Web Search Engines — Reviews-Rehashed

Himaja Rachakonda, Anirudhan J. Rajagopalan
N14633788, N18824115
hr970, ajr619

May 4, 2016

1 Objective

As consumers search online for product information and to evaluate product alternatives, they often have access to dozens or hundreds of product reviews from other consumers. These customer reviews are provided in addition to product descriptions, reviews from experts, and personalized advice generated by automated recommendation systems. Each of these options has the potential to add value for a prospective customer [2]. Online reviews have become one of the most powerful factors which impact the consumer behaviour in the E-commerce industry. Both the consumers as well as the businesses gain a ton of information from the virtual voice of reviews. Online reviews transformed the phenomenon of the simple word-of-mouth feedback into a viral form of virtual feedback, which is not only bringing immense customer satisfaction in relying on them but also showing mind boggling returns in the businesses as well. Therefore, the aim of "Reviews-Rehashed" is to make the relevant reviews more accessible to the users.

Reviews-Rehashed is a specialized search engine to search product reviews pertaining to a product and its features. The search queries are in the form of <product,feature> tuples. This search engine will crawl reviews from various e-commerce websites such as Amazon and BestBuy.com to fetch reviews data of various products. It searches for the user query and presents the links to the results after ranking the documents based on the intrinsic quality of the document as well as the retrieval score. The quality of the document is based on a static score derived from the features of a review like the ratings, review-date, comments etc and also a dynamic score calculated based on the user query. We have implemented the search engine for electronics — mobiles data from Amazon.com. There is a huge scope of expanding this to other products and other domains as well.

2 Data Sources

Amazon.com and BestBuy.com were the options considered to collect information regarding products and their reviews. To build a crawler to fetch the data, a preliminary study was conducted on the following to understand the website page layout and structure —

1. User behaviour to search for a particular product and review.
2. Patterns in the URLs of the websites to reach to a particular product / product category / review / etc.
3. Examining the sitemap urls of Amazon.com and BestBuy.com to get all the crawlable URLs of the domain.

Sitemap is a list of pages of a web site accessible to crawlers or users. It can be either a document in any form used as a planning tool for Web design,

or a Web pagethat lists the pages on a Web site, typically organized in hierarchical fashion. Sitemaps make relationships between pages and other content components. It shows shapeof information space in overview. Sitemaps can demonstrate organization, navigation, and labeling system. [4]

Sitemaps are a useful tool for making sites built in Flash and other non-html languages searchable. If a website's navigation is built with Flash, an automated search program would probably only find the initial homepage; subsequent pages are unlikely to be found without an XML sitemap. [4]

This helped us to make sure all the review pages are searchable by our search-engine and also to find useful patterns which help in identifying each product and reviewwithin each of the domains of Amazon and BestBuy. Because of the vast expense of the data and the crawling required to be done, we have restricted our project to Amazon Review Data pertaining to Mobile Phones under Electronics Sections. This can be scaled easily to other categories and domains seamlessly with appropriate crawler jobs.

All the review pages of a particular product with an ASIN Id can be found at a URL in the following format —

www.amazon.com/<product-title>/product-reviews/<ASIN>/

Every Review of a product has a unique Review ID, and therefore each review can be accessed by a URL in the following format —

http://www.amazon.com/gp/customer-reviews/<Review-ID>/

3 Data Collection

Data Collection was executed in three phases —

Amazon Productsearch API: We used Amazon Product search API to query for highly rated mobile phones for ten famous mobile phones brand.

Amazon Product Crawler: Fetches all the unique Ids for all mobile phone products from Amazon.com. This is required to construct the seedUrls for each product as mentioned above in the Section 2,Data Sources Section.

Reviews Crawler: This crawler take the output of the the Amazon Product Crawler as the input to construct a seed Url for each product Id. All the unique IDs (ASINs) are stored, which are used to fetch Reviews from each product. These reviews can be accessed from this page through the links in the pagination bar. Eachpage consists of 10 reviews and the crawler goes to the depth until which there are no more review pages to be crawled for that product. Therefore, this product level URL is given as a seedURL for every product to crawl across various review pages through the links in the pagination bar.

