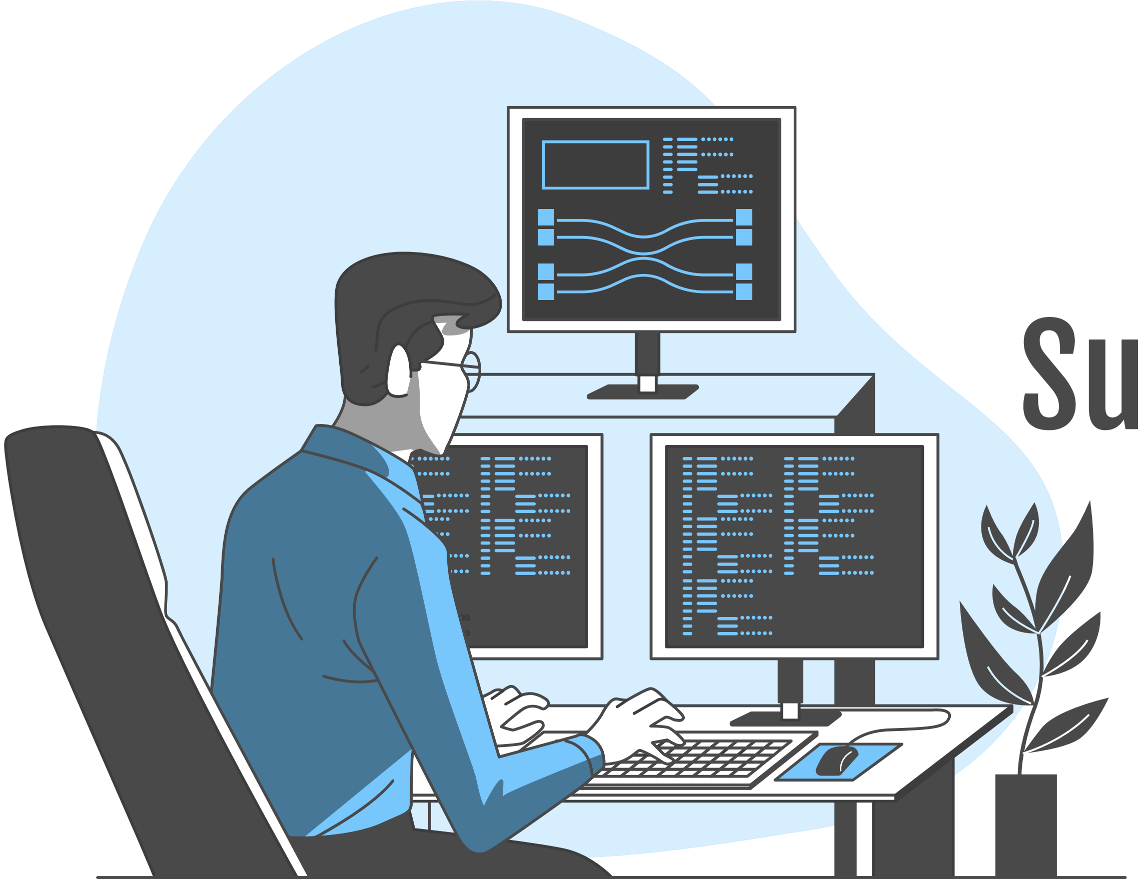
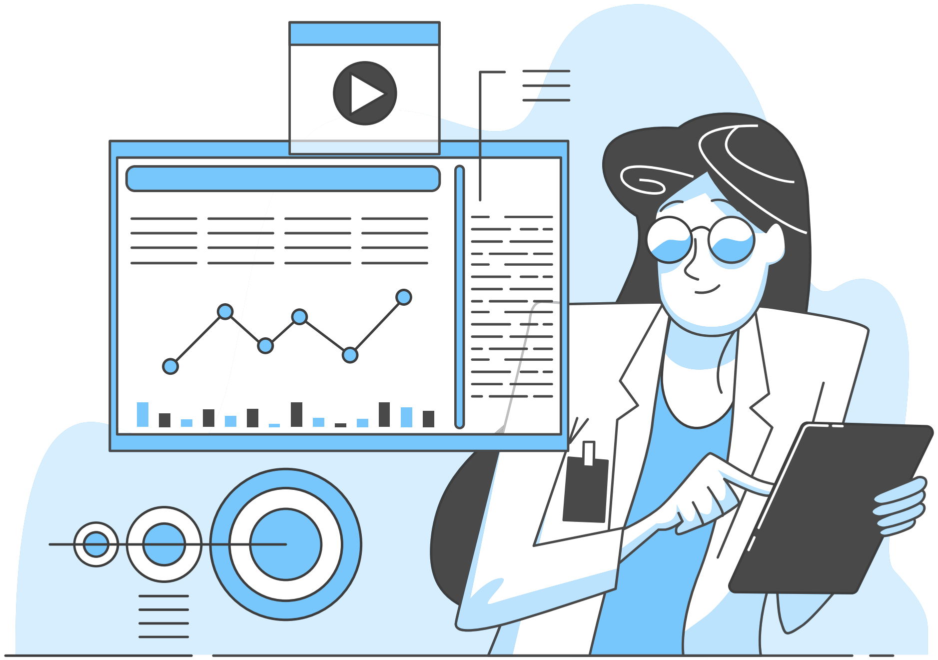


