Selecting a Characteristic Set of Reviews

Theodoros Lappas
Boston University
tlappas@cs.bu.edu

Mark Crovella
Boston University
crovella@cs.bu.edu

Evimaria Terzi
Boston University
evimaria@cs.bu.edu

ABSTRACT

Online reviews provide consumers with valuable information that guides their decisions on a variety of fronts: from entertainment and shopping to medical services. Although the proliferation of online reviews gives insights about different aspects of a product, it can also prove a serious drawback: consumers cannot and will not read thousands of reviews before making a purchase decision. This need to extract useful information from large review corpora has spawned considerable prior work, but so far all have drawbacks. Review *summarization* (generating statistical descriptions of review sets) sacrifices the immediacy and narrative structure of reviews. Likewise, review *selection* (identifying a subset of 'helpful' or 'important' reviews) leads to redundant or non-representative summaries. In this paper, we fill the gap between existing review-summarization and review-selection methods by selecting a small subset of reviews that together preserve the statistical properties of the entire review corpus. We formalize this task as a combinatorial optimization problem and show that it is not only NP-hard, but also NP-hard to approximate. We also design practical and effective algorithms that prove to work well in practice. Our experiments with real-life review corpora on different types of products demonstrate the practical utility of our methods, and our user studies indicate that our methods provide a better summary than prior approaches.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Selection Process; H.2.8 [**Database Management**]: Database applications—*Data Mining*

1. INTRODUCTION

One of the most significant developments in consumer shopping in recent years has been the explosive proliferation of online reviews. This reflects the great utility and trust that individuals ascribe to first-person reviews of products and services, particularly when those reviews come from actual past customers. It also reflects the fact that reviews are often highly detailed: reviews routinely cover many different aspects of a single product, allowing the reader to compile a very thorough picture of the strengths and weaknesses of any given product. Because of the utility and impact of reviews, major e-commerce sites have adopted them as a standard feature (e.g., `Amazon.com`), and a number of popular websites have emerged whose main goal is simply hosting online reviews (e.g., `Yelp.com`).

The immense value of online reviews has led to a vast body of reviews for individual items. For example, Amazon has more than 32,000 reviews for its Kindle e-reader, and there are individual restaurants in Yelp that have more than 100,000 reviews. Having a large number of reviews on the same product is desirable, since a large review-corpus is more likely to provide insights about different aspects of the item at hand. In addition, a large number of reviews increases the reader's confidence in the opinions they express.

However, the vast body of reviews for individual items is also a serious drawback. Consumers can not and will not read thousands of reviews before making a purchase decision. When a body of reviews is massive, its sheer size can prevent readers from extracting its useful information. A number of researchers have therefore begun to look for methods to extract the useful information contained in large review corpora, into a form that users can absorb more easily, despite having limited time or limited screen space.

For example, a significant amount of work has been devoted to the development of *review-summarization* methods [5, 20, 14, 15, 18]. The goal of these approaches is to create a summary of a review corpus that is both compact and representative of the opinions it contains. These summaries are statistical in nature, recording the number of positive and negative opinions that are expressed on each of the item's features. However, the drawback of such summaries is that they lack the immediacy and narrative structure of reviews as written by real users. This sacrifices one of the primary benefits of online reviews: the ability to appreciate the first-person experience of a previous buyer.

In response to this drawback of statistical summaries, another set of methods seek to summarize a review corpus using a subset of 'important' reviews. These important reviews are ultimately the ones shown to the user. For example, there exists a considerable amount of work in the area of *review ranking* [4, 7, 9, 10, 11, 17, 19]. Those approaches first produce a score for each review. They then show the top-k highest-scoring reviews to the user. However, these

KDD'12, August 12–16, 2012, Beijing, China.
Copyright 2012 ACM 978-1-4503-1462-6 /12/08 ...$10.00.

methods also suffer from a serious drawback: they do not seek coverage over the range of features that are important to users, and can therefore be highly redundant. For example, all the top reviews of a new camera may be highly informative about its long-range zoom ability, but mention nothing about how easy it is to use or carry. In other words, by evaluating each review separately, these approaches fail to consider the *complementarity* among reviews.