The data has been collected for around 500 produts with a total of 70000 reviews. The following data about each review is collected as indexable fields based on which the review will be scored during the search time —

1. Review Content
2. Review Title
3. Review Date
4. Number of Comments to the review
5. Number of people found this review helpful
6. Number of images posted by the user in this review
7. Ratings of the review
8. Is the product a Verified Purchase
9. Review Length
10. Review Id
11. ASIN (Product Id)

4 Search Engine Architecture

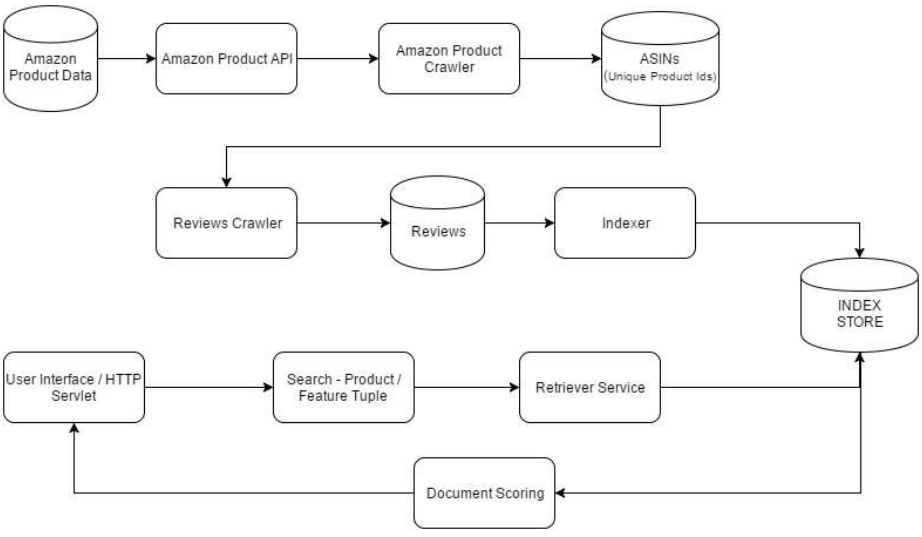

Figure 1: Reviews-Rehashed Search Engine Architecture

The search engine will be built by using the following different components.

Preprocessing module: This module takes care of preprocessing the data and dumps the processed data into Lucene.

- **Amazon WebCrawler:** This module fetches all the ASINs (unique Product Ids) for Mobile phones under Electronics Sections.
- **Reviews WebCrawler:** This module fetches all the reviews of the products from the product Ids retrieved from the Amazon WebCrawler module.

Document Scoring: This module implements the ranking algorithm by which the documents are retrieved in the search algorithm.

Retriever Service: This module receives the search query from the Web interface and interacts with the index store to retrieve results. The retrieved results, based on ranking algorithm in Doc Scoring module, are returned back to the web interface.

Web interface: A web interface for the user to issue the search queries. This will be an interface with two text boxes. One for the product name and another for the feature name.

Index store: This is a Lucene index store, an inverted index, which maintains all indexable data required while retrieving search results. The data will be populated and updated periodically by a crawler job that looks for updates to the data. The data will be stored in lucene after preprocessing.

5 List of softwares

Development environment Java 1.7, gradle.

Libraries used Lucene, Jetty, Jsoup, crawler4j, Spring-framework, slf4j, logback, Gsoup, Joda.

Data analysis Hive for analysing amazon product urls.

6 Document scoring

The documents are scored based on a personalized page rank algorithm which takes the intrinsic quality of the document into consideration as well as the lucene score generated. In many search engines, we have available a measure of quality $g(d)$ for each document d that is query-independent and thus static. This quality measure may be viewed as a number between zero and one. The net score for a document d is some combination of $g(d)$ together with the query-dependent score induced by lucene. [1] This kind of ranking algorithm demands an accumulation of an evidence of a document's relevance from multiple sources.

The personalized pageRank algorithm gives 25% weightage to the quality of the document and 75% weightage to the lucene score of the document. The

quality of the document is again a combination of weights given to each of the dependent facors.

