Product Review Summarization

What did you do and why is it interesting?

For our project, we wanted to create an application that can summarize all the reviews for a product listed on ecommerce websites such as Amazon or Ebay. With our tool, users no longer need to spend time reading and parsing through reviews themselves: all they need to do now is read our application's output which gathers the overall sentiment of all the reviews into a concise summary.

In order to complete this application, we had to use NLP methods and models specifically for summarization and reducing text while keeping context and sentiment. Our main model that we trained for our project is a Sequence to Sequence model (Seq2Seq). We also use other techniques including clustering techniques (DBSCAN, k-means) and pre-trained summarizers (BERT/TextRank) to achieve our goal. In the end, we were able to convert large amounts of reviews into short summaries and review "tags". These "tags" are key aspects of a product that customers weigh or factor the most about while deciding to purchase a product, for example the tags for a snack product could be taste, quantity, texture, nutrition, etc.

How do your model(s) work? (Point out any important details regarding your approaches to unknown words, training, decoding, etc)

Data + Preprocessing:

For our dataset, we chose to use the Amazon Fine Foods dataset. This dataset is a comprehensive collection of reviews for various food products available on the Amazon platform. It has around 500,000 food reviews and a lot of additional metadata about users.

For preprocessing, we performed the usual NLP text processing techniques which include removing stopwords, lowercasing the text, tokenization, and removing punctuation.

Furthermore, for some of our models we performed dataset subsetting, where we only include data points that meet a certain criteria. For example, for our sentence summarization model we wanted our model to generate longer summaries so we filtered out any labels in the dataset that had less than 3 tokens.

Seq2Seq model:

For our Seq2Seq model, we took an existing architecture we found online[1] and fine-tuned its hyperparameters through numerous experiments. Our primary objective was to adapt and optimize the model to specifically cater to summarizing product reviews. The model's architecture consists of an encoder and a decoder. In the encoder, we utilized LSTM layers for their ability in processing sequential data and introduced dropout techniques to prevent overfitting. The encoder effectively captures the context of the input text, which is then passed to the decoder.

The decoder, similarly structured with LSTM layers, begins with an embedding layer. It uses the context provided by the encoder to generate the output text. This setup allows our model to maintain the input text's context, which is crucial for producing coherent summaries. An essential feature of our decoder is the incorporation of an attention mechanism, which aids in focusing on specific parts of the input text while generating each word of the summary. This mechanism, implemented via attention.py, enhances the model's ability to generate relevant and contextually accurate summaries.

For inference, our model employs the decode_sequence function which transforms the contextual information encoded by the model into coherent summaries. This function operates by taking the encoded input text and initiating the decoding process with a start token, signaling the beginning of the summary generation. The decoder, now with the context captured by the encoder and this initial token, begins its task of generating the summary, one word at a time.

At each step of this process, the decoder predicts the next word in the summary based on the current context and the sequence of words generated so far. This is where the attention mechanism comes into play, allowing the decoder to focus on specific parts of the input text that are most relevant to the current word being generated.

As the decoding proceeds, the internal states of the decoder are updated, reflecting the summary at each point in generation. This iterative process continues until the decoder produces an end token, or the maximum summary length is reached. The output is a concise, context-aware summary that accurately represents the input text.

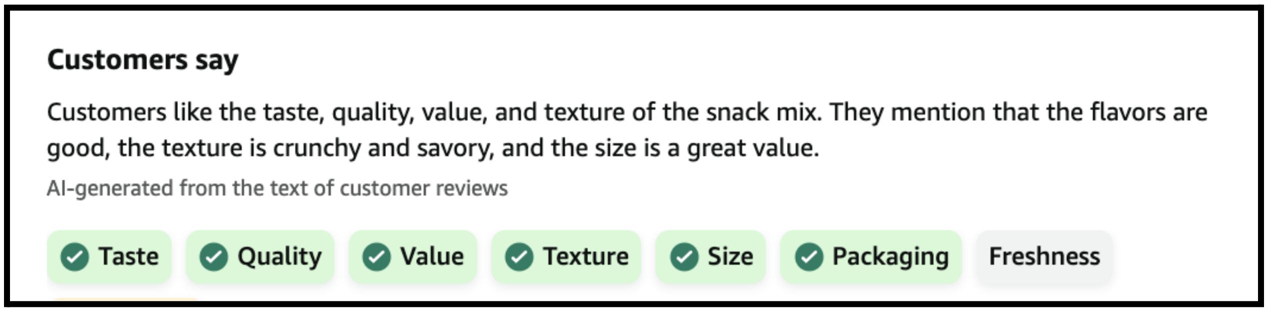
End to End flow:

Here is the an overall flow for our how we obtain our outputs:

1. Gather customer reviews for food products on Amazon.
2. Feed each review into our Seq2Seq model to get short phrases
3. Use DBSCAN to cluster these phrases into their respective types
4. Call our Seq2Seq model again on each cluster of phrases to obtain a final summary for that cluster
5. Compare it to ground truth (the actual Amazon review summaries) to obtain a final sentence similarity metric for evaluation of our model.

What are your results/conclusions?

