PATHFINDER – AN ONLINE SHOPPING ASSISTANT DRIVEN BY DATA MINING

BY

AKSHAT GUPTA

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2017

Urbana, Illinois

Adviser:

Professor Roy Campbell

ABSTRACT

Online Shopping is a household phrase that has been extremely successful in easing the lives of many people across the globe. Online shoppers spend ample amounts of time and money in buying products that they receive at their doorstep in a matter of a few days or, in some cases, a few hours. However, it is not as easy as it looks. People either have plenty of specifics in their mind before buying a product or are just looking to explore a range of products for a particular goal. This adds another layer on top of time and money spent – effort.

PathFinder is a guide that helps shoppers make more informed choices and reach their final product decision faster. It is a hand-in-hand assistant for shoppers that helps them at critical stages of the buying process to ensure that they either reach their specifics without too much research or that they get to explore granular details which they would miss otherwise. Being an online shopping assistant, *PathFinder* aims to reduce the effort spent by online shoppers and eases up the online product purchasing process even further.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

A decade ago, it was normal for people to queue up in stores to buy products. Nowadays, this is no longer the norm as 51% Americans prefer to shop online [7]. It is not hard to think of why this is the case. People can shop for their favorite items in the comfort of their couches and browse through a diverse display of products, each product containing multiple brands, sellers, and options. In the inventories of companies like Amazon, Walmart, eBay, and others, one can find a range of products like electronics, home products, kitchen appliances, video games, books, just to name a few. As a result of this comfort, Americans in metropolitan areas spend about 4.5 hours every week shopping for products online, whereas Americans in rural and suburban areas spend around 5 hours per week doing the same [7]. This small difference might be because it is a little easier for people in urban areas to physically visit stores if they want do not want to buy some product online. Another dimension of this comfort comes in the form of the money spent on online shopping. People in urban areas tend to spend upward of $800 annually and those in the rural areas spend about $700 per year to buy products online [7].

While shopping online, there is a plethora of information that is thrown at a user. This includes product names, product images, product description, reviews, prices and advertisements. Out of these inputs for a product, the set of reviews is a very vital source of information for users. Around 61% of users read reviews before buying a product as they feel that they would make a more informed decision after reading the reviews [6]. Since users tend to look at several similar products before buying what they want, they have to consume, process, and scan through a lot of text information to make their final choice. Such information consumption is important because reading reviews or looking at pictures is the only way to compensate for the inability to physically touch the products. However, that amount of information is a double edged sword – it provides tremendous crowd-sourced detail but at the same time can be a major hindrance. Around 21% of Americans [7] feel that websites become hard to navigate when there is an abundance of information and this abundance can very quickly become a bane from a boon, thereby driving away users.

PathFinder aims to solve the issue of abundance of information that users need to parse through by extracting relevant parts of the content and displaying them in the regular purchasing flow of walmart.com. Websites like amazon.com, walmart.com, and ebay.com have perfected the art of buying flow over the years. Users reach a set of products either by searching for it or by recommendations, look at the description, price, and pictures, read reviews, and after looking at other similar products, make a decision and proceed to payment. This flow is as good as it gets and needs no changing. *PathFinder* uses the same content and the same ordering flow however, it provides the users key highlights along this flow. The system would link key phrases in the product descriptions to the relevant reviews so that users can quickly notice certain features and jump to the reviews that talk about those. Also, based on those key phrases, it would generate related words and highlight the context in which the related words are used in the review. Using this system, in the ideal case, a user would have to read minimally through reviews and after a few clicks and glances over the highlighted parts, be able to make an informed decision. This would in turn make it faster for users to decide on which product to buy. It would also make it easier for them to navigate through websites like walmart.com which have daunting mountains of text, which currently the users have to spend a lot of time parsing and figuring out which parts of the reviews are relevant and which are not. These metrics have been evaluated later in Chapter 6. By using *PathFinder*, users would be able to get condensed information and be able to make faster and more holistic decisions for buying products.

PathFinder is built on top of walmart.com using the Walmart API [8]. It uses the search API provided by Walmart to access product listings. walmart.com does not allow a user to search through the reviews so as the first step, *PathFinder* indexes reviews to create a search engine for product reviews. Using this index, it hyperlinks key phrases in the product descriptions to the relevant reviews that mention those phrases. When the user reaches a set of reviews either through the previously mentioned hyperlinks or by manually searching for keywords, *PathFinder* uses Mutual Information to predict which words are likely to appear together with the phrases or keywords. This thesis will deep dive into each of these components of *PathFinder*, provide a walkthrough of a common search made by users, and explain results of product testing that was carried out.

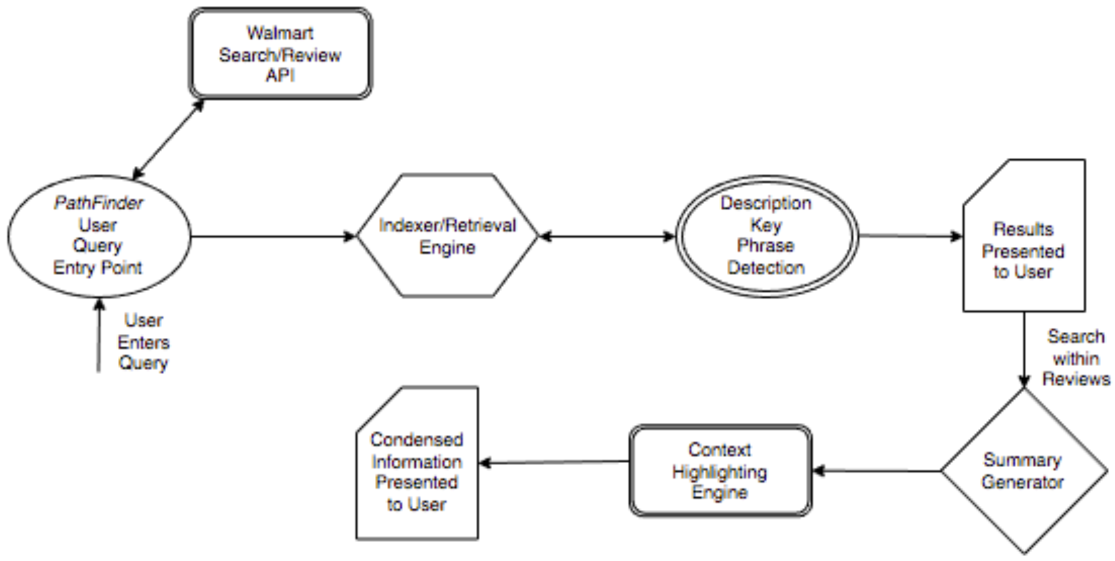

Walmart Search/Review API
PathFinder User Query Entry Point
User Enters Query
Indexer/Retrieval Engine
Description Key Phrase Detection
Results Presented to User
Search within Reviews
Summary Generator
Context Highlighting Engine
Condensed Information Presented to User

Figure 1.1: *PathFinder*'s overall architecture and flow.

CHAPTER 2: SEARCH ENGINE

PathFinder uses the Walmart Search API to query for products requested by a user. The API returns a set of products along with product names, descriptions, and reviews. After retrieving these results, *PathFinder* uses its search indexer to index the reviews.

2.1 BUILDING THE SEARCH INDEX

Suppose there are 3 products namely, p_0, p_1, p_2. For simplicity, let us assume that there is only 1 review for each product. The following are the reviews:

Table 2.1: Example of products and corresponding reviews

Product	**Review**
p_0	Hi is this the product.
p_1	This product is the best.
p_2	How is it so good.

