Report to NSF: Feature Extraction from Textual Datasets

Patrick Moran[*], Bethany Herwaldt[†], and Jeffrey Salter[‡]
Math Department
North Carolina State University, Raleigh, NC
Mentor: Carl Meyer, Ph.D.
Graduate Advisors: Shaina Race and Ralph Abbey

July 31, 2008

Abstract

This paper describes a new process we named Aldteran that extracts topics from sets of reviews. Aldteran takes into account word frequency, semantic similarity, and the proximity of words in a dataset. Existing techniques such as non-negative matrix factorization and graph clustering algorithms are utilized. The Aldteran process was able to identify major topics of reviews on two digital cameras and show differences between them.

1 Introduction

The Internet holds a wealth of new information, which is increasingly taking the form of user-generated content. Many are interested in knowing the general opinions on a given topic without having to read thousands of reviews or articles. Our goal is to find an unsupervised technique that will allow us to identify the topics discussed in a group of documents.

[*]College of Charleston, patrick.a.moran@gmail.com
[†]University of Notre Dame, bherwald@nd.edu
[‡]Albany State University, salter1430@yahoo.com

First we must obtain a list of words that are characteristic of the topics of the reviews. We get a list of words from a non-negative matrix factorization. After this list is obtained, it is filtered using the ratios of the frequencies of words in the document to the frequencies of those words in some large corpus of general English. The words with the highest ratios constitute our list of keywords.

Once we find the keywords, they must be grouped into topics. We create a graph of the words, wherein the distance between two words is determined by semantic similarity and word proximity. The semantic similarity is based on their relationship in WordNet, independent of their context. Word proximity is based on the average number of words between the two words in the document collection. This graph of terms can then be clustered with a traditional clustering algorithm. It is hoped thate ach cluster will be a topic of the collection.

We worked with two datasets during this research, one with 146 reviews about the Leica D-Lux 3 digital camera and the other with 407 reviews about the Canon PowerShot Pro Series S5 digital camera.

2 Aldteran Method

2.1 Obtaining Keywords

The first step of Aldteran is to create a list of keywords from the dataset. These are the words that will be clustered to find the topics. We generate this list using a non-negative matrix factorization (NMF) filtered by a word frequency ratio.

2.1.1 Non-negative Matrix Factorization

Let $A_{n\times m}$ be a non-negative matrix such that each column is a document from the dataset, each row is a term present in the dataset, and each entry a_{ij} is the number of times term i is used in document j. The NMF is a decomposition that will approximately factor the matrix A into two non-negative matrices, W and H. W is an $n \times k$ matrix, and H is an $k \times m$ matrix, where k is a parameter set by the user.

There are many algorithms to find an NMF. Some enforce sparsity on W, H, or both. Enforcing sparsity on a matrix means that most entries in the

matrix are required to be nearly zero. This improves interpretability of the results because only the most significant entries are not close to zero.

The NMF may be interpreted in terms of linear combinations. The i^{th} column of A is approximately equal to a linear combination of columns of W, where the coefficients are the entries in the i^{th} column of H.

$$a_i = H_{1,i}w_1 + H_{2,i}w_2 + \cdots + H_{k,i}w_k$$

NMF has been applied to textual datasets as a clustering algorithm with the hopes of extracting topics. Each column of W can be interpreted as a topic, and the columns of H refer to the degree to which a particular document is about each topic. Specifically, column w_i can be seen as a "pseudodocument" with high entries corresponding to terms that are relevant to the topic.

We decided to apply this method to our datasets of reviews. We chose to use Patrik Hoyer's NMF algorithm with sparsity constraints[6]. This algorithm optimizes the least squares difference between A and WH while enforcing sparsity on W.[6] We implemented Porter's stemming algorithm which is provided in libstemmer_c[9]. This algorithm takes words and reduces them to their bases so that different forms of a word can be compared. For example, "camera", "cameras" and "camera's" all stemmed to "camera".

We also implemented stoplisting, which means that we did not include common words such as "the" and "it" that do not effect the topics of documents. This prevented such useless words from being included in results from the NMF.