The pageRank algorithm is as follows —

$$pageRank(d) = 0.25 * g(d) + 0.75 * (lucenescore) \tag{1}$$

The quality $g(d)$ of the documents is as follows

$$g(d) = 0.6 * H + 0.20 * V + 0.13 * C + 0.07 * I, \tag{2}$$

where $H :=$ Number of people who found the review helpful,
$V :=$ The purchase is a verified purchase,
$C :=$ Number of comments to the review,
$I :=$ Number of images in the review

As all the dependent factors and the lucene score are unbounded non-negative variables with different frequency distributions, it is necessary to normalize them before we use them in the pageRank algortithm in equation (2) to have consistent scores. A sigmoid mathematical function [3] is chosen to normalize these unbounded variables with a multiplacative factor of the x-axis to obtain scaled values between 0 and 1, based on distributions of each of these variables.

A sigmoid function is a bounded differentiable real function that is defined for all real input values and has a positive derivative at each points range is always between 0 and 1.

$$S(t) = \frac{1}{1 + e^{-t}}. \tag{3}$$

The graph in Figure 2 shows the spread of the sigmoid function for various multpicative factors.

7 Experiments and Observations

The reviews of the products in the search results are compared to the top reviews of those products in Amazon.com. We observed that 80% of the times all the search results match with the top reviews of the corresponding products in Amazon. The cases where it performs well and where it doesnot perform well are discussed below The experiments were performed on the following criteria:

1. Query boosting — Changing the boost for the product and feature queries 3.
2. Modifying the scoring algorithm Figure 4.
3. Changing the normalization weights 5.

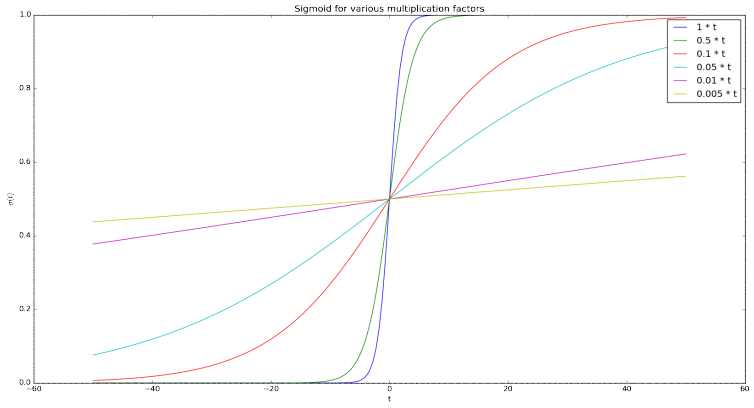

Figure 2: sigmoid variations with different multiplicative factors

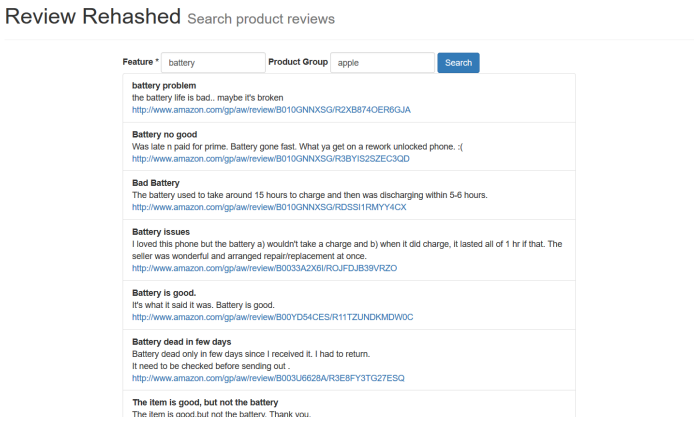

Figure 3: Default lucene scoring with feature query boost of 0.75 for title and 0.25 for content.

7.1 Positive cases

The best results were obtained with the final scoring formula discussed in the Section 5. This works pretty well for most of the queries with valid product name and feature name.