# Text Summarization

Dareen Hussein
Israa Fahmy

# Table of Contents





# 01

# Proposal summary

- Problem statement
- Our application
- Dataset

# Problem statement

Text summarization is a powerful tool to process and compress texts and produce concise, refined and brief content that contains the main information from the original. Neural sequence-to-sequence models have provided a viable new approach for abstractive text summarization. However, they still face challenges when dealing with long text.

# Our application

**Customer reviews** can often be long and descriptive. Analyzing these reviews manually, is really time-consuming. This is where we can apply Natural Language Processing to generate a summary for long reviews.

# Dataset


**Name:** Amazon Fine Food Reviews

**Description:** Reviews of fine foods from amazon.

**Size:** 642.49 MB

**Data includes:** 568,454 reviews

# 02

# Base Model

- Summary
- Base architecture

# Summary of Base Model

Text Summarization to Amazon Reviews

Encoder-decoder RNN with LSTM units and attention to generating headlines.

Abstractive Summarizer

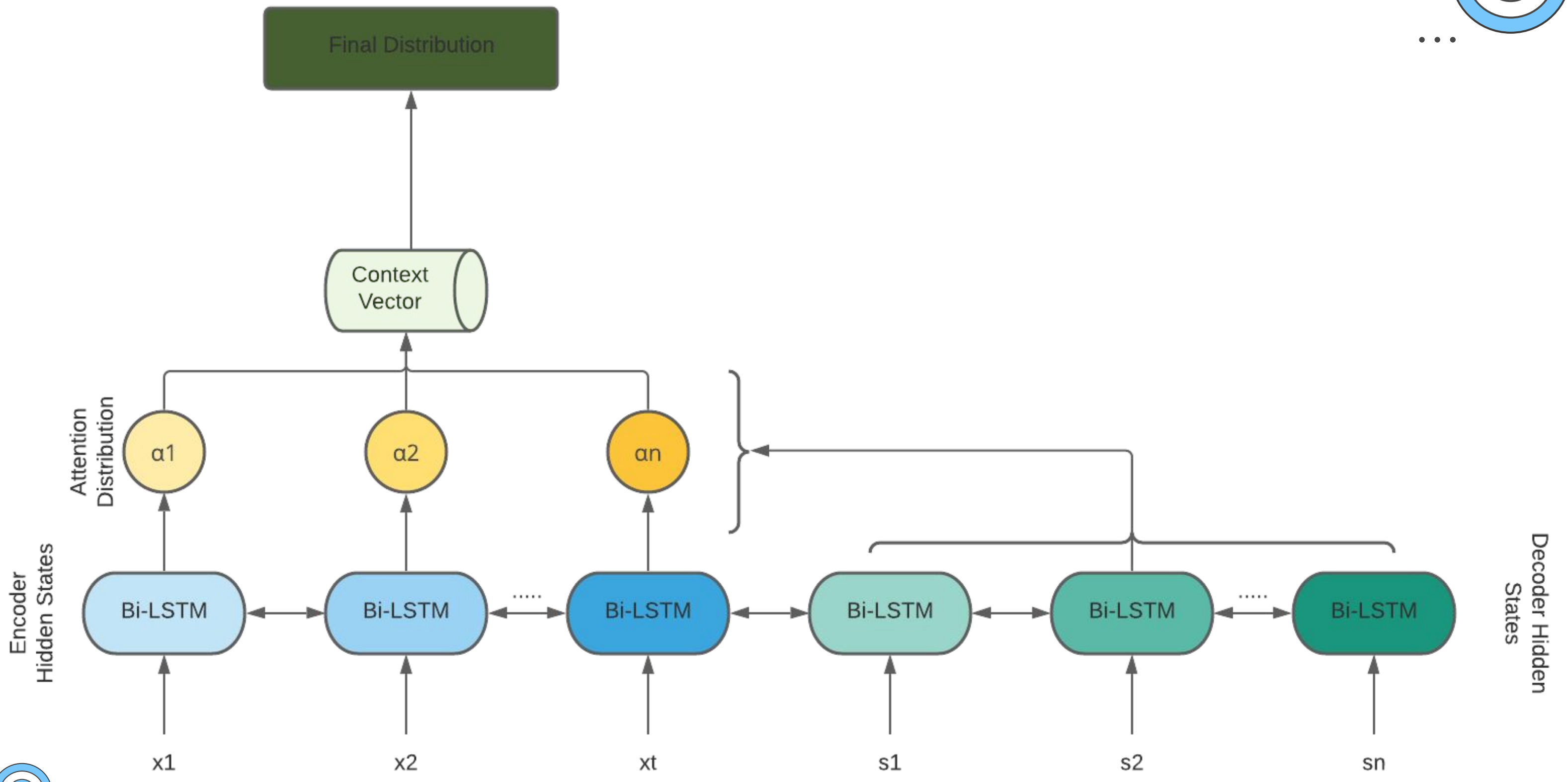
Used the training and holdout loss.

