Toward a Fair Review-Management System

Theodoros Lappas[1,⋆] and Evimaria Terzi[2]

[1] UC Riverside
tlappas@cs.ucr.edu
[2] Boston University
evimaria@cs.bu.edu

Abstract. Item reviews are a valuable source of information for potential buyers, who are looking for information on a product's attributes before making a purchase decision. This search of information is often hindered by overwhelming numbers of available reviews, as well as low-quality and noisy content. While a significant amount of research has been devoted to filtering and organizing review corpora toward the benefit of the buyers, a crucial part of the reviewing process has been overlooked: *reviewer satisfaction*. As in every content-based system, the content-generators, in this case the reviewers, serve as the driving force. Therefore, keeping the reviewers satisfied and motivated to continue submitting high-quality content is essential. In this paper, we propose a system that helps potential buyers by focusing on high-quality and informative reviews, while keeping reviewers content and motivated.

1 Introduction

Item reviews are among the most prominent instances of opinionated text found on the Web. By authoring a review, a reviewer can share his experience and express opinions on the advantages and disadvantages of an item's features. The accumulated corpus of reviews can then serve as a valuable source of information for interested users and potential buyers. Due to their immense popularity and important role in the modern e-commerce model, reviews have been the focus of numerous interesting research problems. The challenges that emerge in a real review-hosting system can be grouped in the following two categories:

Volume and Redundancy: Users are often faced with an overwhelming volume of reviews. As of April 2011, `Amazon.com` hosts almost 21,000 reviews on the popular Kindle reading device. Clearly, it is impractical for a user to go through hundreds, or even thousands, of reviews in order to obtain the information she is looking for. Moreover, a large portion of the reviews are often redundant, expressing the same opinions on the same attributes. A number of approaches have been proposed to address these challenges, mainly focusing on summarization [6,21,12] and search-based methods [11].

Review Quality: As is the case in every system that hosts user-generated content, the quality of the submitted information is a primary issue. Several

⋆ This work was done while the author was visiting Boston University.

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 293–309, 2011.

notions of review quality have been proposed and evaluated in the relevant literature [12,11,14]. The quality of a review is based on the volume and validity of the information it conveys, as well as its readability and structural features.

While significant contributions have been made toward addressing these challenges, we identify two major parts of the review-management process that are still lacking: (a) review presentation and visibility and (b) reviewer motivation and utilization. Next, we discuss these two in more detail.

Review Presentation: In the context of a review-hosting website, the review presentation component determines the visibility of each review in terms of both time and placement on the site's interface. Since the attention span of the users is limited, decisions about which reviews to show can have a major impact on their browsing experience and overall satisfaction.

In major contemporary review portals, reviews are usually sorted based on the date of submission. This approach clearly does not address any of the challenges discussed above. Alternative sorting methods are based on user feedback. On sites like `Amazon.com` and `Yelp.com` users can rate the *helpfulness* of a review. These ratings can be then used for ranking purposes. Further, reviewers inherit the ratings of their authored reviews, providing a measure of expertise. This information can then be considered for future reviews. Recently, `Yelp.com` has introduced a method called YelpSort. The website claims that this measure evaluates reviews based on "recency, ratings and other review quality factors". Even though this is a step toward the right direction, no details on the actual computation are provided. While user ratings can be a useful source of information, they also suffer from significant drawbacks [12]. For example, reviews that already have many votes are more likely to attract even more, since they are granted increased visibility (e.g. ranked higher) by the system. In addition, older reviews have more time to accumulate helpfulness votes and tend to overwhelm new (and potentially superior) reviews.

Reviewer Motivation and Utilization: In existing review-hosting portals, reviewers have no interaction with the system. This leads to two significant shortcomings:

1. *Underutilization of expertise:* Writing a review is an expression of the reviewer's motivation to comment on a particular subset of the item's attributes. However, even though the same person may be perfectly capable to comment on more attributes, he may refrain from doing so due to negligence or lack of motivation. In other words, existing systems fail to get the most out of the reviewer and thus deprive potential customers from informative content.
2. *Lack of motivation:* Fully understanding the process that motivates certain users to review certain items is a non-trivial task. Possible causes include a genuine desire to help others, frustration or excitement due to the reviewed item, the desire to influence others and gain acknowledgment via positive ratings, or simply the need to express one's self. In any case, it is safe to

say that, when a user submits a review on a public website, he does so in anticipation that his opinions will be read by others. Hence, visibility is the primary means that the system can utilize in order to motivate reviewers. For this to work in practice, there needs to be a clear connection between the usefulness of the submitted review and the visibility it is granted. However, current systems either completely disregard the satisfaction of the reviewers, or try to indirectly motivate them with the promise of user-submitted helpfulness votes. We have already discussed the biases that ail such mechanisms earlier in the paper.

1.1 Contribution

In this paper, we present practical and theoretically grounded solutions for addressing the above shortcomings. More specifically, we present a framework that keeps the users informed and the reviewers motivated to contribute high-quality content. Our system guarantees that users are presented with a compact set of high-quality reviews that cover *all* the attributes of the item of their interest. We also guarantee reviewer satisfaction by proposing a mechanism which periodically *shuffles* the existing set of reviews, instead of statically ranking them with respect to certain criteria. The design of the shuffling mechanism is such that the chances of the review to be brought into the spotlight are proportional to its usefulness to the user. Finally, we present a mechanism for suggesting to reviewers how to extend their reviews in order to gain more visibility.

