

Abstractive Text summarization

BY Israa Fahmy & Dareen Hussien

Table of Contents

01 Defining the Problem

03 Results

02 Model Architecture

04 Progress report



Defining the Problem



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: This dataset consists of reviews of fine foods from amazon.

Size: 642.49 MB

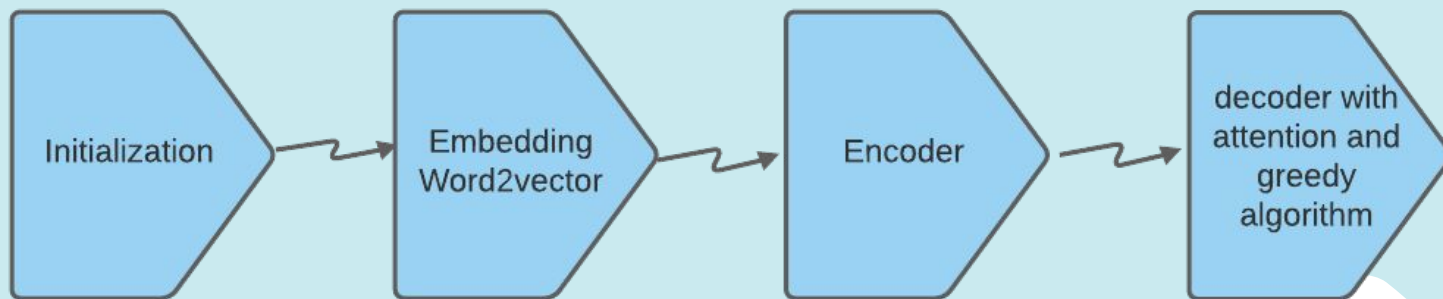
Data includes: 568,454 reviews



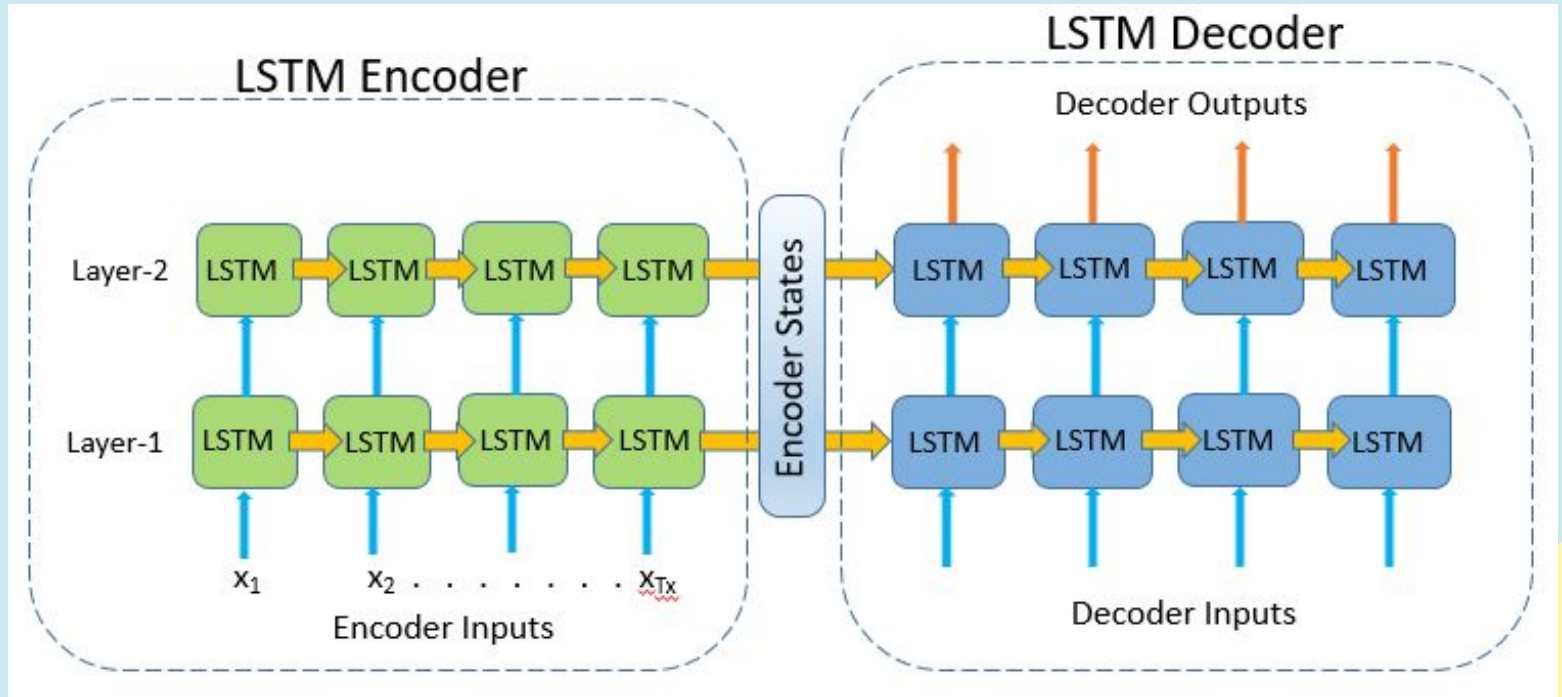
Architecture

Baseline Architecture

- Stacked LSTM encoder-decoder model with imported attention



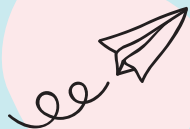
Base Code





Improved Architecture

- ## Pointer Generator Network with Coverage Mechanism



There are two main problems when dealing with summarization techniques:

1. Summaries generated may produce factually incorrect information
 - Used pointing to extract words from source text
2. Summaries generated contain many repeated words
 - Used coverage to keep track of what has been summarized already

• 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.

