Natural Language Processing: Comments Summarization

UC Davis EEC 193A/B Senior Design Winter 2021
Divya Karthik, Jonathan Li, Daiguang Zhang

Abstract—**Public comments are integrated into almost every internet platform today. These lengthy discussion sections, while insightful, can often become overwhelming for the decision-making user to read through. The comment summarization tool we developed allows the user experience to be free of duplicate, biased, and irrelevant data. By filtering through comments based on topics discussed, the user can efficiently gauge public view and sentiment about a feature of the content they are interested in. In the form of a Google Chrome extension, this tool allows users to succinctly explore the gist of the comments section of a website and extract the most meaningful data.**

I. INTRODUCTION

In our world there is data everywhere. People currently must sift data for relevant information. In many internet platforms, public discussion is considered extremely important in terms of providing content feedback and insightful divergent thinking. To capitalize on these insights, YouTube has a dashboard to show creators their analytics. Similar dashboards exist for Amazon Sellers and in Google Analytics.

For consumers/viewers, YouTube is architected to consume people's time and has scrolling comments that keep people on the platform longer and no way to sort comments by date, sentiment, and topic. Comments are often irrelevant: shoutouts, thanks, or troll comments. On The New York Times people want to read the core gist from the news and care not about minor details and quotations. Perhaps they want articles communicated in a couple sentences. Amazon presents customers with product descriptions and top reviews to speed up the buying process. However, buyers may instead want summaries of the one, two, three, four, and five star reviews, a feature which does not yet exist. On Yelp likewise users may want a summary of each of the five stars. If a summary were generated, it would need to be tailored for users to be relevant. For example, some Yelp users care about restaurant customer service while others care about meal portions size.

In general people want a summary. If they do not need to understand the details, then they will not. People care about convenience and want them in bite sized pieces. People want others to make decisions for them and that is why referrals are so common in industry. People want fast and easy. We are inspired by the CNN Business video "Bill Gates Would Start This Kind of Company Today" in which Gates states that the company he would start today would be one that teaches machines to read and understand the world's knowledge. Our motivation is to summarize text, save people time, freeing them up to do higher level thinking instead of waddling in the details of reviews and comments.

Using natural language processing and text summarization, we want to summarize public discussion on platforms to provide the audience with meaningful insight and information so they can more easily make decisions. The trend in technology is for more information to fight for fewer human attention. The challenge is to prioritize human attention to the most relevant information. Other related projects in academia and industry include automatic text summarization using the Bert model and the summarization of clinical information, documents, and videos.

II. PROBLEM STATEMENT

When shopping, entertaining, learning, and consuming news, people have too many options. Reviews and comments proliferate and are being generated every day at a faster rate than people can consume. It is not uncommon to be overwhelmed and to see 2000+ comments on a news article or product. There are several problems with the current user feedback and recommendation systems:

A. Customers have not enough time to read through data.
B. Data is sometimes contradictory, irrelevant, or a burden to process.
C. Bias and prejudice creep into comments.
D. Distraction steers people down information rabbit holes destroying hours of productivity.
E. Duplicate information means people get few new insights.

III. RELATED WORK

In industry there are many approaches to text summarization. One method is to feed an extractive summary into an abstractive summary [1]. We tried this approach when we extracted topics from the comment text using Spacy and they passed all sentences relating a particular topic into Google's Pegasus abstractive summarizer. What we found was that Pegasus abstracted too far and often added extra information to the text. Therefore, we decided not to use Pegasus. Another abstract summarizer BART, a crossover between BERT and GPT, was a viable summarizer model we are considered as it is not as ambitious as Pegasus with adding random information to the summary.

Another idea for text summarization involves using fuzzy lattice reasoning to capture both topic and sentiment [2]. We took inspiration and represented topics using Spacy word vectors that can be compared using cosine distance. Because some domains display the category of the business, product, or service, we made sure to take advantage of that. For example, in Yelp a boba shop may be categorized as bubble tea, juice bars, and smoothies. In that case we would look for topics that match bubble tea, juice bars, and smoothies. As for sentiment, we used the SpacyTextBlob pipeline which integrates into Spacy the TextBlob sentiment analysis library as well as Affin which is another sentiment analysis library.

