CIS 400 Senior Project
Spring 2018 Final Report

1. Student Information

Group #33: Aggregating Insights from Amazon Reviews
Joe Cappadona, Sumit Shyamsukha, Raymond Yin, Carolina Zheng

2. Advisor Information
Prof. Chris Callison-Burch

Advisor Meetings

Mon April 2: Discussed final submitted poster for the demo next week and evaluation. Discussed model schema sync-up.

Mon Mar 26: Discussed poster layout and formatting. Debugged model on NLPgrid. Demoed product comparison features on the web app.

Mon Mar 12: Discussed what we should focus on for the poster - advisor suggested figuring out what deliverables we wanted to show and working backward. Said that we currently had a solid model and should focus on the webapp.

Mon Feb 19: Discussed progress on double propagation algorithm - mostly finished, but worked out a few improvements.

Mon Feb 12: Discussed how to accomplish clustering similar extracted product qualities together. Advisor suggested looking into the word2vec embedding models.

Mon Feb 5: Discuss more advanced double propagation algorithm, determined that we will implement this and see if it improves our model results. Discuss coreference issue: Stanford CoreNLP has a coreference dependency that associates pronouns with the noun phrase (it -> the Sony camera). Demonstrate / discuss webapp deliverables.

Mon Jan 29: First meeting back from break: discussed plans for upcoming weeks.

3. Summary
We built a webapp that aggregates Amazon review opinions about a product's individual attributes to help potential customers make more informed purchasing decisions.

4. Overview of Problem and Approach

Online reviews such as those for Amazon.com products are large, unstructured bodies of text. When a user is looking to purchase a product, they typically don't have time to comb through all the reviews for that particular product. Instead, they will look at the overall star rating and read perhaps the top few reviews. However, the star rating is a one-dimensional measure of the quality of a product and does not tell you anything about individual features. Further, top reviews are individual opinions which may or may not reflect reviewer sentiment as a whole. Potential buyers also lack a way to quickly evaluate overall reviewer sentiment regarding individual product features.

Our goal is to create an efficient shopping experience for the end-user, and concisely describe how products perform in different aspects. For an illustrative example, if a user wants to buy new headphones and especially values their durability and bass, we want to summarize what thousands of reviews said about the durability and bass of each product, and allow the user to compare these qualities across different pairs of headphones. This way, the user does not have to read through dozens of reviews, searching for those that mention durability and bass.

To accomplish this goal, our project consisted of building a system to automatically infer product qualities from reviews, aggregating reviewer opinions about individual product qualities, and displaying these results in a web application interface for users to browse.

Our solution can be broken up into two parts: the model that aggregates the reviews and the web application infrastructure that serves this data to users.

5. Implementation

Our implementation is composed of a NLP-based machine learning model and a React- and Django-based web application, as shown below in Figure 1:

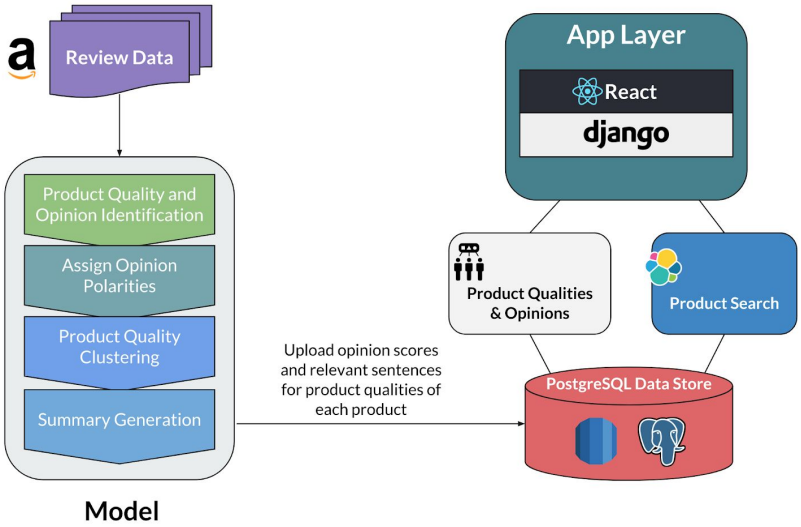

Figure 1: Architecture of the model and web application components.