1.2 Roadmap

The rest of the paper is organized as follows: we review the related work in Section 2. After a brief introduction of our notation in Section 3, we present our methods for review shuffling and user motivation and utilization in Sections 4 and 5. In Section 6 we provide a set of experiments that demonstrate the practical utility of our approach. We conclude the paper in Section 7.

2 Related Work

Our work is the first to propose a complete review management system that balances the satisfaction of customers, as well as reviewers. Nonetheless, our methodology has ties to areas relevant to the domain of item reviews.

Review Quality: The problem of formalizing and evaluating the quality of a review has attracted considerable attention. A large volume of work has been devoted to evaluating the *helpfulness* of reviews [14,20], typically formulating the problem as one of classification or regression. Jindal and Liu [8] focus on the detection of spam (e.g. duplicate reviews). Liu and Cao [12], formulate the problem as a binary classification task, assigning a quality rating of "high" or "low" to reviews. In recent work, Lu et al. [13] discuss how mining information from social networks can be used toward the evaluation of review quality. A

different notion of quality deals with the readability of a review, as defined by as its structural characteristics. The Flesch Reading EASE [9] is indicative of this line of work. Although our framework includes a component for the evaluation of review quality, our ultimate goal is to build a system that balances user and reviewer satisfaction.

Attribute Extraction and Opinion Mining: Given a review corpus on an item, opinion mining [4,7,16,17], looks for the attributes of the item that are evaluated in each review, as well as the respective opinions expressed on each attribute. For our experiments, we implemented the technique proposed by Hu and Liu [7] for attribute extraction.

Review Management: The accumulation of overwhelming volumes of online reviews has created the need for methods to manage and present such data. A relevant field is that of opinion summarization [10,12,21], in which the review corpus is processed to produce a statistical summary with information on the distribution of positive and negative opinions on the attributes of the item. Other relevant approaches [11,22] propose methods for finding a compact and informative set of reviews. Finally, the Information Systems community has explored the problem of review placement and its effect on customers [18]. Contrary to our own approach, none of these methods take into consideration the motivation and satisfaction of the reviewers. We present a framework that helps customers deal with the large number of reviews, while keeping reviewers motivated to submit high-quality and informative content.

3 Notation

We use A to denote the set of attributes associated with the given item. We also use R to denote the set of all reviews that have been written for the item. We assume that $|A| = m$ and $|R| = n$. Every review $r \in R$ is represented as a subset of the item's attributes; that is, $r \subseteq A$. We assume that every reviewer writes a single review for each item, and therefore every review r uniquely identifies its author.

4 Spotlight Shuffling

In this section, we present our method for selecting the set of reviews to be shown to the users. We call the set of reviews shown at any point in time to the visitors of the host site, the *spotlight set*. If R is the set of all the reviews of an item, then the spotlight set S is a subset of the reviews from R (i.e., $S \subseteq R$). Our mechanism for review selection is based on creating different spotlight sets at different points in time. Therefore, the output of our method is a sequence of N spotlight sets, $\mathcal{S} = \{S_1, S_2, \ldots, S_N\}$. Ideally, we would like the following properties for the spotlight sequence.

- **Attribute coverage:** each spotlight set needs to cover *all* the attributes of the item, in order to provide a thorough presentation to the interested user.
- **Review Quality:** each spotlight set has to include only high-quality reviews, that are both informative and easy to read.
- **Fair spotlight share:** each eligible review needs to have a fair chance of inclusion in the spotlight sequence, according to the information it conveys.
- **Compactness:** every spotlight set of $\mathcal{S}$ needs to be compact, so that users can read through it in a reasonable amount of time.

In the remaining of this section, we identify concepts and methods that will allow us to produce spotlight sequences that satisfy the above requirements.

4.1 Attribute Coverage

We first define the notion of the *cover*, which is central to our analysis:

Definition 1. *Given the corpus of reviews R on an item with attributes A, a a subset of the reviews $R' \subseteq R$ is a* cover *of the item if every attribute of A is evaluated in at least one of the reviews in R'. That is, $\cup_{r \in R'} r = A$.*

In order to satisfy the attribute-coverage requirement we require every set in the spotlight sequence to be a cover. In fact, for the rest of the discussion, we will use the terms *spotlight set* and *cover* interchangeably.

4.2 Review Quality

The second requirement demands that all the spotlight sets in the sequence consist only of high-quality reviews. That is, if $q(r)$ is a numeric measure of the quality of the review r, then we say that a spotlight set S is *high-quality* if for every $r \in S$ $q(r) > \tau$. Here, τ is a minimum threshold imposed on the quality of the reviews in $\mathcal{S}$. The next step is to find an intuitive definition that accurately captures the quality of a review. Our methodology is compatible with any approach for review-quality evaluation. We refer the reader Section 2 for a thorough overview on related work. In our implementation, we adopt the *Flesch Readability Ease* (FRE) formula [9] . Formally, the FRE score of a given (English) review r is defined as:

$$FRE(r) = 206.835 - 1.015 \times \frac{words(r)}{sents(r)} - 84.6 \times \frac{syllables(r)}{words(r)},$$

where $words(r)$, $sents(r)$ and $syllables(r)$ denote the number of words, sentences and syllables in d, respectively. This is very popular formula, the weights of which have been derived by means of regression on training data. The formula yields numbers from 0 to 100, expressing the range from "very difficult" to "very easy", and is meant to be used for measuring the readability of texts addressed to adult language users. In our experiments, we discard reviews that score less than $\tau = 60$, a typically used threshold for FRE [9]. We choose FRE for its popularity and its use as a standard for readability by many organizations (e.g., by the U.S. Department of Defense).

