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 still be considered a top review simply because of visibility.

To solve this issue, we propose to automatically extract relevant information from the entire set of reviews with their related sentiment and present it in a way that is easier to digest and to compare different products.

3. Statement of Problem and Key Questions

Using user reviews, help consumers easily and accurately decide whether a product meets their requirements.

Key Questions:

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

4. General 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: Extract relevant product features
   - 1.1. In order to determine if a review contains relevant features, we will extract the product specifications from the product description using a web scraper. Using TFIDF from NLTK we will generate a list of words that have high frequency. These, in addition to the words sound, audio, comfort, base, highs, mids, tones, durability, quality, blocking, noise, sharp, deep, and tangle will be considered important aspects.
   - 1.2. 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. Using SVM or Naive Bayes annotate each sentence with a sentiment score based on the previously created list of relevant features for the product.
   - 2.2. 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. Using a hashmap store the phrase with the sentence it came from. Find the frequency of all these phrases in the entire set of sentences from reviews.
3. Post-processing stage: Aggregation
   - 3.1. Average the sentiment scores from each sentiment the aspect was tagged in to give an output sentiment score for the aspect. We can then output the sentence with highest sentiment score tagged with that aspect with same sign as output sentiment score for the aspect.

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.

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 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, and 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 pre-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.

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

   3.3.

IS THE BELOW GOING TO BE COMPLETE? IF SO PLEASE STATE IN PRESENT TENSE. IF NOT I'M NOT SURE WHY IT'S HERE....SINCE THE GENERAL APPROACH HAS BEEN DESCRIBED ABOVE? UNLESS YOU ARE ELABORATING. Following this, we will use a binary classifier to determine if the sentence's sentiment polarity is neutral or has sentiment. If the polarity is too low then we will discard the sentence.
At this point pre-processing is complete and we will get into the main part of the system. Next, we will tag each sentence with any aspect features it may contain and create an inverted index of aspect features mapped to the sentences they occurred in. An aspect feature is defined as a phrase or group of words that describe a specific feature of the product. Following this we can rank the aspect features based on their frequency of occurrence in the data set.
At this point we need to aggregate the sentiments associated with each aspect feature, a simple approach could be based on averaging the sentiment scores towards a product.

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 December 4^{th}, the due date of the poster, complete:

- aggregate sentiments using the dashboard to deal with conflicting sentiments for same feature by aggregating and displaying them.

  For instance: If 4 people say they like the texture of a phone case, but 6 people say it slips around too much, display 40% like texture, 60% dislike texture.

- rank sentiments in terms of relevance to consumer requirements (can use combination of either product description, weight from all the reviews that expressed the sentiment, or an alternative method to relate the sentiment back to corpus of reviews and use the frequency to determine rank)

Final report due: December 10^{th}