Model

We built a machine learning model that accepts a corpus of product reviews as input, determines the product qualities and associated sentiments that were mentioned most in the reviews, and exports the data to the web application data store. Our pipeline consisted of 4 parts: (1) dependency parsing with the Stanford CoreNLP library, (2) target and opinion word extraction via the double propagation algorithm, (3) target word clustering via k-means clustering, and (4) exporting to the web application.

(0) Dataset: The data we used was the "Amazon product data" data set published by Julian McAuley of the University of California, San Diego [1]. We chose to focus on the set of electronics product reviews because of the many product qualities that can be associated with a single electronics product. The Electronics product category had 7.5 million reviews across 500,000 products.

(1) NLP parsing: The Stanford CoreNLP library is a Java-based Natural Language Processing (NLP) API that gives programmers access to high quality part-of-speech (POS) tagging, dependency parsing, coreference detection, and more. The review data was passed through an instance of the CoreNLP POS tagger and dependency parser in order to prepare the data for the target and opinion word extraction.

(2) Double propagation algorithm: This algorithm was drawn from "Opinion Word Expansion and Target Extraction through Double Propagation" by Qiu, Liu, Bu, and Chen [2]. The algorithm begins with a known opinion lexicon of positive, negative, and neutral sentiment opinion words. By following the dependencies between nouns and adjectives, the algorithm iteratively discovers target words (in our case, product qualities) and the associated opinion words and sentiments. Figure 2 below provides an example:

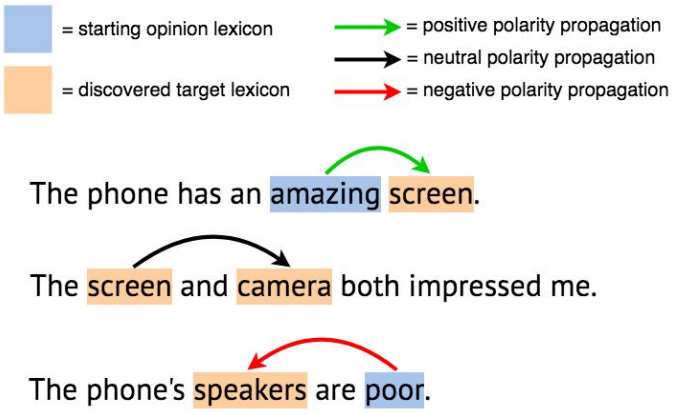

Figure 2: An example of target and opinion word discovery on successive iterations of the double propagation algorithm.

On the first double propagation iteration, we are able to extrapolate from the word "amazing" in our starting opinion lexicon to discover the target word (product quality) "screen" and see that it has positive sentiment in this sentence. On the second double propagation iteration, we are able

to extrapolate from the newly discovered target word “screen” to discover another target word “camera” due to their connection via the word “and”; however, we are not able to propagate sentiment from “screen” to “camera” because we are not yet sure of the sentiment of “screen” in the sentence. The last sentence is an example of negative sentiment propagation from a word in the starting target lexicon (“poor”) to a newly discovered target word (“speakers”). See the appendix for sample model output.

(3) Product quality clustering: We trained the word2vec model [3] on the entire Electronics reviews dataset to output word vectors of length 300 for all words with at least 100 occurrences across all reviews. We ran k-means clustering on the resulting word vectors to produce 500 classes, such that words with the most similar word vectors would be grouped together, and saved the output classes. Once the double propagation algorithm obtains the product qualities, we group them into clusters based on whether or not they share the same class from the clustering output. The sentiment score we display in the webapp is based on an aggregate for all product qualities in a given cluster; for example, sentiments for the product qualities “price”, “pricing”, and “cost” are all aggregated together.