4.3 Fair Spotlight Share

For a given item, one can pick a high-quality spotlight set S by simply appending to S high-quality reviews until all the attributes in A are covered. Of course, one can see that there are exponentially many distinct high-quality spotlight sets. For the rest of the discussion, we denote the complete collection of spotlight sets by $\mathcal{C}$. The question that we consider here is: "how can we *fairly* construct a spotlight sequence by picking spotlight sets from $\mathcal{C}$?".

In order to address this problem, we propose a shuffling scheme that constructs a spotlight sequence $\mathcal{S}$ in a fair way that ensures the satisfaction of both customers and reviewers: customers see a compact and informative set of reviews, while each reviewer receives a *fair* share of participation in the spotlight sequence. Clearly, the design of the shuffling method largely depends on the definition of "fair participation". In a fair shuffling scheme, valuable reviews should be rewarded. Intuitively, a review is valuable if it evaluates a large number of attributes or if it comments on significant attributes that are overlooked by the majority of the reviewers. As an example, consider an expert review that provides insightful opinions on attributes that are too hard for the average reviewer to evaluate (e.g. the advanced technical features of a laptop computer). Taking the above into consideration, we formalize the *spotlight privilege* of a given review r as follows:

Definition 2. *Let $\mathcal{C}$ be the collection of all covers of a particular item. Then, the spotlight privilege $p(r)$ of a given review r is equal to the number of spotlight sets in $\mathcal{C}$ that include r:*

$$p(r) = |\{S \in \mathcal{C} \mid r \in S\}|.$$

Conceptually, the more spotlight sets a review participates in, the higher its spotlight privilege. Thus, in the shuffling scheme, every time we need to extend the spotlight sequence, it is sufficient to choose a new spotlight set from $\mathcal{C}$, uniformly at random. If we repeat this sampling process for an appropriate number of times, the *expected* number of times a review r is included in the spotlight sequence will converge to $p(r)$.

The Spotlight Shuffling Algorithm: Given the above discussion, the next issue is how to sample uniformly at random from the collection of covers $\mathcal{C}$. An intuitive algorithm is the following: pick random subsets $R' \subseteq R$ and check whether R' is a spotlight set. If it is, then it is presented to the users and the algorithm proceeds in the same fashion to pick the spotlight set to show in the next timestamp. Although this algorithm is both natural and simple, its time complexity is exponential. The reason for this is that $\mathcal{C}$ can be very small compared to the 2^n possible subsets of R. Therefore, the algorithm needs to pick many subsets before actually finding a valid spotlight set.

An alternative is to explicitly construct spotlight sets from $\mathcal{C}$. Such an algorithm could initialize the spotlight sequence $\mathcal{S}$ with the spotlight set $S_1 = R$ (i.e. the full set of reviews). Spotset S_i is then created from spotset S_{i-1} by removing from S_i one review at a time, as long as the remaining set is still a spotlight

Algorithm 1. The `ImportanceSampling` algorithm.

Input: Set of minimal covers $\mathcal{M}$, number of desired samples N.
Output: Spotlight sequence $\mathcal{S}$ of length N .

$\mathcal{S} \leftarrow \emptyset$
while $|\mathcal{S}| < N$ **do**
  pick M_i from $\mathcal{M}$ with probability $\frac{2^{n-|M_i|}}{\sum_{M \in \mathcal{M}} 2^{n-|M|}}$
  Generate a superset $S \in \mathcal{C}_i$ of M_i by appending each review $r \in R \setminus M_i$ with probability 1/2.
  Let i^* be the *Canonical Representative* of S.
  if $i^* = i$ **then** $\mathcal{S} = \mathcal{S} \cup \{S\}$
return $\mathcal{S}$

set. Although this algorithm guarantees the construction of many spotsets in polynomial time, it still does not solve our problem. This is because it does not guarantee uniform sampling from $\mathcal{C}$. In fact, by using this algorithm, several elements from $\mathcal{C}$ might never be discovered.

In order to address the problems discussed above, we employ *importance sampling* [15] in order to uniformly sample solutions from $\mathcal{C}$. Next, we discuss how the technique can be applied to our setting.

The `ImportanceSampling` Algorithm: The importance sampling technique requires as input the set of all *minimal* spotlight sets. Recall that a spotlight set S is minimal if it is not a proper super-set of any other possible spotlight set. In other words, every review in a minimal spotlight set S is the only review in S that covers at least one of the item's attributes. As before, we use $\mathcal{C}$ to denote the collection of all possible covers of the attribute-set A. We also use $\mathcal{M}$ to denote the set of all *minimal* spotlight sets. It is easy to see that every cover in $\mathcal{C}$ is a superset of at least one of the covers in $\mathcal{M}$. Even though $|\mathcal{M}| << |\mathcal{C}|$, computing $\mathcal{M}$ can still be a non-trivial task. For the sake of our analysis, we assume that $\mathcal{M}$ is available. Towards the end of this section, we show how to effectively sample from $\mathcal{M}$.

