CIS 522 – Final Project – Technical Report

Re:Summary

April 2022

Team Members:

- Yuying Fan; yuyingf; Email: `yuyingf@seas.upenn.edu`
- Rehaan Furniturewala; rehaan; Email: `rehaan@wharton.upenn.edu`
- Andrew Raine; andrewlr; Email: `andrewlr@seas.upenn.edu`

Abstract

E-commerce websites frequently contain an overwhelming number of reviews for a specific product, and a serious online shopper has to spend a significant amount of time and energy reading these reviews in order to extract useful information. To address this problem, we develop a summary system that summarizes multiple Amazon product reviews into a word cloud with words selected and weighted by a combination of sentiment score and frequency. The system consists of a star-rating prediction model which predicts review rating from review text, a masked-review generator to mask adjectives, and a sentiment-score calculator that compares the rating prediction difference between a masked review and its original review. The word cloud summary contains adjectives with the highest sentiment score and highest frequency in the input reviews. We experiment with multiple model choices for the star-rating prediction model including a random forest, a 5-layer stacked RNN, and an advanced model using pre-trained BERT followed by a multi-layer RNN with dropouts. We find that the pre-trained BERT model predicts with the highest accuracy, and a system using the BERT model produces meaningful word cloud summaries for products. Our system can be deployed to e-commerce websites and adapted into domains including business and research to make information extraction from online text more efficient.

1 Introduction

Popular e-commerce websites like Amazon.com feature millions of products, many of which solve the same need. Each product page features a gallery of images, a description of the product, an average rating out of 5 stars produced by previous purchasers of the product, and customer reviews. At its core, each

customer review consists of a text description of the customer's opinion of the product as well as a numerical rating out of 5 stars. For each type of product, there are dozens and sometimes hundreds of competing models from different manufacturers at different prices. For example, searching up "mug" on Amazon produces 7 pages of results, featuring a total of more than 300 models of mugs. To pick which model to purchase, Amazon customers use a combination of product images, product descriptions, as well as customer reviews, and ratings.

Since the product descriptions are written by the seller of the product themselves, there is a financial incentive for the seller of the product to make their product look better than it is. As a result, the product descriptions can be biased. On the other hand, customer reviews are generally unbiased, since the customer cannot gain or lose from the consequences of someone else reading their review. As a result, customers value reviews as a core part of their purchasing decision.

However, even though customers can read thousands of reviews about each product to learn about them, customers lack the time to read each review in order to make the optimal purchasing decision. As a result, many customers only read a few review as a proxy for how good the product is, and do not benefit from the insights of the majority of the reviews, which could possibly change a user's purchasing decision. In order to help Amazon users save time and understand a product better, Amazon shows the "top reviews" for a product before showing all the reviews for the product. While these "top reviews" may be more useful than the average review, the "top reviews" are still written by individuals and do not necessarily reflect what the majority of purchasers of a product thought about the product. Currently, the 5-star rating, which averages each person's individual rating for the product, is the only statistic that summarizes how everyone felt about the product. While informative, the 5-star rating fails to provide a potential customer with any qualitative information about how the average purchaser of the product felt.

In order to improve the purchasing experience on Amazon and other e-commerce websites, we built an algorithm to summarize every review for a product into a word cloud. We aimed for our word cloud to help the customer learn the most important aspects about a product in as little time by factoring in each review. If successful, our method could be incorporated into every e-commerce product page and revolutionize the way that people buy products online. While others have written papers about summarizing an individual review into a sentence, we could not find any papers using deep learning to summarize multiple reviews into a single "super-review". Currently there are methods to summarize reviews and display sentiment aggregates for reviews, however, a way to create "super-reviews" does not exist; this is the aim of our project.

Our approach entails encoding product reviews, predicting star rating scores

based on those encodings, sorting adjectives in each review based on their importance ("rating impacts" moving forward), and then generating word clouds based on said adjectives. This would give customers an idea of the most important traits of the products they are purchasing from crowd-sourced reviews, and offer an unbiased and more efficient way of gauging product quality. The path to achieving this goal is described henceforth in this paper and in our code repository.

2 Related Works

Related work most similar to ours entails product review summarizing and phrase frequency based word clouds. Lu Yang[1], David Currie[2], Ravali Boorugu et al.[3] all use some form of embedding generation fed into a RNN model to compress product reviews into short summaries (click on names to view sources). Hua Shi[4] splits all product reviews into three sets, negative ($\leq$2 stars), neutral (3 stars), and ($\geq$ 4 stars) and then generates word clouds using word frequency for each subset of reviews. While Hua Shi's project generates a visual "super-review" like ours, the words in her review have no qualitative meaning other than how many times they appear in all reviews.