An embedding layer that transforms the word into a distributed representation

Evaluated using ROUGE metrics for performance.

**Base code:** Text-Summarization-with-Amazon-Reviews

# Baseline Architecture



# 03
# Our progress

- Proposed Model
- Modifications on proposed model
- Final Model
- Comparison

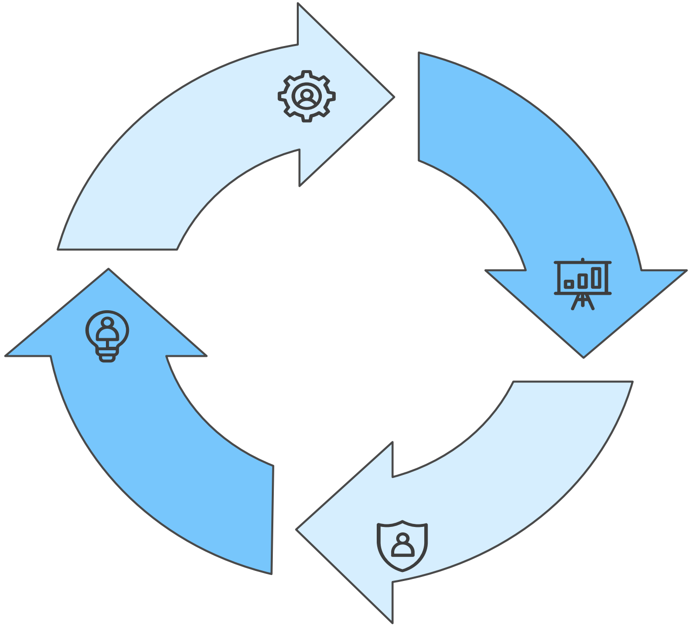
# Proposed Solution

## Abstractive Summarizer

Generate new sentences from the original text.

## Seq2Seq

The input is a sequence of words and the output is a sequence of tags for every word in the input sequence



## Greedy Vs Beam Search

Beam search saves computational powers and performs well..

## Encoder-Decoder Architecture

An Encoder LSTM reads the entire input sequence. The decoder is trained to predict the next word in the sequence given the previous word.

## Attention Mechanism

It aims to predict a word by looking at a few specific parts of the sequence only, rather than the entire sequence.

# Model Modifications

## Pointer Generator Network

$$p_{\text{gen}} = \sigma(w_{h^*}^T h_t^* + w_s^T s_t + w_x^T x_t + b_{\text{ptr}})$$

$$P(w) = p_{\text{gen}} P_{\text{vocab}}(w) + (1 - p_{\text{gen}}) \sum_{i:w_i=w} a_i^t$$

- At each decoder timestep, a generation probability is calculated.
- Decides the probability of generating words versus extracting words from source text.
- Vocabulary and attention distribution are weighted and summed to make prediction.

