Lumière

A Smart Review Analysis Engine

Ruchi Asthana
Nathaniel Brennan
Zhe Wang

Purpose

A rapid increase in Internet users along with the growing power of online reviews has given birth to fields like opinion mining and sentiment analysis. Today, most people seek positive and negative opinions of a product before making a purchase. Customers find information from reviews extremely useful because they want to know what people are saying about the product they want to buy. Information from reviews is also crucial to marketing teams, who are constantly seeking customer feedback to improve the quality of their products. While it is universal that people want feedback about online products, they are often not willing to read through all the hundreds or even thousands of customer reviews that are available. Therefore our tool extracts the information both vendors and customers need so they can make the best decision without having to read through any reviews. Lumière does this by going through all product reviews and extracting: (1) popular features being commented on, (2) sentiment that identifies how people feel about popular features, and (3) a representative summary that gives insight into what people are saying about popular features. Our platform is composed of two interfaces: a buyer interface and a seller interface, which we have shown in Figure 1.

Buyer Interface

The buyer interface is a smart search engine that takes in a product type and the features of highest importance to the buyer. It outputs products that are ranked in order of how strongly they perform on the buyers' prioritized features.

Seller Interface

The seller interface provides a space for marketing teams to find information relevant to their products' performance. The seller interface does this by displaying: (1) popular features, (2) sentiment associated with each feature (from reviews), (3) feature specific summaries, and (4) feature based comparison with other products in the category. The seller

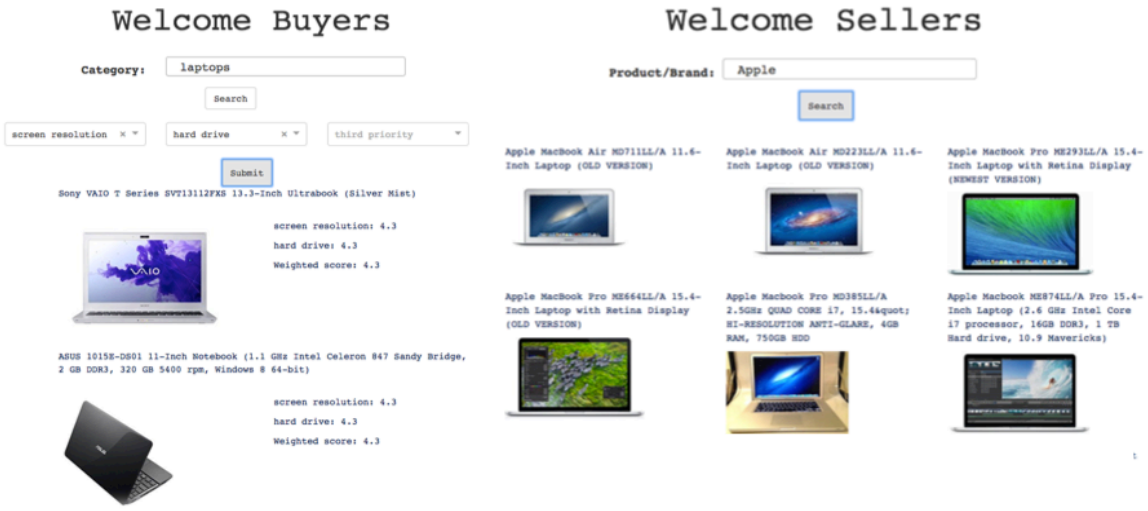


Figure 1. Displays the buyer and seller interfaces.

interface also includes demographic information about the gender of the buyers and their location across the world, which can be helpful for targeted marketing (Appendix).

Data Loading

For our dataset, we used the Amazon review data from Computer Science Professor Julian McAuley at University of California San Diego (UCSD). The dataset contains information for 82 million Amazon reviews, including the full text of the review. It also contains ancillary data points such as the reviewer username, the score from one to five stars, and product information such as title, description, price, brand, and categories. This data is provided in document format. DocDB is a new feature of the InterSystems IRIS Data Platform which allows for the easy storage and querying of document data. Using this new tool, we were able to seamlessly insert the raw review data and query the fields we needed, such as the review text, rating, reviewer, and product info. After our raw review data was loaded for easy accessibility, we faced the challenge of designing a database that could store that raw data alongside the results of our natural language analysis. We needed a simple way to create and insert new data points and to handle one-to-many and many-to-many relationships. For this we employed Caché Object Script. We had several types of data that could easily be translated into objects. Our dataset revolves around reviews, reviewers, products, and product categories. We created a class for each of these with the fields we could extract directly from the raw data and the new properties from our analysis. Our new data includes extracted product features, a sentiment score for each feature, and a summary for each feature. We did this analysis on a per-review, per-product, and per-category basis. Therefore, we decided to make serial objects to contain this feature information and put them in lists within our review, product, and category objects.

