping semantic relationship intact (Dropout used)

Used to take transpose of the output matrix so that Time-Distributed layer operates on i_{th} neuron of WM-vector of each word

$$\text{WD-vector} = \sum_{i=1}^{seq-length} w_i * \text{WM-vector}_i$$

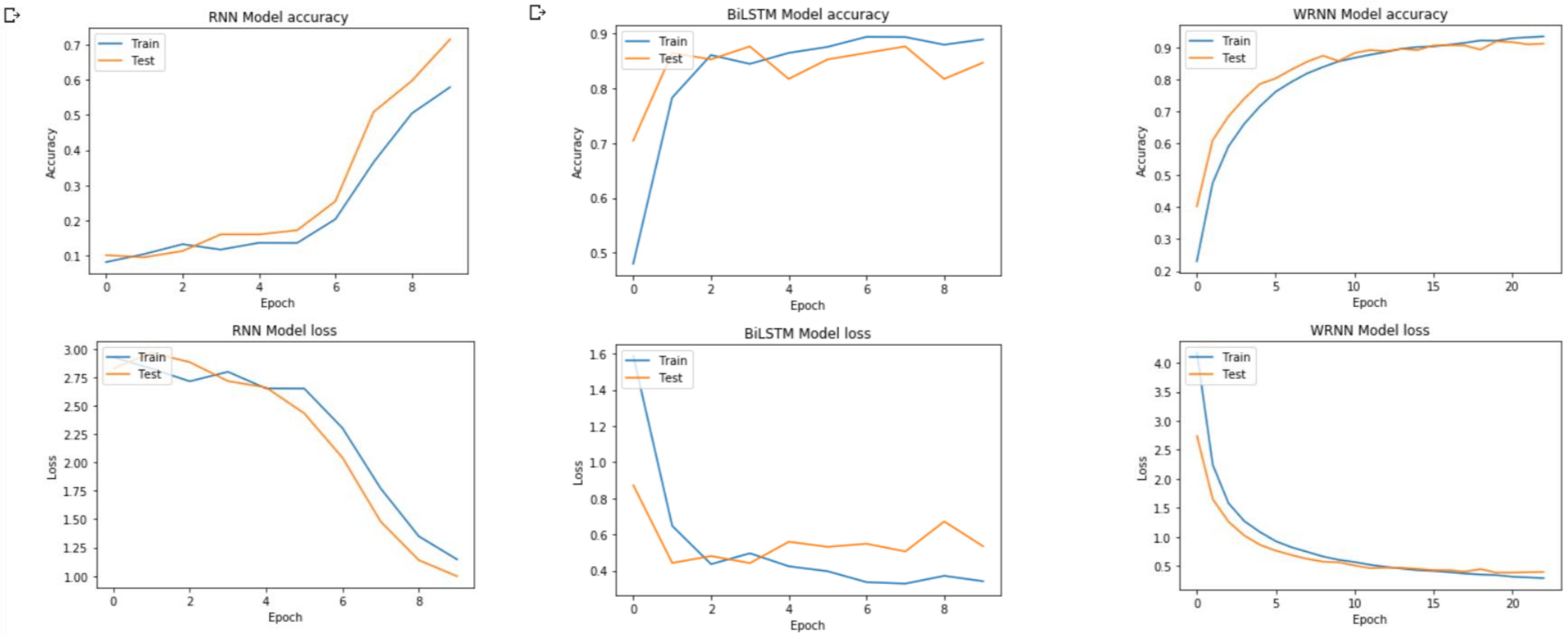
To make the output one dimensional to feed to the dense layer

1st dense layer (hidden layer) for classification

Output layer (20 one-hot neurons for 20 categories) – Regularization (l1=0.01, l2=0.01)

* Red font colour/fill – architectures different (but used for same purpose, or for improvement) from ones specified in the paper

Results (Accuracy and Loss Function)