7.2 Negative cases

1. The results are not very accurate when conjuctive queries are given in the feature. An example can be “Camera Battery” as a search query for the feature of a product. We expect to see reviews containing both camera

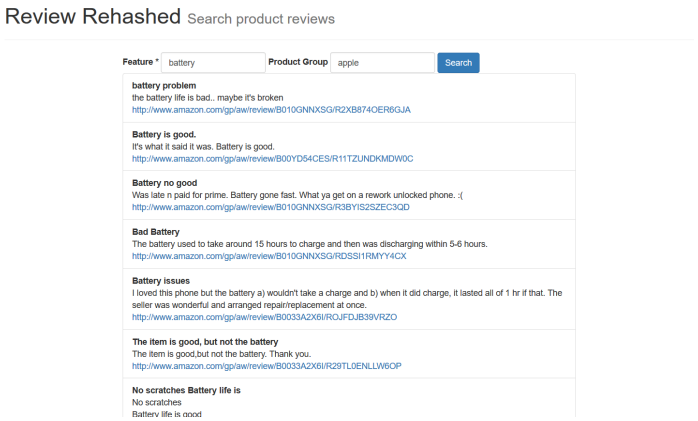

Figure 4: Default lucene scoring without boosting.

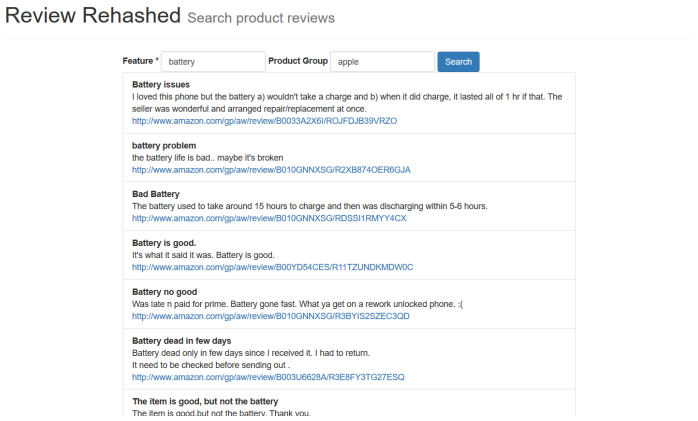

Figure 5: When the normalization weights are equal.

and battery in the review 7, however we see some reviews containing only battery and some reviews down the list, containing camera. This bahviour is erratic and the search results are not always consistent

2. Search results are better without the scoring algorithm when the product query is null. For example, if we just give "Great phone, good battery life" for feature 8, search results did not show results which have the exact same line in the review title. This is because of the other factors we consider for determining the static score/quality of the document. However, in this scenario, we also expect to see some direct matches with reviews as well.

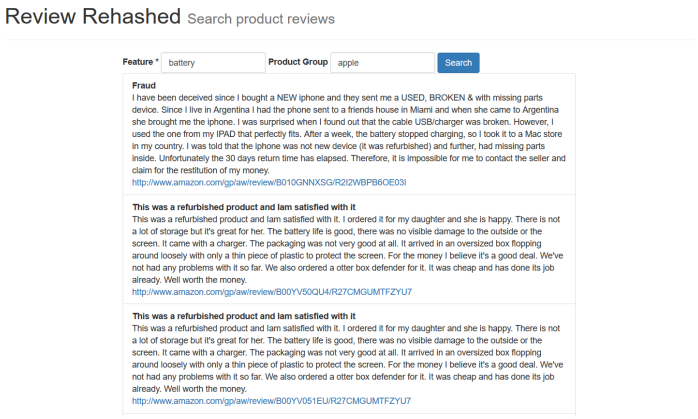

Figure 6: Scoring with different multiplicative factors for the dependent factors of the score — numHelpful=0.05, isVerifiedPurchase=100, numComments=0.1, numImages=1

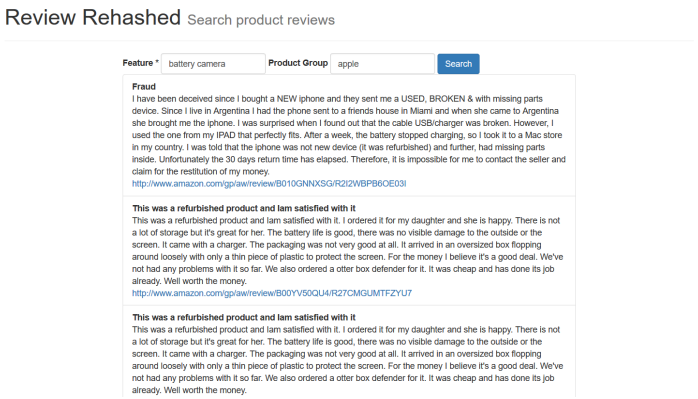

Figure 7: Review with Conjuctive queries — “battery camera”.