Feature Extraction

The goal of feature extraction was to identify the product features being commented on. In order to do that we took customer reviews from all the products in a particular category. The output of our pipeline was a list of the most talked about features in the given category. The review text contains misspelled words and punctuation errors that may compromise the performance of our natural language processing algorithms. Thus, our first step was to use the Ginger API to correct grammar and punctuation mistakes in the customer reviews. The ideal review dataset for our analysis are opinion sentences about product features. However, the Amazon reviews also contain objective sentences that describe why and when a product was bought. Thus we developed a Naive Bayes classifier to filter out these objective sentences. We used subjectivity dataset v1.0 introduced in Pang/Lee ACL 2004 as the training set, and unigrams as features. We turned all the sentences into a sparse matrix in which a value of 1 represented that a

certain word existed in a certain sentence, and then fed it to the classifier. Next we used part of speech tagging provided by Natural Language Toolkit for Python to obtain the most frequent unigrams and bigrams. We tested extracting different types of unigrams and bigrams and found that unigrams that are nouns and bigrams that consist of two consecutive nouns or one adjective followed by one noun performed the best. When we counted the unigram and bigrams, we stemmed them so that "noise cancelling" and "noise cancellation" both counted toward the bigram "noise cancel". Then if "noise cancellation" occurred most often among all the forms of the bigram "noise cancel", we output "noise cancellation" as the feature.

Feature Pruning

Feature pruning was conducted to remove meaningless entities. We filtered out the brand names (attribute of dataset), stop words and redundant features. For example, we removed "batterylife", and added to the count for "battery life". All of our natural language processing algorithms were written in Python. We took advantage of the Spark Connector and PySpark to execute SQL statements in Python, and got the data we needed from InterSystems IRIS in an efficient manner. We also used multiprocessing to utilize the 16 cores of our cloud instance, and improve the runtime of these algorithms. Figure 2 gives an example of part of speech tagging and the results of feature extraction conducted on customer reviews from the laptop category.

Sentiment Analysis

By themselves, features are not very useful. The goal of our sentiment analysis is to provide a numeric score for each product feature indicating whether people talked about it positively or negatively in reviews. To determine how people feel about a given feature, we examined the words they use to describe this feature (descriptors). To extract