Hence, a number of recent efforts has focused on the problem of *review selection based on feature coverage*. For example, Lappas and Gunopulos [8] select a subset of reviews that collectively comment upon of *all* the features of the product at hand. Similarly, Tsaparas *et al.* [16] select a set of reviews that collectively provide both the negative and the positive aspects of each commented feature. While these methods do diversify the reported set of reviews, they fail to accurately capture the *proportion* of opinions (both positive and negative) in the underlying collection. For example, the summary constructed by method in [16] will always contain at least one positive and at least one negative opinion on each feature, even if one kind of opinion is significantly more common in the reviews of the underlying collection.

In this paper, we fill the gap between existing methods for review summarization and selection, by providing a subset of reviews that collectively present a statistically accurate summary of the entire review collection. More specifically, we preserve the integrity, coherence, and immediacy of actual reviews by providing users with a subset of reviews written by real previous customers. At the same time, we also ensure that the selected set of reviews preserves the statistics of the underlying corpus with respect to the positive and negative opinions expressed for different features. On a high level, the problem that we address in this paper is the following: "Given a corpus of Reviews $\mathcal{R}$ on item, find a small subset $\mathcal{S} \subseteq \mathcal{R}$ of reviews that accurately capture the proportion of opinions of the item's features." To the best of our knowledge, we are the first to formally define and study this problem, which we call the Characteristic-Review Selection problem.

In our work, we provide a formalization and thorough analysis of the problem, and identify and overcome the challenges that arise in efficiently finding its solution. We present novel algorithmic techniques, and show that they are effective through a rigorous evaluation. We show that our methods can accurately represent large review corpora, using only a small subset of reviews (e.g., from 5 to 20 reviews). Finally, we confirm the utility of the reported sets through a user study in which actual users judge the relative value of our approaches compared to the state of the art.

In addition to our main contribution of formalizing and solving the Characteristic-Review Selection problem, we also address an important precursor problem. A statistical summary is only meaningful if the distribution of positive and negative opinions in the corpus is stable at the time the summary is formed. If the state is unstable or immature, then the reported set could also be misleading. Taking this into consideration, we include an analysis of real review corpora in our experiments, and explore how the allotment of opinions in such corpora changes through time. Our analysis provides valuable insight on the converge of opinions, and allows us to identify when a body of reviews is sufficiently stable so as to serve as the basis of an informative summary.

Roadmap: The rest of the paper is organized as follows. First, we review related work in Section 2. We formally define our problem in Section 3, and in Section 4 we describe our algorithms for solving it. In Section 5, we present a thorough experimental evaluation of our methods. We conclude the paper in Section 6.

2. RELATED WORK

The goal of our work is to select a characteristic subset of reviews from a given corpus, in order to provide users with a compact and representative source of information. Although – to the best of our knowledge – the exact problem formulation we are proposing has not been considered before, our work clearly has ties with existing work in the domains of review summarization and review selection.

Review Selection: A recent line of work has addressed the problem of selecting a subset of reviews from a given corpus. Lappas and Gunopulos [8] proposed the selection of a set that represents the *majority opinion* on each feature. The drawback of this approach is that it overlooks the minority opinion, regardless of how significant it is. In a follow-up work, Tsaparas *et al.* [16] adopted an alternative formulation: given an upper bound on the number of the selected reviews, their goal is to select the set of reviews that covers as many features as possible. In their work, a feature is considered covered if the reported set includes at least one positive and at least one negative opinion on it. The shortcoming of this formulation is that it disregards the ratio of the number of positive and negative opinions on a feature. Thus, the reported set can be misleading to the user, who cannot use this set to deduce the actual proportion of positive and negative opinions across product features. The set of characteristic reviews reported by our methods does not have such drawbacks, since our goal is to accurately emulate the opinion distribution in the underlying corpus.