Academic literature offered us potential NLP techniques to try out. One journal discusses summarizing text using aspect extraction, sentiment analysis, and topic modeling [3] and we went with the latter two techniques since the first strategy of aspect extraction does not have a solid agreed upon definition. Another paper suggests that selecting YouTube comments by frequency captures most of the meaning [4]. Keeping this idea in mind, we went with a hybrid approach: choosing topics from the comments based both on frequency and relevance to the category business, product, or service that the comments or reviews are describing. Another paper in tourism management brought up weighing hotel reviews by helpfulness in addition to sentiment. This sparked us to consider that perhaps we can eventually weigh comments based on the reputations of the reviewers. Most importantly we realized we can rate our review based on conclusions already drawn from domains. For example, we can benchmark our overall sentiment score against Amazon and Yelp's five-star rating system and YouTube's like-to-dislike ratio.

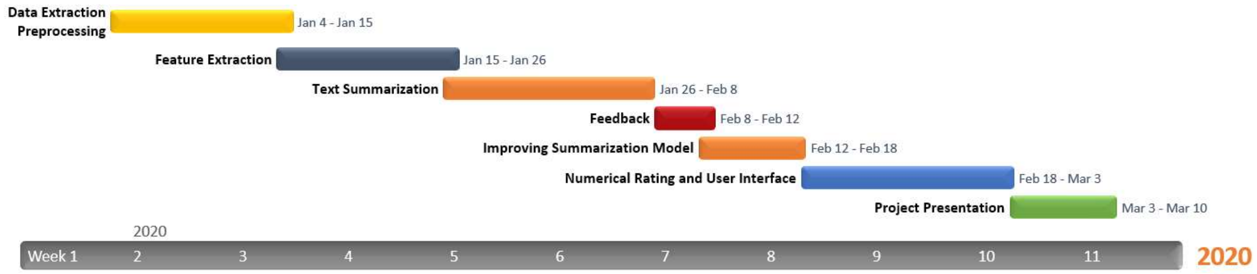

FIGURE 1: GANTT CHART TIMELINE

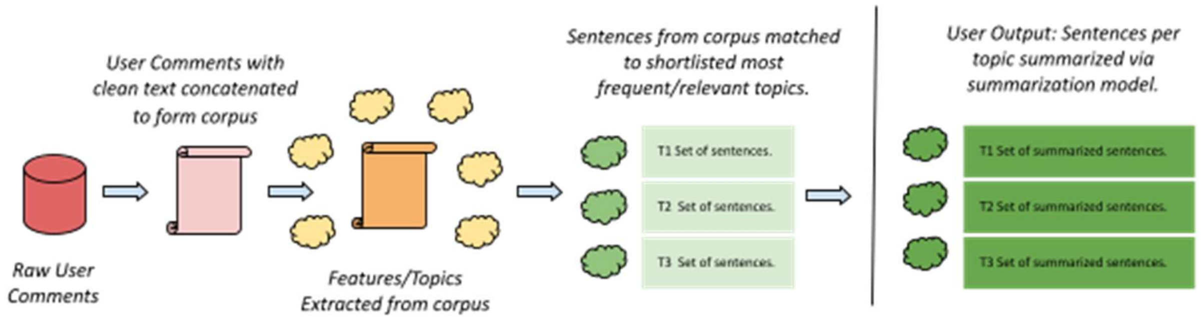
User Comments with clean text concatenated to form corpus
Sentences from corpus matched to shortlisted most frequent/relevant topics.
User Output: Sentences per topic summarized via summarization model.
T1 Set of sentences.
T2 Set of sentences.
T3 Set of sentences.
T1 Set of summarized sentences.
T2 Set of summarized sentences.
T3 Set of summarized sentences.
Raw User Comments
Features/Topics Extracted from corpus

FIGURE 2: PIPELINE DIAGRAM

V. DESIGN METHODOLOGY

A. Project Pipeline

We followed a waterfall design methodology as we went through linear steps one at a time. Before beginning we created a Gantt chart to map our progress and a pipeline visual to represent how we would summarize the text.