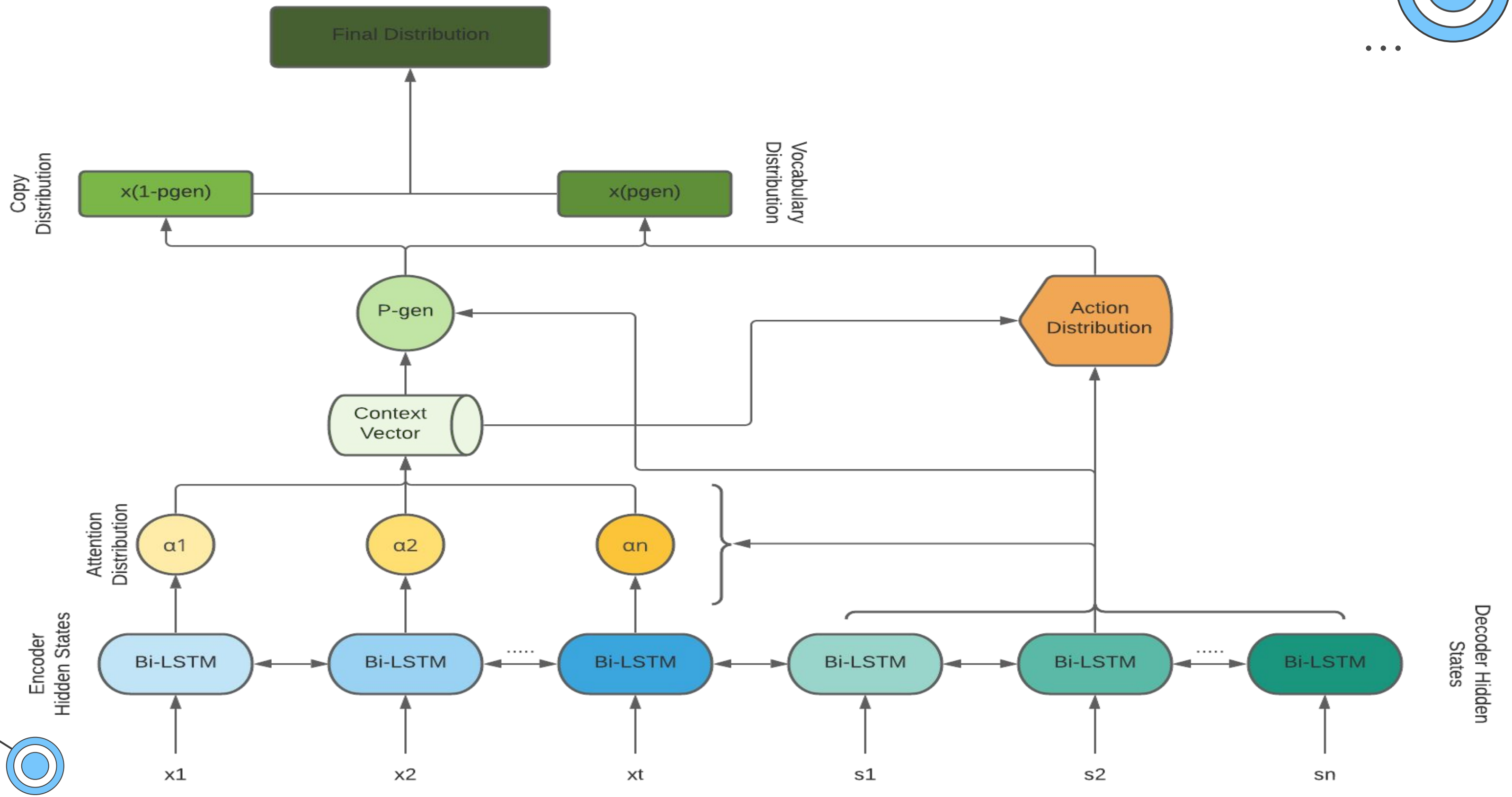
## Coverage Mechanism

**Coverage vector** $c^t = \sum_{t'=0}^{t-1} a^{t'}$

$$e_i^t = v^T \tanh(W_h h_i + W_s s_t + w_c c_i^t + b_{\text{attn}})$$

- We build on top of the model with a coverage, that sums attention distributions over all previous decoders and introduce an extra loss term.
- This penalizes the network for attending to the same words again.