Let $M_i \in \mathcal{M}$ be a minimal cover and $\mathcal{C}_i$ be the collection of all supersets of M_i. In order for the technique to be applicable, the following three conditions need to be met [15]

1. We can can compute $|\mathcal{C}_i|$ in polynomial time.
2. We can sample uniformly at random from $\mathcal{C}_i$.
3. Given any subset of reviews $R' \subseteq R$, we can verify in polynomial time if $R' \in \mathcal{C}_i$.

In our case, all 3 conditions are satisfied: for (1) we have $|\mathcal{C}_i| = 2^{n-|M_i|}$. For (2), we can sample uniformly from the sets in $\mathcal{C}_i$ by appending each review in $R \setminus M_i$ with probability 1/2. For (3), given any $R' \subseteq R$, we can verify if R' is included in $\mathcal{C}_i$ by simply checking if R' is a superset of M_i.

The importance sampling technique is based on considering the multi-set $\mathcal{U} = \mathcal{C}_1 \uplus \ldots \uplus \mathcal{C}_{|\mathcal{M}|}$, where the elements of $\mathcal{U}$ are pairs of the form (C, i), corresponding

to a cover $C \in \mathcal{C}_i$. In other words, for every cover $C \in \mathcal{C}$, $\mathcal{U}$ contains as many copies of C as there are $\mathcal{C}_i$'s for which $C \in \mathcal{C}_i$. The multi-set $\mathcal{U}$ is then divided into equivalence classes, where each class contains all pairs (C, i) that correspond to the same cover $C \in \mathcal{C}$. A *single pair* (C, i) is defined to be the *canonical representation* of each class. The intuition is that, instead of sampling from the space of size 2^n that contains all possible subsets of reviews, we sample only from $\mathcal{U}$. This guarantees that the sampled subsets are indeed review covers. The algorithm is polynomial to the number of minimal reviews in $\mathcal{M}$ [15].

Selecting a Seed of Minimal Covers: So far, we have assumed that the set of minimal covers $\mathcal{M}$ is available. However, enumerating all the minimal covers[1] is computationally expensive. In fact, using any of the existing algorithms [2,3,5] we were unable to generate all minimal covers, even for very small datasets. Therefore, we propose to execute `ImportanceSampling` with a smaller *seed set*. We abuse notation and denote this set by $\mathcal{M}$. We generate elements of $\mathcal{M}$ by starting with set R and randomly removing elements until we reach a minimal solution. We repeat this process until we generate a seed of the desired size. Our intuition is that even a small seed is adequate to capture a subset for most of the solutions in $\mathcal{U}$. As we show in our experiments, $|\mathcal{M}| = O(n)$ is sufficient.

Covering Opinions: The standard definition of the spotlight set requires the covering of all the attributes of an item. An interesting alternative is to cover *opinions*. Given a review r, we first apply a method to extract the expressed opinions, where an opinion is defined as a mapping of an attribute to a positive or negative polarity. Then, a review with a positive opinion on attribute a_1 and a negative opinion on attribute a_2 would be represented by $\{a_1^+, a_2^-\}$, instead of $\{a_1, a_2\}$. It is important to note that our methodology is entirely compatible with this formulation (see also the example in Section 6.2).

4.4 Compactness

Compact spotlight sets that cover all the attributes of the item in a small number of reviews are preferred by the users, since they require much less effort to process. In order to give precedence to covers with a small number of reviews, we modify `ImportanceSampling` to give higher preference to small covers. For this, we associate with every cover S a weight $w(S)$ that only depends on the number of reviews in S. In our implementation, we use $w(S) = e^{-\lambda x_s}$, where $\lambda > 0$ and x_s is the size of S. In order to sample correctly from this weighted sample space, we need to modify the `ImportanceSampling` algorithm as follows: first, the minimal elements from $\mathcal{M}$ are sampled with probability proportional to the sum of the weights of their supersets. Second, the selected minimal set $M \in \mathcal{M}$ is extended as follows: Using an exponential prior, we select x_s with probability proportional to $e^{-\lambda x_s}$. Then, we form S by expanding M using x_s random reviews from $R \setminus M$. The rest of the algorithm proceeds as shown in the

[1] This is also known as the *transversal hypergraph* problem.

pseudocode in Algorithm 1. We explore the effect of the exponential prior in the experimental evaluation (Section 6).

5 Reviewer Motivation and Utilization

In this section, we discuss our methodology for improving the utilization of reviewer expertise. Our goal is to motivate reviewers to submit more high-quality content. Given a review r, we want to recommend to the author of the review a set of attributes $Q \subseteq \{A \setminus r\}$, such that the new extended review $r' = r \cup Q$ has better spotlight privileges. That is, r' appears in more spotlight sets in the spotlight sequence than the original review r.

Our attribute-recommendation system is based on the observation that the spotlight privileges of a review r increase with the number of reviews in R that it *dominates*. We say that a review r_i *dominates* a review r_j if $r_j \subseteq r_i$. That is, every attribute from A that is covered by r_j is also covered by r_i. We use $D(r)$ to denote the number of reviews from R that r dominates. Then, given a review r, our goal is to recommend its extension with attributes $Q \subseteq \{A \setminus r\}$ such that $D(r \cup Q)$ is maximized. Our attribute-recommendation system can be activated after the review is submitted (and parsed to extract attributes) or before (by asking the user to state the attributes he intends to review).