B. Deliverables and Results

1) Text Preprocessing

Before doing any feature extraction or summarization on the text we first had to clean up the text. For our raw data, we got Amazon and Yelp through publicly available test sets, for YouTube it was from web scraping and for New York Times it was from Kaggle. From the csv files we extracted particular rows and stitched them together to form a big paragraph of text. Then we used the Spacy python library which we selected for its speed to do the preprocessing. We made the text lowercase and stripped first person pronouns, emojis, and html tags.

2) Feature Extraction

The TextBlob python library produced a list of popular noun phrases and out of them we selected the most frequent not too polarizing phrases to be our “topics”. Our decision to use TextBlob can be credited to its accurate compilation of noun phrases and ability to detect sentiment and subjectivity in text. While there were several options to extract topics from text, TextBlob’s subjectivity filter allowed us to eliminate heavily opinionated topics that would be inappropriate to include in the discussion’s final summary, which is intended to be sorted by the content’s features rather than the user’s sentiment. Once we had these popular noun phrases that described the commentary’s gist, we used a combination of their frequency in the corpus and their relevance to the content to shortlist them and extract their associated sentences.

3) Summarization

We aimed to finalize a summarization model to paraphrase these sentences respective to their topics. We researched the hyperparameters and resources for different models and chose BART, BERT, and T5 to test with our focus group. We asked them to select the best model and tell us how they preferred the summarized results be presented to them.

4) Customizing different domains

With our summarization model developed, we looked to tailor our results for different domains. Our goal was to benchmark our results against existing metrics such as the Amazon and Yelp star rating, the YouTube like to dislike ratio, and the New York Times article category. After looking into all four domains, we would select one domain to be our focus for demonstration and evaluation: Yelp.

5) Sentiment Analysis

We produced our overall summary sentiment using a combination of the Affin and TextBlob python libraries. To check if it was reasonable we decided to compare our summary sentiment with the Yelp star rating to see if they trended together.

6) Chrome Extension JavaScript Front End

To see our summarization in action we decided to build a Google Chrome browser extension. Since we were following the waterfall design methodology, our first step was just to get a chrome extension that could grab an element from an HTML page. Our plan was to scrape the yelp reviews from the client side. If that did not work we would scrape the yelp reviews from the server side and simply use the client side to pass the yelp business url and number of paginated pages to the backend Flask API server.

7) Flask API Python Back End

In our backend which would be an API, we made it a Flask API which takes a POST request with two arguments: yelp url and number of paginated pages. Once this information was received we used python's requests library to grab Yelp pages associated with a particular Yelp business. We cached these pages as not to burden the Yelp website. We then extracted the reviews from the Yelp HTML pages and put them into a big paragraph and passed it to the summarizer. The summarizer would do the summarization and return selected topics and their associated sentences in json format.

8) Connecting Chrome Extension and Flask API

We designed our Chrome extension frontend to query the Flask API backend, receive the json topics and then present the topics and their associated summaries in a pretty way. The first step was to have the Chrome extension call the Flask API using POST request and the second step was for it to receive the response.

9) Designing a UI Chrome Extension

Now that we have received a json with topics and their associated sentences, our final step was to convert the json to topic buttons that displayed their associated sentences. We looked online for solutions to see whether we would have JavaScript fill out fields in popup.html or whether it would straight up inject HTML elements into popup.html.

