

## **Deep Learning**

### **LAB Assignment-3**

#### Introduction

The main aim is to implement the text classification using CNN model, RNN model, LSTM model with new data set and compared results obtained.

#### Objectives

To compare the accuracy results between RNN, CNN, LSTM.

#### Approaches/Methods

Upload the dataset from the regret and later load it.

Later train it and evaluate it.

By setting the parameters and hiding layers run the code to get wanted results.

#### Workflow

- Required libraries are to be imported such as Numpy, Pandas, Keras, text CNN, RNN, LSTM
- Parameters are considered for Data, Model and Training
- Data to be used for pre-processing
- Load the wanted data and build vocabulary then shuffle randomly
- Splitting is done as the test and train data
- Cross validation must be done after training the data
- Train the procedure
- Placeholders are created.
- Summarize the loss and accuracy
- Evaluate parameters
- Observe the results

#### Datasets

The data set considered is regret.txt

## Parameters

### For CNN

```
Parameters:
ALLOW_SOFT_PLACEMENT=True
BATCH_SIZE=64
CHECKPOINT_EVERY=100
DEV_SAMPLE_PERCENTAGE=0.01
DROPOUT_KEEP_PROB=0.5
EMBEDDING_DIM=128
EVALUATE_EVERY=100
FILTER_SIZES=3,4,5
L2_REG_LAMBDA=0.0
LOG_DEVICE_PLACEMENT=False
```

### For RNN

```
# Parameters
learning_rate = 0.001
training_iters = 1000
display_step = 500
n_input = 3
```

### For LSTM

```
# Parameters
learning_rate = 0.001
training_iters = 1000
display_step = 500
n_input = 5
```

## Configuration

Pycharm

Python: 2.7.13

Tensor Flow

Keras

## Evaluation & Discussion

### Output for LSTM



See `tf.nn.softmax_cross_entropy_with_logits_v2`.

```
2018-05-09 22:03:00.246904: I T:\src\github\tensorflow\tensorflow
Iter= 500, Average Loss= 4.986780, Average Accuracy= 6.40%
['marrying.', 'She', 'had', 'never', 'been'] - [in] vs [a]
Iter= 1000, Average Loss= 3.444704, Average Accuracy= 15.20%
['yet', 'lived', 'to', 'regret', 'it.'] - [So] vs [in]
Optimization Finished!
Elapsed time: 2.84920072555542 sec
Run on command line.
5 words:
```

## Output for RNN

```
STM RNN RNN

See tf.nn.softmax_cross_entropy_with_logits_v2.

2018-05-09 21:57:34.274596: I T:\src\github\tensorflow\tensorflow\core\p
Iter= 500, Average Loss= 5.262385, Average Accuracy= 1.40%
['few', 'cows,', 'a'] - [couple] vs [hat]
Iter= 1000, Average Loss= 4.626032, Average Accuracy= 3.00%
['crops,', 'and', 'the'] - [fowls,] vs [had]
Optimization Finished!
Elapsed time: 2.0332632064819336 sec
Run on command line.
3 words:
```

## Output for CNN

```
n train
2018-04-23T17:12:27.838248: step 982, loss 0.500326, acc 0.8125
2018-04-23T17:12:28.311586: step 983, loss 0.379159, acc 0.84375
2018-04-23T17:12:28.775909: step 984, loss 0.473596, acc 0.765625
2018-04-23T17:12:29.248970: step 985, loss 0.307726, acc 0.859375
2018-04-23T17:12:29.726602: step 986, loss 0.356266, acc 0.859375
2018-04-23T17:12:30.211947: step 987, loss 0.423783, acc 0.78125
2018-04-23T17:12:30.692788: step 988, loss 0.539414, acc 0.71875
2018-04-23T17:12:31.157760: step 989, loss 0.380394, acc 0.859375
2018-04-23T17:12:31.602589: step 990, loss 0.411435, acc 0.833333
2018-04-23T17:12:32.075926: step 991, loss 0.349379, acc 0.84375
2018-04-23T17:12:32.561771: step 992, loss 0.392038, acc 0.828125
2018-04-23T17:12:33.020096: step 993, loss 0.287705, acc 0.890625
2018-04-23T17:12:33.507943: step 994, loss 0.381357, acc 0.8125
2018-04-23T17:12:33.980279: step 995, loss 0.399409, acc 0.8125
2018-04-23T17:12:34.438605: step 996, loss 0.271033, acc 0.921875
2018-04-23T17:12:34.903121: step 997, loss 0.386839, acc 0.875
2018-04-23T17:12:35.367450: step 998, loss 0.288989, acc 0.875
2018-04-23T17:12:35.842789: step 999, loss 0.37101, acc 0.859375
2018-04-23T17:12:36.316695: step 1000, loss 0.423611, acc 0.796875
```

## Conclusion

From the above screenshots we can observe that CNN performance is too good compared than the other too. So, CNN is best and Later LSTM compared to RNN. The ranking will be CNN, LSTM, RNN.