Clearly, if any reviewer was able to provide high-quality comments for all the attributes of a product, then the above problem would have a trivial solution: simply set $Q = A \setminus r$ and make the extended review r' dominate all the reviews in R. However, not all reviewers have the potential to comment on all attributes. Instead, the typical reviewer has the background and experience to comment on a subset of the item's attributes. For example, some reviewers might be in a better position to comment on the hardware parts of a laptop, while others may be able to provide a more comprehensive review on the accompanying software. To a certain extent, the ability of a reviewer to comment on a set of attributes is also encoded on his original review r. That is, the attributes that r covers are indicative of the attributes that this reviewer can comment on. Formally, we use $\Pr(a \mid r)$ to denote the probability that the author of review r is able to provide high-quality comments on a certain attribute $a \in A$. For now, we assume that these probabilities are given. Toward the end of the section, we discuss how we can compute them from the available data. Given a set of attributes $Q \subseteq A$, and the individual probabilities $\Pr(a \mid r), \forall a \in Q$, we compute the probability that a reviewer that wrote r can effectively evaluate the attributes in Q using the following independence assumption:

$$\Pr(Q \mid r) = \prod_{a \in Q} \Pr(a \mid r).$$

Given the above, the attribute-recommendation problem can now be formalized as follows.

*Problem 1 (*Attribute Recommendation - AR). Given an item with attributes A and a review $r \subseteq A$, find the set of attributes $Q \subseteq \{A \setminus r\}$ to recommend to the author of r such that

$$\textsc{Ed}(r \cup Q) = \Pr(Q \mid r) D(r \cup Q) = \left(\prod_{a \in Q} \Pr(a \mid r) \right) D(r \cup Q). \quad (1)$$

is maximized.

We call the quantity $\textsc{Ed}(r \cup Q)$ the *expected dominance* of the extended review $r' = r \cup Q$ and we refer to Problem 1 as the Attribute Recommendation problem (or AR for short). Note that the trivial solution $r' = A \setminus r$ is no longer optimal for the AR problem. In fact, as Q becomes larger, the $\Pr(Q \mid r)$ decreases. Also, as Q becomes larger, the second part of the objective (i.e., $D(r \cup Q)$) increases. This is because more reviews are getting dominated by $r \cup Q$.

Solving the AR Problem: Although we do not know the particular complexity of the AR problem, we know that the version of the problem where $\Pr(a \mid r) = 1$ for every $a \in A$ and the goal is to pick k attributes to form set Q such that $D(r \cup Q)$ is maximized is an NP-Complete problem.[2]

In our experiments, we deploy the following `Greedy` heuristic for the AR problem. At first, the algorithm sets $Q = \emptyset$. At iteration t, the algorithm forms set Q^t, by extending set Q^{t-1} with the single attribute that maximizes $\textsc{Ed}(r \cup Q^t) - \textsc{Ed}(r \cup Q^{t-1})$. The process repeats until none of the remaining attributes benefits the objective function. A single iteration of this algorithm has running time $O(nm)$. In the worst case scenario, the `Greedy` algorithm can have at most n iterations. In practice, however, the number is much smaller than n.

Computing the Probabilities $\Pr(a \mid r)$ for $a \in A$: Next, we discuss how we can compute $\Pr(a \mid r)$, i.e., the probability that the author of review r is able to provide high-quality comments on attribute a of an item. This is a challenging problem on its own. While we propose and experimentally evaluate a way to address this, we do not claim that we have exhaustively explored this particular problem. Instead, our purpose is to advocate the concept of attribute-recommendation for improved expertise utilization. In fact, our overall methodology is compatible with *any* method that can effectively compute $\Pr(a \mid r)$.

Consider the following motivating example. In the domain of digital cameras, the attributes *camera lens, digital zoom* and *optical zoom* are very often discussed in the same review, due to their close connection and interdependence. Intuitively, a person with particular interest for the zoom capabilities of a camera, is much more likely to also focus (and comment) on the camera's lens. We capture this intuition by computing the probability $\Pr(a \mid r)$ as follows:

$$\Pr(a \mid r) = \max_{s \in \mathcal{P}(r)} \frac{\text{freq}(s \cup a)}{\text{freq}(s)}, \quad (2)$$

where $\mathcal{P}(r)$ is the power set of the attributes in r and $\text{freq}(s)$ the number of reviews in the corpus that cover all the attributes in a given set s. Observe that Equation (2) is the *confidence* measure, as defined in the context of mining

[2] The reduction is from the well-known Set Cover problem.

association rules [1]. Hence, we can apply well-known rule-mining techniques to pre-compute the probabilities and reduce the processing time.

One can also take into account the reviewer's inherent ability and expertise on the item's domain. Then, if the reviewer has authored a number of well-received reviews on a topic, the probability that he is able to expand his initial review is elevated accordingly. This can be easily incorporated into Equation (2), by multiplying the left-hand side by a prior that captures the relevant expertise of the given review's author. The computation of such a factor falls within the area of expertise mining, and is orthogonal to our work. In our experiments, we assume a uniform prior distribution for all reviewers.

6 Experiments

In this section, we experimentally evaluate the methods presented in this paper. All the components of our system assume that we know the attributes that are discussed in every available review. To extract these attributes, we use the method proposed by Hu and Liu [7], which also mines opinions. An additional pass was made over the produced attribute-lists to verify their validity and also to address synonymy issues (e.g. *bathroom=restroom=toilet*).

