CS4780 Project Proposal

1. Size of Team

Size of team is four people.

2. Motivation

Before making a purchase on Amazon.com, buyers typically read reviews of the product to determine if it will meet their requirements. Although reviews are a great resource, it is very time consuming to read through many to find the information that is relevant to buyers. Furthermore, buyers must filter out a large number of unhelpful reviews. Although Amazon attempts to ease the problem by ranking its reviews in order of helpfulness, the top reviews may not address all the product features that the buyer is interested in, may be too old to still be reliable, or may be considered a top review simply due to visibility.

To solve this issue, we propose to automatically extract sentiments from the entire set of reviews and aggregate it in a concise summary.

3. Statement of Problem and Key Questions

Using user reviews, help consumers easily and accurately understand the important sentiments contained in the reviews.

Key Questions:

1. What are the characteristics of the product that reviewers find important?
2. How do reviewers rate these features?

4. Refined Approach

We will restrict our focus to one type of product (headphones) to make evaluation easier. We see 3 main stages to this project:

1. Preprocessing stage:
   Using the scraper extract all user reviews and split by sentence to generate a list of sentences from all the reviews per product.
2. Processing stage
   - 2.1. Determining Important Features of each product
     This will be done with a combined static and dynamic technique.
     - 2.1.1. Static Technique
       Generate important aspects from product descriptions scraped from Amazon.com by using a TFIDF weighting scheme from NLTK. These words, along with the words sound, audio, comfort, bass, highs, mids, tones, durability, quality, blocking, noise, sharp, deep, and tangle will be considered important aspects automatically.
     - 2.1.2. Dynamic Technique
       Using a parse tree split each sentence into bigrams and trigrams with the grammar format of <adj><noun> to represent candidate product aspects of importance to reviewers. We can filter this list using stopword filters and other grammatical based rules to get rid of false positives. We can create an inverted index of these terms and the sentences they occurred in. We can also easily use the list of sentences to find the frequency of aspects in the review corpus.
   - 2.2. Using SVM or Naive Bayes annotate each sentence with a sentiment score based on the previously created list of relevant features for the product. If a sentence has a score below a low threshold then it will be discarded.
3. Post-processing stage: Aggregation
   - 3.1. We need to summarize the aspect information we have and present sentences from the corpus that best represent the corpus. One way to do this is to rank the aspects based on frequency and then calculate an averaged sentiment score, which can be used to determine which sentiment polarity the majority of the reviews have with respect to that aspect. We can then take a sentence to represent this aspect. A simple scheme could be to find the sentence with the greatest magni-

tude sentiment score with the same polarity as the aspect average score.

Evaluation: Compare the results with annotated libraries of training and testing data from a project on reviews to find our accuracy on tagging aspect features and sentiment intensity/polarity. After that we will have to do manual surveying from people of the different summarization techniques that we can use.

5. Progress

1. Refinement of Problem and General Approach

   1.1. We have decided to change the problem and our approach to it to focus more on the text summarization aspect of the project rather than the dashboard GUI. We have refined our project to be an attempt to make a framework to summarize the Amazon reviews with respect to the features of the reviewed product. We no longer focus on the ability to compare other brands. We specifically have chosen headphones to shrink the scope of the problem and focus on one specific product area. Given a corpus of reviews for a headphone product, our framework will be able to output a list of sentences that best summarize the sentiment in the reviews towards the product and its features.

   1.2. Our new approach adapts a technique described in a 2008 paper about using a lexicon based system for summarizing review sentiment in relation to local services such as restaurants and hotels. To improve on this technique, we will also be doing feature engineering to improve the accuracy using prior mined information about the product. This takes place in the processing stage described above, in which we generate relevant features from product descriptions and prior knowledge about headphones.

2. Project Progress

   2.1. Data collection
   The scraper is operational for collecting Amazon review data for any given product. We have already gathered 3,152 reviews and have parsed them into sentences, broken them up into tokens and tagged each token with part of speech. The scraper is currently being updated to intake product descriptions to generate the relevant features using the frequency distribution of words in the combined product descriptions. For analytical purposes, we will be copy and pasting directly into the program for parsing, tagging, and finding the frequency distribution until the scraper is updated.

3. Problems

   3.1. Review Syntax
   Reviews may contain misspelled words, slang, or contractions that must be escaped. To resolve this issue, we will be using NLP's similar words function and slang libraries to match up the words with associated relevant features. To escape contractions, the text will be pre-processed before being parsed and matched up with the list of important aspects.

   3.2. Lack of Product Description
   If an item has no product description, we plan to simply parse its name and add adjectives from it into the list of important aspects and give it a generic list of words (as listed in the general approach) as its important features.

6. Resources

Python libraries: Sci-Kit Learn, NLTK, Numpy, Scipy

Data for benchmarking and testing accuracy

- Web scraper written in Node.js to parse relevant data (reviews, product descriptions, etc)

- Annotated data from SemEval2014-Task4 on aspect feature based sentiment analysis
  `http://alt.qcri.org/semeval2014/task4/index.php?id=data-and-tools`

- Amazon product review data mined by Stanford
  `http://snap.stanford.edu/data/web-Amazon-links.html`

Readings on Parsing & Sentiment Analysis:

- Stanford paper on parsing & sentiment analysis:
  `http://nlp.stanford.edu/~socherr/EMNLP2013_RNTN.pdf`

- Multi-aspect sentiment analysis
  `http://www.cs.cornell.edu/home/cardie/papers/masa-sentire-2011.pdf`

- Catching the Drift: Probabilistic Content Models, with Applications to Generation and Summarization
  `http://www.aclweb.org/anthology/N04-1015`

7. Schedule

By November 16th, complete:

- the web scraper to scrape product descriptions
- the static technique of the processing stage
- the annotation of sentences with SVM or Naive Bayes

By November 22nd, complete:

- the dynamic technique of the processing stage

By November 29th, complete the post-processing aggregation stage.

By December 4th, the due date of the poster, complete:

- ranking of sentiments in terms of relevance to consumer requirements
- charts and tables of data
- begin writing the final report

Final report due: December 10th