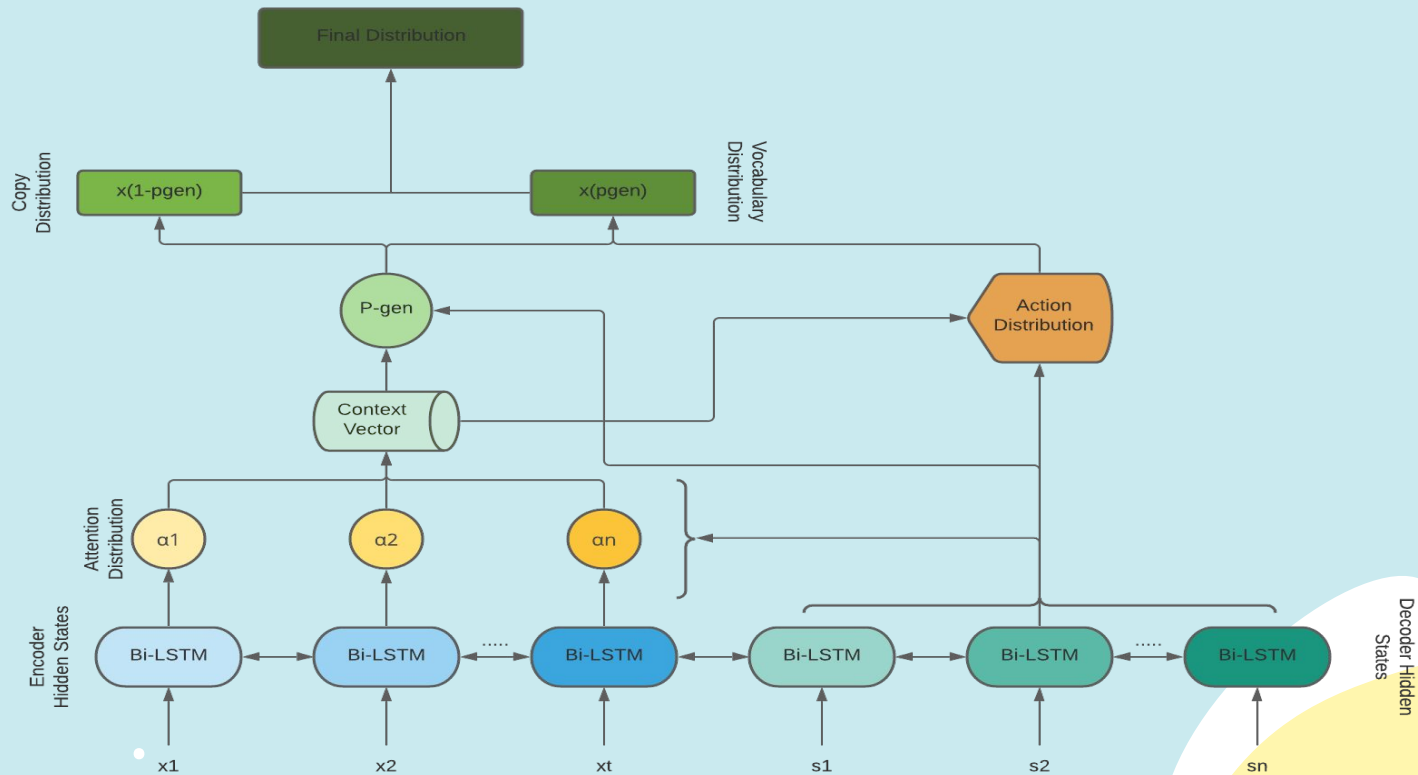
Coverage Mechanism

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.

Model Architecture



Highlighting main differences

	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	Pointer generator with Coverage mechanism



Results

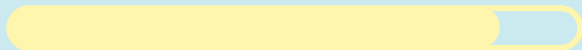
Output of Milestone 1



Sentence: great coffee br excellent service
br best way to buy k cups br stock up
before coffee prices go up again.

Predicted Summary: great great coffee

Actual Summary: great coffee excellent
service



Sentence: our dog has to have insulin shots
twice daily and these liver treats make
giving the shots easier and less stressful
for the owner and dog

Predicted Summary: the dog loves

Actual Summary: great reward for my dog



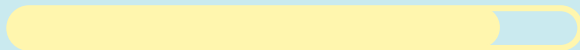
Output of Milestone 2



Sentence: great coffee br excellent service
br best way to buy k cups br stock up
before coffee prices go up again.

Predicted Summary: great coffee

Actual Summary: great coffee excellent
service



Sentence: our dog has to have insulin shots
twice daily and these liver treats make
giving the shots easier and less stressful
for the owner and dog

Predicted Summary: my dog loves it

Actual Summary: great reward for my dog





Progress



Progress Report

1

Experimented
different
hyperparameters
over the baseline

2


Added pointer
generator network
with coverage
mechanism

3

Replaced greedy