We observed that the NMF returned words that were important topics of the datasets, but did not cluster them well. Here is an example of results from the NMF. Each bullet refers to one cluster.

- noise buy sensor panasonic silly fuji
- quality manufacture pay operational lens
- format shoot flash slowlag promise automotive flashoth side equipment inside
- image color clarity small size alternative mk lightweight sturdy c-lux
- camera amazing happy menu master photo mp close

We decided to develop a new method to obtain topics, but still incorporate the NMF as one way to obtain a list of keywords.

We sorted each column of W in decreasing order and noticed that the first entry, or the entry with the highest value, usually referred to an important keyword. Therefore, we decided to make the first entry from each column a keyword. The next entries in the second through tenth rows were stored as a separate set. We repeated the NMF six times (Since it uses a random initialization, more runs provide more words.) and maintained the two sets of unique words. The set of words from the second through tenth rows were often important keywords, but there were also many low-quality words. We decided to filter these results with a frequency ratio, as explained in the next section.

2.1.2 Comparing Frequencies

We have found, as expected, that words that are present in much higher proportions in our dataset compared to general English are typically related to the topics of the dataset. For example, in a dataset of camera reviews, words like "lens" and "sensor" will be present in a much higher proportion than the English language, but words like "is" and "the" will be present in similar frequencies to general English language.

The number of times a word with index i is present in the dataset is denoted d_i. We also find a large corpora of documents reflecting typical usage of the English language. The number of times the word i is present in the corpora is denoted c_i. For all of the words found in the second through tenth rows of W, we find the proportions $p_i = \frac{d_i}{c_i}$ for each unique word. We then order these proportions and find the β highest proportions, where β is a parameter, and this list of β keywords is added to the list of the words that were at the top of each column to create our final list of keywords.

Note that another important decision in this step is choosing a large corpora of English text. We chose to use a corpus of movie and television show scripts[1]. We wanted to have a sample of informal language in a conversational style, since we were comparing it to internet reviews. We expected that including blogs[11] in our corpora would improve our results. However, our success drastically decreased. We believe this may be because blogs are often about technological subjects, so camera-related words may have abnormally high frequencies in blogs. It is important that we compare the dataset frequencies to a body of English text that covers a wide variety

of topics so that the inherent topics of our dataset become evident.

At one point in our research, we were trying to use these ratios as a source of keywords. Unfortunately, we discovered that misspellings such as "promiss" would appear to be keywords because the misspelt term had a very low frequency in the English corpus, and thus the word had a large ratio. We attempted to fix this problem in a couple of ways, before these attempts were made obsolete because we returned to using the NMF as our source of words.

First we experimented with other ratios, such as $p_i = d_i^2/c_i$, $p_i = d_i \log(d_i)/c_i$, $p_i = d_i^2/\log(c_i)$, and $p_i = d_i/\log(c_i)$. Of these, only $p_i = d_i^2/c_i$ gave comparable results. It is inconclusive whether this is better than a simple ratio, and we kept the original ratio for its better interpretability.

Second, we implemented stemming as discussed in the nonnegative matrix factorization section.

2.2 Graphing Keywords

At this point, we have a list of keywords. Many of the words are related to each other, such as "options" and "menu." We want to group these words into topics by creating a weighted, undirected graph wherein the distance between two words are determined by two measures, semantic similarity and word proximity. We combine these two measures into one metric $distance = \alpha s + (1 - \alpha)p$, where α is a parameter. We set $\alpha = 0.5$, but it's value is a topic of future research. This graph will later be clustered so that the clusters generated can be interpreted as topics.