Review Summarization: Our work has ties to the extensive research devoted to review summarization. The problem of summarization was essentially introduced by Hu and Liu [5], who proposed a method for opinion-extraction, and applied the mined knowledge to create a statistical summary that informs the user of the number of positive and negative opinions in the corpus. A similar approach to the problem was taken by Zhang *et al.* [20]. In that work, the authors focused on the review domain, enriching their summary with opinionated sentences from the corpus. Instead of reporting the number of positive and negative opinions, Meng and Wang [14] report the terms that are used to characterize each feature (e.g. "small" or "expensive"). Another extension is proposed by Shimada *et al.* [15], who enrich the produced summary with objective information on the item, mined from web sources (i.e. Wikipedia). Finally, Liu *et al.* [9] explore ways to filter out low-quality reviews, in order to prevent them from corrupting the produced summary.

While the proposed approaches use statistical findings and isolated snippets to represent the given corpus, our own formulation asks for the selection of a characteristic set of actual reviews. We consider reviews as cohesive linguistic constructs, which provide users with an intuitive and user-friendly representation of the corpus. Nonetheless, in the context of a review-hosting website, our selection paradigm can be easily complemented by a corresponding statistical summary, in order to provide users with the maximum amount of information.

Review Quality and Ranking A significant amount of work has been devoted to evaluating different aspects of review quality [1, 11, 17]. These methods assign a quality score to every review, which is then used for ranking the reviews. The most prolific line of work in this field has focused on finding ways to emulate the user-assigned helpfulness votes that are present in review-hosting websites [19, 10, 7, 19, 4]. The main shortcoming of these approaches is that they evaluate the quality of each review separately. Hence, they completely dismiss the potential of reviews to complement each other and cover as many of the item's features as possible. High-quality reviews can still be redundant, reiterating the same opinions on the same features, and thus conveying no additional information to the user. Our selection paradigm overcomes such shortcomings, by considering the complementarity among reviews. Nonetheless, our methodology is fully compatible with quality-evaluation techniques, which can be used to filter out substandard reviews.

3. PROBLEM DEFINITION

In this section, we describe the notational conventions that we will use throughout the paper. We also provide a formal definition of the CHARACTERISTIC-REVIEW SELECTION problem and discuss its computational complexity.

3.1 Preliminaries

We use $\mathcal{R}$ to denote the collection of n reviews of the item of interest. We also assume that every item is associated with a set of z features $\{f_1, \ldots, f_z\}$. One should think of the features as the characteristics of the item that reviewers comment upon. For example, the features of a restaurant may be the `food quality`, `ambiance`, `service`, and so forth. A typical review only comments on a subset of these features, express either positive or negative opinions. In fact, the same review may have a positive opinion about one feature and a negative opinion about another. Hence, in order to describe the opinions that appear in reviews, we form the set of binarized feature opinions, which we refer to simply as the *opinions* of the item. That is, for each feature f_i, there are two possible opinions: positive and negative. Hence, the opinion vector of a item has cardinality $m = 2z$; we denote it by $A = \{a_1, \ldots, a_m\}$.

Throughout the paper, we assume that features and opinions can be automatically extracted. There are a number of methods for this task. In our experiments, we use the method by Ding *et al.* [2]. Therefore, every review $R \in \mathcal{R}$ can be represented as a subset of opinions from A; we say that R *covers* all the opinions that appear in R.

Given a collection of reviews $\mathcal{S} \subseteq \mathcal{R}$, we use $\pi(\mathcal{S})$ to denote the m-dimensional vector of that represents the fraction of the reviews in $\mathcal{S}$ that cover each opinion. That is, $\pi(\mathcal{S}, i)$ denotes the fraction of reviews in $\mathcal{S}$ that cover opinion a_i. We refer to $\pi(\mathcal{S})$ as the *opinion vector* of $\mathcal{S}$.