approach with
Beam search
strategy

4

Tried different
word embeddings
and decided on
fasttext





Technical issues resolved

1

We mounted google drive on Google Colab to reduce uploading time.

2

We used pickle to save/load our model.



Members contribution

Dareen Hussein

Implemented Beam search algorithm

Tried different hyperparameters to improve original baseline.

Co-implemented Pointer generator and coverage

Imported and used pickle

Israa Fahmy

Implemented Word embedding

Tried different hyperparameters to improve original baseline.

Co-implemented Pointer generator and coverage

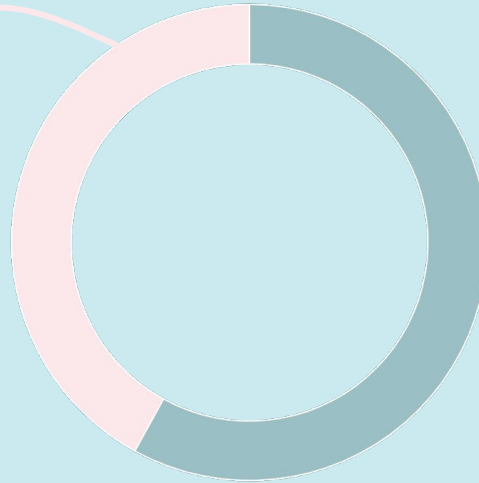
Added customized attention.

Timeline

Upcoming week

Evaluate the
Model using
BLEU or ROUGE
Scores

Try different
customized
Attention
methods



Accomplished work

Model generator
network with stacked
Bi-LSTM encoder
decoder model with
Beam Search
decoding strategy and
attention along with
pointer generator
with coverage
mechanism

Resources

- See, A., Liu, P. J., & Manning, C. D. (2017, April 25). Get To The Point: Summarization with Pointer-Generator Networks. arXiv.org. <https://arxiv.org/abs/1704.04368>
- Ilya Sutskever, Oriol Vinyals, and Quoc Le. 2014. Sequence to sequence learning with neural networks. *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. *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.
- 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/>



Thanks!