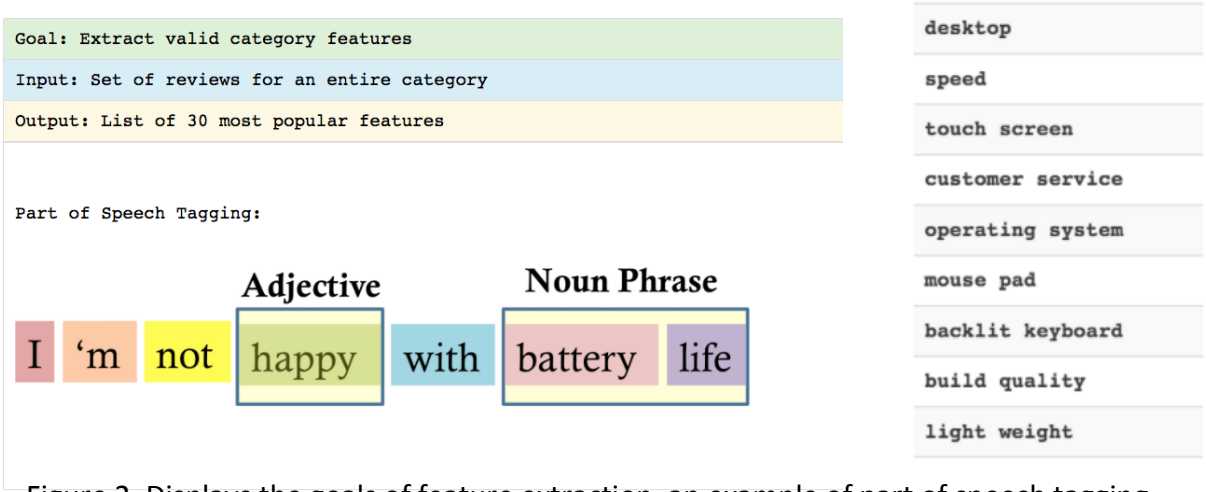


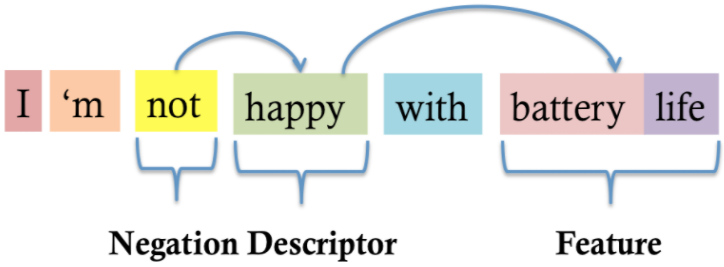
Figure 2. Displays the goals of feature extraction, an example of part of speech tagging, and the output of feature extraction and feature pruning on a set of laptop reviews.

Goal: Numerical representation of feature likeability

Input: Set of reviews for an product

Output: Score between 1-5 of how much people liked the feature

Dependency Parsing:



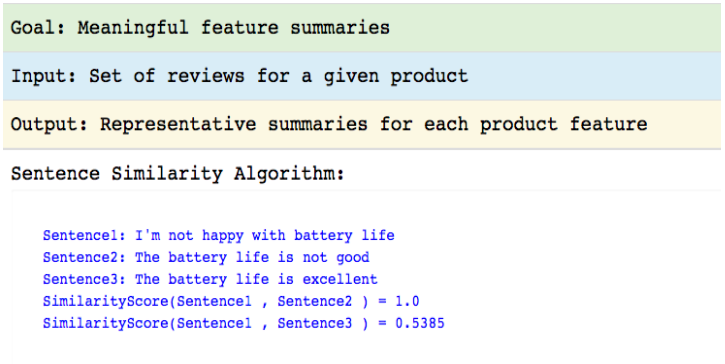
Feature	Sentiment	Popularity
battery life	3.6	3.8
window	3.2	3.1
size	3.6	2.9
software	3.3	2.8
operating system	3.6	2.7
speed	3.6	2.7

Figure 3. Displays the goals of sentiment analysis, an example of part of dependency parsing, and output from sentiment analysis on features extracted from a set of laptop reviews.

descriptors, we used a natural language processing technique called dependency parsing. Dependency parsing transforms a sentence into a tree of grammatical relations, such as nsubj (nominal subject), nmod (nominal modifier), and amod (adjectival modifier). This allowed us to reasonably guess what words are being used to modify or describe other words. We manually curated a custom set of dependencies that usually indicate this type of relation. When a word is related to a product feature (that we already had extracted in the previous step) by one of our selected dependencies, we marked the other word as a descriptor. We also used dependency parsing to extract words that modify the descriptors, such as negations and adverbs. This allowed us to more accurately determine the sentiment of a particular sentence with respect to the feature of interest. Now that we had all the descriptors of a feature, we could analyze these descriptors with SentiWordNet, a tool that provides a sentiment score for every possible use of a word in English text. Taking a weighted sum of these scores provided an accurate estimate for the sentiment of that descriptor, before taking into account negations. If a descriptor was negated, then we inverted its score. The score of a feature is calculated by taking the average of the scores of all the descriptors of the feature, as displayed in Figure 3.

Feature Specific Summaries

Feature specific summaries help give insight on what people were saying about a particular feature of a product. These summaries are included in the seller interface to help marketing teams know what to improve about a given feature. In order to get the summaries, we first isolated all product reviews (PR) that mentioned feature (F). Then, we used InterSystems IRIS Natural Language Processing to get a summary for each review that is related to what people are saying about that feature.



Feature	Summary
battery life	Impressive battery life, faster than the iMac (you have to love solid state drives)
window	As the return window expired a little bit, if I return the laptop, I have to pay 20% restocking fee and shipping
size	Just bought this for my boyfriend- It's the perfect size, very light, very easy to use, and battery lasts a long time and doesn't heat up
software	Great software and speed!

Figure 4. Displays the goal of feature specific summaries, the results of the sentence similarity algorithm, and the results of feature specific summaries for the features from the set of laptop reviews.