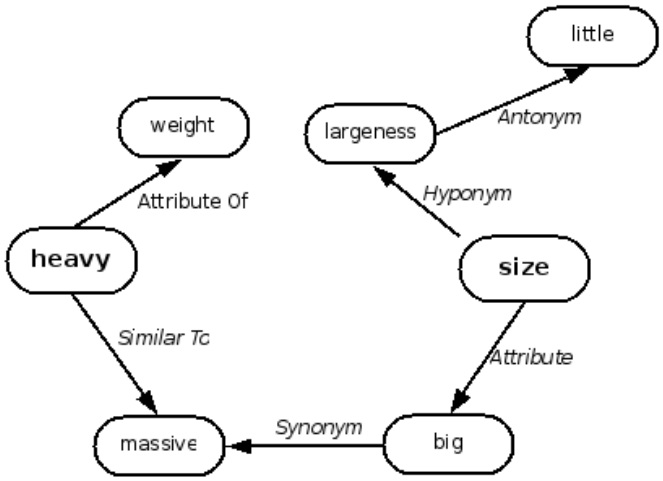
2.2.1 Semantic Similarity

WordNet is a hyperlinked database of the English language developed at Princeton University[4]. Each word is linked to its synonyms, antonyms, attributes, derivationally related forms and words related under many other relationships. We use WordNet to approximate the semantic similarity of pairs of words.

At first, we used a preexisting metric of semantic similarity, called Hirst-St. Onge[5]. However, due to implementation details, the computation intensity of this method proved prohibitive, so we developed our own metric.

To find the semantic similarity of two words based on WordNet, we consider WordNet to be a unweighted, undirected graph where words are nodes

and there exists an edge between two nodes if those two words have any relationship. For example, an extremely abbreviated subgraph of WordNet might resemble the following:



To determine the semantic similarity of two words, we search for the shortest path between the two by performing a breadth-first search from both nodes and waiting for the two subgraphs to connect. We note the length of the shortest path between the two words. This is based on the idea that if two words are semantically similar, they will be close together on the WordNet graph.

Then we continue the breadth-first search one more step from both directions and count the total number of paths between the two. This is done for robustness, because WordNet is very complete and contains even obscure meanings for words. Suppose we wish to find the distance between two words which both have obscure meanings, and that these obscure meanings are closely related to each other even though the dominant meanings of these words are not. In this case, the minimum path length will indicate that these words are very related, when that is not the desired result. When this is the case, if we continue the breadth first search one more step, we will not gain many more paths between the two words. However, if two words truly are related, then we expect that continuing the search another step will reveal many new paths.

These numbers must then be scaled and combined appropriately. We decided that if the length of any shortest path was greater than 6, we would set that length to 6. This allowed us to subtract 1, since no path has length 0, and divide by 5 to scale our results to be in the range $0 \leq n \leq 1$. Then

we decided that if a pair of words had more than 20 paths after 1 additional iteration of the breadth-first search that we set its number of paths to be 20. This allows us to divide by 20 to scale this metric to the range $0 \leq n \leq 1$. We subtract this value from 1 so that lesser values indicate stronger relationships as does the shortest path length. These two metrics are then combined with an arithmetic mean to get a single measure of semantic similarity.

2.2.2 Word Proximity

We tried three metrics of word proximity in our datasets. First, we went through the dataset for each pair of words, and for each instance of one of those words, we recorded the number of words between it and the nearest instance of the other word. These distances were summed up and divided by the sum of the number of instances of both words. This metric performed relatively well, but was surpassed by the next attempt.

Next we tried measuring the number of times in our dataset the two words appeared with fewer than 4 words in between them. We choose this measure because Cui, Mittal, and Datal[8] have concluded that there is an insignificant relationship between words more than 5 words apart (with 4 words between, or members of a 6 words phrase). This number is then divided by the minimum of the number of instances of the first or second word. This provided the best results, and is the metric implemented in the final algorithm.

Finally, we tried to improve upon the previous algorithm by counting the number of times the two words were in the same sentence rather than the number of times they were within six words. However, this provided inferior results and was abandoned.

2.3 Clustering the Graph

Finally, we cluster graph of keywords. Ideally, each cluster is a topic of a camera review. We experimented with various clustering algorithms, including k-means and principal direction gap partitioning[3]. We found that projecting the data down into a lower dimensional space with a singular value decomposition, then applying principal direction divisive partitioning and using k-means as a post-processing step works well for this type of problem. However, any effective graph clustering algorithm may be used.

3 Results

We applied all of these techniques to our datasets of camera reviews, weighting WordNet distance and word proximity equally.

From the Leica dataset, we found a list of 57 keywords. We asked our clustering algorithm for 15 clusters and got the following clusters.