The first step before indexing this data is to clean it. The indexer of *PathFinder* removes all punctuations and converts all words to lowercase as part of the cleaning process. After cleaning this data, an Inverted Index is created. Inverted Index is a mapping of documents (in this case documents are products) to the different unique tokens (in this case tokens are the words in the reviews) in them. Along with unique tokens, *PathFinder*'s indexer also stores the positions of the tokens in the documents. So, the indexer indexes products $p_i\ where\ 0 \leq i \leq n$ and n is the total number of products. At the end of this stage, this is what the Inverted Index looks like:

Table 2.2: Inverted Index after cleaning the data

Product	Words and Positions
p_0	hi: [0], is: [1], this: [3], the: [4], product: [5]
p_1	this: [0], product: [1], is: [2], the: [3], best: [4]
p_2	how: [0], is: [1], it: [2], so: [3], good: [4]

The final step in creating the search index is to combine the mappings from the Inverted Index to create a Forward Index. A Forward Index would then be a mapping of words in the reviews to the documents they occur in, along with the corresponding positions. The Forward Index looks as follows.

Table 2.3: Forward Index as a result of combining the mappings from the Inverted Index

Words	Products and Positions
hi	p_0: [0]
is	p_0: [1], p_1: [2], p_2: [1]
this	p_0: [2], p_1: [0]
the	p_0: [3], p_1: [3]
product	p_0: [4], p_1: [1]
best	p_1: [4]
how	p_2: [0]

Table 2.3 (cont.)

Words	**Products and Positions**
it	p_2: [2]
so	p_2: [3]
good	p_2: [4]

The Forward Index is what is used by *PathFinder*'s Search Engine to perform different types of queries on the reviews. Currently, it supports Simple queries, Phrase queries, and Proximity Search queries. To explain each of these types of queries, consider a query q containing m terms such that for any given term q_i in $q, 0 \leq i \leq m$.

2.2 SIMPLE QUERIES

Simple Queries are the type of queries that simply search for the documents that the query terms appear in. For example, let the query be "is". Thus, q_0: "is". For every term in the query q, which in this case is just q_0, *PathFinder* would use the Forward Index to form a union of all documents containing the query terms. Since, q_0 is present in p_0, p_1, p_2, the set of documents returned would be p_0, p_1, p_2. Similarly, for query term "product", the result would be p_0, p_1. As a final example, taking "this product is" as a query, the set of documents for q_0 ("this") is p_0, p_1 and the set of documents for q_1 ("product") is p_0, p_1 and for q_2 ("is") is p_0, p_1, p_2. Therefore, the union of all these documents would form the final result i.e. p_0, p_1, p_2.

Even though this simple queries algorithm is useful in extracting documents based on the occurrence of words, it does not work well in capturing the context in which each of the query terms are used and also assumes that each term in the search query is equally important [1]. For example, for a query "this bad product", the simple query algorithm would return all documents where “this”, “bad”, and “product” are used. However, it is not necessary that the term “bad” is used in the context of “product” and the returned document(s) would not be useful to the user.

2.3 PHRASE QUERIES

Phrase Queries do slightly better than Simple Queries in terms of capturing the context of the query terms in the query. The phrase query algorithm ensures that it only retrieves documents which contain all the terms in the query, unlike the simple query algorithm which only checks for occurrence. Apart from checking for the presence of all query terms, it also checks whether the terms appear consecutively or not in the document. If they do not, then the document is not returned. Since it uses the same Forward Index, retrieval is more efficient than substring matching.

Suppose the query is “this product”. Thus q_0 is “this” and q_1 is “product”. According to the simple query algorithm, documents p_0 and p_1 would be returned. The Phrase Query algorithm would check the positions of q_0 and q_1 , which in this case would be positions 2 and 4 for p_0 and positions 0 and 1 for p_1. Since the positions in p_0 are not continuous, p_0 would not be retrieved and since they are continuous in p_1, so only p_1 would be retrieved.

Going back to the example of the query being “this bad product”, using the Phrase Query algorithm, only the documents which contain these terms consecutively would be returned. This captures the context in which the terms are being used and thus solves the limitation of Simple Queries.

2.4 PROXIMITY OF SEARCH QUERIES

The Proximity Search Query algorithm is similar to the Phrase Query algorithm but instead of checking for continuous occurrence of query terms, it checks if the query terms lie within a pre specified proximity of each other. *PathFinder* uses an online search query size survey [9], to determine the appropriate proximity between the search query terms.

As an example, taking the same query "this product" and a proximity window of 1, the Proximity Search Query algorithm would return the same documents as the Phrase Query algorithm i.e. only p_1. This is because in p_0, "this" and "product" are separated by a distance of 2. However, if the proximity window is increased to 2, then the algorithm would return p_0 *and* p_1 since "this" and "product" lie within a distance of 2 in both these documents.

PathFinder uses Proximity Search under the hood of the review search engine. Another key component of this algorithm is selecting the proximity window size. According to [9], more than 60% of the queries have 1 or 2 words and around 95% of the queries have 5 words of less. Since the proximity window is just an upper limit, it is selected to be 5 to allow for a smoother cushion even if the query terms are less than 5 terms apart.

The reason for using Proximity Search is because users would not necessarily use all the query terms together while writing reviews. That being said, query terms that appear within a certain distance of each other are likely to be used in the same context. So Proximity Search retrieves documents which capture the context of usage of the query terms even if the terms do not appear consecutively in the documents. Using this search engine, users can now simply search through the reviews quickly without having to read through them manually one after the other.

CHAPTER 3: DESCRIPTION KEY PHRASES DETECTION

While buying products, users read product descriptions before even reading the reviews. As discussed above, users have either specific details that they are looking for or they are simply exploring varied features. The product descriptions provide a concise summary of the product features and details but currently, there is no bridge between the descriptions and the reviews. So users would look at a certain detail in the description, search for it in the reviews, and similarly go back and forth between the description and the reviews till they are satisfied or they reject the product. *PathFinder* simplifies this process and connects the descriptions with the reviews by detecting key words and phrases in the descriptions and linking them to the relevant reviews. Within a given product description, not all words or sub phrases are important for a user. *PathFinder*'s key phrases detection algorithm is built on the assumption that users would be interested in specific details in the description that other users are also talking about.

3.1 SLIDING WINDOW MECHANISM

Suppose there is a product description d. d contains words or terms along with several punctuations. Also, several words in d would be English 'stop words' or words that are very commonly used in English language. *PathFinder* maintains a sliding window which is bounded by an upper limit size of 3, thus can be of size 0, 1, 2, and 3. This sliding window glides over the description for each possible window size and plugs part of the description that is within the sliding window into the Phrase Query algorithm in the review search engine. It then checks if the search engine returned any document and since it performs this operation for each possible window size, it would consider a phrase to be a key phrase if it is bounded by the maximum possible window size for which there is at least one document that is returned.

As the first step in this process, *PathFinder* cleans the description d by removing all the punctuations and then tokenizes d by splitting it on spaces. For this example, consider the tokenized version of d to be d_t, thus d_t is a list of tokens after splitting d by spaces. If the starting position of the sliding window points at a token which is a stop word, then the sliding window skips over that position to the next position. From a particular starting position of the sliding window s, the possible phrases are given by the following equation:

$$d_{t[s,\ s+k]} \text{ where } 1 \leq k \leq 3 \tag{3.1}$$

This equation would create a list of sub-phrases from d_t and query the search engine using the Phrase Query algorithm to check if any review is returned. *PathFinder* would plug in each sub-phrase from the generated list and select the one which is of maximum length and for which more than 1 review is returned by the search engine. After selecting a particular key phrase, *PathFinder* creates a hyperlink to the reviews that are returned by the search engine. The sliding window then jumps to the starting position of the phrase that lies right after the selected phrase. This process takes place till the end of d_t is reached and at the end, *PathFinder* would have converted the original description into a description which has the same text as before, but with certain key phrases hyperlinked to the relevant reviews that those key phrases appear in.

CHAPTER 4: SUMMARY GENERATION

At this point, *PathFinder* is equipped to allow users to search for products, search within the reviews of all products, and also highlight key words or phrases in the description and link them to the relevant reviews. *PathFinder* then aims to generate related words to the word or phrases that the user searched for or clicked from the description hyperlinks using Word Association Mining methods described in [1]. In addition to this, it would also highlight the context in which the related words occur along with them within reviews, thereby creating a specific summarized view of the information. Thus, the two main tasks in summary generation are to generate the related words and to highlight the context.