This is called the feature summary (FS). Once we had feature summaries for each review, we used the REST API to collect them. We then implemented a sentence similarity algorithm in Python to find the most representative review. Our sentence similarity algorithm takes two sentence and gives a numerical representation from 0 to 1 (0 = not related, 1 = highly related) of how similar they are. For each feature summary, it added all similarity scores of related sentences and then ultimately output the sentence under 100 words with the maximum score.

Comparison to Google API

We researched similar products and the only one we found that came close to as comprehensive as ours was Google's Cloud Natural Language API. We found that it could extract meaningful entities from text and the sentiments associated with them. If we feed the reviews to the API, these entities could potentially be product features. Therefore, we tested the API and our pipeline on the same set of reviews, all of the laptop reviews. The top features are listed in Table 1. Compared to Google API, our pipeline could extract more meaningful feature phrases, like mouse pad and touch screen. In addition, Google API charges 1000 dollars for just one category with around 100 products, while our pipeline is built entirely on open source tools.

Google NLP	Lumiere NLP
price	software
screen	battery life
keyboard	size
battery life	port
problems	hard drive
battery	window
performance	graphic
money	speaker
purchase	button
games	desktop
hard drive	speed
model	touch screen
system	customer service
software	light weight
work	operating system
unit	mouse pad

Table 1. Displays the top features extracted from the Google API and our feature extraction

method. As you can see the results are comparable.

Conclusion

We have developed an interface for everyone. Our platform allows both buyers and vendors to access the information they need to make good choices regarding what purchases to make and how to make a product more favorable to consumers. They key features of the interface are rooted in: (1) feature extraction, (2) sentiment analysis, and (3) feature specific summaries. Furthermore we highlighted several InterSystems technologies including InterSystems IRIS, Multimodel data platform (includes SQL, Object Model, DocDB), REST API, InterSystems IRIS Natural Language Processing Tool, and InterSystems Cloud Manager. In the future we plan to expand our product by extending its use to other sites with reviews, like Yelp and TripAdvisor. Additionally we will filter out reviews that are redundant, sarcastic, and auto-generated. Our product has immense potential to become the most widely trusted review analysis engine. So you can stop scrolling and go back to doing what really matters.

References

http://jmcauley.ucsd.edu/data/amazon/links.html

https://www.cs.utah.edu/~riloff/pdfs/cicling05.pdf

http://www.getginger.jp/
https://github.com/maciejkula/glove-python

https://github.com/akhilram/product-profiler/tree/master/featureExtraction

https://nlp.stanford.edu/courses/cs224n/2007/fp/johnnyw-hengren.pdf

https://www.cs.uic.edu/~liub/publications/kdd04-revSummary.pdf

https://pdfs.semanticscholar.org/ee6c/726b55c66d4c222556cfae62a4eb69aa86b7.pdf

https://turing.cs.washington.edu/papers/emnlp05_opine.pdf

https://pdfs.semanticscholar.org/d62e/06da793f86d7216058fe377fdb30a67d877f.pdf

https://www.researchgate.net/profile/Lorin_Hitt/publication/220079805_Self_Selection_and_Information_Role_of_Online_Product_Reviews/links/54f505180cf2f28c1362df5c/Self-Selection-and-Information-Role-of-Online-Product-Reviews.pdf

Appendix

Features | Demographics

Apple MacBook Air MD711LL/A 11.6-Inch Laptop (OLD VERSION)

Compared to other laptops

Feature	Popularity	Comparision
software	4.4	better than 74%
battery life	4.3	better than 41.4%
program	4.2	better than 16.7%
size	4.1	better than 42.7%
port	4.1	better than 41.9%

Feature analysis based on 450 reviews

Feature	Sentiment	Popularity	Summary
battery life	3.6	3.8	Impressive battery life, faster than the iMac (you have to love solid state drives)
size	3.6	2.9	Just bought this for my boyfriend- It's the perfect size, very light, very easy to use, and battery lasts a long time and doesn't heat up
software	3.3	2.8	Great software and speed!
operating system	3.6	2.7	Very nice laptop, but suffers from Apple's clunky operating system
speed	3.6	2.7	This is my first SSD laptop and the speed performance, despite having a slower processor chip, exceeds my expectations
program	2.8	2.6	I'm in a nursing program right now and this computer serves the purpose due to the Light weight and the speed of the computer
desktop	3.7	2.6	Beautiful desktop and icons
speaker	3.2	2.5	30 min later after presenting the buzzing sound on the right speaker, it stopped functioning
button	3.2	2.4	When I press the power button the computer is on SO quickly, it's unreal

Figure 1. This image displays the first part of our seller interface. It contains the results of feature extraction, sentiment analysis, and feature specific summaries for a given product sold on Amazon. This information can be very helpful to marketing teams, who want to know what people are thinking and saying about their product and its features.