VI. EVALUATION

A. Google Survey

We first evaluated our progress by surveying a focus group. With the user survey, our goal was to gauge user interface preferences and to select the best summarization model for our final product. In researching summarization models, we observed that extractive and abstractive summarization models had their pros and cons. Abstractive summarization models had more coherent outputs but did so by adding extra - often unrelated/false - sentences and statements. Extractive Models chose the most salient comments from the input, but had an unorganized output built from the concatenation of the salient sentences. Considering these pros and cons and adjusting the hyperparameters to find a balance, we shortlisted three summarization model candidates: Bert (Extractive), BART (Abstractive), and T5 (Extractive). We presented these selected summarization models and their respective summaries towards the same group of comments to our focus group. The survey also included questions to measure the efficacy of our progress and results. Based on 25 responses we learned that:

A. 59% of users are overwhelmed by scrolling through comments

B. Despite the information overload, 86.4% of users are interested in knowing what other users are saying about the content they are viewing. Over 50% of users read comments to make a decision about the content and gauge the public sentiment.

C. Users prefer summaries for platforms with reviews, news articles and YouTube videos. Users prefer to scroll through comments on Reddit, TikTok, Twitter and Instagram.

D. User pointed out it's important to provide different topics of information for different summaries rather than one long paragraph of summary.

In terms of the performance of different models we learned that model preferences change with the domain in question:

Domain	Best Model	Average Model	Worse Model
Amazon	T5	BERT	BART
New York Times	BERT	BART	T5
YouTube	BERT	BART	T5
Yelp	BERT	BART	T5

FIGURE 3: SUMMARIZATION MODELS COMPARED ACCORDING TO USER FEEDBACK

After taking into consideration the survey feedback, we decided to choose Bert as our final choice for the summarization model. We also modified our summarization output from one long paragraph to multiple short paragraphs associated with each topic.

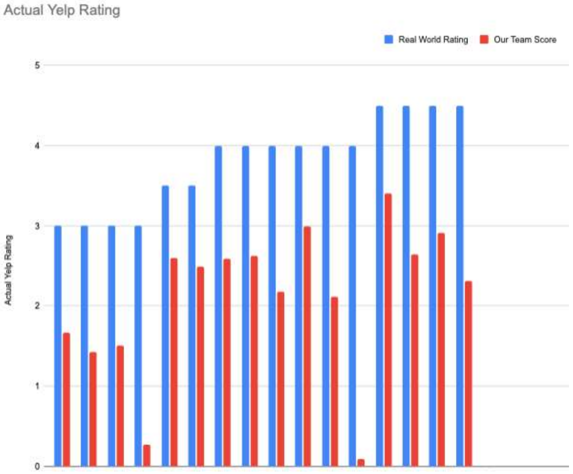

FIGURE 4: SUMMARY SENTIMENT VS YELP RATING

A. Sentiment Analysis

We also evaluate our summarized texts by running TextBlob sentiment analysis on all the topic summaries and comparing the score it generates with the actual Yelp ratings score. By comparing the two, we were able to gauge how accurate the sentiment of our summarization was with respect to the original comments.

From the normalized data in the chart above, we can see that the sentiment score for the summary generated by our model generally correlates to the actual Yelp rating. Especially when we compare Yelp comments with a rating of 3 star to Yelp comments with a rating of 4.5 stars, the sentiment scores from 4.5 stars summary are consistently higher than the sentiment scores from 3 stars summary. For sentiment analysis with very low scores, it is because some businesses have very few comments so only one topic is generated, and the summary of that topic is made up of mostly unhappy comments.

VII. RESULTS

A. Intermediate Results

1) Text Preprocessing

While preprocessing our text we learned the csv files are large. We solved this problem extracting yelp reviews from the csvs and putting each review into its own text file. We also merged a csv containing yelp business and another one containing yelp reviews into new csv. In the new csv each row corresponded to a yelp business and its associated reviews.

2) Feature Extraction

While originally we used the NLTK python library to select topics by looking for bigram and trigram frequencies, we soon realized the TextBlob python library could do a better job.