All models were run using early-stopping as call-back w.r.t. validation loss

Confusion Matrices and Metrics – LSTM Model

```
Accuracy: 0.678125
[[47  0  0  0  0  0  0  0  1  1  0  0  1  8  0 38  2  0  2  0]
 [ 0 30  8  6  2 32  3  0  0  1  0  4  5  2  6  0  0  0  0  0]
 [ 0  2 48  4  8  4  0  0  0  0  0  0  0  2  0  0  0  0  0  0]
 [ 0  3  4 41 47  0  3  0  0  1  0  0  1  0  0  0  0  0  0  0]
 [ 0  1  1 18 69  0  6  0  0  0  0  0  4  1  0  0  0  0  0  0]
 [ 0 17 10  0  0 67  1  0  0  0  0  0  0  5  0  0  0  0  0  0]
 [ 0  0  1  1 17  0 75  2  0  0  0  0  3  1  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  4 46 39  0  0  0  5  3  1  0  2  0  0  0]
 [ 0  0  0  0  0  0  1  7 81  4  0  0  3  2  0  0  1  0  1  0]
 [ 0  1  0  0  0  0  4  0  0 79 16  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0 17 84  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  4  0  0  0  0  0 90  0  6  0  0  0  0  0  0]
 [ 0  1  1 11  2  3  5  2  0  0  0  1 68  4  2  0  0  0  0  0]
 [ 0  0  0  1  0  0  0  1  0  0  0  0  5 87  2  1  0  0  3  0]
 [ 0  1  0  0  0  1  0  0  1  2  0  0  2  1 89  0  0  0  3  0]
 [ 7  0  0  0  0  0  0  0  0  0  0  0  0  0  0 92  0  0  0  1]
 [ 1  0  0  0  0  0  0  0  1  0  0  1  0  3  0  0 90  1  3  0]
 [ 1  0  0  0  0  0  0  0  0  1  1  0  0  1  0  4  0 88  4  0]
 [ 0  0  0  0  0  0  1  0  1  0  0  0  0  7  6  3 48  4 30  0]
 [15  0  0  0  0  0  0  0  0  0  0  1  0  2  0 24  6  0  3  1]]
```

	precision	recall	f1-score	support
0	0.662	0.470	0.550	100
1	0.536	0.303	0.387	99
2	0.658	0.706	0.681	68
3	0.500	0.410	0.451	100
4	0.476	0.690	0.563	100
5	0.604	0.670	0.635	100
6	0.728	0.750	0.739	100
7	0.793	0.460	0.582	100
8	0.653	0.810	0.723	100
9	0.745	0.790	0.767	100
10	0.832	0.832	0.832	101
11	0.928	0.900	0.914	100
12	0.701	0.680	0.690	100
13	0.644	0.870	0.740	100
14	0.840	0.890	0.864	100
15	0.568	0.920	0.702	100
16	0.604	0.900	0.723	100
17	0.946	0.880	0.912	100
18	0.612	0.300	0.403	100
19	0.500	0.019	0.037	52
accuracy			0.678	1920
macro avg	0.676	0.662	0.645	1920
weighted avg	0.681	0.678	0.660	1920