In addition to the opinion vector, we also use the notion of a *target vector* τ. The target vector is also defined over the set of opinions in A and, thus, it is also an m-dimensional vector. We quantify the closeness between two vectors π and τ using the L_2^2 norm of their difference. We denote this by

$$D\left(\pi, \tau\right) := L_2^2\left(\pi - \tau\right) = \sum_{i=1}^{m} \left(\pi(i) - \tau(i)\right)^2.$$

Given a set of reviews $\mathcal{S}$ and a target vector τ we call the value of $D(\pi(\mathcal{S}), \tau)$ the *error* of the collection $\mathcal{S}$.

3.2 The CRS problem

Given a collection of reviews $\mathcal{R}$ about a item and a target vector τ, our high-level goal is to select a subset of reviews $\mathcal{S} \subseteq \mathcal{R}$ such that $D\left(\pi(\mathcal{S}), \tau\right)$ is small. At the same time, we also want the subset $\mathcal{S}$ to consist of a small number of reviews. The encoding of these two goals is incorporated in the following problem definition.

PROBLEM 1 (CHARACTERISTIC-REVIEW SELECTION (CRS)). *Given a collection of reviews $\mathcal{R}$, a target vector τ and an integer number k, find $\mathcal{S} \subseteq \mathcal{R}$ such that $|\mathcal{S}| \leq k$ and $D\left(\pi(\mathcal{S}), \tau\right)$ is minimized.*

Problem 1 would be trivial if all the reviews contained at most one opinion. Without such unrealistic restrictions, however, the problem is hard. Specifically, our analysis below demonstrates that the CRS problem is not only NP-hard, but it is also NP-hard to approximate. In order to see this, lets consider the decision version of the problem which is defined as follows: Given a collection of reviews $\mathcal{R}$, and real number L and integer K does there exist a subset $\mathcal{S} \subseteq \mathcal{R}$ with $|\mathcal{S}| \leq k$ and $D(\pi(\mathcal{S}), \tau) \leq d$? We call this decision version of the problem DECISION-CRS. The following lemma shows that this problem is NP-complete.

LEMMA 1. *The* DECISION-CRS *problem is NP-complete.*

PROOF. We will reduce and instance of X3C (EXACT COVER BY 3 SETS) [3] to DECISION-CRS. An instance of X3C consists of a universe of n items U and a set of sets $\mathcal{C} = \{C_1, \ldots, C_M\}$ where $|C_i| = 3$ for every $i = 1, \ldots, M$ and $C_i \subseteq U$. Given integer K, the decision version of the problem asks whether there exists a subset $\mathcal{C}' \subseteq \mathcal{C}$ such that $|\mathcal{C}'| = K$, U and every element in U is covered by exactly one set in C'.

We transform an instance of X3C to an instance of DECISION-CRS by setting the set of opinions A to be the universe U. In this case, every set $C_i \in \mathcal{C}$ is represented by a review $R_i \in \mathcal{R}$; review R_i covers opinion $a_j \in A$ if set C_i contains the j-th element of U. To complete the instance of the DECISION-CRS problem, we set the target vector τ to consist of n dimensions all equal to $1/K$, and we also set $k = K$ and $d = 0$. Then, it is easy to see that the answer to the X3C decision problem is "yes" if and only if there exists a solution to the instance of the DECISION-CRS problem, which we created above. □

By the fact that we used X3C for our reduction, we can show that the problem is NP-hard even if all the reviews of the collection have exactly three opinions. Another immediate corollary of Lemma 1 is the following.