Our project essentially combines both of the methods mentioned above when to generate informative, impact-weighted word clouds for Amazon products. Similar to the first few researchers, we use a deep learning model to embed text reviews. Where they used seq2seq and RNNs to encode text, we use a pre-trained BERT encoder fed into a custom feed-forward network outlined in section 4.1. Where Hua Shi used word frequency to create her word clouds, we calculated score impact ratings for adjectives and then generated product-specific word clouds, described in section 4.3. In doing so, our project is able to offer customers more holistic view of the key qualities belonging to the products they are purchasing.

3 Dataset and Features

We used a dataset of Amazon reviews from a research group at the University of California San Diego[5]. Each review is of the format: (overall_score, verified_purchase, review_time, reviewer_id, product_id, style, reviewer_name, review_text, summary). The only features we used to train our models were overall_score, which is a whole number rating of the product from 1 to 5 stars, and review_text, which is a text review of the customer's thoughts on a given product. While we did not use the product_id as a feature for the models, we did use product_id to group our reviews by their unique product id, so that we could produce our "super-review" on that product.

Our dataset used reviews from the following product categories on Amazon:

'All-Beauty', 'AMAZON-FASHION', 'Appliances', 'Arts-Crafts-and-Sewing', 'Automotive', 'Books', 'CDs-and-Vinyl', 'Cell-Phones-and-Accessories', 'Clothing-Shoes-and-Jewelry', 'Digital-Music', 'Electronics', 'Gift-Cards', 'Grocery-and-Gourmet-Food', 'Home-and-Kitchen', 'Industrial-and-Scientific', 'Kindle-Store', 'Luxury-Beauty', 'Magazine-Subscriptions', 'Movies-and-TV', 'Musical-Instruments', 'Office-Products', 'Patio-Lawn-and-Garden', 'Pet-Supplies', 'Prime-Pantry', 'Software', 'Sports-and-Outdoors', 'Tools-and-Home-Improvement', 'Toys-and-Games', 'Video-Games'. Since we did not have infinite compute resources, we limited our dataset so that it could only use a maximum of 200,000 reviews per category.

The subset of the full dataset that we downloaded for training contains 75,257,650 reviews. Due to our limit of 200,000 reviews per category, our dataset shrunk to 4,247,792 reviews. We then removed duplicate reviews, reviews with empty text, and reviews with more than 100 characters, which further limited our dataset to 1,029,752 reviews. We removed reviews with more than 100 characters, because we later hypothesized that our model was better at summarizing shorter reviews compared to longer reviews, since we thought that longer reviews would be more repetitive and harder for our model to summarize than shorter reviews. However, the distribution of reviews based on their star ratings was heavily skewed. There were more than 28 times the number of five-star reviews compared to two-star reviews in our dataset. As a result, we shrunk our dataset further in order to ensure there was an equal amount of reviews for each star ratings, thus leaving the dataset at 136,530 reviews.

asin	starRating	reviewText
B0000CFN90	2	"didn't stick"
B000X4OLS0	5	"It works ok it did what I bought it for."
B000HDKZK0	1	"I found them unsatifying and the taste just ok. Note that you cannot return."
B0007M1ZGE	4	"worked as advertised I put a cargo hauler on min to put generator satilite works good"
B000XZTIIM	5	"wonderful product excellent service"'

Figure 1: Sample Reviews

4 Methodology

For all of the following models we used a 90/10 split on the dataset described in section 3.

4.1 Architectures

We began by establishing a non deep learning benchmark using a random forest which comprises 100 decision trees. After training on 90% of our data we reached a validation accuracy of 20.93%. This will be the baseline we try to beat moving forward.

For a simple deep learning model, we used a stacked RNN with 5 layers and

a hidden size of 50, and then fed the hidden layer outputs to a linear layer. Our best RNN model achieved a validation accuracy of 39.65%.

For our advanced deep learning model, we used a pre-trained BERT model[6] to extract review embeddings and then, initially, fed those embeddings through a single linear layer. Our second iteration of this design fed the embeddings through two linear layers with a ReLU activation layer. Our third, and final, iteration fed embeddings through four linear layers, with ReLU activations after the first three layers, dropout(0.2) on the first linear layer input, dropout(0.5) on subsequent inputs, and skip connections from the first input to the second output and third output. Each of these achieved accuracies in the low 60%s, with a caveat described in the Results section.