6.1 Datasets

We use four item-review datasets provided by Lappas and Gunopulos [11]. The **GPS** and **TVS** datasets include the complete review corpora from `Amazon.com` for 20 GPS systems and 20 TV sets, respectively. The **VEG** and **SFR** datasets include the complete review corpora from `Yelp.com` for 20 Las Vegas Hotels and 20 San Francisco restaurants, respectively. The average number of reviews per item for **GPS**, **TVS**, **VEG** and **SFR** was 203.5, 145, 266 and 968, respectively.

6.2 Qualitative Evidence

First, we present qualitative evidence that demonstrate the validity of the spotlight sets produced by `ImportanceSampling`. Figure 1 shows examples for two different items. For each item, we present the set of considered attributes, which are underlined in the reviews of the respective spotlight set. For lack of space, we consider a subset of the complete set of each item's attributes. We also anonymize the reviews for the sake of discretion. The first spotlight set corresponds to an item from the **TVS** dataset, and contain at least one evaluation for each considered attribute. The second spotlight set follows the variation we discussed at the end of Section 4. In this case, we want to include at least one positive and one negative opinion for each of the item's attributes. In both examples, the spotlight sets produced by our method successfully covers the attributes.

Item 1 (**TVS**), Attributes: { *picture, price, warranty, sound, design, menu*}:

"...Of all the LCD Tvs the ∗ ∗ ∗ overall seemed to have a brighter picture, has 120Hz, 2 year warranty, reasonably priced and..."

"...The ∗ ∗ ∗ delivers outstanding picture quality and sound..."

"...Intuitive menu, easy to plug and play with most any hook up and source... The design of the TV is stunning, beautiful work all around..."

Item 2 (**SFR**), Attributes: {*food, price, staff (service), restrooms (bathrooms)* }

"...The food is delicious, prices are fair, venue is nice, staff is friendly, restrooms are clean..."

"...BAD SERVICE, WORSE ATTITUDES, AND EXTREMELY HIGH PRICES ..."

"...The food was substandard, unfortunately..."

"...the only drawback were the bathrooms..."

Fig. 1. Examples of spotlight sets

6.3 Evaluation of `ImportanceSampling` on the Spotlight-Shuffling Task

Next, we compare our `ImportanceSampling` algorithm against three alternative baselines for spotlight set selection. The considered approaches are evaluated in terms of how well they can balance the compactness of the produced spotlight sets and the visibility they offer to the reviews in the corpus. **(1)** `GreedySampling` begins by selecting a review from the corpus uniformly at random. It then greedily appends reviews, picking the review that covers the most new attributes at every step, until all the attributes at covered. The random choice in the first step is required to introduce variety in the spotlight sequence. The greedy choice at every step is also diversified by randomly picking among all the reviews that maximize the number of new attributes. **(2)** `RandomSampling` populates the spotlight set by picking reviews from the corpus uniformly at random, until all the attributes are covered. **(3)** `HelpSampling` works in a similar manner, except that the probability of choosing a review is proportional to the number of (user-submitted) helpfulness votes it has accumulated.

First, we pick the item with the most reviews from each of the four datasets (the results for the remaining items were similar and are omitted for lack of space). We then use each approach to sample 1000 spotlight sets for each item. To account for compactness, we allow for a maximum of 10 reviews per spotlight set. This is also the number of reviews that can fit in a typical webpage `Amazon.com` and is thus in tune with the limited attention span of the average user. If an approach reaches the bound without covering all the attributes, it stops and returns the incomplete set of reviews. To account for this, we report for each approach the percentage of reported sets that were incomplete.

Table 1. Evaluation on the spotlight-set shuffling task

	0	[1-20)	[20-40)	[40-60)	[60-80)	80 ≤	Inc. %
			TVS				
`ImportanceSampling`	0.0	0.67	0.17	0.06	0.05	0.05	8%
`GreedySampling`	0.78	0.14	0.03	0.03	0.0	0.02	0%
`RandomSampling`	0.0	0.0	0.32	0.68	0.0	0.0	90%
`HelpSampling`	0.41	0.19	0.18	0.09	0.02	0.1	48%
			GPS				
`ImportanceSampling`	0.0	0.81	0.08	0.02	0.05	0.04	11%
`GreedySampling`	0.88	0.03	0.03	0.02	0.01	0.04	0%
`RandomSampling`	0.0	0.02	0.92	0.06	0.0	0.0	98%
`HelpSampling`	0.47	0.31	0.09	0.04	0.02	0.07	79%
			SFR				
`ImportanceSampling`	0.15	0.82	0.01	0.02	0.0	0.0	16%
`GreedySampling`	0.93	0.04	0.01	0.0	0.0	0.02	0%
`RandomSampling`	0.0	1.0	0.0	0.0	0.0	0.0	100%
`HelpSampling`	0.5	0.37	0.08	0.02	0.01	0.01	100%
			VEG				
`ImportanceSampling`	0.01	0.81	0.1	0.04	0.02	0.02	14%
`GreedySampling`	0.67	0.27	0.02	0.02	0.0	0.02	0%
`RandomSampling`	0.0	0.09	0.91	0.0	0.0	0.0	99%
`HelpSampling`	0.43	0.15	0.22	0.1	0.05	0.05	97%

The results are shown in Table 1. The second column shows the percentage of reviews that were not included in *any* spotlight set. Columns 2-6 show the number of reviews that were included a number of times that falls within the corresponding intervals (i.e. between 20 and 40 times for column 2). The intervals in Table 1 are chosen based on a step of 20. The prevalence of our method is not affected by this choice. The last column shows the percentage of the reported sets that were incomplete (i.e., they did not cover all the attributes).