Confusion Matrices and Metrics – Bi-LSTM Model

```
Accuracy: 0.879167
[[72  1  1  0  0  0  0  0  0  0  0  0  2  0  0  3  0  0  0 21]
 [ 0 84  3  1  3  2  0  0  0  1  0  0  1  0  0  1  0  1  0  2]
 [ 0  4 58  4  1  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  3  5 77 10  0  1  0  0  1  0  0  3  0  0  0  0  0  0  0]
 [ 0  1  0  8 87  0  2  1  0  0  0  0  1  0  0  0  0  0  0  0]
 [ 1  4  5  0  1 89  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  2  2  6  6  0 72  9  0  1  0  0  2  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0 97  0  0  0  0  1  1  1  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  3 96  1  0  0  0  0  0  0  0  0  0  0]
 [ 1  0  0  0  0  0  0  0  0 92  3  0  0  1  1  2  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  2 99  0  0  0  0  0  0  0  0  0]
 [ 0  1  0  0  0  0  0  0  0  0  0 97  0  0  0  0  1  0  1  0]
 [ 0  3  2  5  0  0  1  0  0  0  0  0 88  1  0  0  0  0  0  0]
 [ 1  1  0  0  0  1  0  0  0  0  0  1  1 92  0  3  0  0  0  0]
 [ 0  1  0  0  0  0  0  0  0  0  0  0  2  1 93  1  0  0  2  0]
 [ 1  0  0  0  0  0  0  0  0  0  0  0  0  0  1 98  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  1  0  0  1  0  0  0  1 93  0  4  0]
 [ 0  1  0  0  0  0  0  0  0  1  0  0  0  0  0  4  1 87  3  3]
 [ 0  0  0  0  0  0  0  0  1  0  0  0  0  1  1  0 21  0 69  7]
 [ 1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  2 48]]
```

```
              precision    recall  f1-score   support

           0      0.935     0.720     0.814       100
           1      0.785     0.848     0.816        99
           2      0.763     0.853     0.806        68
           3      0.762     0.770     0.766       100
           4      0.806     0.870     0.837       100
           5      0.967     0.890     0.927       100
           6      0.935     0.720     0.814       100
           7      0.882     0.970     0.924       100
           8      0.980     0.960     0.970       100
           9      0.929     0.920     0.925       100
          10      0.971     0.980     0.975       101
          11      0.980     0.970     0.975       100
          12      0.871     0.880     0.876       100
          13      0.948     0.920     0.934       100
          14      0.959     0.930     0.944       100
          15      0.867     0.980     0.920       100
          16      0.802     0.930     0.861       100
          17      0.989     0.870     0.926       100
          18      0.852     0.690     0.762       100
          19      0.593     0.923     0.722        52

    accuracy                          0.879      1920
   macro avg      0.879     0.880     0.875      1920
weighted avg      0.888     0.879     0.880      1920
```

Confusion Matrices and Metrics – W-RNN Model

```
[30] Accuracy: 0.894792
[[ 75   1   0   0   0   0   0   0   1   0   0   0   1   1   1   1   0   1
    1  17]
 [  1  81   2   2   3   4   1   0   0   0   0   0   1   0   1   1   0   1
    1   0]
 [  0   5  49   5   4   2   2   0   0   0   0   0   0   1   0   0   0   0
    0   0]
 [  0   1   3  79  10   0   7   0   0   0   0   0   0   0   0   0   0   0
    0   0]
 [  0   0   0   2  86   1  10   0   0   0   0   0   0   1   0   0   0   0
    0   0]
 [  0   1   0   0   0  99   0   0   0   0   0   0   0   0   0   0   0   0
    0   0]
 [  0   0   0   2   1   0  96   1   0   0   0   0   0   0   0   0   0   0
    0   0]
 [  0   0   0   0   0   0   4  92   1   0   0   0   1   2   0   0   0   0
    0   0]
 [  0   0   0   0   0   0   0   0  98   1   1   0   0   0   0   0   0   0
    0   0]
 [  0   1   0   0   0   0   2   0   0  95   2   0   0   0   0   0   0   0
    0   0]
 [  0   0   0   0   0   0   0   0   0   1 100   0   0   0   0   0   0   0
    0   0]
 [  0   1   0   0   0   0   0   0   0   0   0  99   0   0   0   0   0   0
    0   0]
 [  0   3   0   2   0   0   7   0   0   0   0   1  85   2   0   0   0   0
    0   0]
 [  0   1   0   0   0   1   0   0   0   0   0   0   0  96   0   2   0   0
    0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   1   2  97   0   0   0
    0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   1  99   0   0
    0   0]
 [  0   0   0   0   0   0   0   1   1   0   0   0   0   0   0   2  95   0
    1   0]
 [  1   0   0   0   0   0   0   0   0   2   0   0   0   0   0   4   2  88
    2   1]
 [  0   0   0   0   0   0   0   0   0   1   0   1   0   0   1   0  24   0
   70   3]
 [  4   0   0   0   0   0   0   0   0   0   0   0   0   1   0   1   3   0
    4  39]]
```