8 Challenges

8.1 Product development challenges

Understanding the structure of Amazon and BestBuy websites is a huge task, given the large number of product categories and urls that can be crawled 1.

Web Crawling to fetch data was a major obstacle as our requirement was restricted to review pages of various products. The decision of choosing the set of seedURLs was a confusion, in the beginning. To solve this problem, we analyzed the patterns in the URLs for various catergories,

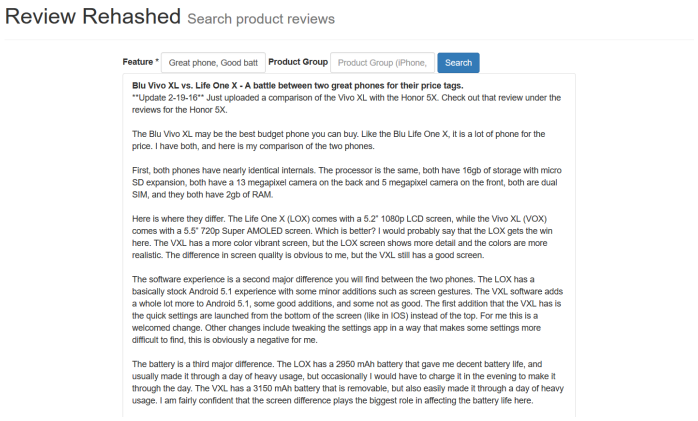

Figure 8: When no products are given.

	Total urls	English product urls
Amazon	917262698	917262698
Bestbuy	1493896	754179

Table 1: Amazon & Bestbuy product listings

products and then reviews. This analysis led us to choose product URLs to be the seedURLs to fetch reviews of each product.

In order to score the documents, one major challenge was to understand how helpful a review is to a consumer. We have gone through hundreds of reviews to see patterns of consumers and understand customer behaviours. A proper statistical analysis on these determining factors has to be conducted to find out which factor affects the customers the most. We came up with a personalized page Rank based on a heruristic we had drawn from the sample of reviews we observed.

8.2 Technical challenges

File formats Crawling the sitemaps with nested folder structure was a difficult process as the files were tar.gz format which has to be extracted to get another list of urls and so on.

Customizing crawler4j We had to do huge amount of customization on crawler4j for fetching only the links pertaining to reviews from a page. Crawler4j is not easily extensible and changing the behaviour involved using java reflection in a few places.

Changing lucene scoring Changing lucene's scoring and simialrity functions are pretty hard as lucene uses the similarity metric during the indexing

time as well. Since we had a few custom queries, writing a custom similarity class proved to be hard.

9 Future work

NLP module: This will essentially be a machine learning model that will be pre trained using the data collected by the crawler. We will use this to rank and retrieve the list of all matching sentences for a given query. This module can be developed to have the intelligence to do the following tasks—

Summary Extraction: A summary from the top — K reviews is presented to the user, giving an overall picture of the opinion of the crowd on this product's feature.

Opinion Mining: A module which shows the positive and negative reviews of a product

Data Analytics: A module to collect data for analysis of the consumer behaviour in learning about a product and / or its features.

Tuning Score parameters: Score Parameters can be tuned to more accurate values by training the data and running models, to improve the current scoring algorithm. Machine Learning to Rank the documents based on these parameters may give a better performance with respect to the relevancy of the search.

References

[1] Christopher D Manning, Prabhakar Raghavan, Hinrich Schütze, et al. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[2] Susan M Mudambi and David Schuff. What makes a helpful review? a study of customer reviews on amazon. com. *MIS quarterly*, 34(1):185–200, 2010.

[3] Wikipedia. Sigmoid function — wikipedia, the free encyclopedia, 2016. [Online; accessed 5-May-2016].

[4] Wikipedia. Site map — wikipedia, the free encyclopedia, 2016. [Online; accessed 5-May-2016].