4.1 RELATED WORDS GENERATION

Consider the following documents:

Table 4.1: Sample documents to illustrate Related Words Generation

Product	**Review**
d_0	The pan is made of non stick material.
d_1	The spatula is made of steel.
d_2	Great to cook meat on the pan.
d_3	Set includes both the pan and the spatula.
d_4	The spoons are made of silver.

Generating related words of a given word is all about mining associations between the words from a given set of documents. In the field of Information Retrieval, there are two types of word associations – Paradigmatic relation and Syntagmatic relation.

According to [1], in a Paradigmatic relationship, two words w_1 and w_2 can be used in place of each other while still keeping the overall meaning of the sentence the same. These words belong to the same semantic or syntactic class. In documents d_0 through d_4, an example of paradigmatic relationship would be between “spatula” and “spoon” since they can be used interchangeably. If we replace “spatula” with “spoon” in d_3, then d_3 would become “The set includes both the pan and the spoon.”, which still conserves the general context of the sentence. However, words like “pan” and “set” do not share a paradigmatic relationship because their interchange or substitution would not lead to a meaningful sentence.

In a Syntagmatic relationship [1], two words w_a and w_b are semantically related i.e. they can be combined together to form a meaningful phrase. In documents d_0 through d_4, (“pan” and “made”) and (“meat” and “cook”) are some examples of pairs of words that share a syntagmatic relationship. The context that would be created by (“pan” and “made”) could be thought of as “made in a pan” and the same created by (“meat” and “cook”) could be “cook meat”. Such pairs of words or phrases do not necessarily have to fit perfectly with each other but rather create a well perceivable context by their combination.

PathFinder makes use of Syntagmatic relationships between words and phrases to generate related words. In the case of online product shopping, suppose users search for “crockery” and then search for “pan” within the reviews of the product listing, they probably want to understand the contexts in which “pan” is used by other users in their writing. The users would be better off looking at related words like “set” or “made” instead of “spoon” or “spatula”. As described above (“pan” and “set”) creates the context of information about a set of pans and (“pan” and “made”) creates the context of “things made in a pan”. (“pan” and “spoon”) on the other hand, does not provide as much information about the “pan” as “set” and “made” do. These reasons and examples warrant the use of Syntagmatic relationships between words and phrases for the generation of related words.

4.2 SYNTAGMATIC RELATIONSHIPS EXTRACTION

Words that co-occur tend to have a syntagmatic relationship with each other. Considering the documents in Table 4.1 and for extracting syntagmatic relationships among words, the most important question is given word w_1, what are the other words that are likely to occur in the same context. For example, if w_1 is "pan", some words that are likely to co-occur are "made" and "set". If w_1 is "meat", then the co-occurring words would be "cook" and "pan".

4.2.1 CONDITIONAL ENTROPIES

The first method of extracting syntagmatic relationships is by using Conditional Entropies described in [1]. Given a word w', the task is to figure out what words tend to co-occur with w'. Suppose X_w is a random variable which simply depicts whether the word w is present or not, thereby the domain of X_w is $\{0, 1\}$. Although we need to check all words against w', for simplicity assume that we calculate the tendency of w co-occurring with w'.

One idea to perform this extraction would be to simply calculate the probability of the occurrence of w without taking anything into account about w'. Therefore, one would just need to calculate the following values when w is "meat" and w' is "cook":

$$P(X_{meat} = 1) \tag{4.1}$$

This could be performed for all words in the corpus and simply take the one that has highest probability. However, such an approach does not say anything about the co-occurrence of the 2 words "meat" and "cook". Thus, in the second approach, consider the following equations:

$$P(X_{meat} = 1 \mid X_{cook} = 1) \text{ and } P(X_{meat} = 0 \mid X_{cook} = 1) \tag{4.2}$$

The equations above now try to predict the occurrence or non-occurrence of “meat” given that “cook” is present. Using these equations, we will extract syntagmatic relations with the help of conditional entropies.

The conditional entropy of “meat” and the conditional entropy of “meat” given that “cook” is present or not present are defined as follows:

$$H(X_{meat}) = -P(X_{meat} = 1)\log_2(X_{meat} = 1) - P(X_{meat} = 0)\log_2(X_{meat} = 0) \quad (4.3)$$

$$H(X_{meat} \mid X_{cook} = 1) = -P(X_{meat} = 1 \mid X_{cook} = 1)\log_2(X_{meat} = 1 \mid X_{cook} = 1) - P(X_{meat} = 0 \mid X_{cook} = 1)\log_2(X_{meat} = 0 \mid X_{cook} = 1) \quad (4.4)$$

$$H(X_{meat} \mid X_{cook} = 0) = -P(X_{meat} = 1 \mid X_{cook} = 0)\log_2(X_{meat} = 1 \mid X_{cook} = 0) - P(X_{meat} = 0 \mid X_{cook} = 0)\log_2(X_{meat} = 0 \mid X_{cook} = 0) \quad (4.5)$$

Entropy is a measure of calmness within a system. In terms of syntagmatic words extraction, the entropy would be high if two words do not tend to co-occur and vice versa. For example, the entropy of just “meat” would be higher than the entropy of “meat” given that “cook” is also present because the presence of “cook” brings the prediction closer to predicting “meat”. Therefore, $H(X_{meat} = 1)$ is higher than $H(X_{meat} = 1 \mid X_{cook} = 1)$. On the extreme case, predicting “meat” given that “meat” is present would be 0 since the prediction would be 100% accurate.

This approach seems reasonable but has a major shortcoming. While using conditional entropies for syntagmatic relationships extraction, one needs to iterate over all the words in a corpus and calculate the value of the conditional entropy. This is given by the following equation:

$$H(X_w = 1 \mid X_{w'} = 1)\ \forall w\ \in V \quad (4.6)$$

where V, w, w' are the vocabulary of the corpus, word in the corpus, and the word for which we need to find co-occurrences respectively.

After getting these values, a method to extract the top words that share a syntagmatic relationship would be to define a cutoff and ignore words that lie below it. The problem with this approach is that the values that are calculated for a particular w cannot be compared across different w's. Each would have a different cutoff and renders this approach inconsistent.

4.2.2 MUTUAL INFORMATION

Since the conditional entropy scores are not comparable across different words in the approach described above, it is challenging to find the words that correlated or co-occur with each other the most. To solve this problem, *PathFinder* uses Mutual Information as described in [1]. Mutual Information is depicted by the following equation:

$$I(X;Y) = H(X) - H(X \mid Y) \tag{4.7}$$

$I(X;Y)$ stands for a reduction in the entropy of X once the presence of Y is confirmed. The reason why this allows the comparison of different words is because it measures a magnitude of reduction, a metric that can be used to estimate usefulness or in this case, degree of co-occurrence. Thus, while ranking words that co-occur with a given word X, one would choose the top-k words that lead to the greatest reduction in $H(X)$. Consider the following relation:

$$I(X_{meat}\,; Y_{cook}) > I\big(X_{meat}\,; Y_{spoon}\big) \tag{4.8}$$

This relation indicates the presence of “cook” reduces the entropy of “meat” more than the reduction caused by “spoon”. Thus, “cook” co-occurs to a greater extent with “meat” than “spoon” does in the corpus.