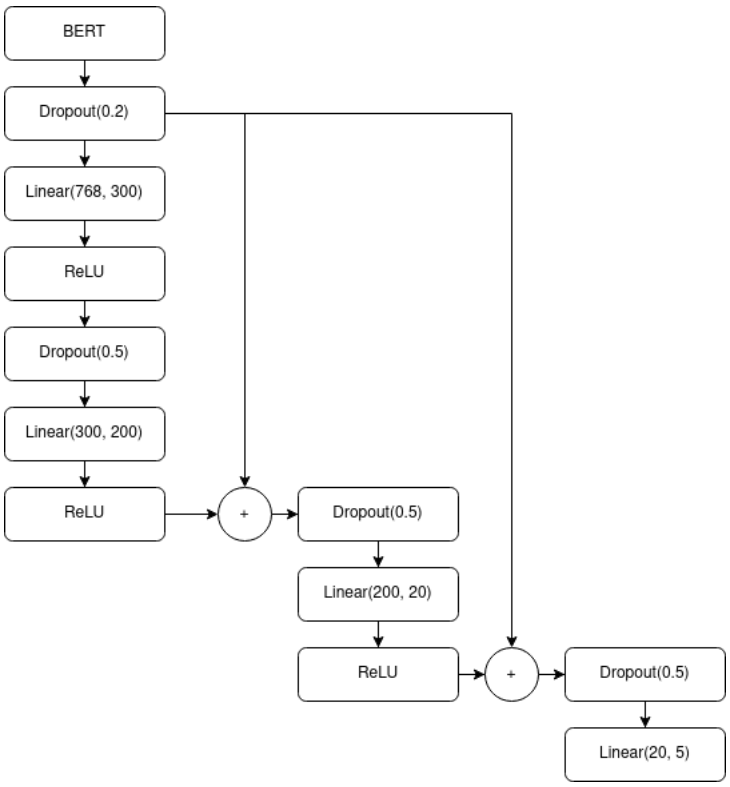

Figure 2: Our Final Architecture

4.2 Hyperparameter and Optimizer Optimization

Due to GPU and time constraints, we only tested the effects of doubling batch size, learning rates, and using SGD vs. ADAM,. We found that doubling the batch size from 16 to 32 reviews yielded a smoother validation accuracy

vs. time graph and offered a small accuracy improvement, that the most effective learning rate was 2e-5 (possibly due to the fact that our BERT model is pre-trained), and ADAM performed immensely better than SGD for our task (25.53% accuracy vs. 61.84% after one epoch).

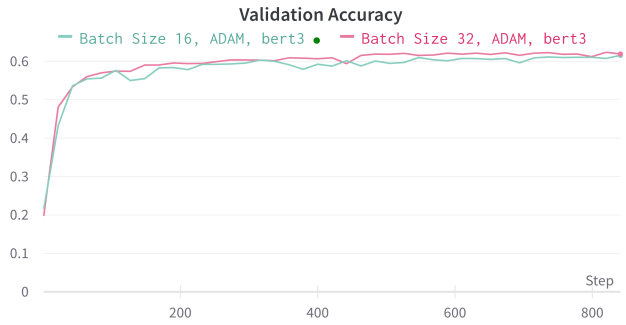

Figure 3: The Effect Of Batch Size On Validation Accuracy

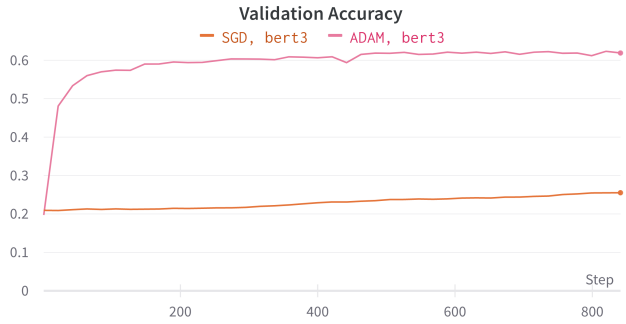

Figure 4: The Effect Of Optimizer Choice On Validation Accuracy

If we had more GPU power and runtime we would test the effects of a larger range of learning rates, batch sizes, optimizers, and even different architectures for our simple deep learning model. A larger range of learning rates would ensure that we are not getting stuck in local minima, allowing us to 'jump' over them or 'sink' into global minima. A larger batch size could reduce the noise for each gradient step we take, essentially giving us a 'true' estimate of the correct gradient descent direction. Although we conjecture that ADAM is the best optimizer for our task, testing out many others may yield positive results. Deepening our RNN architecture may help it detect higher level features in our dataset and thus perform better.