(4) Export script: The format of the model output was designed to match the information we wanted to display on the web application. Accordingly, the output was organized into a product quality clusters table, which included information on the classes generated by the product quality clustering in step (3); a product qualities table, which included information on the most mentioned product qualities for a given product; and a review snippets table, which included snippets of review text and associated sentiments corresponding to product qualities in the product qualities table.

Web Application

We built a web application (originally hosted on http://review-notes.com) that allows users to compare up to 5 Amazon products at a time, based on product qualities that we learn by running our model on Amazon review data.

Figure 3: Screenshot of the web application to compare information about Amazon products.

Backend: We created a Django-powered backend to host and serve data to the app. The model uploads Amazon product metadata, product qualities, and review snippets to a PostgreSQL database. The product metadata (title, description, category fields) is additionally uploaded and indexed in an ElasticSearch cluster. The backend server accesses DB and ElasticSearch and offers a simple REST API for the frontend to make requests from. The PostgreSQL cluster was hosted on AWS RDS, ElasticSearch was hosted on AWS ElasticSearch, and the Django server ran on EC2 via AWS ElasticBeanstalk.

Frontend: The frontend is a React app that retrieves data from the backend REST API for its comparison view and search components. We implemented "search-as-you-type" functionality, which quickly returns relevant products based on title, description and category matches via ElasticSearch. The comparison table groups product qualities with the same product quality class ID together in the same row. The comparison table shows product qualities that most of the displayed products have in common first, and for each product displays the percentage of positive reviews over total reviews. When clicking on a row, it expands to show (1) the count of positive and negative reviews and (2) review snippets pertaining to the product quality for each product. Many of the UI views were built using components from Blueprint, a React-based UI toolkit. The frontend React app was compiled using Webpack and hosted on Amazon S3.

6. Evaluation

Model

To evaluate the effectiveness of the predictions of our model, we examined model outputs generated for products with over 1,000 reviews; after filtering outliers, we extracted 2,854 new opinion words and 62 product qualities overall from the sample. Using this data and a sample of 62 new opinion words, we evaluated product qualities, opinion words and linked review text snippets by hand to measure the following:

1. **Precision of product quality discovery: of the product qualities we identify, how many are valid?**
   Of 34 unique product qualities we identified for the products, 27 (79.4%) were evaluated to accurately identify the product, demonstrating our model's efficacy. We recognize room for improvement in filtering techniques to improve this precision.

2. **Precision of new opinion word discovery: of the new opinion words we identify, how many are valid for the product?**
   Over our sample of new opinion words, 69.4% were found to be valid. While the model's algorithm can consistently identify new opinion words, we may want to add an additional filtering step powered by crowdsourcing or a separate opinion word classification model. Another improvement could be building a more comprehensive blacklist of opinion words which are not classified correctly. We would also like to explore whether some discovered opinion adjectives can be identified as product qualities, since our approach only considers nouns as product qualities. For example, "waterproof" is a product quality but is classified

as an opinion word, since it is an adjective.

3. **Accuracy of opinion word sentiments: of the opinion words we identify, how often is our assessment of sentiment accurate?**
   In our sample of 62 new opinion words, we found that we assessed sentiment correctly 58.1% of the time. One drawback to explain this accuracy is that the review text corpus did not have perfect formatting, grammar and syntax. In the future, we could explore augmenting our sentiment outputs obtained from the double propagation algorithm with modern sentiment analysis methods for better performance.

Web Application

We conducted a user study by sending a Google form out to 20 of our friends (students at Penn). We provided a link to our webapp and asked them to provide text feedback as well as rate yes/no whether they would use our app to help inform future purchasing decisions on Amazon. 17 out of 20 respondents responded "yes", indicating that overall, our sample of young millenials thought that our app was helpful. The most common feedback we received in the text response was that the users wanted to see more review snippets, particularly the negative ones. This coupled with the yes/no feedback indicated that while the users found the quantitative aggregation to be helpful, they also wanted more context as to what the reviewers were saying. Given more time to modify our webapp, more snippets/more sophisticated snippet filtering is a feature that we could definitely add to enhance the product.