1. image, images, color, quality, clarity
2. lens, optics, decent, sturdy
3. use, its
4. delicate, shipping, raw, mode, ratio
5. size, post, noise, flash, screen
6. canon, nikon, sony, packaging, mp
7. feature, format, shoot, lightweight
8. pictures, candid, landscapes
9. digital, compact, complicate, swears
10. options, menu, item, manual, settings, sensor, photos, photographer, worlds, shoots
11. camera, cameras
12. love, very, great, also, expensive
13. everyday
14. grandchildren
15. aspect

The results are mixed, but we are happy with this, given that it was an unsupervised process. The first cluster is clearly an "image quality" cluster. The second cluster is about the optics, but with "sturdy" thrown in. Even "sturdy" might belong, as a colleague has said that it is a term used in photography meaning that the camera takes clear pictures even if you have an

unsteady hand. The sixth cluster is about alternative cameras. The eighth refers to things that people take pictures of with their Leica. The tenth is the cluster about the "bells and whistles" of the camera. The twelfth expresses the common sentiment that the camera was great, but overpriced.

We quantify the success of our clustering by using entropy, a measure of cluster goodness. For a given cluster X whose proportions of topic i is x_i, the entropy of X is given by

$$H_X = -\sum_i x_i log_2(x_i)$$

We can use this metric to find an entropy of the entire clustering by taking a mean of all the cluster entropies, weighted by the number of words in the cluster. If there are n clusters, we can then divide this result by $log_2(n)$ to scale our result H to the range $0 \leq H \leq 1$, with 0 being the ideal clustering and 1 being the worst possible clustering.

When calculating the entropy of a clustering, a human must decide what is the "right" clustering. To mitigate the influence of these subjective decisions, we calculate two entropies - one in which we always make any unclear decisions in the most optimistic manner, and one in which we always make unclear decisions in the most pessimistic manner.

Our most optimistic normalized entropy is 0.2042, and our most pessimistic normalized entropy is 0.3247. This compares favorably with the expected entropy of a random clustering of these words, 0.440.

We also used the silhouette metric[7] to measure the quality of our clusterings. Specifically, we tried to use the silhouette to optimize α and the number of clusters to request. Due to the nature of the silhouette metric, as the number of clusters increases, silhouette increases. Therefore, silhouette was not an effective way to optimize the number of clusters. Silhouette was also not a useful measure to compare values of α. This is because silhouette measures the quality of the clustering, not the quality of the data that the clustering algorithm is given.

From the Canon dataset, we found a list of 55 keywords. We asked our clustering algorithm for 20 clusters and got the following clusters.

1. zoom
2. lens
3. stabilized, blurry
4. series, product, digital, canon, mp
5. purchase, outstanding
6. quality
7. reviews, como, muy, bueno, buena
8. cap
9. pictures, image, optical
10. review, shipping, mode
11. great, user
12. camera, cameras
13. photo, awesome, shots
14. price, noise
15. very, lo, owned, photographs, purchased
16. photos, display, videos, controls
17. video, recommend, battery
18. x, es, por, su
19. love, flash
20. much, nice, s, excellent

One notices immediately that there are some Spanish words in these clusters. This is because there were some Spanish reviews mixed in with English reviews on the internet. A possible goal in the future would be to more effectively filter out non-English words. The string "x" is listed as a keyword because the "12x optical zoom" of the camera is frequently mentioned, and numbers are not included in our dictionary, so "12x" is stored as "x". Similarly, "s" is a keyword because the name of the camera is the "Canon PowerShot Pro Series S5." There are several good clusters in these results. For example, the third cluster is clearly about image stabilization. The ninth is about image quality.