COROLLARY 1. *The* CRS *problems is NP-hard and NP-hard to approximate. That is, it is NP-hard to find a polynomial - time approximation algorithm for the* CRS *problem.*

PROOF. We will prove the hardness of approximation of the CRS problem by contradiction. Assume that there exists an α-approximation algorithm for the CRS problem. Then if $\mathcal{S}^*$ is the optimal solution to the problem and $\mathcal{S}^A$ is the solution output by this approximation algorithm, it will hold that $D(\pi(\mathcal{S}^A), \tau) \leq \alpha D(\pi(\mathcal{S}^*), \tau)$. If such an approximation algorithm exists, then this algorithm can be used to decide decision instances of the Crs problem for which $d = 0$. However, this contradicts the proof of Lemma 1, which indicates that these problems are also NP-hard. Thus, such an approximation algorithm does not exist. □

So far, we have discussed the Crs problem for an arbitrary target vector τ. For a collection of reviews $\mathcal{R}$, the most natural instantiation of the target vector is the *mean opinion vector*, denoted by $\pi(\mathcal{R})$. Figure 1 shows an example of a review corpus and the optimal characteristic subset that minimizes $D(\pi, \pi(\mathcal{R}))$. The corpus consists of 6 reviews $R_1, R_2, R_4, R_4, R_5, R_6$. These reviews comment on 3 distinct features f_1, f_2, f_3. A plus (resp. minus) denotes a positive (resp. negative) opinion on the feature. For example, review R_1 expressed positive opinions on features f_1 and f_2 and a negative opinion on feature f_3. In this example, the mean opinion vector corresponding to the included opinions $f_1^+, f_1^-, f_2^+, f_2^-, f_3^+, f_3^-$ is $\pi(\mathcal{R}) = (4/6, 2/6, 4/6, 0/6, 2/6, 4/6)$. We observe that the opinion vector of the selected characteristic set $\{R_4, R_5, R_6\}$ is identical: $(2/3, 1/3, 2/3, 0/3, 1/3, 2/3)$. Therefore, this set is optimal with respect to the objective function, when the target is the mean vector.

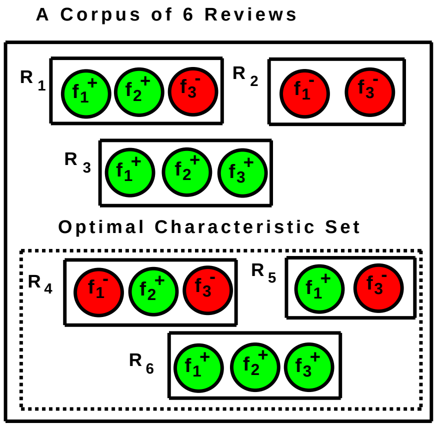


Figure 1: An example of a review corpus and the optimal characteristic set that minimizes $D(\pi, \pi(\mathcal{R}))$, where $\pi(\mathcal{R})$ is the mean opinion vector.

This instantiation of the target vector – and the Crs problem – corresponds to a very natural geometric problem: Given a set of points in m-dimensional space, select a subset of them of size k whose centroid approximates the centroid of the collection as well as possible. In the case of Crs the m-dimensional vectors are also binary, but the general problem where vectors lie in $\mathbb{R}^m$ is also interesting and – to the best of our knowledge – unexplored. Another natural target vector is the distribution of features in $\mathcal{R}$. However, one can observe that this is actually a simple rescaling of the mean opinion vector $\pi(\mathcal{R})$. Although in all our experiments we use $\pi(\mathcal{R})$ as our target vector, we note that alternative instantiations may also arise in practice. For example, when the goal is to provide personalized sets of characteristic reviews, then one may choose to approximate a specially-selected set of reviews (e.g. reviews written by reviewers with similar preferences as the interested user). Hence, in such cases the target vector would be tuned by the characteristics of the particular user.

4. ALGORITHMS

Although the Crs problem is NP-hard and NP-hard to approximate, our experiments with real datasets demonstrate that, in practice, there exist heuristics that work extremely well. We describe such heuristics below. With the exception of `Random` (described in Section 4.3), all our heuristics work for arbitrary target vectors τ.

4.1 The `Greedy` algorithm

The `Greedy` algorithm is an iterative algorithm for Crs. At every iteration t, the algorithm picks one review R to form subset $\mathcal{S}^t$. The review R is picked so that the distance $D\left(\pi\left(\mathcal{S}^{t-1} \cup \{R\}\right), \tau\right)$ is minimized. The pseudocode of the `Greedy` algorithm is shown in Algorithm 1.