The first observation is that `RandomSampling` and `HelpSampling` consistently report high percentages of incomplete spotlight sets, reaching up to 100%. This is due to the fact that they do no consider the complementarity among reviews. As a result, they can require an arbitrarily large number of reviews to be included in order for all the atttributes to be covered. On the other hand, `GreedySampling` reports no incomplete spotlight sets, since it ensures complete coverage with a minimal set of reviews. However, this approach fails to fairly distribute visibility privileges to the reviews. In fact, a consistently high percentage of the reviews were not included in any spotsets. The greedy nature of the algorithm forces it to focus on a small subset of the review population, while completely overlooking the majority. Finally, our `ImportanceSampling` method successfully balances a consistently low percentage of incomplete spotlight sets and a fair distribution of visibility; the percentages of completely neglected reviews (column 1) was

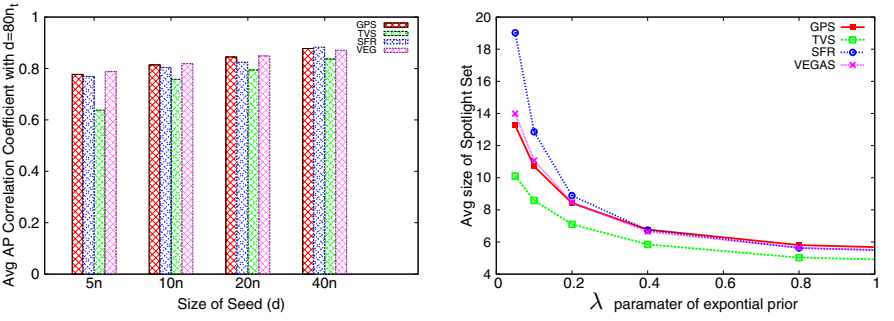

Fig. 2. Figure 2(a): Stability of `ImportanceSampling` with respect to seed size. Figure 2(b): y-axis: Size of spotlight set, x-axis: value of the parameter λ.

consistently low, going down to zero for two of the items. In general, even though some reviews were rightly sampled more often than others by our approach, almost all the reviews were given a fair chance to be in the spotlight.

6.4 The Effect of the Seed of Minimal Covers on `ImportanceSampling`

As discussed in Section 4, we compute only a subset of the complete set of minimal covers required by `ImportanceSampling`. We call such subset a *seed* and denote it by $\mathcal{M}$. Next, we evaluate the effect of $|\mathcal{M}|$ on the algorithm's results. We used the version of the algorithm given in Section 4.4, with $\lambda = 0.5$. The results for different values of λ were similar and are omitted for lack of space.

Given the review corpus R on an item, we use the method discussed in Section 4 to obtain a seed $\mathcal{M}$ of size d. We then use `ImportanceSampling` to sample spotlight sets from R, until convergence. The standard principles of `ImportanceSampling` are applied to determine convergence [15]. Let count(r) be the number of times a review r was included in a sampled spotlight set. We then rank the reviews by their respective counts. The process is repeated to obtain a new ranking for different values of $d = |\mathcal{M}|$. Figure 2(a) shows the AP correlation coefficient [19], computed by comparing the rankings obtained for $d \in \{n, 5n, 10n, 20n, 40n\}$, with the ranking given for $d = 80n$. AP takes values in $[-1, 1]$ and is a variant of Kendall τ used to compare two rankings by giving more importance to the top of the ranks. Figure 2(a) shows the AP values (y-axis) obtained for different values of d (x-axis). The results clearly show that a seed of size linear to the number of reviews is sufficient to approximate the counts. The AP values steadily increase with d, until, for $d = 40n$, they reach a near-perfect value of 0.9.

6.5 Compactness Evaluation

Here, we evaluate the spotlight sets produced by the `ImportanceSampling` with respect to their compactness. We sample spotlight sets using the pseudocode

given in Algorithm 1, using a seed $\mathcal{M}$ of size $40|R| = 40n$. We repeat the process for different values of $\lambda \in [0.05, 0.2, 0.4, 0.8, 1.6]$, the parameter of the exponential prior. Figure 2(b) shows the average cardinality of the sampled spotlight sets, taken over all items in each dataset (y-axis) as a function of the λ values (x-axis). As expected, increasing the value of λ leads to smaller spotlight sets. We observe that setting $\lambda \in [0.1, 0.2]$ consistently produces spotlight sets of size around 10. This is also the number of reviews that can fit in a typical webpage of `Amazon.com` and is thus in tune with the limited attention span of the average user. In any case, the λ parameter is an intuitive way to tune the size of the presented spotlight sets according to one's own specifications.

6.6 Evaluating the Attribute-Recommendation System

Next, we evaluate our attribute-recommendation system, presented in Section 5. We first pick the item with the most reviews from each of the four datasets. For each item, we randomly pick 100 reviews from its corpus and extend each review r with the set of attributes Q suggested by our method. We then record the ratio of the review's *expected dominance* after the extension, over the number of reviews that r initially dominates (i.e. $\frac{\text{ED}(r \cup Q)}{D(r)+1}$).