For words w and w', where the task if to find the reduction in entropy [1] in w caused by w', *PathFinder* uses the following equation:

$$I(X_w\,;X_{w'}) = \sum_{u\in\{0,1\}}\sum_{v\in\{0,1\}} P(X_w = u, X_{w'} = v)\log_2 \frac{P(X_w=u,X_{w'}=v)}{P(X_w=u)P(X_{w'}=v)} \quad (4.8)$$

The numerator within the log is the joint distribution and the denominator is the expected joint distribution on the assumption that X_w and $X_{w\prime}$ are independent. This equation is a form of the KL-Divergence method which estimates the the difference between the actual joint distribution and the expected joint distribution on the basis of independence of X_w and $X_{w\prime}$. Taking a look at the two summation operation, mutual information uses all possible values of occurrence i.e. present or not present thus, 1 or 0 respectively and sums over all these values. It is clear that the the higher the divergence is, the greater the mutual information would be. In order to compute the mutual information using the KL-Divergence approach, the following are the probabilities that are important:

$$P(X_w = 1) + P(X_w = 0) = 1 \quad (4.9)$$

$$P(X_{w\prime} = 1) + P(X_{w\prime} = 0) = 1 \quad (4.10)$$

Since marginal probabilities sum up to 1,

$$P(X_w = 1, X_{w'} = 1) + P(X_w = 1, X_{w'} = 0) + P(X_w = 0, X_{w'} = 0) + P(X_w = 0, X_{w\prime} = 1) = 1 \quad (4.11)$$

Also,

$$P(X_w = 1, X_{w\prime} = 1) + P(X_w = 1, X_{w\prime} = 0) = P(X_w = 1) \quad (4.12)$$

$$P(X_w = 1, X_{w\prime} = 0) + P(X_w = 0, X_{w\prime} = 0) = P(X_{w\prime} = 0) \quad (4.13)$$

$$P(X_w = 0, X_{w\prime} = 0) + P(X_w = 0, X_{w\prime} = 1) = P(X_w = 0) \quad (4.14)$$

$$P(X_w = 1, X_{w\prime} = 1) + P(X_w = 0, X_{w\prime} = 1) = P(X_{w\prime} = 1) \quad (4.15)$$

These probabilities can be plugged into the following equation:

$$I(X_w\,;X_{w'})\ \forall w' \in V \quad (4.16)$$

This would return a list of (word, score) pairs where the word is every word in the vocabulary and the score is the mutual information score. *PathFinder* simply sorts this list and takes the top-5 words to be the words that have highest co-occurrence with w.

4.2.3 OPTIMIZATIONS AND SMOOTHING

PathFinder also makes optimizations suggested in [1] in terms of computation of these probabilities. The only values that need to be calculated are $P(X_w = 1)$, $P(X_{w\prime} = 1)$, and $P(X_w = 1, X_{w\prime} = 1)$. The following are the optimizations:

1. $P(X_w = 0)$ does not need to be calculated as it is $1 - P(X_w = 1)$.
2. Similarly, $P(X_{w\prime} = 0)$ does not need to be calculated as it is $1 - P(X_{w\prime} = 1)$.
3. $P(X_w = 1, X_{w\prime} = 0)$ can be computed using $P(X_w = 1) - P(X_w = 1, X_{w\prime} = 1)$.
4. $P(X_w = 0, X_{w\prime} = 0)$ is calculated by doing $P(X_{w\prime} = 0) - P(X_w = 1, X_{w\prime} = 0)$.
5. $P(X_w = 0, X_{w\prime} = 1)$ can be calculated using $P(X_w = 0) - P(X_w = 0, X_{w\prime} = 0)$.

Such optimizations not only improve the time complexity of calculating the values but also reduce the amount of storage space required for the different variables.

To compute the probability, a simple idea is to using MLE or Maximum Likelihood Estimation with the help of the following equations:

$$P(X_w = 1) = \frac{count(w)}{N} \qquad (4.17)$$

$$P(X_{w\prime} = 1) = \frac{count(w\prime)}{N} \qquad (4.18)$$

$$P(X_w = 1, X_{w\prime} = 1) = \frac{count(w,w\prime)}{N} \qquad (4.19)$$

where w and $w' \in V$, N is the number of documents, $count(.)$ is the number of documents containing the word or joint occurrence of words.

The problem with this approach is that there might be certain probabilities that might be 0. To avoid this issue, *PathFinder* uses smoothing according to the following modified equations:

$$P(X_w = 1) = \frac{count(w)+0.5}{N+1} \tag{4.20}$$

$$P(X_{w\prime} = 1) = \frac{count(w\prime)+0.5}{N+1} \tag{4.21}$$

$$P(X_w = 1, X_{w\prime} = 1) = \frac{count(w,w\prime)+0.25}{N+1} \tag{4.22}$$

By adding a small noise like 0.5 and 0.25, it is guaranteed that no probability would be 0 even if there are some words or word combinations whose count is 0 and also ensures that words or combinations that have 0 occurrences are not rewarded by a big amount.

4.3 CONTEXT HIGHLIGHTING

After generating related words of a search query or phrase, *PathFinder* aids the users further by highlighting the context in which these related terms are used with the search key words. Suppose a user enters a search query q and *PathFinder*, using the Mutual Information algorithm described previously, outputs a set of related terms. For each of these related terms, *PathFinder* would perform the highlighting process when the user clicks on a given related term r.

It is simple to extract the indices of the first occurrence of q and r and just highlight the part of the review spanning those indices. However, there might be multiple occurrences of either q or r or both. So it is important to highlight the span that pertains to the context of q and r. *PathFinder* computes the index of the first occurrence of q, suppose q_i and then extracts the index of the first occurrence of r, suppose r_i that is closest to q_i. The last step is the highlight the terms from q_i to r_i.

It is also possible that the search query consists of multiple terms $q_0, q_1, \dots, q_n$ where n is the number of terms in the search query. In this case, *PathFinder* would extract the indices of the first occurrence of each query term to form a list of indices. Suppose this list is $[q_{0i}, q_{1i}, \dots, q_{ni}]$. The next step is to append to this list the index of the first occurrence of r, called r_i. Thus the list becomes $[q_{0i}, q_{1i}, \dots, q_{ni}, r_i]$. Since this list is not necessarily sorted, *PathFinder* simply takes the minimum value and maximum value, which are nothing but the starting index and ending index of the span respectively. The range gives *PathFinder* the correct context pertaining to the search query and the generated related word.

The idea behind highlighting such contexts is that it is not enough to just show the users the related words. A related word might give a certain association to the user related to the search query but will not be able to further the association in terms of whether it is a positive or negative association, or how the two entities are actually related to each other. For example, if the search query is “battery” and the related word generated is “life”, then without any highlighting the users would get to know that other users have associated “battery” with “life” but will not know any more information about the association. By highlighting the context, users would be able to know how “life” is used with “battery” by reading a few words around “battery” and “life” in the review and be able to sense the context without reading the whole review.

CHAPTER 5: PATHFINDER WALKTHROUGH

5.1 LANDING PAGE

The first part of *PathFinder* is the search page, on which users can perform searches for products of their choice. Below are the screenshots for the landing page and then followed by a few queries:

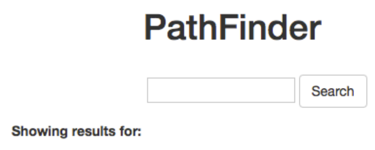
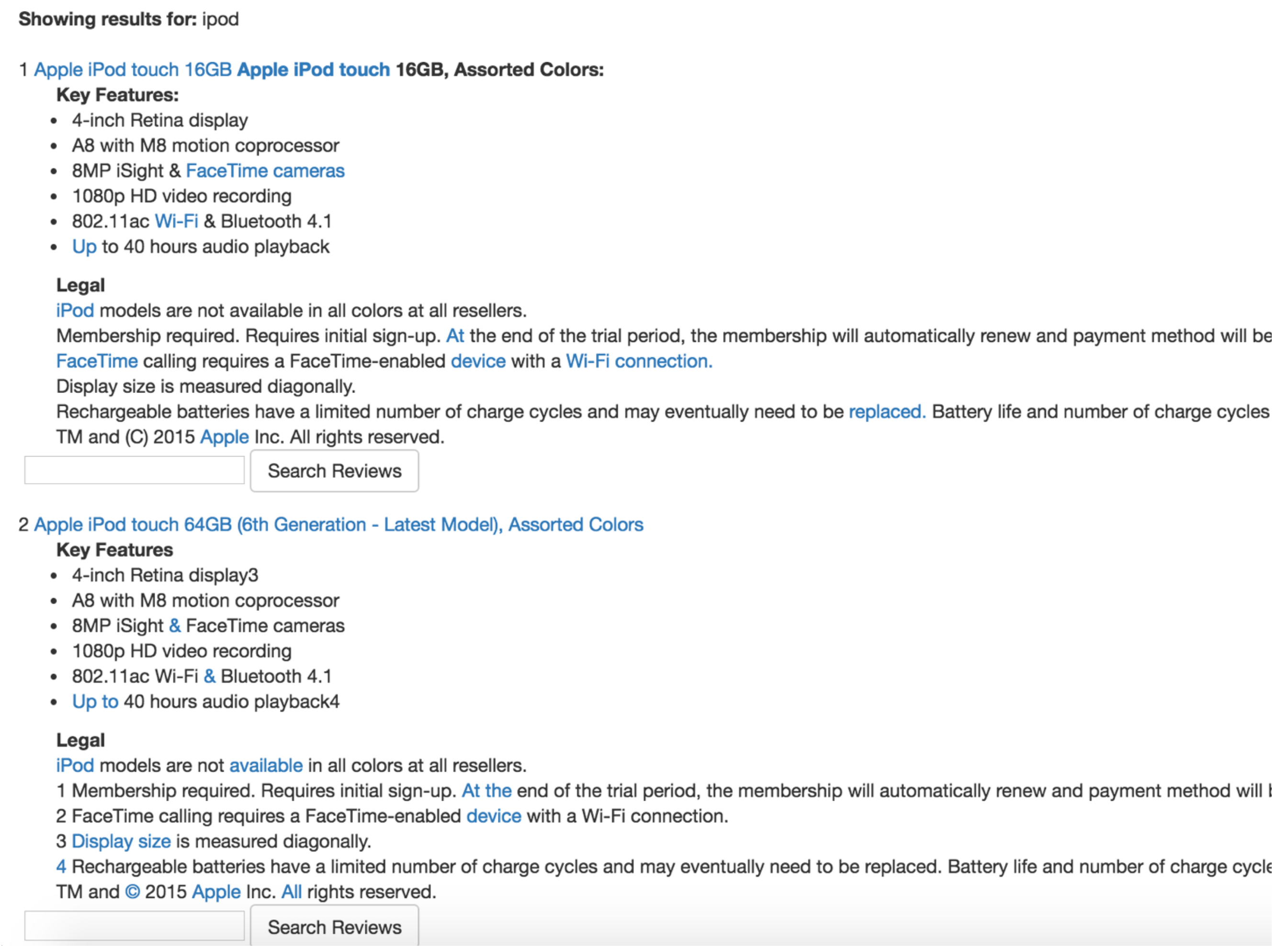


Figure 5.1: *PathFinder*'s landing page.

The user can then start to type in the query in the search box and press the "Search" button. As part of the walkthrough, we will make searches for "ipod", "zoom camera", and "desk lamp". These queries are based on common searches made by users online i.e. for electronics and appliances. The reason for using common themed search queries is because they are bound to have more information in the form of reviews and descriptions, which would aim in underscoring the features of *PathFinder*.

5.2 SEARCH RESULTS AND DESCRIPTION KEY PHRASES DETECTION



Showing results for: ipod

1 Apple iPod touch 16GB **Apple iPod touch 16GB, Assorted Colors:**

Key Features:

- 4-inch Retina display
- A8 with M8 motion coprocessor
- 8MP iSight & FaceTime cameras
- 1080p HD video recording
- 802.11ac Wi-Fi & Bluetooth 4.1
- Up to 40 hours audio playback

Legal

iPod models are not available in all colors at all resellers.
Membership required. Requires initial sign-up. At the end of the trial period, the membership will automatically renew and payment method will be
FaceTime calling requires a FaceTime-enabled device with a Wi-Fi connection.
Display size is measured diagonally.
Rechargeable batteries have a limited number of charge cycles and may eventually need to be replaced. Battery life and number of charge cycles
TM and (C) 2015 Apple Inc. All rights reserved.

2 Apple iPod touch 64GB (6th Generation - Latest Model), Assorted Colors

Key Features

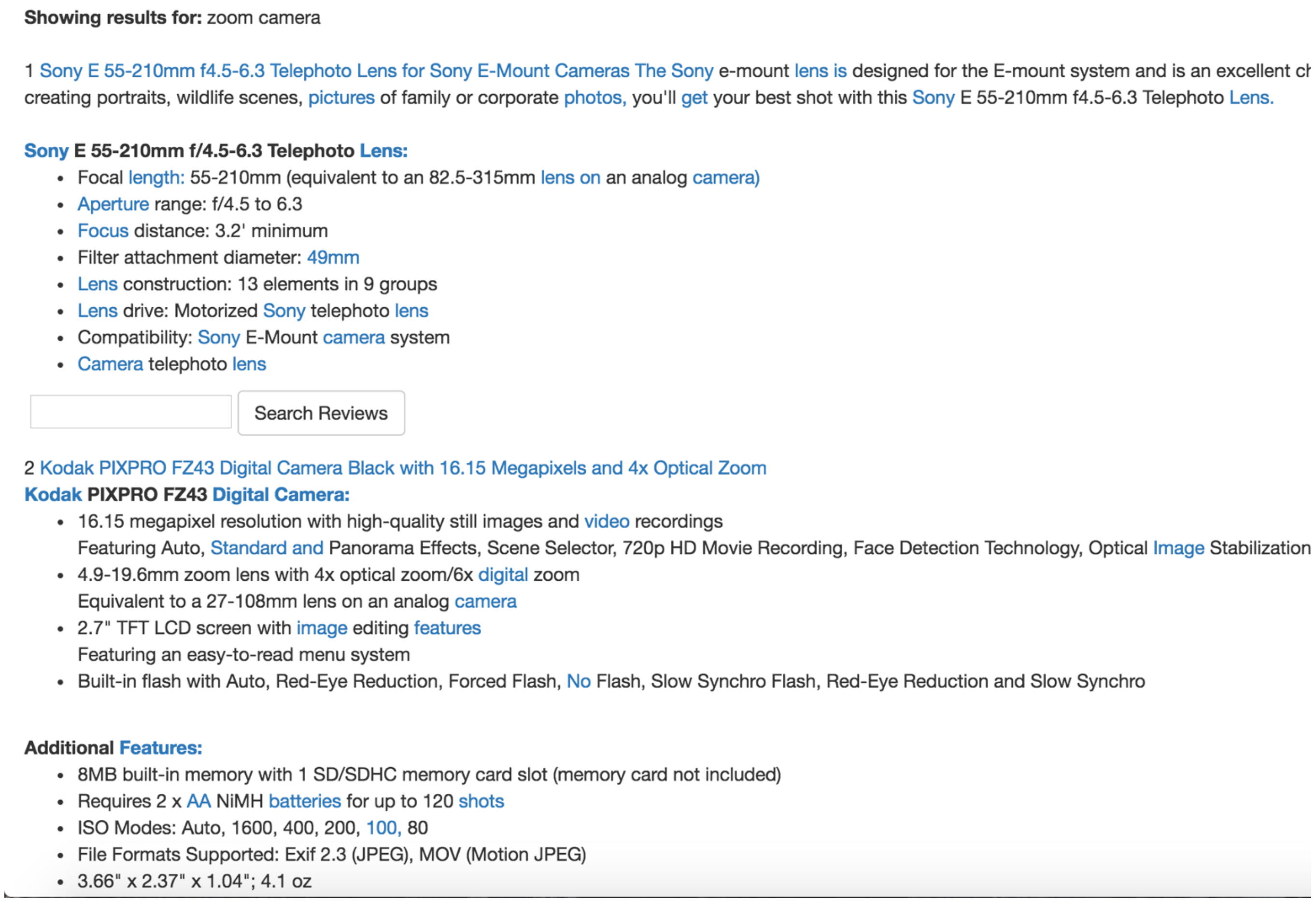
- 4-inch Retina display3
- A8 with M8 motion coprocessor
- 8MP iSight & FaceTime cameras
- 1080p HD video recording
- 802.11ac Wi-Fi & Bluetooth 4.1
- Up to 40 hours audio playback4

Legal

iPod models are not available in all colors at all resellers.
1 Membership required. Requires initial sign-up. At the end of the trial period, the membership will automatically renew and payment method will
2 FaceTime calling requires a FaceTime-enabled device with a Wi-Fi connection.
3 Display size is measured diagonally.
4 Rechargeable batteries have a limited number of charge cycles and may eventually need to be replaced. Battery life and number of charge cycle
TM and © 2015 Apple Inc. All rights reserved.