In the end, we were able to successfully train our models and convert reviews into shortened summaries. Our application produces two types of outputs: 1-4 short sentences that summarizes all the reviews and a list of positive/negative tags. To evaluate how well our outputs are, we used Amazon's review summaries (a few sentences) and tags as a ground truth to compare our summary to. Here is an example below:


Customers say

Customers like the taste, quality, value, and texture of the snack mix. They mention that the flavors are good, the texture is crunchy and savory, and the size is a great value.

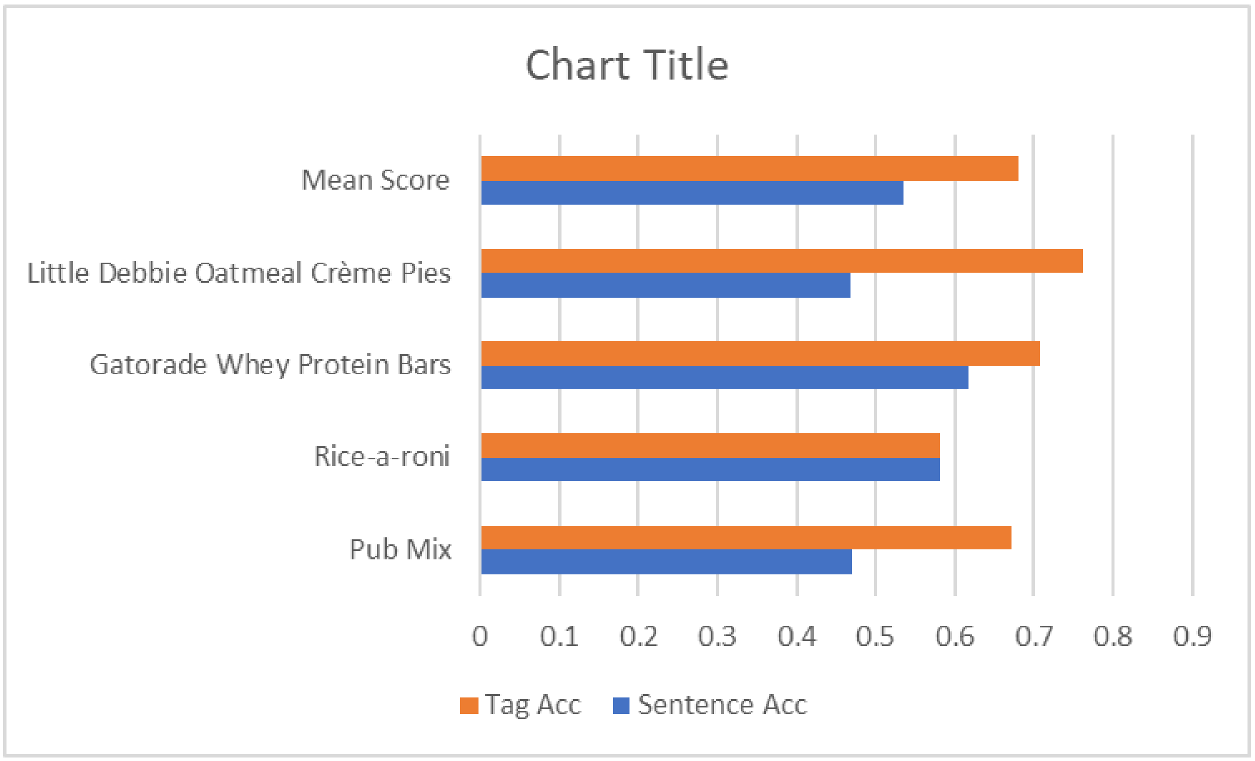
AI-generated from the text of customer reviews

For our metric, we used word embedding similarity which generates a score from -1 to 1, where the higher the score indicates the more semantically close in meaning a sentence is to another.

We created our own dataset for evaluation, where we visited Amazon products and collected customer reviews along with Amazon's sentence summary + tags.

For tags:
Our tags compared to Amazon's tags (both converted to a simple sentence using a template) achieve a mean score of 0.681.

For sentence summaries:
Our sentence summarization model's output compared to Amazon's sentence summaries achieve a mean score of 0.534 in similarity.



Future experiments/work? (What would you want to do next, given more time?)

Given more time, we would definitely want to train another Seq2Seq model on a dataset that has sentences instead of review titles as our labels. A TA did suggest for us to use a GPT-2 model to generate sentence summaries for labels for every review in our dataset, but that approach did not work too well. If we had more time, one option to achieve this goal is to have a more complex model such as GPT4 create sentence labels for us.

Also, for evaluation, it would be nice if we were able to have more products and reviews so we can test our application out in various scenarios.

We believe this type of application has a lot of use cases apart from product reviews. For instance, this tool could be trained to also summarize service, restaurant, or location reviews. If we had a viable dataset, fine-tuning our model for these types of reviews would definitely be something we are interested in trying in the future.

Works Cited

1. https://www.kaggle.com/code/singhabhiiitkgp/text-summarization-using-lstm/notebook
2. https://huggingface.co/learn/audio-course/chapter3/seq2seq
3. https://www.analyticsvidhya.com/blog/2018/11/introduction-text-summarization-textrank-python/#:~:text=TextRank%20is%20an%20unsupervised%20graph,represent%20the%20relationships%20between%20them.
4. https://www.kaggle.com/code/nulldata/fine-tuning-gpt-2-to-generate-netlfix-descriptions