Algorithm 1 The `Greedy` algorithm

```
   Input: $\mathcal{R}$, target vector $\tau$ and integer $k$
   Output: $\mathcal{S} \subseteq \mathcal{R}$, $|\mathcal{S}| = k$
1: $\mathcal{S} = \emptyset$
2: for $i = 1 \ldots k$ do
3:     $R = \arg\min_{R' \in \mathcal{R}} D(\tau, \pi(\mathcal{S} \cup \{R'\}))$
4:     $\mathcal{S} = \mathcal{S} \cup \{R\}$
5:     $\mathcal{R} = \mathcal{R} \setminus \{R\}$
6: end for
7: return $\mathcal{S}$
```

For a collection $\mathcal{R}$ of n reviews, the running time of `Greedy` is $O(knT_D)$, where T_D is the time required to compute the distance D required in line 3 of Algorithm 1. In our case, T_D requires $O(m)$ time, and therefore the running time of `Greedy` is $O(knm)$.

4.2 The `Integer-Regression` algorithm

The `Integer-Regression` algorithm finds the set of k characteristic reviews by first solving a continuous version of the Crs problem, and then transforming the obtained continuous solution into the closest discrete one. This is a well known strategy that has been shown to be effective for combinatorial optimization problems.

To describe the `Integer-Regression` algorithm, we use matrix notation. We start with the $m \times n$ binary matrix $\mathbf{R}$ in which entry $\mathbf{R}_{ij} = 1$ iff opinion a_i appears in review R_j. The Characteristic-Review Selection problem can then be restated as: find a 0-1 vector $\mathbf{s}$ such that $D(\mathbf{Rs}, \tau)$ is minimized, and $\mathbf{s}$ contains at most k 1s. The fact that $\min_{\mathbf{s}} D(\mathbf{Rs}, \tau)$ has the form of a linear regression is what inspires our approach.

In reality, $\mathbf{R}$ can contain duplicate columns; these are irrelevant for regression, so we remove them to form $\tilde{\mathbf{R}}$ having size $m \times n'$ and consisting of distinct columns. However, duplicate columns in $\mathbf{R}$ correspond to distinct reviews that may be useful in approximating τ; hence we keep track of the number of such duplicate columns by remembering for each column i of $\tilde{\mathbf{R}}$ its *multiplicity* c_i in $\mathbf{R}$.

The `Integer-Regression` algorithm works in two steps, which are repeated for all values from 1 to k:

For $\ell = 1$ to k

Step 1: Form a nonnegative real-valued vector $\mathbf{x}$ such that $D(\tau, \tilde{\mathbf{R}}\mathbf{x})$ is small, and the number of nonzero elements of $\mathbf{x}$ is not larger than ℓ.

Step 2: Form a nonnegative integer-valued vector $\tilde{\mathbf{s}}$ representing k reviews that together approximate $\mathbf{x}$ in distribution. That is, find $\tilde{\mathbf{s}}$ such that $\forall\, i$, $\tilde{\mathbf{s}}_i \leq c_i$, $||\tilde{\mathbf{s}}||_1 \leq k$, and $\| \frac{\tilde{\mathbf{s}}}{||\tilde{\mathbf{s}}||_1} - \frac{\mathbf{x}}{||\mathbf{x}||_1} \|_1$ is minimized.

Step 1 solves a regression problem, with the constraint that the solution has no more than ℓ positive coefficients. In statistics, regression with this sort of constraint is referred to as a *subset selection* problem; in signal processing it is called *sparse signal reconstruction.* We adopt an algorithm from signal processing for this step, namely Nonnegative Orthogonal Matching Pursuit (`NOMP`) [12, 13]. However, a variety of algorithms are known for this problem, and we expect that our general approach can make use of other algorithms as well. `NOMP` works iteratively; at each iteration t, it greedily selects the (not-yet selected) column from $\tilde{\mathbf{R}}$ that has the largest dot-product with the residual of the target vector τ. Then, the corresponding coefficients $\mathbf{x}^t$ are computed via least-squares, and a new residual is computed as $\tau - \tilde{\mathbf{R}}\mathbf{x}^t$.