The results are shown in Table 2. The first column shows the average ratio over all 100 reviews picked per item. The second column shows the respective standard deviation. The third and fourth columns show the same quantities, calculated only over the subset of the reviews that benefited by the extension. The fifth column shows the maximum ratio observed for each item, and the sixth column shows the percentage of reviews that benefited from the extension. For smoothing purposes, all the average and standard-deviation computations were done after removing the top and bottom 5% of reviews, with respect to the ratio.

Table 2. Results of the Attribute-Recommendation System

	Avg	StDev	Avg*	StDev*	Max	Improved (%)
TVS	1.5	1	2.1	1.2	21	50%
GPS	1.7	1.5	2.5	1.9	8.8	58%
VEG	1.3	0.7	2.2	0.9	12.2	34%
SFR	1.8	1.2	2.6	1.1	20.1	56%

The results show that a significant portion of the reviews consistently benefited from the extension. The average improvement ratio was between 1.3 and 1.8 for all reviews, rising to 2.1 and 2.6 respectively when considering only the benefited reviews. Further, the observed maximum ratios show that an appropriate extension can have a tremendous effect on the dominance of a review. The effects of the recommendation system depend largely on the computation of the $Pr(a|r)$ probabilities (see Section 5); higher probabilities increase the expected dominance (see Equation (1)) and thus make extensions more beneficial.

7 Conclusion

In this paper, we presented a novel review-management system that considers the satisfaction of the customers, as well as the reviewers. We showed how informative and compact sets of reviews can be sampled from a corpus in way that takes into consideration the contribution and quality of each review. Further, we proposed an attribute-recommendation system that can help reviewers improve their reviews in order to gain visibility in the system. We concluded the paper with a thorough experimental evaluation on real review data. Overall, our framework considers both users and reviewers and try to optimize both the user satisfaction and utilization of reviewer expertise while motivating reviewers to submit high-quality content.

Acknowledgments. This research was supported in part by NSF award #1017529 and a gift from Microsoft. We also thank Aris Gionis for useful discussions.

References

1. Agrawal, R., Imieliński, T., Swami, A.: Mining association rules between sets of items in large databases. SIGMOD Rec. 22, 207–216 (1993)
2. Bailey, J., Manoukian, T., Ramamohanarao, K.: A fast algorithm for computing hypergraph transversals and its application in mining emerging patterns. In: ICDM, pp. 485–488 (2003)
3. Eiter, T., Gottlob, G.: Identifying the minimal transversals of a hypergraph and related problems. SIAM J. Comput. 24, 1278–1304 (1995)
4. Ghani, R., Probst, K., Liu, Y., Krema, M., Fano, A.: Text mining for product attribute extraction. SIGKDD Explorations Newsletter (2006)
5. Hébert, C., Bretto, A., Crémilleux, B.: A data mining formalization to improve hypergraph minimal transversal computation. Fundam. Inf. 80, 415–433 (2007)
6. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: SIGKDD (2004)
7. Hu, M., Liu, B.: Mining opinion features in customer reviews. In: AAAI (2004)
8. Jindal, N., Liu, B.: Opinion spam and analysis. In: WSDM 2008 (2008)
9. Kincaid, J., Fishburne, R., Rogers, R., Chissom, B.: Derivation of new readability formulas for navy enlisted personnel. In: Research Branch Report 8-75. Naval Technical Training, Millington (1975)
10. Ku, L.-W., Liang, Y.-T., Chen, H.-H.: Opinion extraction, summarization and tracking in news and blog corpora. In: AAAI-CAAW (2006)
11. Lappas, T., Gunopulos, D.: Efficient confident search in large review corpora. In: Balcázar, J.L., Bonchi, F., Gionis, A., Sebag, M. (eds.) ECML PKDD 2010. LNCS, vol. 6322, pp. 195–210. Springer, Heidelberg (2010)
12. Liu, J., Cao, Y., Lin, C.-Y., Huang, Y., Zhou, M.: Low-quality product review detection in opinion summarization. In: EMNLP-CoNLL (2007)
13. Lu, Y., Tsaparas, P., Ntoulas, A., Polanyi, L.: Exploiting social context for review quality prediction. In: WWW 2010 (2010)
14. Min Kim, S., Pantel, P., Chklovski, T., Pennacchiotti, M.: Automatically assessing review helpfulness. In: EMNLP 2006 (2006)

15. Motwani, R., Raghavan, P.: Randomized Algorithms. Cambridge University Press, Cambridge (1995)
16. Popescu, A.-M., Etzioni, O.: Extracting product features and opinions from reviews. In: HLT 2005 (2005)
17. Riloff, E., Patwardhan, S., Wiebe, J.: Feature subsumption for opinion analysis. In: EMNLP (2006)
18. Li, M.X., Tan, C.H., Wei, K.K., Wang, K.L.: Where to place product review? an information search process perspective. In: ICIS (2010)
19. Yilmaz, E., Aslam, J.A., Robertson, S.: A new rank correlation coefficient for information retrieval. In: SIGIR, pp. 587–594 (2008)
20. Zhang, Z., Varadarajan, B.: Utility scoring of product reviews. In: CIKM (2006)
21. Zhuang, L., Jing, F., Zhu, X., Zhang, L.: Movie review mining and summarization. In: CIKM (2006)
22. Tsaparas, P., Ntoulas, A., Terzi.: Constructing a comprehensive set of reviews. In: SIGKDD (2011)