4.3 Adjective Rating Impacts & Word Clouds

For each Amazon product, we run all text review through our trained model to collect score predictions. Then we mask each adjective in each review (for example the review "This product was wonderfully designed." is masked to "This product was [MASK] designed.") and get the predicted scores for those too. With those two sets of predicted scores, for each adjective we can assign a mean "rating impact score", which is found by taking the absolute value of the difference between the masked review score and the original review score. In essence, we are recording how much of an impact each adjective has on the its review rating (1-5).

With a mean rating impact score for adjectives from any particular product, we can sort by impact scores and thus generate word clouds for each product (depicted in section 5).

5 Results

Architectures

Our benchmark to beat was the non-deep learning model of a random forest, with an accuracy of 20.93%, on par with random guessing. The 5 layer RNN that we used as a simple deep learning model achieved a peak accuracy of 39.65% after several iterations of training. We say 'peak' accuracy for the following reason: we trained multiple instances of the exact same RNN architecture using different, random weight initializations. All instances but one could not surpass random guessing accuracy (20%), and the one that did achieved the high accuracy above. This speaks to the 'lottery' aspect of weight initializations, where lucky initializations can achieve greater accuracy.

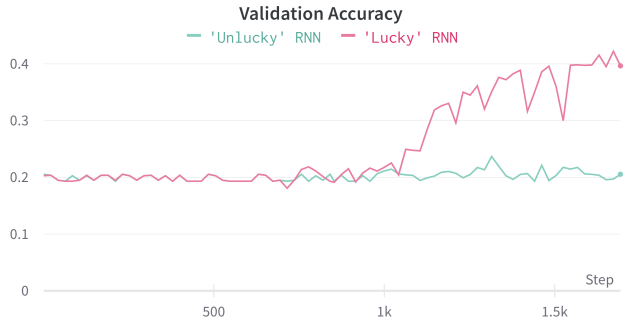

Figure 5: "Lottery" Weight Initialization Effect

For the three advanced deep learning architectures we tested, explained in section 4.1, we achieved roughly similar accuracies in the low 60% range. Al-

though they all performed similarly, the adjective impacts found by each varied widely. This is notable as our project aims to discover which words in a review are the most impactful to the review rating. The first two models produced few impactful adjectives (adjectives comprising only 1-2% of the review texts) while the final model produced many more (adjectives comprising 10-20% of the review texts).

The random forest model was trained using Gini impurity as a loss function (inbuilt to the Sklearn implementation that we used). While the RNN and custom BERT based model used the negative log likelihood loss. In our prototyping stages we also tested Cross-Entropy loss but found that the former achieved greater accuracies.

Adjective Rating Impact & Word Clouds
Using our trained networks to predict ratings on the original review texts and masked review texts, as described in section 4.3, we are able to determine the rating impact that each adjective for every product has. An example word cloud generated from adjective impacts for the product "The Hobbit" is shown below.

Figure 6: Word Cloud For *The Hobbit* Fantasy Novel

6 Discussion

We trained a random forest, a simple stacked RNN and a complex model using pre-trained BERT followed by a RNN to predict star-ratings based on review text for Amazon product reviews. We then ran our advanced model on modified reviews each with one adjective masked and compare the predictions with

their original review to identify adjectives with the greatest influence to review ratings. By doing so for multiple reviews of the same product, we can extract a list of significant adjectives from the reviews, each with its score for influence to star-rating prediction. These words can then be aggregated in a word cloud to generate a summary for the multiple reviews for a certain product.

6.1 Findings

Our random forest model achieved an accuracy of only 20.93% in predicting star rating from review text. Given that the label space is $\{1, 2, 3, 4, 5\}$, the random forest gives virtually no improvement to a model that simply makes random predictions. This is likely due to the high variance in our data: the reviews for different products can differ drastically, and even reviews for the same product can focus on different aspects of the product. There is no "typical" review for a specific star rating. As a result, the decision trees trained likely don't have sensible split standards. With the high variance, the validation set can also differ greatly from the training set. Together, these factors cause the random forest to be incapable of predicting star ratings from the review texts. The 5-layer stacked RNN achieved 39.65% validation accuracy in its best run, which is better than chance and than the random forest model. Our advanced model using BERT performed the best, with validation accuracy in the low 60% for multiple runs. Comparing the two deep learning models, the key for the performance boost in the advance model is likely the longer memory span and the attention mechanism. In the stacked RNN, the impact of previous tokens quickly diminishes after processing new ones, which means the sentiment prediction relies largely on the few ending words in a long review. Such short memory clearly limits the model's capability to grasp the sentiment of the entire review. Also, most words in a review do not carry a sentiment, such as pronouns, prepositions, and words that help set up the sentence or the context. The stacked RNN's performance is also hindered by distractions given by these words. In comparison, the advanced model has an attention mechanism pre-trained in BERT, so it is capable of processing words with selective weights to reduce the amount of distraction and keep track of the earlier tokens better.

