Final Report
Abstractive Summarization for Amazon Reviews

Rushab Munot, 14405
Shibhansh Dohare,14644

20^{th} November 2016

Introduction

Summarization is the task of reducing the length of a document/article or any text so that the meaning of the original text is retained as far as possible. Within Natural Language Processing, summarization is broadly classified into two parts - Extractive Summarization and Abstractive Summarization.

The basic task of summarization is to shorten the length of the text and Extractive Summarization does exactly that. In Extractive Summarization, unwanted or irrelevant sentences (or words) are simply removed from the original text. This may seem naive but it produces quite relevant output summaries. Abstractive Summarization on the other hand takes a much wider and general approach. The output summaries produced need not have same words from the input neither do they need to follow the same order. It can be seen as - the computer extracts the meaning of the text provided and then uses this meaning to produce an output it its (computer's) own words. Though this is a vague description it is what it intuitively means.

The general task of summarization can be formalized as follows:
Given an input string $x = x_1, x_2, x_3, \ldots, w_n$ where each $x_i \in \mathcal{V}$ where $\mathcal{V}$ is our vocabulary or $x \in \bar{\mathcal{V}}$ where $\bar{\mathcal{V}}$ is the set of all sequences that can be formed by words from $\mathcal{V}$. The task is to generate a new sequence $y = y_1, y_2, y_3, \ldots, y_m$ such that each $y_i \in \mathcal{V}$ such that

$$y = \arg\max_{y' \in \bar{\mathcal{V}}} s(x, y')$$

Here $s(x, y)$ denotes the similarity between x and y.

In this project we consider abstractive summarization. There are a few basic challenges that needed to be overcome while tackling the problem of abstractive summarization. The first one is that the output produced must not only be similar to the input but also be correct grammatically. Second, the lengths of the input text and the output summary may vary. The first prroblem is solved by using models for learning sequential data, that is methods that a learn a map from one sequence to another. The second problem is addressed by learning a fixed size representation

for the input no matter what the input length is.

Here we constrain our input and output lengths by an upper bound. We trim any input text longer than 100 words to the first 100 words of the text. The output is constrained to generate at most 10 words of output. This part becomes clearer when we describe the problem statement in depth.

Problem Setup and Data

We address a task related to sentence summarization in the sense that we try to summarize text containing about 2-4 sentences whereas the output is in 2-5 words, a short sentence at the most. Given a set of such sentences the task is to capture the most relevant parts of the text in the more or less the least number of words possible.

In particular, we run our experiments on the Amazon Fine Food Reviews Dataset, which contains short food reviews, with 2-5 words summaries. These reviews, as the name suggests, are typically centered on food products (for humans and pets, mostly cats and dogs) along with other information about the product. There are about 5,00,000 food reviews. These reviews were collected on the products sold by Amazon before October 2012. Further, every review has a score attached with it, ranging between 1 and 5, 1 signifying dissatisfaction. Other fields include Id, ProductId, UserId, ProfileName, Helpfulness Numerator (number of users who found the review helpful), Helpfulness Denominator (number of users who indicated whether they found the review helpful) and Time. In our experiments we concern ourselves with the Review, its Summary and the Score.

Here are certain observations about the dataset

- Majority of the data consists of positive reviews.
- Only about 10 to 15% of the data is negative (and it is not uniformly distributed, hence a range 10 to 15%).
- Coffee and tea as also dog and cat food form a large part of the reviews.
- Reviews basically focus on the taste of food, and many-a-times consist of an extreme description of the reviews. For example, 'This is the best coffee in the world', 'Going to have this every morning', 'My cat hates it', 'It tastes like mud', etc. Some reviews focus on the shipping and packaging.

Background and Previous Work

Sequence to Sequence Learning with Neural Network (1)

This model by Sutskever et al. (1)is basically an approach that tackles sequence mapping, basing the experiments on machine translation. We using a similar model for our task. More details about this model model are given in the 'Model Proposed' section.

