Opinion Generation using Abstractive Text Summarization Techniques

Sumedh Sankhe, Soumitra Mishra and Rajath Kashyap

1 Changes

We have made minor changes to our understanding of the project. We intend to create an abstractive summary of the different review's and then draw out an opinion based on our summary. Due to the unavailability of the CNET dataset on their website. We will be using the Yelp Review dataset as our primary dataset. The dataset was downloaded from Kaggle since all the data was pre-segregated into different files Yelp Dataset[1].

2 Pre-Processing

The yelp dataset contains reviews for multiple services and establishments. In our project we will be focusing on the review that restarurants have received. The data set is broken down into multiple json files which incorporate the business id's, reviews and user id's. The volume of the dataset is enormous, even after restricting to restaurant review we faced the challenge of identifying reviews that are not in english and marking them unsuable.

We extracted the key attributes from the business json file (business id, category, review counts) filtered it down till we only had restaurants. These restaurants were then used to search through the review's json and extract all the relevant reviews. In all we extracted around 59,000 restaurants, although there is stark difference in the reviews. Some restaurants have very long and detailed review and some at best are a few words long. For the purpose of further usage in our models we have identified the start and end of review and tagged it appropriately. To remove and filter any reviews that are not in english we will be using an of the shelf implementation from called TextBlob to identify those reviews. Franchise restaurants such as Hooters/Domino's are removed from out dataset since their reviews are pretty varied from location to location.

Our key features for the initial model that we will be building out will be the n-grams. We will be considering upto a trigram model to help us create some good summaries. Unigrams will be generated by stripping off the stop words and we will also generate an n-gram model, where the n-grams are varied based on the model performance.

3 Model

The model performs the proposed task in two folds. First, it uses a Neural Attention Model[3] which takes a list of reviews of particular product as input and assigns attention weights to trigram phrases. Then, we apply beam search algorithm to generate an abstractive summary which is expected to give a general opinion of the product or the service in consideration.

For this project, we will try using Neural Language Model, a Feed-Forward Neural Language model as well as a Recurrent Neural Language model. Then we will evaluate and compare results from each of the model to finalize the one with highest accuracy.

4 Results Evaluations

To get our hands dirty and see how our output would look like, we made use of an pre-trained off the shelf model that is available in the textgenrnn library. We passed the individual reviews file for a restaurant named Gordon Ramsay Steak to the neural network. The neural network was a bi-directional RNN with 64 nodes in it. We ran it only for 1 iteration on our file, since the model is already pre-trained on a large corpus of data.

```
###################
Temperature: 1.0
###################
Foie:  when it was all I an also ordered amazing... pepperse 13,
enjuy is notimate your routhel dish was agoly. <end>  <start> The
Strain Boup i he saw I have ever wanted it enortry this last I
wasn't anioce was really answardly. This is the best did no the
polic though too wsharing the other flavo
```

By glancing at the summary that was generated the number of training epoch were clearly not enough the network did create some garbled summary but it is unreadable and makes no sense..

5 What is Working and What is Wrong

So we know that using a bidirectional RNN's would help us in a big way since they are capable of capturing the syntax and the grammar that persists in a sentence. Increasing the training epoch of the RNN and having a larger number of nodes in the hidden layer may improve the performance of our model. But we will focus on just increasing the number of epochs since time taken for training 1 epoch on 1060 GTX GPU was around 15 minutes. We will try to implement an approach that will leverage the different features that we will be generating on the corpus to leverage better and more consice summaries.

6 Future Work

After getting an abstractive summary, we will perform a sentiment analysis on it to rank the product in 5 levels : 1 = Hated it, 2 = Didn't like it, 3 = It was OK, 4 = Liked it, 5 = Loved it.
We intend to perform this sentiment analysis using various methods including Naïve Bayes, MaxEnt and SVM. Then we will evaluate and compare the results of each of them to finalize the one with highest accuracy.

References

[1] https://www.kaggle.com/yelp-dataset/yelp-dataset,

[2] Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saeid Safaei, Elizabeth D. Trippe, Juan B. Gutierrez, and Krys Kochut. 2017. Text Summarization Techniques: A Brief Survey. In Proceedings of arXiv, USA, July 2017,

[3] Alexander M. Rush, Sumit Chopra, Jason Weston. 2015. A Neural Attention Model for Abstractive Sentence Summarization, Facebook AI Research / Harvard SEAS, September 2015,

[4] Proceedings of NAACL-HLT 2016, pages 93–98, San Diego, California, June 12-17, 2016. c 2016 Association for Computational Linguistics Abstractive Sentence Summarization with Attentive Recurrent Neural Networks, Sumit Chopra, Michael Auli, Alexander Rush/