Extracted Features for Test Domains	
Yelp: South Point Hotel	Place, hotel, hotel room, staff, everything, strip. Resort casino, hotel rate, buffet, mandalay bay hotel, pool, vegas, hotel room elevator, vegas hotel, hotel price, hotel guest, star hotel, food, hotel casino, south point hotel, big room, price, reservation desk, time, lot, hotel registration, casino, room, bar
YouTube: Easy origami dinosaur tutorial	Step, tutorial, origami, split, dino, foil color origami, origami claw, clip, origami artist, paper, origami paper, version, dinosaur
New York Times: Senate Republicans	Nuclear option, democrat, filibuster, even filibuster, garland, gop, government, justice, majority, McConnell,

Deploy 'Nuclear Option' to Clear Path for Gorsuch	politically motivated rejection, nominee, republican, nuclear "yes" vote, party, option, people, denial strategy, power, president, proposed filibuster, rule change
Amazon: Red Heart Super Saver Yarn, Orchid	Craft, simple easy craft, red heart yarn, quilting, price, local craft store, skein, amazon, rainbow art project, crochet project, crochet lesson, color, art blanket, afghan color, art, brand, kid's project, learner, patrick's day craft project, cmas decorating, product store

FIGURE 5: SAMPLE TEXT FEATURES EXTRACTED USING TEXTBLOB FROM TEST DOMAINS.

3) Summarization

From survey feedback of 25 responses from Facebook Davis Computer Science Club and Slack EEC193B Group, we found that most users preferred the BERT model. On the one occasion when T5 was voted as best, BERT came in at second place. Because BERT performed best overall, we elected it as our summarizer model. We also found users preferred filterable topics over a long paragraph summary.

Sample Summary Outputs: Yelp Comments	
Bart	The pool was very nice and had a wading pool for the kids . It gives the best stone massages in the world, they have a steam room, sauna, pool, jacuzzi, and relaxation room .
Bert	The pool was very nice and had a wading pool for the kids. It gives the best stone massages in the whole world, they have a steam room, sauna, pool, jacuzzi, and a relaxation room. every pool was over-crowded with KIDS KIDS KIDS, even the large juquzzi Spa.
T5	the pool is nothing special Bingo, slots, tables, sportsbook, food, pool, everything is good enough. every pool was over-crowded with KIDS kiDS, even the large juquzzi Spa.

FIGURE 6: SAMPLE SUMMARIES USING BART, BERT, AND T5 MODELS FOR SOUTH POINT HOTEL ON YELP

Sample Summary Outputs: YouTube Comments	
Bart	Mark Zuckerberg is wasting a lot of his money just for the sake of over-glorifying his own image to the public . Facebook use to be a place where people could keep up with friends and make new friends..but money got involved..and they were naive .
Bert	yes he didn't want Hillary at the WH because he needed to make money of the Russians and get his huge tax cut which Trump gave. its disturbingZuckerberg is more interested in making money than anybody else's rights to privacy and correct informationIf Mark is really just wasting a lot of his money just for the sake of over-glorifying his own image to the public. , Money cant buy Stop treating your users as commodities and making money by selling them!Facebook crash? and now it's all about the money.
T5	facebook use to be a place where people could keep up with friends and make new friends. but money got involved..and they were naive

FIGURE 7: SAMPLE SUMMARIES USING BART, BERT, AND T5 MODELS FOR MARK ZUCKERBERG INTERVIEW ON YOUTUBE

4) Tailoring Summarization to Different Domains

We noticed that reviews were more focused on Amazon and Yelp where most customers discussed ideas relating to that particular product or business in question. On New York Times and YouTube, comments were unfocused, covered a wide variety of topics, and were largely unrelated to each other. Each platform also had its own benchmarks that could guide us to provide better summarization. On Amazon benchmarks included predetermined popular topics such as filters and the number of stars rating. On Yelp benchmarks included the business category and the number of stars rating. On YouTube we could use the like to dislike ratio and the video tags. In the New York Times we saw the general category of a news article such as Finance, Opinion, or Politics but not much more information. After considering all options we decided to narrow our focus on Yelp as it had focused reviews on a particular business and unlike Amazon it did not have precomputed topics.