# Final Solution



# Comparison between Baseline and Final Model

| | Baseline | Solution |
|---|---|---|
| **Model** | LSTM encoder-decoder | Bi-LSTM encoder-decoder |
| **Attention** | Third party imported layer | Customized attention layer |
| **Decoding algorithm** | Greedy algorithm | Beam search algorithm |
| **Word Embedding** | Word2vec | Fasttext |
| **Behaviour** | Abstractive mechanism | Pointer generator with Coverage |

# 04
# Results

- Experiments summary
- Evaluation Results
- Sample Outputs

# Hyperparameters Experiments

| Parameter | Experiment 1 | Experiment 2 | Experiment 3 | Experiment 4 |
| --- | --- | --- | --- | --- |
| Dropout | 0.4 | 0.4 | 0.4 | 0.3 |
| Activation | softmax | softmax | softmax | softmax |
| L1 | 1e-5 | 1e-5 | 1e-2 | 1e-2 |
| L2 | 1e-4 | 1e-4 | 1e-2 | 1e-2 |
| Optimizer | RMSprop | Adam | Adam | RMSprop |
| Loss function | Sparse categorical cross entropy | Sparse categorical cross entropy | Sparse categorical cross entropy | Sparse categorical cross entropy |

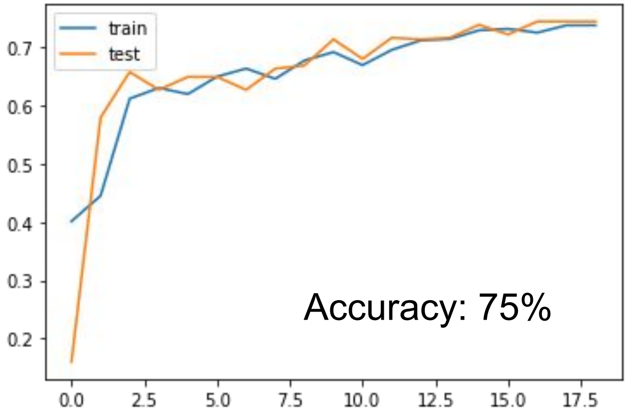
# Hyperparameters Experiments (Cont.)

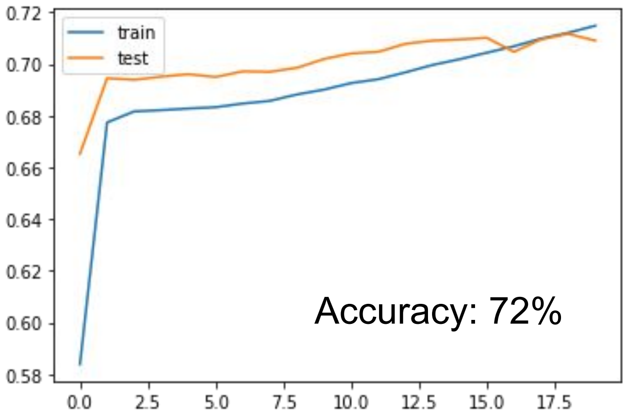
| Parameter | Experiment 1 | Experiment 2 | Experiment 3 | Experiment 4 |
|---|---|---|---|---|
| Stacked LSTM layers | 1 | 3 | 3 | 5 |
| Loss | 2.79 | 1.99 | 1.6 | 1.2 |
| Accuray | 0.43 | 0.57 | 0.7.09 | 0.745 |