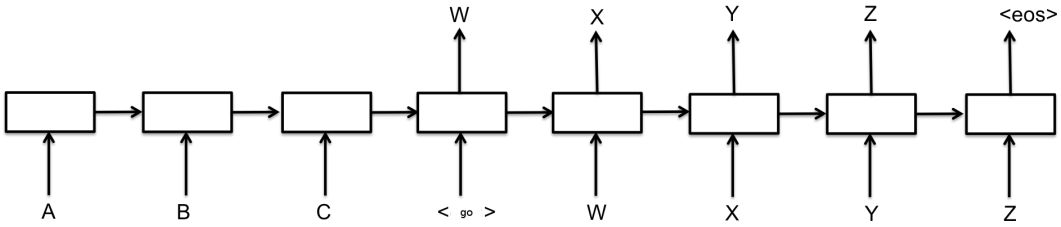

Figure 1: Pictorial representation (taken from (1))

Neural Attention model for Sentence Summarization(6)

This method proposed by (6) utilizes a local attention-based model that generates each word of the summary conditioned on the input sentence. This method scales to a large amount of training data. The core of this model is a language model for estimating the probability of the next word. This language model is taken from a standard feed-froward neural network language model. Then model used various encoders the most basic being the BoW model. On using a deep convolutional encoder local interaction between words can be captured, thus it shows improvement over the BoW encoder.Best results are achieved using attention based contextual encoder that constructs a representation based on the context.

Model proposed

We use the model proposed by Sutskever et al.(1) in 2014 which basically creates a map between input sequences and output sequences. Though they base their experiments on Machine Translation, the underlying idea is mapping sequences.

The task at hand is similar, given an input sequence of words we need to output a shorter sequence of words with certain constraints. There are multiple tasks involved. Similar to Sutskever et al. we use an encoder - decoder network. We first create a fixed size encoding of the input. This means that even though the input length varies, the size of this encoding remains the same. This input encoding is then passed on to the decoder. The task of the decoder is to generate an output sequence of words, the length of which may vary depending on the input encoding.

We use a deep LSTM network to encode the input as well as to decode the encoded input. The input fed to the encoder one by one in time i.e. the first word is fed, and an encoding is generated by the input. Now the next input to the LSTM is the second word and the encoding of the first word. The encoding thus created remembers information from the first as well as second word. The input after that is the third word and the combined encoding of the first two words. This continues till the entire input gets encoded. This fixed size input encoding is then passed on to the decoder. The decoder then receives a special $< go >$ symbol.

The first output word is generated using this $< go >$ symbol and the input encoding. The second word is generated using the first word and the encoding. This goes on till the $< eos >$ symbol is generated when the decoder stops generating

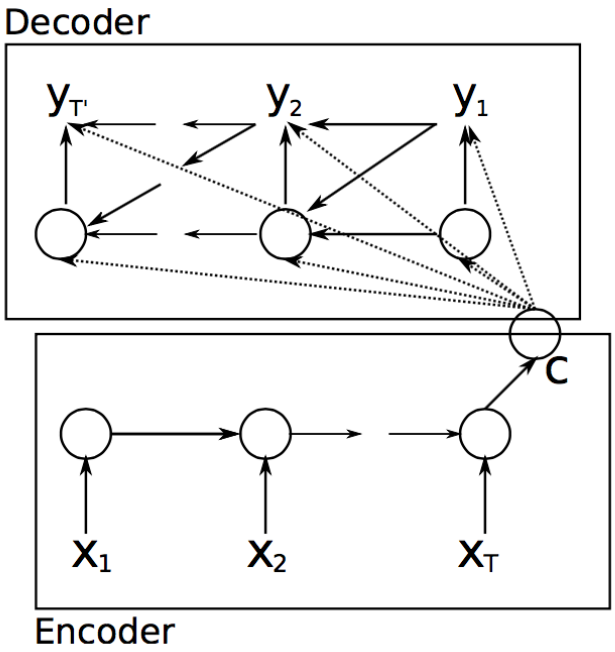

Figure 2: Pictorial representation (taken from (1))

output. The model is shown is Figure 2.

As mentioned above the number of negative samples in the dataset is quite less compared to positive samples. This can checked using the Score for the review. We call a review negative if its score is less than 3. We make three copies of each negative review so as to increase its proportion in the dataset. Without this the model may give poor results on certain negative reviews.