Currently, there exists tools for summarizing individual reviews and for generating most frequent phrases from reviews, but generating summary words from multiple reviews is a novel attempt. In addition to considering word frequency, our model takes semantic meaning into account and emphasizes words that likely carry more sentiment to produce a more informative summary word cloud. Our work can be deployed on e-commerce websites to greatly reduce the overhead of reading dozens of reviews in search for useful comment on the product. Our work has the potential to dramatically reduce the amount of time human individuals spend in search for useful data in this age of information overload.

6.2 Limitations and Ethical Considerations

The main limitation to the functionality of our model is that we currently only use a small portion of the available data in review texts. Specifically, we only used the reviews whose lengths are under 100 characters. Longer reviews are more difficult to handle with our current design as the effect of masking a single adjective diminishes as the length of the review increases, so it becomes harder to identify influential adjectives in long reviews. We also frequently include generic words such as "great," "good," and "bad" in our final summary; these words indeed carry sentiment, but they are not as informative to the users as descriptive words like "fragile" or "sticky."

Ethically, a potential problem that can come with deploying our model to e-commerce websites is that malicious attack using spam comments becomes easier. When humans read the entire comment, a spam comment may sometimes reveal to be obviously fake. However, if the machine is the reader, spam comments are processed the same way as real comments, and an attacker may control what shows up in the summary word cloud by adding a lot of fake comments with certain keywords. If the human solely relies on the summary word cloud, it's much easier for them to become misguided by the fake comments.

6.3 Future Research Directions

There are a number of things we can try to improve our current model. First, during data preparation, we can split long reviews into multiple text pieces each with the same rating label instead of discarding all long reviews. This will greatly increase the amount of information we are able to take into account for when generating a review summary for a specific product. Second, we can generate a list of generic words including ones like "good" and "bad" and ask the model to filter these words out of the final result to produce more informative summaries. Third, we should train a model that distinguishes genuine reviews from the spam ones and use it to filter which reviews to process when generating a summary. This will defend against the malicious spam attack. If we have more time and compute power, we should also train the star-rating prediction model on a greater number of examples, as our model's performance continuously increased when trained on more examples, and there is still a large fraction of reviews from the dataset that we did not use.

Our pipeline can be further extended to domains outside e-commerce to give summaries to any groups of text when coupled with a sentiment analyzer, such as generating overviews of survey results. Hence, with some adaptation, the model can be useful in the fields of market research and social research as well.

7 Conclusions

In this study, we developed a system to generate summary word clouds from multiple Amazon reviews. We find that a deep-learning model using pre-trained BERT followed by RNN gives the best performance on star-rating prediction based on review text compared to simple machine learning methods and simple RNNs. Using the star-rating prediction model, our system is able to factor semantic meaning of words into account and generate more informative summaries than the state-of-art frequency word cloud. Our system has the potential to aid effective information extraction in an era of information overload.

8 References

[1] Yang, L. (n.d.). Abstractive summarization for Amazon Reviews. Retrieved April 28, 2022, from https://cs224d.stanford.edu/reports/lucilley.pdf

[2] Currie, D. (2017, May 9). Text summarization with Amazon Reviews. Medium. Retrieved April 28, 2022, from https://towardsdatascience.com/text-summarization-with-amazon-reviews-41801c2210b

[3] Boorugu, R., Ramesh, G., amp; Madhavi, K. (n.d.). Summarizing Product Reviews Using NLP Based Text Summarization. INTERNATIONAL JOURNAL OF SCIENTIFIC amp; TECHNOLOGY RESEARCH. Retrieved from http://www.ijstr.org/final-print/oct2019/Summarizing-Product-Reviews-Using-Nlp-Based-Text-Summarization-.pdf

[4] Shi, H. (2020, June 27). Natural language processing (NLP) - Amazon Review Data (part II: EDA, data preprocessing and model... Medium. Retrieved April 28, 2022, from https://melaniesoek0120.medium.com/natural-language-processing-nlp-amazon-review-data-part-ii-eda-data-preprocessing-and-model-3866dcbdbb77

[5] Ni, J. (n.d.). Amazon Review Data (2018). Amazon review data. Retrieved April 28, 2022, from https://nijianmo.github.io/amazon/index.html

[6] Jafer, N. (2020, March 16). Fine tuning Bert for Amazon Food Reviews. Medium. Retrieved April 28, 2022, from https://medium.com/analytics-vidhya/fine-tuning-bert-for-amazon-food-reviews-32e474de0e51