Figure 5.2: Search results for search query "ipod". The image has been cropped from the right to increase the size of the image.

The first line of each product listing is a hyperlink to the product on walmart.com. As discussed in Chapter 3, *PathFinder* detects key words and phrases in the descriptions and hyperlinks them to the relevant reviews. In this listing, "FaceTime cameras", "device", "Wi-Fi connection", and "replaced" are some terms and phrases that are key to the "ipod". There are also some terms like "At the" and "Up to" that do not add benefit for the users. Overall, by looking at this page, users would easily be able to glance over these key phrases and jump to the reviews to read more about the context of these key phrases.


Showing results for: zoom camera

1 Sony E 55-210mm f4.5-6.3 Telephoto Lens for Sony E-Mount Cameras The Sony e-mount lens is designed for the E-mount system and is an excellent ch
creating portraits, wildlife scenes, pictures of family or corporate photos, you'll get your best shot with this Sony E 55-210mm f4.5-6.3 Telephoto Lens.

Sony E 55-210mm f/4.5-6.3 Telephoto Lens:
- Focal length: 55-210mm (equivalent to an 82.5-315mm lens on an analog camera)
- Aperture range: f/4.5 to 6.3
- Focus distance: 3.2' minimum
- Filter attachment diameter: 49mm
- Lens construction: 13 elements in 9 groups
- Lens drive: Motorized Sony telephoto lens
- Compatibility: Sony E-Mount camera system
- Camera telephoto lens

Search Reviews

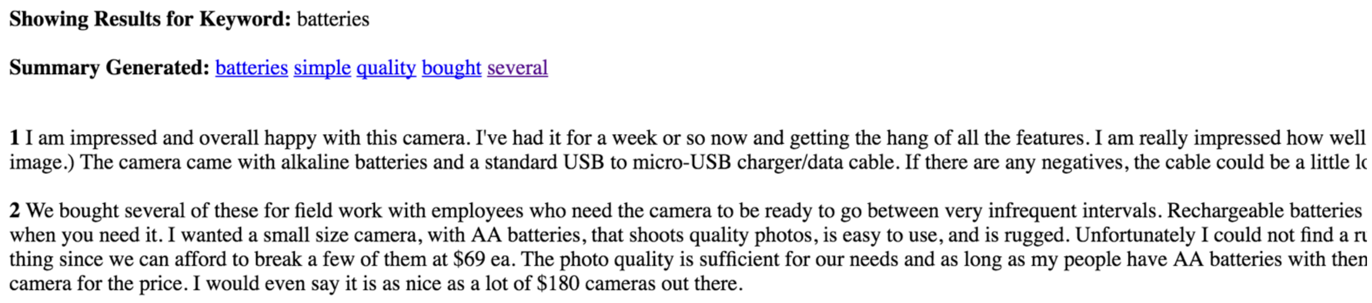
2 Kodak PIXPRO FZ43 Digital Camera Black with 16.15 Megapixels and 4x Optical Zoom
Kodak PIXPRO FZ43 Digital Camera:
- 16.15 megapixel resolution with high-quality still images and video recordings
  Featuring Auto, Standard and Panorama Effects, Scene Selector, 720p HD Movie Recording, Face Detection Technology, Optical Image Stabilization
- 4.9-19.6mm zoom lens with 4x optical zoom/6x digital zoom
  Equivalent to a 27-108mm lens on an analog camera
- 2.7" TFT LCD screen with image editing features
  Featuring an easy-to-read menu system
- Built-in flash with Auto, Red-Eye Reduction, Forced Flash, No Flash, Slow Synchro Flash, Red-Eye Reduction and Slow Synchro

Additional Features:
- 8MB built-in memory with 1 SD/SDHC memory card slot (memory card not included)
- Requires 2 x AA NiMH batteries for up to 120 shots
- ISO Modes: Auto, 1600, 400, 200, 100, 80
- File Formats Supported: Exif 2.3 (JPEG), MOV (Motion JPEG)
- 3.66" x 2.37" x 1.04"; 4.1 oz

Figure 5.3: Search results for the search query “zoom camera”.

Similarly, in the case of “zoom camera” as the search query, there are several key phrases that *PathFinder* managed to detect from the descriptions. For example, users can quickly look at “Aperture”, “Focus”, “Lens”, “49mm”, “Camera”, “digital”, “image”, and others, each of which are essential terms when one researches about buying a camera. Since these terms are hyperlinked, on underlined in blue, the users’ eyes would notice such terms and they could click on these terms to read more about them in the reviews. Even if one does not want to read through the reviews, one can read about these features in the description itself. As an example of this, one can be informed of the “Aperture” being in the range of “f/4.5 to 6.3” of the first listing. Thus, users would not be required to read through the whole description to get a sense of each product.

5.3 REVIEWS SEARCH AND SUMMARY WORDS GENERATION



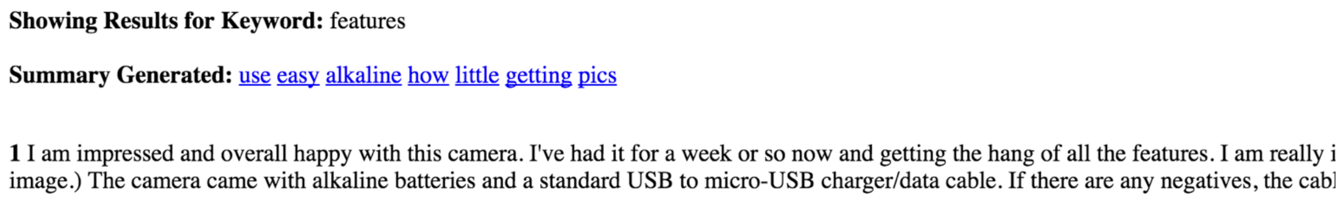
Showing Results for Keyword: batteries

Summary Generated: batteries simple quality bought several

1 I am impressed and overall happy with this camera. I've had it for a week or so now and getting the hang of all the features. I am really impressed how well
image.) The camera came with alkaline batteries and a standard USB to micro-USB charger/data cable. If there are any negatives, the cable could be a little lo

2 We bought several of these for field work with employees who need the camera to be ready to go between very infrequent intervals. Rechargeable batteries
when you need it. I wanted a small size camera, with AA batteries, that shoots quality photos, is easy to use, and is rugged. Unfortunately I could not find a ru
thing since we can afford to break a few of them at $69 ea. The photo quality is sufficient for our needs and as long as my people have AA batteries with then
camera for the price. I would even say it is as nice as a lot of $180 cameras out there.

Figure 5.4: Filtered reviews when the user searches for "batteries" within the reviews from the product listings page of *PathFinder*. Other reviews have been cropped to enlarge the image.