	precision	recall	f1-score	support
0	0.926	0.750	0.829	100
1	0.853	0.818	0.835	99
2	0.907	0.721	0.803	68
3	0.859	0.790	0.823	100
4	0.827	0.860	0.843	100
5	0.925	0.990	0.957	100
6	0.744	0.960	0.838	100
7	0.979	0.920	0.948	100
8	0.970	0.980	0.975	100
9	0.950	0.950	0.950	100
10	0.971	0.990	0.980	101
11	0.980	0.990	0.985	100
12	0.955	0.850	0.899	100
13	0.906	0.960	0.932	100
14	0.960	0.970	0.965	100
15	0.900	0.990	0.943	100
16	0.766	0.950	0.848	100
17	0.978	0.880	0.926	100
18	0.886	0.700	0.782	100
19	0.650	0.750	0.696	52
accuracy			0.895	1920
macro avg	0.895	0.888	0.888	1920
weighted avg	0.901	0.895	0.894	1920

ANALYSIS OF RESULTS

	LSTM (RNN)	Bi-LSTM	W-RNN
Mean Precision	**67.6%**	**87.9%**	**89.5%**
Mean Recall	**66.2%**	**88%**	**88.8%**
F1 Measure	**64.5%**	**87.5%**	**88.8%**

Classification Results of 21 Documents

	Predicted	Actual	Status
0	comp.sys.ibm.pc.hardware	misc.forsale	InCorrect
1	comp.graphics	comp.sys.ibm.pc.hardware	InCorrect
2	sci.electronics	sci.electronics	Correct
3	rec.sport.baseball	rec.sport.baseball	Correct
4	talk.politics.guns	talk.politics.misc	InCorrect
5	misc.forsale	sci.electronics	InCorrect
6	misc.forsale	misc.forsale	Correct
7	comp.windows.x	comp.windows.x	Correct
8	rec.autos	rec.autos	Correct
9	misc.forsale	misc.forsale	Correct
10	comp.os.ms-windows.misc	comp.os.ms-windows.misc	Correct
11	talk.religion.misc	alt.atheism	InCorrect
12	misc.forsale	comp.graphics	InCorrect
13	comp.sys.mac.hardware	comp.sys.ibm.pc.hardware	InCorrect
14	comp.graphics	sci.electronics	InCorrect
15	talk.religion.misc	talk.religion.misc	Correct
16	sci.crypt	sci.crypt	Correct
17	comp.windows.x	comp.windows.x	Correct
18	soc.religion.christian	soc.religion.christian	Correct
19	comp.graphics	comp.graphics	Correct
20	talk.politics.guns	talk.politics.guns	Correct

An Improvement to the above model (Additional to the Paper)

- Since Bi-LSTM gives a significant improvement as compared to the standard LSTM architecture, we used the Bidirectional LSTM to obtain the WM-vectors, which would then be weighted to obtain the document vector

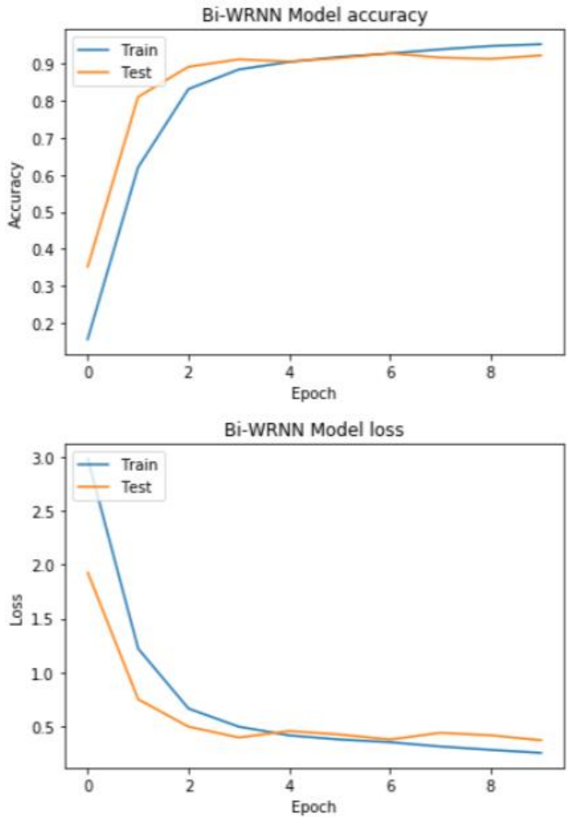
- This would further increase the accuracy of classification as each WM-vector would contain information of both, the past and the future.