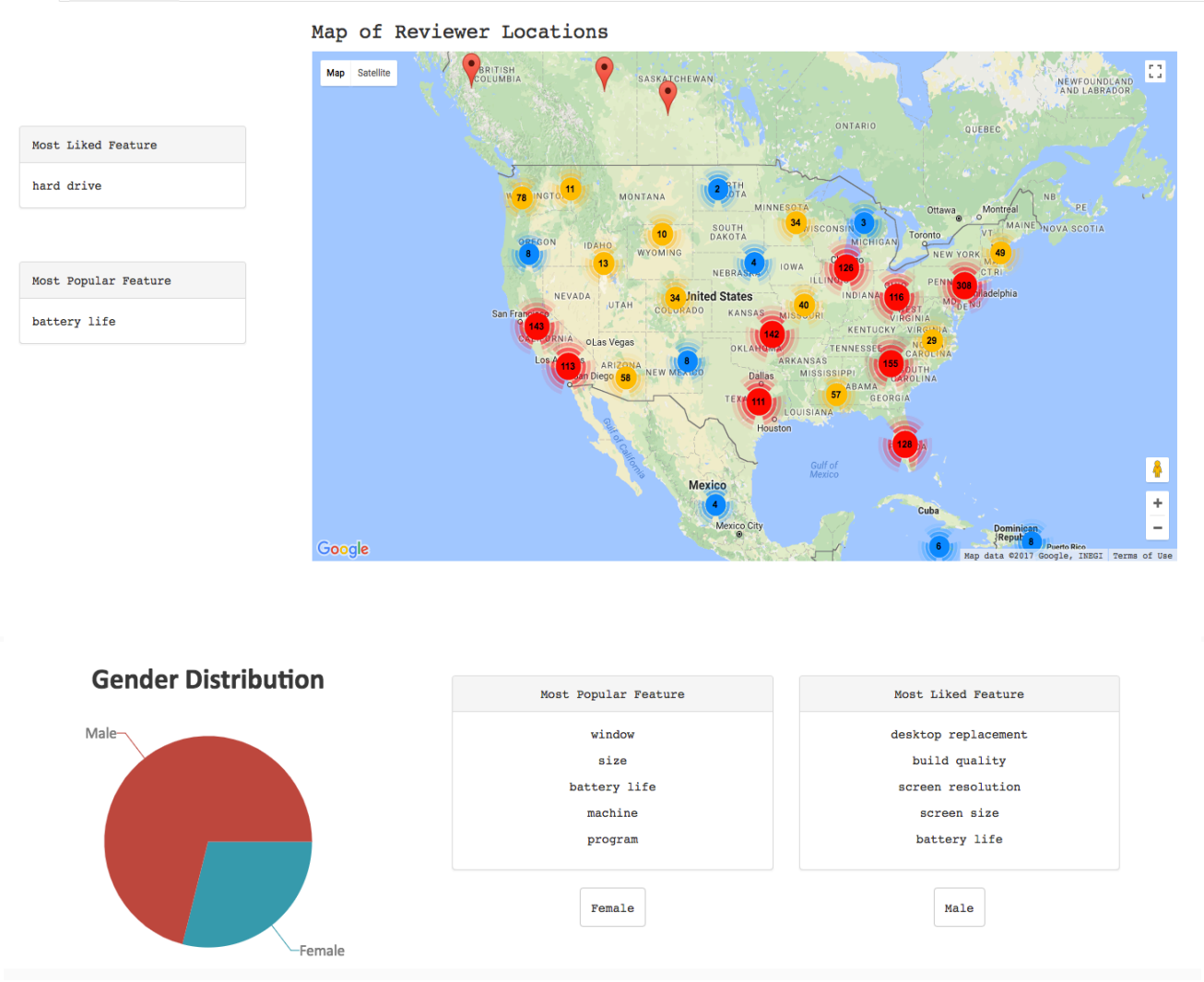


Figure 2. This image displays the second part of our seller interface. It contains demographic information that tells who and where people are when they are commenting about a product and its features. If you hover over a region on the map, you can see what feature of the product was most popular and what feature was most liked in that area. Below the pie chart shows what percentage of reviewers were male and what percentage were female. The boxes to the right of the pie chart contain the most liked and most popular features for each gender.