Step 1 returns a nonnegative real-valued vector $\mathbf{x}$ such that $D(\tau, \tilde{\mathbf{R}}\mathbf{x})$ is minimized. Intuitively, the entries of $\mathbf{x}$ encode the "proportions" of each column of $\tilde{\mathbf{R}}$ that are needed in order to accurately approximate the target vector τ. The vector $\mathbf{x}$ is real-valued, and does not take into account the multiplicity of reviews c_i. Hence, $\mathbf{x}$ cannot be used directly as a solution to the Crs problem; instead, we use $\mathbf{x}$ to guide us to an actual set of reviews from the original collection $\mathcal{R}$. Accordingly, we transform the proportions of reviews encoded in $\mathbf{x}$ into integer values in $\tilde{\mathbf{s}}$ such that $||\tilde{\mathbf{s}}||_1 = k$ and for every element $\tilde{\mathbf{s}}_i$ of $\tilde{\mathbf{s}}$ it is guaranteed that $\tilde{\mathbf{s}}_i \leq c_i$ (i.e., each column in $\tilde{\mathbf{R}}$ cannot be used more times than its multiplicity in $\mathbf{R}$). This translation from proportions to integers is accomplished in **Step 2** of the algorithm.

The reason that we perform these steps for $\ell = 1, ..., k$ is that a trade-off exists between the sparsity of $\mathbf{x}$ and the hard limit k on the total number of reviews. When ℓ is small (say, 1) then it is difficult to find an $\mathbf{x}$ such that τ is well approximated by $\tilde{\mathbf{R}}\mathbf{x}$; after all, $\mathbf{x}$ can only have ℓ nonzero elements. On the other hand, when ℓ is large (say, k) then it is difficult to approximate the real-valued $\mathbf{x}$ with k integer-valued reviews. Intuitively, we expect the best value of ℓ to be somewhere in between 1 and k; in practice, we simply test all values of ℓ between 1 and k.

As we have already discussed, **Step 2** constructs the closest possible discrete approximation to the output of **Step 1**. The problem can expressed as follows: Let the number of nonzero elements in $\mathbf{x}$ be p. Given the nonnegative real vector $\mathbf{v} \in \mathbb{R}^p$ having $||\mathbf{v}||_1 = 1$, and a set of integers $c_1, c_2, ..., c_p$, output a nonnegative integer vector $\tilde{\mathbf{s}} \in \mathbb{R}^p$ such that $\forall i \; \tilde{s}_i \leq c_i$ and $|| \frac{\tilde{\mathbf{s}}}{||\tilde{\mathbf{s}}||_1} - \mathbf{v}||_1$ is minimized.