5) Sentiment Analysis

They were not exactly similar but they both trended the same in terms of being overall positive or negative sentiment. To continue our work we decided we could either scale our sentiment score to try to match the Yelp star rating or we could graph our sentiment against the Yelp star rating for different Yelp businesses to see we find a correlation.

6) Chrome Extension JavaScript Front End

In building the extension we learned about various files including the manifest, popup.html, popup.js, content.js, background.js, and favicon. In our case background.js was rarely used. The original idea was to have JavaScript on the client side scrape all the review data from Yelp through content.js and pass it up to popup.js which would put the data in popup.html to be displayed. However we learned that Yelp pages were paginated and so to get all the reviews a user would have to open all the paginated pages in different tabs. We decided this would be a hassle for the user and so we pivoted to a different plan: scraping Yelp through the backend using Python.

7) Flask API Python Back End

We looked at extensions that did this including one that used a TypeScript library called Axios. However this was complicated and we did not want to learn TypeScript so we opted instead to use the plain JavaScript XML HTTP Request library included will all browsers.

8) Connecting Chrome Extension and Flask API

After trial and error in debugging code, we eventually succeeded in calling the Flask API from the Chrome extension.

9) Designing a UI Chrome Extension

Most extensions by others modified popup.html, effectively using Chrome extension JavaScript to fill in existing fields of an HTML document. Since our number of topics was variable in number, we decided it would not be a good idea to predetermine the number of topics in popup.html. Therefore we choose to dynamically generate elements in popup.html using JavaScript.

VIII. DISCUSSION

Our original goal focused on summarizing thousands of user comments to a simple paragraph on any given platform. Over the course of ten weeks, this implementation was modified to incorporate the new results from each pipeline and user evaluation of our progress. Our final model focuses on presenting users with popular topics and associated summaries (rather than a block of paraphrased text), for any given content that involves public discussion. In terms of the technicalities, this implementation allows for a faster runtime on the backend since the summarizer deals with a smaller input size (inputs are sentences per topic). It improves relevance as it reduces the burden on the summarizer to produce a long output of coherent text given a large corpus of unrelated sentences as input. In terms of product usability, the output of this model is more comprehensive, customizable, and readable.

Compared to the paragraph summary method, this model handles the initial problem of information overload more efficiently. By presenting users with multiple short and readable topics, this method of summarization effectively communicates the right amount of information. Users also have the option to filter by and further read more about a feature of the content that interests them. Furthermore, as any other product does, our final model has use cases beyond the expected scope. Summarization in the form of filtered topics uncovers many features particular to the content being discussed. For a restaurant this highlighted feature may be a famous dish, for a movie review it may be a particular scene in the movie, for a new article it may be a reference to a similar historical event etc. In our original approach of a block text summary, these subtle, but important features, would get lost or condensed. Lastly, our product can be used as a side-by-side comparator tool for users trying to decide between content. An example of this would be picking a destination, where with the topic summaries the user would be able to compare certain features they are concerned about.

With these outcomes, our product has made significant progress in removing irrelevant and duplicate data and eliminating the overwhelm in the processing of user comments.

IX. FUTURE WORK

Further improvements on this product can be done through customization per domain and fine tuning of the summary model. For each domain there are several different characteristics that can be analyzed and customized. Some characteristics we studied were open vs. closed discussion platforms, text vs. video content, content genre, content length etc. Apart from the domains, the summarization model can also be fine-tuned.