Preprocessing the Data

Another problem faced was the vocabulary size which is approximately 50,000 words. Due to limited GPU memory available (2GB), we introduced a new $< unknown >$ tag for words with less than 4 occurrences in the entire training set. This reduces the vocabulary size to about 30,000 words.

Due to memory and time issues we limited ourselves to training over a maximum of 20,000 food reviews. These reviews were selected randomly from the dataset and duplicated (as mentioned above) wherever necessary.

Experiments and Results

The model was put to test on vocabulary sizes of 500, 2000 and 20000. The deep LSTM network for the encoder was varied over 2 to 3 layers and with sizes of 300, 500, 1000. The results with deeper layer were better than shallow ones.

Here are some qualitative results on a model trained on 20k reviews with a deep LSTM network of 3 LSTMs of size 500 for the encoder, and a single LSTM layer of size 1000 for the decoder:

Successes

- **These are great chips and taste was a little spicy , exactly what I wanted . If you like cheese then you will definitely like them . Thank you amazon for the nice packaging and a good delivery of the chips.**
  ⇒great chips !
- **When I bought this juice from the near by store was sweet but what Amazon sold tasted sour and bitter . do not buy from amazon .**
  ⇒will not buy again
- **This tea tastes very bad . No flavor and no color .**
  ⇒ bitter dirt
- **Got this yesterday . Amazing flavor . Love them .**
  ⇒ best flavor by far

Failures

- **I hate this coffee . Do not buy from amazon**
  ⇒ stale cookies
- **Very bad coffee . Did not like it .**
  ⇒not too bitter and smoky

ROUGE Scores

These are scores when trained over 20,000 Reviews (only, couldn't train properly for more), encoder = 3 LSTM layers of size 500, decoder = one LSTM layer of size 500

6 Epochs:
ROUGE-1: 4.883; ROUGE-2: 1.297; ROUGE-L: 4.883

9 Epochs:
ROUGE-1: 6.128; ROUGE-2: 1.657; ROUGE-L: 6.128

12 Epochs:
ROUGE-1: 5.218; ROUGE-2: 1.511; ROUGE-L: 5.218

Problems

We faced a lot of problems due to lack of resources. We had to restrict the complexity of our model. Further We had to restrict the number of Reviews used for training. Much of our time was spent in memory management rather than improving the model. If we were to train our model on a sufficiently large dataset, we would definitely get better results.

The predictions fail (sometimes miserably) on very small (< 10 words) or large reviews (>50 words). Due to the small training set it becomes difficult to correctly evaluate the results. We wrote our code in torch which has very little documentation. Hence we had to write lots of code.

Code Repository

https://github.com/Rushab1/AbstractiveSummarization

Acknowledgements

We would like to thank Prof. Karnick and our TA Pawan Kumar, for their efforts in helping us complete this project.

References

[1] Ilya Sutskever, Oriol Vinyals, Quoc V. Le. 2014. *Sequence to Sequence Learning with Neural Networks*,

[2] Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) 2014. *Learning Phrase Representations using RNN Encoder Decoder for Statistical Machine Translation.*,

[3] Lin, C. (2004) ROUGE: A Package for Automatic Evaluation of Summaries. Proceedings of Workshop on 2004. *Text Summarization Branches Out, Post-Conference Workshop of ACL 2004. Barcelona, Spain.*

[4] Amazon Fine Food Reiews. Kaggle *https://www.kaggle.com/snap/amazon-fine-food-reviews.*

[5] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean. 2013. *Distributed Representations of Words and Phrases and their Compositionality.*

[6] Alexander M. Rush, Sumit Chopra, Jason Weston 2015. *A Neural Attention Model for Sentence Summarization*

[7] Chopra, S., Michael Auli, Alexander Rush. 2016. *Abstractive Sentence Summarization with Attentive Neural Networks, NAACL 2016*

[8] https://github.com/Element-Research/rnn/blob/master/examples/encoder-decoder-coupling.lua *Seq2seq model implementation*