GENERATING SEQUENCES USING RECURRENT NEURAL NETWORKS

TEAM NAME – **BOUNDED EXCEPTION**

TEAM MEMBERS – **RITVIK GARG (2020201071)**

PRANAV MODI (2020202021)

SACHIN KUMAR GOYAL (2020201073)

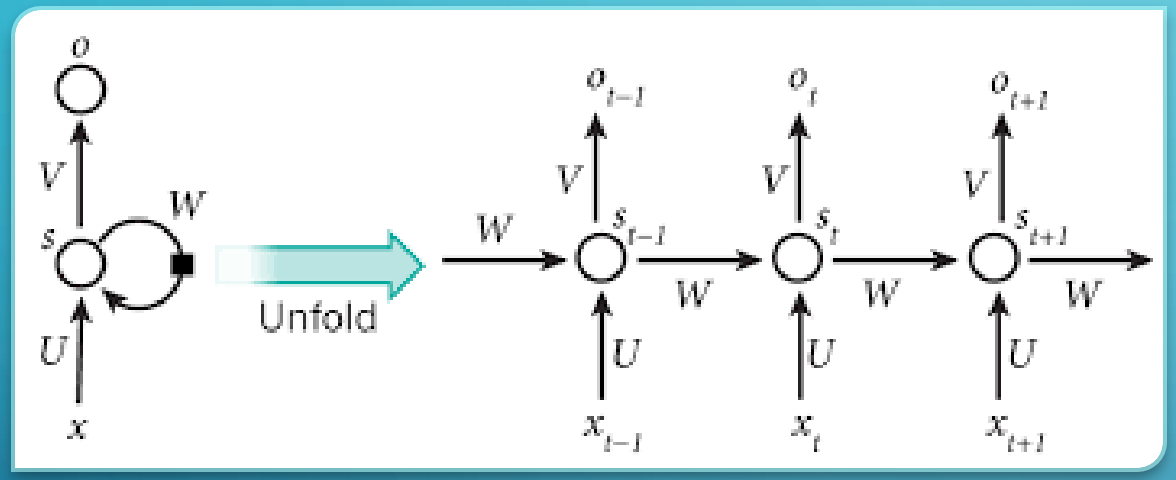
AIM OF THE PROJECT

- To create a generative model for text, word-by-word using RNN and LSTM neural network model.
- To create a generative model for text, character-by-character using LSTM in python.

RNN

- Recurrent Neural Networks are rich class of Dynamic models used to generate sequences in time series data.
- It can be trained on real data sequence one step at a time and predict what comes next.
- Limitations of Static Models:
  - One sample at a time with no track of order of input
  - Fixed dimension input

ARCHITECTURE OF RNN



ARCHITECTURE OF RNN

- Two major equations:
- $H_t = f(Ux_t + Wh_{t-1})$
- $O_t = \text{Softmax}(Vh_t)$

APPLICATION AND LIMITATION

- Application of RNN -
  - Language Translation – Uni-model or Multi-model
  - Stock Pricing prediction, Trajectory prediction
  - Sentiment Analysis
  - Generating Image Description
- Limitation of RNN -
  - Unable to store information about past inputs for very long

LSTM

- Overcomes the limitation of RNN
- LSTM can use its memory to generate complex, realistic sequences containing long-range structure.
- It has a cell state which works as a memory unit
- There are three gates namely Input gate, Forget gate, output gate.

MODEL DEFINITION

- Model – Sequential
- Layers
  - Embedding: input – vocabulary size
  - LSTM : Number of neurons – 64
  - Gaussian Noise : Standard deviation – 0.075
  - Dense : output – one hot encoding, activation – softmax
- Optimizer
  - SGD – learning rate = 0.0001, momentum = 0.99
  - Adam
- Loss – Categorical Cross Entropy

DATASET DESCRIPTION (PAPER DATASET)

- **THE PENTREE BANK**, or PTB for short, is a dataset maintained by the University of Pennsylvania. It is *huge* — there are over **four million and eight hundred thousand** annotated words in it, all corrected by humans.
- It is widely used as a language modelling benchmark.

DATASET DESCRIPTION (PAPER DATASET)

- The training set contains 930,000 words, the validation set contains 74,000 words and the test set contains 82,000 words.
- The dataset is divided in different kinds of annotations, such as piece-of-speech, syntactic and semantic skeletons.

EXPERIMENT

- Model1 : Word-level Simple-RNN
- Model2 : Character-level Single-layer LSTM
- Model3 : Character-level Multi-layer LSTM
- Model4 : Word-level Single-layer LSTM
- Model5 : Word-level Multi-layer LSTM
- Model6 : Word-level Multi-layer LSTM (did hyperparameter-tuning)

RESULTS

Char/Word Level	Model	# Epochs	# Validation Loss
Word	RNN	20	10.009
Char	Single-Layer LSTM	20	2.150
Char	Multi-Layer LSTM	20	1.597
Word	Single-Layer LSTM	50	11.09
Word	Multi-Layer LSTM	20	10.21
Word	Multi-Layer LSTM (optimized)	20	6.19

```
predict_next_word(mlstm,tokenizer,['jones', 'friday', 'industrial', 'average', 'stock'],10,num_features)
```

```
'index futures trading and chief executive officer of the u.s.'
```

ANALYSIS

- RNN performs better for short sequential data.
- For long sequential data, LSTM is preferrable.
- Multi-Layer LSTM performs better than Single-Layer LSTM.
- Word-level model predict more meaningful sentences as compared to Character-level model.
- SGD optimizer shows better result as compared to all other optimizers.
- Added noise to the model to avoid overfitting, and thus the model gave better result on testing data.

CONTRIBUTION

- Ritvik & Pranav -
  - Implemented character-level single layer and multilayer LSTM
  - Implemented word-level single layer Simple RNN
- Sachin & Pranav -
  - Implemented word-level single layer and multilayer LSTM
  - Understood theory and working of RNN and LSTM
- Ritvik & Sachin -
  - Performed multiple experiments taking different value of epochs, parameter values
  - Worked on Documentation

THANK YOU !!