A. Domain Customization

Beyond our original testing domains, our model can usefully serve several other domains, Rotten Tomatoes, Angie's List, AllRecipes.com, Twitch for example are namely few. These platforms however, come with certain required customizations for quality summaries. We observed that in platforms like Amazon and Yelp where there is a closed discussion setting, where a service/product/content is being reviewed, there is significantly less irrelevant data, since the discussion is centered around people's opinion of that topic. Meanwhile in YouTube or a news platform like New York Times, information gets buried under people's sharing of personal experiences or extreme sentiment toward the content's protagonist. Even in the large pool of comments of open discussions, there is valuable information on open discussion platforms that is worth being summarized. In a news source, this can be a commonly referenced event in the comments that could help bolster the reader's understanding of the article or in YouTube this can be a discussion of a controversial feature of the video. Based on this observation, we eliminated sentences with first person pronouns that were indicative of subjective opinions, however there can be research done further to employ more strategies. While we found no major differences in text and video content-based comments, entertainment content often tends to be in the form of videos. If the nature of the content is entertainment, like satire, comments will include jokes which cannot be summarized or grouped into topics. More work is required to identify the genre and label the content.

B. Summarization Model Fine Tuning

Most users from our focus group had constructive feedback to make about our summarization model. Due to the timelines of this project, we were not fully able to customize the hyperparameters of our summarization model. We explored extractive and abstractive models and identified contradicting pros and cons for each. Both these model types did have a similar issue, they both produced incoherent sentences at some points since we were sending in a set of comments from different users. This also caused some comments to duplicate. More research is needed to determine how we can fine tune and train a model for our comments analysis purposes and find a middle ground between extractive and abstractive models.

REFERENCES

[1] R. Atanda, "Text Summarization of Customer Reviews Using Natural Language Processing," thesis, 2020.

[2] H. Jelodar, Y. Wang, M. Rabbani, S. B. B. Ahmadi, L. Boukela, R. Zhao, and R. S. A. Larik, "A NLP framework based on meaningful latent-topic detection and sentiment analysis via fuzzy lattice reasoning on YouTube comments," *Multimedia Tools and Applications*, vol. 80, no. 3, pp. 4155–4181, 2020.

[3] C. Musto, G. Rossiello, M. D. Gemmis, P. Lops, and G. Semeraro, "Combining text summarization and aspect-based sentiment analysis of users reviews to justify recommendations," *Proceedings of the 13th ACM Conference on Recommender Systems*, 2019.

[4] E. Poche, N. Jha, G. Williams, J. Staten, M. Vesper, and A. Mahmoud, "Analyzing User Comments on YouTube Coding Tutorial Videos," *2017 IEEE/ACM 25th International Conference on Program Comprehension (ICPC)*, 2017.

[5] C.-F. Tsai, K. Chen, Y.-H. Hu, and W.-K. Chen, "Improving text summarization of online hotel reviews with review helpfulness and sentiment," *Tourism Management*, vol. 80, p. 104122, 2020.

ACKNOWLEDGEMENTS

Thank you to the people who contributed responses to our survey and provided us valuable insights and feedback about the applications of our text summarization by topic project idea. A big thank you to TA Weitai Qian, who guided us through aspects of the project and written reports. Also, a big thank you to Professor Venkatesh Akella for the help and guidance on areas of the project to focus on.

We also thank Google Collaboration Notebooks for providing us GPUs on which to run our text summarization models and thank UC Davis for bringing together the essential resources and people to our project.

APPENDIX

Daiguang Zhang

1. Data Collection (YouTube)
2. Summarization Model Research
3. Summarization Model Testing
4. Sentiment Score Generation
5. Sentiment Score Evaluation
6. Final Report and Presentation

Divya Karthik

1. Data Collection (Yelp, Amazon)
2. Text processing
3. Feature Extraction
4. Summarization Model Pipelining
5. Feedback Collection UI enhancement
6. Project Structure and Planning
7. Final Report and Presentation
8. Final Demo Video

Jonathan Li

1. Data Collection (New York Times)
2. Text processing
3. Feature Extraction
4. Summarization Model Tuning and Testing
5. User Interface Design (Chrome Extension Implementation)
6. Leading code structure and organization
7. Final Report and Presentation
8. Final Demo Video