Showing Results for Keyword: features

Summary Generated: use easy alkaline how little getting pics

1 I am impressed and overall happy with this camera. I've had it for a week or so now and getting the hang of all the features. I am really i
image.) The camera came with alkaline batteries and a standard USB to micro-USB charger/data cable. If there are any negatives, the cabl

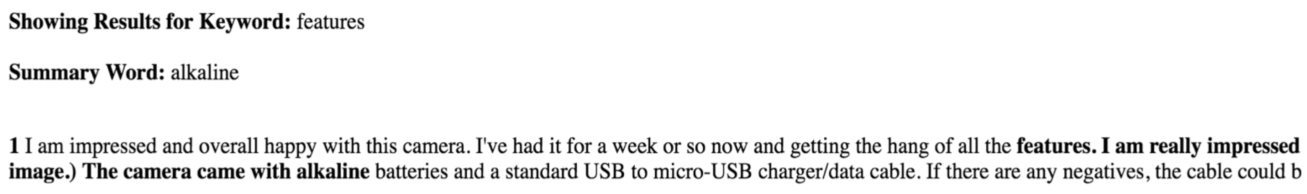
Figure 5.5: Filtered reviews when the user searches for "features" within the reviews from the product listings page of *PathFinder*.

Users can search for key words or phrases within the reviews from the product listings page. They could also simply click on the key phrases that *PathFinder* detects and *PathFinder* would directly search for those key words or phrases within the reviews. In this case, the user sees a filtered list of reviews, those that contain that keyword "batteries" and "features" in them. After landing on this page, users can read reviews and opinions of different people before they make their decision about the product. The word "batteries" is present in both reviews, in the second line of both reviews 1 and 2 in Figure 5.4 and the word "features" is present in the first line of review 1 in Figure 5.5. However, users would still need to manually glance through the reviews to find these words and then read about them.

As discussed in Chapter 4 to solve this issue, *PathFinder* also generates a list of related words or a summary. The summary would help the users get a better idea of the terms and ideas people typically use around the word they searched for. In this case, a user would get a broader view of what other users are writing about "batteries" and "features". Taking a look at the summary words generated for "batteries", one could relate "simple" with the batteries being simple to replace or install, "quality" with high or low quality batteries, and "several" with the batteries lasting several hours or even several batteries being needed to be changed within a few days. In the summary words for "features", one can relate "easy" with the ease with which photos can be clicked, or easy to carry around, other relations, "little" might be used to refer to size of the camera and "pics" could be related to the ability of the camera to click good or bad pictures.

5.4 CONTEXT HIGHLIGHTING PAGE

Such hypothetical relations mean nothing till the they are read in the actual context. In order to view the original context in which such related words were used by other reviewers, users can click on a particular related word and *PathFinder* would highlight or bold the context of use.



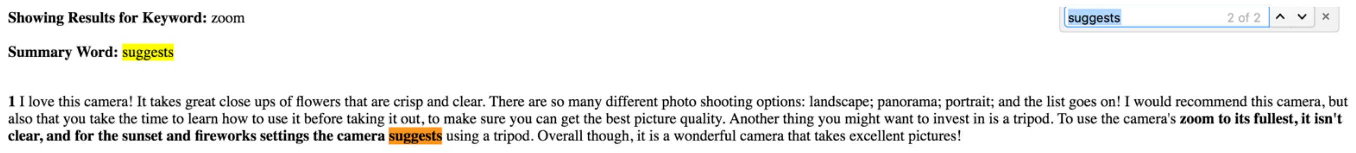
Showing Results for Keyword: features

Summary Word: alkaline

1 I am impressed and overall happy with this camera. I've had it for a week or so now and getting the hang of all the **features. I am really impressed image.) The camera came with alkaline** batteries and a standard USB to micro-USB charger/data cable. If there are any negatives, the cable could b

Figure 5.6: Context for "features" and "alkaline" highlighted in bold.

Before analyzing this context highlight, we will also make a review search with search query "zoom" and then click on a related word generated called "suggest".


Showing Results for Keyword: zoom

suggests 2 of 2

Summary Word: suggests

1 I love this camera! It takes great close ups of flowers that are crisp and clear. There are so many different photo shooting options: landscape; panorama; portrait; and the list goes on! I would recommend this camera, but also that you take the time to learn how to use it before taking it out, to make sure you can get the best picture quality. Another thing you might want to invest in is a tripod. To use the camera's **zoom to its fullest, it isn't clear, and for the sunset and fireworks settings the camera suggests** using a tripod. Overall though, it is a wonderful camera that takes excellent pictures!

Figure 5.7: Context for "zoom" and "suggest" highlighted in bold.

In Figure 5.6, the intention of the user is to find the context in which "alkaline" is used with "features". Since *PathFinder* highlighted the text in between these two words, the user can easily figure out that one of the features of the camera is that it comes with alkaline batteries. Even though this is not exactly a feature of a camera, it still gives one a better understanding of the overall product package that the user will buy. Similarly, in Figure 5.7, the user wants to understand the context of "suggest" when used with "zoom". Reading the bold text, it is easy to figure out that to optimize the zoom of the camera, it is better to use a tripod.

Such details might be missed if not explicitly displayed to the user. Apart from these insights getting missed, it is also tedious to manually read through all the reviews and see what is important for a user and what is not. Also, a particular user might not be interested in reading about everything that other reviewers have to say. Thus, the description key hyperlinking along with summary words generation allow users to select what specific information they want to learn more about and put lesser effort in gathering the relevant information.

CHAPTER 6: PATHFINDER TESTING RESULTS

PathFinder is a tool that connects users to the products they want to buy using intelligent data mining. However, to test the effectiveness of the system as a whole, there is no set method as each user would have different needs and varied experiences with the system. We took the typical searches that users make on online shopping forums [10] and performed the same searches on *PathFinder*.

6.1 FEEDBACK QUESTIONS

As part of the testing process, we did 2 parallel searches – one on *PathFinder* and the other on walmart.com. There were a total of 10 queries [10] used in the testing phase i.e. “headphones”, “PC”, “phone”, “kindle”, “TV”, “toys”, “Christmas”, “wearables”, “gaming”, and “movies”. Before performing these searches, we also kept a list of questions that one would typically expect while working with an online shopping forum. These questions included the following:

1. What we were specifically looking for while buying products
2. Whether we were able to reach specific information quicker than searching through walmart.com
3. Whether we were able to find some additional information that they would not have found otherwise.

After searching and exploring the 2 platforms, we report our experiences which helped us in understanding how well the system is working in terms of our initial hypothesis and how it could be improved further in future.

6.2 TESTING RESULTS

It is known that at the time of researching about products to buy, some users have features that they most care about while others are simply looking to explore a range of products. In this test, for 60% of the searches we made, we had no pre-conceived goal in mind to decide which products to buy. In the next step, we checked if our decision making time improved over the buying process of wamart.com. In around 70% of the searches we felt this specifically due to the key phrase highlighting within the descriptions and the summary generation with the filtered reviews. In all the searches we performed, we needed to do minimal reading as the hyperlinks proved to be easy to navigate pointers which at the end of the day, helped us in reaching our decision quicker. In about 30% of the searches, we also thought that we got some additional information about the product with the help of the summary generation feature. Even though the reviews were filtered, it is still tedious to read through all of them. Thus, the summary generation feature guided us through the different contexts within reviews and made it easy to read through important parts of the reviews.

There are also some vital nuances that we felt would help make *PathFinder* more useful and easier to use. One improvement would be to implement product name autofill feature in the search bar with a dropdown. It is simple to implement this feature provided walmart.com provides a list of all the products they have indexed. After entering the search query, we felt that *PathFinder* took too long to process the results and present them. A part of this could be attributed to the HTTP request that is sent to the walmart.com API which takes the most amount of time in the loading process. Other improvements to the search indexing and retrieval algorithms have been suggested in Chapter 7. Having images and prices of the products along with the description would also be a great addition. This will definitely make *PathFinder* more visually appealing and more beneficial. Also, we found that there were too many phrases in the description that were being highlighted. Apart from the number of phrases that were being highlighted, there were phrases like “at the” that were considered to be key by *PathFinder*. This is because “at the” is a combination of stop words in the English language and *PathFinder* simply checks for stop words and not for stop phrases. However, checking for both would allow for more in-depth analysis of the description and would highlight fewer number of key phrases as stop phrases are very prevalent in the descriptions. Even though the summary generated was useful, it could have been presented in a more concise manner. This is definitely doable by either representing them according to the frequency with which they appear in the reviews or by inserting words in between the summary words to bring the summary closer to natural language. Out of the total number of contexts that were highlighted, about 60% of them had between 4-6 words in them, about 30% of them had between 7-12 words and the remaining 10% had more than 12 words. After reading the highlighted contexts, we felt that the fewer the number of words highlighted, the more the conciseness. Thus, in a majority of cases, *PathFinder* was able to provide the contexts in a brief but meaningful highlight.