Shortcomings and Legal Considerations

While we believe that our model should work for any product category, it is possible that certain categories lend themselves better to learning product qualities than others. We have made the assumption that review text serves as the best way to learn product qualities that reviewers care about, but there may be some products that do not have many qualities to be learned. For example, we could learn several product qualities from electronics products, but may not be able to learn as many product qualities for food products. Although we tested our model on a few different categories and found that it learned product qualities well, there may be categories we are unaware of on which the model does not perform as well.

The main legal consideration of our project is that we are analyzing and displaying results obtained from proprietary reviews written on Amazon.com; although we are using a public dataset for academic use, the legality of this may be reevaluated in case Amazon changes their legal policy regarding review data. We believe this is unlikely, since Amazon recently released their own public official review dataset, affirming their commitment to "further[ing] research in multiple disciplines related to understanding customer product experiences."

7. Individual Contributions

Carolina Zheng: Discovered the paper and wrote the first iteration of the double propagation algorithm to determine that it was a good fit for the project; trained the models for product quality clustering and integrated into model pipeline; helped refactor the model into a single Python script; worked on evaluation of the model together with Joseph.

Raymond Yin: Created the webapp architecture; managed application setup for frontend and backend apps; setup deploy workflows; integrated Blueprint with React; worked on product comparison view; built search service and indexing pipeline using ElasticSearch; created database schema; developed data import UI and functionality to upload model data; worked on evaluation of webapp with Sumit.

Sumit Shyamsukha: Worked on the front end and back end for the web app, including the single product view and comparison view. Worked on the evaluation of the web app along with Raymond.

Joseph Cappadona: Built data pipeline; implemented first version of naive product quality extraction; implemented the sentiment propagation portion of the double propagation algorithm; refactored pipeline to interface with the web application; worked on evaluation of the model with Carolina.

Works Cited

[1] McAuley, Julian. *Amazon Product Data*. University of California San Diego, jmcauley.ucsd.edu/data/amazon/.

[2] Qiu, Guang, et al. “Opinion Word Expansion and Target Extraction through Double Propagation.” *Computational Linguistics*, vol. 37, no. 1, 2011, pp. 9–27., doi:10.1162/coli_a_00034.

[3] Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." *Advances in neural information processing systems*. 2013.

Appendix

Sample model output

Sentry HO278 Retro High Performance Stereo Headphones, Blue (3.5 stars on Amazon)
52 reviews

Feature clusters:

```
[[['quality'], 20],
 [['sound', 'bass'], 16],
 [['pair', 'set'], 5],
 [['price', 'pricing', 'value'], 4],
 [['style', 'design'], 3]]
```

Selected feature and opinions (positive opinions are marked with 1, negative with -1):

```
'bass': [('deep', 1), ('good', 1)]
'sound': [('good', 1), ('nice', 1), ('great', 1), ('great', 1),
('crisp', 1), ('great', 1), ('balanced', 1), ('loud', 1), ('nice', 1),
('clean', 1), ('comfortable', 1), ('ok', -1), ('tinny', -1), ('muted',
-1)]
'quality': [('poor', -1), ('great', 1), ('phenomenal', 1), ('superb',
1), ('great', 1), ...]
'pricing': [('reasonable', 1), ('amazing', 1)]
'style': [('great', 1)]
'design': [('great', 1), ('cute', 1)]
'fit': [('comfortable', 1), ('tight', 1)]
'value': [('great', 1)]
'plastic': [('cheap', -1)]
'volume': [('uncomfortable', -1)]
```

Business Plan

Problem/Need

E-commerce is a fast-growing industry, with both new players and traditional retailers paying increasingly more attention to this space. Amazon is worth more than the largest brick-and-mortar retailers - Walmart, Target, and Macy's - combined, with a market capitalization of $764 billion. Warby Parker's incredible success in the eyewear industry can be partially attributed to their decision to cut costs by bypassing traditional retailers by selling their product online.

74% of adults state that considering online reviews before buying a new product is at least somewhat important, with 32% believing it to be "extremely important." [1] Research has shown that there exists a strong correlation between product sales and aggregated review scores for that product, considering review aggregators such as Metacritic or Rotten Tomatoes. [2] Thus, companies have a strong interest in the sentiment of consumer/expert reviews of their products.