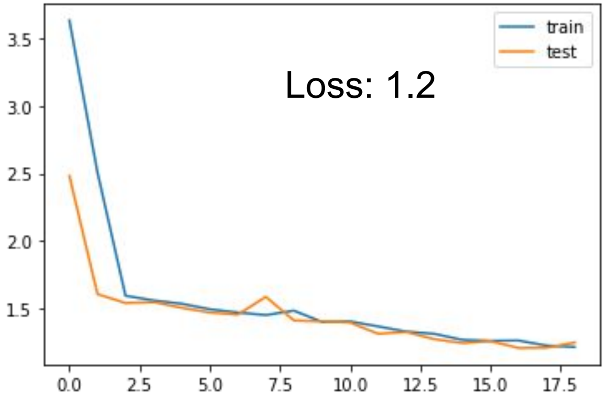
# Evaluation Metrics

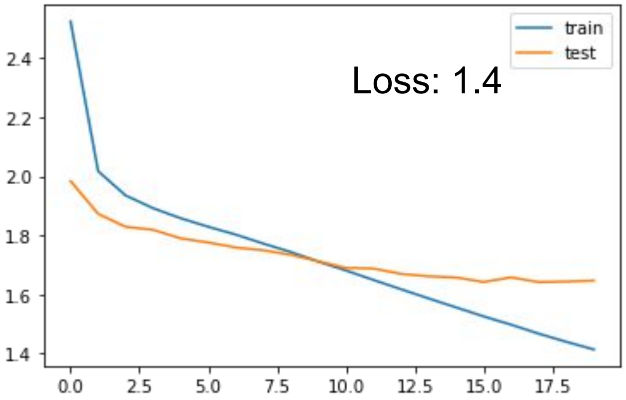
| Model | ROUGE 1 | ROUGE 2 | ROUGE L | BLEU |
|---|---|---|---|---|
| abstractive model | 35.46 | 13.30 | 32.65 | - |
| seq-to-seq + attn baseline (150k vocab) | 30.49 | 11.17 | 28.08 | - |
| seq-to-seq + attn baseline (50k vocab) | 31.33 | 11.81 | 28.83 | - |
| pointer-generator | 36.44 | 15.66 | 33.42 | - |
| Our Solution | 43.95 | 22.22 | 40.14 | 78.67 |

# Best Result Graphs



# Sample Output

**Sentence:** wonderful flavor would purchase this blend of coffee again light flavor not bitter at all and price was great the best i found anywhere
**Our model  Summary:** great coffee flavor
**Baseline Summary:** good flavour
**Actual Summary:** wolfgang puck k cup breakfast in bed.

**Sentence:** the pepper plant habanero extra hot california style hot pepper sauce 10 oz has great flavor as all the pepper plants do i just love it it is a bit pricey but worth it
**Our model  Summary:** great seasoning
**Baseline Summary:** great flavour
**Actual Summary:** wonderful love it

**Sentence:** once more amazon was great the product is good for kids even though it has a little bit more sugar than needed
**Our model Summary:** good as expected
**Baseline Summary:** good
**Actual Summary:** as expected

# 05

# Conclusion

- Progress report
- Future work
- Lessons Learnt

# Progress report

1. Compared all different previous solutions and decided to deploy best 2
2. Added additional layers to increase accuracy and attached attention layer
3. Added pointer generator network with coverage mechanism
4. Evaluated the model using BLEU and ROUGE score and did more testing

# Future Work

## 01 Train whole data

Expand the architecture to improve the quality of the generated summaries.

## 02 Web/Mobile Application

Take the model one step further and apply it on a web or mobile application to be user friendly.

# Lessons Learnt

- Better understanding of seq2seq models and their different types.
- More sense in choosing the suitable hyperparameters.
- Making sure of code availability when looking for base models.
- How to understand and modify existing coded.
- Difference between different evaluation metrics like BLEU and ROUGE.
- Hand on experience on Machine language translation and NLP library.
- How to mount from driver and use third party libraries.
- Know the development environment and its limitations.

# Members Contribution

## 01

### Israa

- Added attention layer to base code.
- Deployed base model and changed parameters.
- Changed stacked LSTM layers.
- Implemented Word embedding
- Co-implemented Pointer generator and coverage
- Added customized attention.
- Added BLEU score.

## 02

### Dareen

- Loaded and checked different base line codes.
- Deployed the other base model and changed parameters.
- Downloaded Dataset.
- Co-implemented Pointer generator and coverage
- Imported and used pickle
- Implemented Beam search algorithm.
- Added ROUGE score.

# Resources


- Ilya Sutskever, Oriol Vinyals, and Quoc Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*.
- Vishal Gupta and Gurpreet Singh Lehal. 2010. A survey of text summarization extractive techniques. *Journal of Emerging Technologies in Web Intelligence* , 2(3):258–268.
- Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. In *arXiv preprint arXiv:1609.08144* .
- Ian Good fellow, Aaron Courville ,and Yoshua Bengio. Deep learning. Book in preparation for MIT Press, 2015.
- Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. CoRR, abs/1506.03099, 2015.
- William Chan, Navdeep Jaitly, Quoc V. Le, and Oriol Vinyals. Listen, attend and spell. CoRR, abs/1508.01211, 2015.

# Resources (Cont.)


- Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. Recurrent neural network regularization. CoRR, abs/1409.2329, 2014.
- Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5 – rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 2012.
- Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. 2016. Pointing the unknown words. In Association for Computational Linguistics.
- Aravind PaiAravind is a sports fanatic. His passion lies in developing data-driven products for the sports domain. He strongly believes that analytics in sports can be a game-changer. (2020, May 10). Text summarization: Text summarization using deep learning. Retrieved March 30, 2021, from https://www.analyticsvidhya.com/blog/2019/06/comprehensive-guide-text-summarization-using-deep-learning-python/
- Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, C ̧aglar Gulc ̧ehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In Computational Natural Language Learning.


Thank you!
Dareen & Israa
Github repo