- We would call this as **Bidirectional Weighted RNN, or simply, Bi-WRNN**

The implementation of this model is also given in the source code

Results of Bi-WRNN Model

- Reaches below 0.5 loss in just 2 epochs, which takes around 15 epochs in case of W-RNN.
- It has much **higher rate of convergence**, takes less than ½ the number of epochs as compared to W-RNN to reach optimum accuracy
- Performs **as good as** the W-RNN model when compared to our implementation of both
- Shows 89% precision and recall, 5**% better** than the metric quoted for W-RNN as per the paper



Confusion Matrix and Metrics – Bi-WRNN

```
Accuracy: 0.891667
[[67  1  0  1  1  0  0  1  0  0  0  0  0  0  0  3  0  1  1 24]
 [ 0 77  7  4  1  3  0  0  0  0  0  0  2  0  3  1  0  0  0  1]
 [ 0  2 60  3  2  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  3 88  5  0  3  0  0  0  0  0  1  0  0  0  0  0  0  0]
 [ 0  0  1  8 85  0  4  0  0  1  0  0  1  0  0  0  0  0  0  0]
 [ 0  2  4  2  0 92  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  3  1  4  2  0 88  1  0  0  0  0  1  0  0  0  0  0  0  0]
 [ 0  1  0  0  3  0  3 87  0  0  0  0  4  0  2  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  1 95  3  1  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  1  0  0  0 95  3  0  0  0  1  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  3 98  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0 97  2  0  0  0  0  0  1  0]
 [ 0  4  0  5  1  1  2  0  0  0  0  1 86  0  0  0  0  0  0  0]
 [ 0  0  0  1  0  1  1  0  0  0  0  1  2 90  0  3  0  0  1  0]
 [ 0  0  0  0  1  0  0  0  0  0  0  1  1  0 96  0  0  0  0  1]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  1 98  0  0  0  1]
 [ 0  0  1  0  0  0  0  0  0  0  0  1  0  0  0  0 94  0  3  1]
 [ 0  0  1  0  0  0  0  0  0  2  0  0  0  0  0  1  1 95  0  0]
 [ 0  0  1  0  0  0  0  0  0  0  0  0  0  0  2  0 15  1 78  3]
 [ 2  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  2  0  1 46]]
```

	precision	recall	f1-score	support
0	0.971	0.670	0.793	100
1	0.856	0.778	0.815	99
2	0.759	0.882	0.816	68
3	0.759	0.880	0.815	100
4	0.842	0.850	0.846	100
5	0.939	0.920	0.929	100
6	0.871	0.880	0.876	100
7	0.967	0.870	0.916	100
8	0.990	0.950	0.969	100
9	0.913	0.950	0.931	100
10	0.961	0.970	0.966	101
11	0.960	0.970	0.965	100
12	0.860	0.860	0.860	100
13	1.000	0.900	0.947	100
14	0.914	0.960	0.937	100
15	0.916	0.980	0.947	100
16	0.839	0.940	0.887	100
17	0.979	0.950	0.964	100
18	0.918	0.780	0.843	100
19	0.597	0.885	0.713	52
accuracy			0.892	1920
macro avg	0.891	0.891	0.887	1920
weighted avg	0.900	0.892	0.892	1920

THANK YOU!