However, small retail brands may not have the spare personnel nor technical infrastructure to keep abreast of consumer reviews. Most existing review aggregators collect only expert opinions and for relatively well-known releases in media industries such as games, comics, and movies. On websites such as Amazon where third-party businesses sell their products, the only aggregation done is a star rating, which, taken alone, provides little useful information. There exists an untapped market for a technical solution that algorithmically extracts reviewer sentiment from consumer reviews.

Value Proposition

We have built a scalable, low-cost analytics technology that can aid small retailers in gaining valuable insights about consumer opinions toward their products.

Stakeholders

End customers: These are relatively small businesses whose scale does not warrant conducting extensive customer research. However, they could still benefit from a consumer opinion-driven understanding of their existing products' strengths and weaknesses. They sell either through their own website or a platform such as Amazon that supports customer reviews.

Review platforms: We may need to regularly crawl third-party review platforms such as Amazon (crawling means to download Amazon web pages in order to obtain new reviews). This may be somewhat difficult as Amazon has been known to block crawling, but we could able to get around this by limiting the rate of requests. Ideally, the retailer would have their own website with product reviews.

Employees: We are a small team of engineers. We would handle marketing in-house by contacting small retailers directly and/or gaining exposure on tech media platforms such as TechCrunch or ProductHunt.

Competition
There is little existing competition as applying natural language processing/machine learning techniques to real-world datasets is still a very new idea and an active area of research. Although there are a few experimental product review aggregators [3], they have not been monetized.

We believe that in the future, our primary competitor will be Amazon itself. They easily have the capability to develop a similar technology and have already taken steps toward algorithmically aggregating review content (such as having keywords that filter reviews). However, our product has the potential to be a much more customized solution for individual businesses, as well as provide more detailed analytics than Amazon will offer through their product review interface. We also believe that this is a relatively niche product whose profits would not scale well enough for a giant such as Amazon to take an interest in.

Market Research
Over a million small businesses sell their products through Amazon [4]; however, in 2015, only 47% of retailers utilized customer analytics to drive operating decisions [5]. This indicates that there is a significant untapped market that combines data analysis with retail decision-making.

Revenue Model
We believe that software-as-a-service (SaaS) is the ideal revenue model: this would be a subscription service in which customers would pay a variable fee based on the level of service they use. Small retailers would most likely want to select certain analytics to focus on (e.g. simply wanting to know which product attributes were most important to consumers vs. a more sophisticated sentiment analysis of a large bodies of reviews); we could offer variable pricing based on which measures they are interested in.

In addition, we would use cloud computing platforms such as AWS to run our models and host our web application, as these services offer pay-as-you costing that is much more efficient than managing our own cluster of servers.

Works Cited

[1] Perez, Sarah. “79 Percent of Americans Now Shop Online, but It's Cost More than Convenience That Sways Them.” *TechCrunch*, TechCrunch, 19 Dec. 2016, techcrunch.com/2016/12/19/79-percent-of-americans-now-shop-online-but-its-cost-more-than-convenience-that-sways-them/.

[2] Greenwood-Ericksen, Adams, et al. “On the Validity of Metacritic in Assessing Game Value.” *Journal for Computer Game Culture*, vol. 7, no. 1, 2013, www.eludamos.org/index.php/eludamos/article/viewArticle/vol7no1-6/7-1-6-html.

[3] “Review Summarizer for Gadget Reviews - 2018.” *The Review Index*, thereviewindex.com/us.

[4] Stevens, Laura. “Amazon Says More Than a Million U.S. Small Businesses Sell on Its Site.” *The Wall Street Journal*, Dow Jones & Company, 3 May 2018, www.wsj.com/articles/amazon-says-more-than-a-million-u-s-small-businesses-sell-on-its-site-1525341606.

[5] Brodsky, Matthew. “Retail Could Make Better Use of Customer Analytics.” CMSWire.com, 27 July 2015, www.cmswire.com/analytics/retail-could-make-better-use-of-customer-analytics/.