Our most optimistic entropy is 0.1261, and our most pessimistic entropy is 0.2034. This compares favorably with the expected entropy of a random clustering of these words, 0.2208.

e can see that the effectiveness of Aldteran is consistent over the two datasets. However, it is interesting to note the differences between them. A common topic in the Canon dataset is the lens cap because it falls off of that camera easily. Note that "cap" is a feature word for the Canon and not the Leica camera. The Canon does not come with a rechargeable battery and many customers did not like the placement of the battery door, so "battery" is listed for the Canon and not the Leica. The Canon has an exceptionally good zoom, unlike the Leica, so "zoom" and "x" are listed with the Canon. The Canon is a typical size camera, and referred to as "compact," whereas the Leica is considered "ultra-compact." The Canon weighs more than twice as much as the Leica does. The Leica is a more expensive camera and is often bought by people who are used to DSLRs. They repeatedly mention in the reviews that they enjoy the small size of the Leica and their ability to slip it in their pockets and use daily. It is notable that "lightweight," "size," "candid" and "everyday" are listed only for the Leica. Another frequent comment for the Leica camera is that it is overpriced, and the word "expensive" appears only in the Leica list. The words "raw" and "format" are only keywords for the Leica camera and the Leica is the only one that has an option of using the RAW format for a photograph instead of JPEG. Customers also commented that the Leica has a vintage metal structure that is very sturdy. "Sturdy" is only a keyword for Leica.

4 Conclusion

We have presented an unsupervised method of extracting topics from textual datasets. This method works in a three-step process, each of which can be improved independently of the others. We tested the method on two datasets of product reviews for digital cameras and got acceptable accuracy with both. We believe that with refinements to our methods, we can further improve these results.

5 Future Work

Our results are promising, but need improvement before becoming practical for real-world applications. We present here some possible improvements.

- Improve the quality of our corpora of general English language.
  - Expand the corpus with more scripts or other types of documents.
  - Clean non-words and misspellings out.
- Replace our word proximity measure with more sophisticated natural language programming.
  - Use natural language programming techniques to determine which words are actually related by the structure of the sentence.
- Apply dictionary-based stemming to the datasets.
  - Treat all valid forms of a word as the simplest form of that word. That is, no distinction should be made between "camera", "cameras," and "camera's."
- Develop an automated technique for selecting the number of clusters that we should request from our clustering algorithms.
- Integrate a spell-checker with replacement suggestions to minimize the error due to user misspellings.
- Investigate which word relationships are most important for measuring semantic relationships.

- Find a technique for optimizing the following parameters.
  - The relative weights with which semantic similarity and word proximity are combined.
  - The number of columns of W to request in our non-negative matrix factorization.
  - The number of words to add to the pool from each column of W in the non-negative matrix factorization.
  - The number of terms to keep from the pool of words generated by the NMF.

The next logical step of this research is to use these topics to score the datasets for the positivity or negativity of their opinions on these topics. We have only begun preliminary work on this problem, and much more can be done to this end.

6 Acknowledgments

This research project was funded by the National Science Foundation. It was mentored by Carl Meyer, Shaina Race, and Ralph Abbey at North Carolina State University.

References

[1] Word frequencies from tv and movie scripts. `http://en.wiktionary.org/wiki/Wiktionary:Frequency_lists`, 2006.

[2] John W. Eaton. *GNU Octave Manual*. Network Theory Limited, 2002.

[3] Meyer Race Abbey et al. Search engines and data clustering. 2007.

[4] Christiane Fellbaum. *Wordnet: An Electronic Lexical Database*. Bradford Books, 1998.

[5] Hirst and St. Onge. Lexical chains as representations of context for the detection and correction of malapropisms. *Fellbaum*, pages 305–332, 1998.

[6] Patrik O Hoyer. Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research*, (5):1457–1469, 2004.

[7] Jacob Kogan. *Introduction to Clustering Large and High Dimensional Data*. Cambridge University Press, 2007.

[8] Cui Mittal and Datar. Comparative experiments on sentiment classification for online product reviews. *AAAI*, 2006.

[9] Porter and Boulton. Snowball stemming c library. `http://snowball.tartarus.org/`.

[10] Jason Rennie. Wordnet::querydata: a Perl module for accessing the WordNet database. `http://people.csail.mit.edu/~jrennie/WordNet`, 2000.

[11] M. Thelwall and R. Prabowo. Identifying and characterising public science-related fears from rss feeds. *Journal of the American Society for Information Science and Technology*, 58(3):379–390, 2007.