Such testing is critical for evolving and maintaining a system like *PathFinder* because the common interests of users change over time and it is imperative to adjust the system based on that. Since *PathFinder* is a very application oriented system, it is beneficial to perform subjective testing as that is what would bring out the pitfalls and highlight possible improvements that could be made to the system.

CHAPTER 7: RELATED WORK AND FUTURE WORK

As shown in in Chapter 1, *PathFinder* has been built with several components on top of each other. It starts off with the Walmart Search and Reviews API [8] which is followed by *PathFinder*'s Indexer. The Indexer is used by Information Retrieval algorithms to extract and process relevant reviews and also perform summary generation. We will discuss parts of these components which have been inspired by works of other researchers and authors and how each component can be extended further as part of future work.

The data for *PathFinder* is retrieved from the Walmart database using their Search and Reviews Open API [8]. Based on the search query specified by the user, the Search API returns an 'item id' which is then fed into the Reviews API to retrieve the reviews of the product. amazon.com is another entity that could be used for gathering data. It hosts a plethora of products, description, and reviews however, they are not free for use. It is also possible to integrate the listing of these websites together and let the user decide the website from which he or she wants to buy the product after looking at the features offered by *PathFinder*.

After retrieving the data, the Indexer indexes the reviews and the products using inverted indices and forward indices [1]. This textbook [1] also discusses several scoring techniques like term-at-a-time scoring using forward indices and inverted indices and document-at-a-time scoring. A disadvantage of the document-at-a-time scoring that is discussed is that the size of the total scores gathered would be the size of the number of reviews that match at least one term. Taking a look at the term-at-a-time scoring using forward indices, it is evident that this is an efficient scoring mechanism. One would need to go over all the documents one by one in order to get the score for a particular term. This means that the algorithm would also touch documents where that particular term has not even occurred, resulting in a wastage of resources and processing time. Thus, *PathFinder* uses term-at-at-time scoring using Inverted Indices where terms are mapped to the documents they appear in and their positions in those documents. This approach is more efficient as one would only hit documents in which the terms occur. *PathFinder* uses Proximity Search [2] to retrieve documents in which the entered query terms are within a certain distance apart from each other in no specific order. The current approach works well for walmart.com. However, as the amount of data gets bigger, it would be imperative to modify the indexing approach. [1] discusses an Index Sharding method where several indices are built and kept apart in a distributed fashion on one node or on multiple across different machines. In order to perform term-at-a-time scoring, multiple threads would gather a particular term's mapped value from different shards and the final step would be to combine the gathered values using Map Reduce. Thus, even though the single inverted index gets partitioned, the final output remains the same. Similar review search has been implemented by Yelp [11] however, that search engine simply returns the reviews that contain the occurrence of the search terms without any further annotations.

As part of future work, *PathFinder* could be deployed publicly for testing purposes and the feedback of users could be gathered to make several improvements in the system. Based on overall metrics like CTR or Click Through Rate and session level metrics like time spent on a particular section of the results page, one can estimate the goodness or effectiveness of the search result ordering. Such metrics can be combined as features to evaluate MAP [3]. Other ideas for improvement would be to use feedback mechanisms like Rocchio feedback [1] which aim at increasing the search engine's recall. Summary Generation of *PathFinder* is harder to evaluate because there are no ground truth labels which specify what the correct summary is. It is possible to use human judges who would pick a list of top words that are an effective summary and then use that list to evaluate the accuracy of the list generated by *PathFinder*. Apart from this, it is also important to consider the time-accuracy [1] tradeoff which balances the processing time and accuracy.

There are other stretch ideas about *PathFinder* that could make it more effective and holistic. [4] discusses approaches which can be used to judge how user-item interactions change over a given period of time. In a space like Online Shopping, users' affinity to certain products may keep changing with time, or due to some developing story around the product. Such information can be useful for shoppers because a newly developing trend is essential and might be overshadowed by the mountains of information lying above it that point to some other trend. Generating image summaries [5] is a very useful way to extract more information from content posted by users. Day by day, the amount of visual data is increasing on the internet. A lot of users post their product pictures on such online shopping websites but some of them do not write descriptions for the images. As described in [5], it possible to generate a natural language summary for the image and use is as an input to the components of *PathFinder*, for example by appending it to the corresponding indices. Such ideas and improvements would ensure that *PathFinder* becomes more scalable and more intelligent than the current version.

CHAPTER 8: CONCLUSION

PathFinder is built with the aim of making it easier and faster for online shoppers to make a decision regarding the products they want to buy as compared to buying products. After studying the results of the subjective user study that was conducted to evaluate the system, it is indeed the case that a majority of the users are able to reach their product decision faster. This is mainly due to the intelligent retrieval and presentation of information in the same flow as that of walmart.com. Users get more information in a concise manner in *PathFinder* that helps them gather varied types of data about the products they are searching for. However, walmart.com is a very reputable and established source with an already large base of users. Thus, *PathFinder* is not and was never meant to be a replacement of online shopping places like Walmart and Amazon. As *PathFinder* evolves with time and further improvements as suggested in Chapter 7, it can be thought of as a one stop shop to explore a plethora of products indexed from each online shopping website and using *PathFinder* to eventually visit websites like walmart.com and amazon.com to complete the purchase. Thus, users would be able to search for all their favorite or new products at one place, gather information in a quick manner, and proceed to existing and widely used websites to enhance their online ordering experience.

BIBLIOGRAPHY

[1] Zhai, C., & Massung, S. (2016). *Text data management and analysis: a practical introduction to information retrieval and text mining*. New York: ACM Books.

[2] Proximity Operators. (2009, January 13). Retrieved from https://library.alliant.edu/screens/proximity.pdf

[3] Evaluation of Ranked Retrieval Results. (2009, April 07). Retrieved from https://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-ranked-retrieval-results-1.html

[4] Wang, Y., Nan, D., Trivedi, R., & Song, L. (2017). Coevolutionary Latent Feature Processes for Continuous-Time User-Item Interactions. [online] Available at: http://papers.nips.cc/paper/6480-coevolutionary-latent-feature-processes-for-continuous-time-user-item-interactions.pdf.

[5] Karpathy, A., & Fei-Fei, L. (2017). Deep Visual-Semantic Alignments for Generating Image Descriptions. [online] Available at: http://cs.stanford.edu/people/karpathy/cvpr2015.pdf.

[6] Morrison, K. (2017). 81% of Shoppers Conduct Online Research Before Buying [Infographic]. [online] Adweek.com. Available at: http://www.adweek.com/digital/81-shoppers-conduct-online-research-making-purchase-infographic.

[7] Wallace, T. (2017). Ecommerce Trends: 147 Stats Revealing How Modern Customers Shop in 2017. [online] www.bigcommerce.com. Available at: https://www.bigcommerce.com/blog/ecommerce-trends/.

[8] Walmart Open API - Search and Reviews. (2017). Walmart.

[9] Anon. (2017). Average number of search terms for online search queries in the United States as of July 2017. [online] www.statista.com. Available at: https://www.statista.com/statistics/269740/number-of-search-terms-in-internet-research-in-the-us/

[10] Kulach, K. (2017). How buyers conduct product searches on Amazon vs. Google: new insights for online sellers. [online] Available at: https://webinterpret.com/us/blog/ecommerce-product-searches-amazon-google/.

[11] Yelp Search Engine. Available at: www.yelp.com (2017). Yelp.