We use an efficient algorithm to solve the above problem in $O(Cp)$ time, where $C = \sum_{i=1}^{p} c_i$. The basic idea is 'conditioning' on the sum of $\tilde{\mathbf{s}}$ elements, i.e., $||\tilde{\mathbf{s}}||_1$. That is, we augment the problem with the requirement that $||\tilde{\mathbf{s}}||_1 = N$. We can solve this efficiently as follows. First, we observe that the $\tilde{\mathbf{s}}$ that minimizes $|| \frac{\tilde{\mathbf{s}}}{||\tilde{\mathbf{s}}||_1} - \mathbf{v}||_1$ also minimizes $||\tilde{\mathbf{s}} - N\mathbf{v}||_1$. Next, we set aside the constraints c_i and solve the unconstrained problem. For that, let $U = \sum_{i=1}^{p} \lceil N\mathbf{v}_i \rceil$ and $L = \sum_{i=1}^{p} \lfloor N\mathbf{v}_i \rfloor$. If $N \leq L$ (resp. $N \geq U$) then the solution is to set each $\tilde{\mathbf{s}}_i$ value below (resp. above) the corresponding value of $N\mathbf{v}_i$. If $L < N < U$, then compute $X = N - L$. Let the set $\mathcal{L}$ be the elements of $\mathbf{v}$ having the X largest values of $N\mathbf{v}_i - \lfloor N\mathbf{v}_i \rfloor$. For elements $i \in \mathcal{L}$, set $\tilde{\mathbf{s}}_i = \lceil N\mathbf{v}_i \rceil$. For the other elements, set $\tilde{\mathbf{s}}_i = \lfloor N\mathbf{v}_i \rfloor$. In order to solve the constraint version of the problem, we first fix $\tilde{\mathbf{s}}_i = c_i$ for all entries i such that $c_i < N\mathbf{v}_i$. Then, we solve the unconstrained version of the problem for the remaining elements. For any given N, this yields the optimal $\tilde{\mathbf{s}}$ in $O(p)$ time. Since the maximum allowable value of $||\tilde{\mathbf{s}}||_1$ given the constraints is C, we only need to run this algorithm for each $N \in 1, ..., C$ to find the optimal value of $\tilde{\mathbf{s}}$.

The computational complexity of **Step 1** of the `Integer-Regression` algorithm requires $O(k^3)$ in general. However, since `NOMP` works column-wise, incremental algorithms exist that lower the complexity of the regression to $O(k^2)$. Combined with the outer loop over k, this brings the overall complexity to $O(k^3)$. Thus, for small values of k that are most appropriate for review summaries, the running time of this step is negligible. As we have already discussed, the running time of **Step 2** is $O(Cp)$. As $C = O(n)$ and $p = O(k)$ the worst-case running time of this step is $O(nk)$. For the very small values of k that we consider in practice, the running time of this step is almost linear in n.

4.3 The `Random` algorithm

When the target vector τ is the mean opinion vector of the collection of reviews $\mathcal{R}$ (denoted by $\pi(\mathcal{R})$) we have the following observation.

Observation 1. *A random sample $\mathcal{S}$ of $\mathcal{R}$ of size k guarantees that* $D(\pi(\mathcal{R}), \pi(\mathcal{S})) = 0$ on expectation.

In other words, for a random sample of reviews $\mathcal{S}$, $\pi(\mathcal{S})$ is an unbiased estimator of $\pi(\mathcal{R})$. However, the algorithm that randomly samples k reviews from $\mathcal{R}$, which we call `Random`, performs poorly in practice. This is because its results have high variance, which also leads to higher error. In order to reduce the variance, we also propose an iterative version of `Random` – which we call `Iterative-Random`. The `Iterative-Random` selects a reasonably large number of random samples from $\mathcal{R}$ and reports the one with the best valuation of the objective function of the Crs problem.

5. EXPERIMENTS

Here, we present a thorough experimental evaluation of our methodology using review corpora from Amazon. In all our experiments, we use the mean opinion vector of the underlying collection of reviews as our target vector

We start our experimental evaluation in Section 5.2, where we present a study of the convergence of the opinion vectors of different corpora. We then evaluate the performance of the different algorithms with respect to the objective function of Crs in Section 5.3. We conclude our experimental study in Section 5.4, by presenting a user study that demonstrates the appeal of our summaries to real users.

5.1 Datasets

In our experiments, we use six different datasets of reviews from Amazon.com. The data was extracted from the collection of reviews introduced by Jindal and Liu [6]. This collection, crawled in June 2006, covers a wide range of different product from Amazon.com. It includes 5.8 million reviews, 2.14 million reviewers and 6.7 million products. For our own experiments we extract six datasets from six different domains: MP3 players, Digital Cameras, Coffee Makers, Printers, Books, and Vacuum Cleaners. We refer to these as `MP3`, `CAM`, `COF`, `PRINT`, `BOOK` and `VAC`. These datasets include reviews on 238, 233, 40, 112, 11447, and 62 different items, respectively. The mainstream nature and popularity of these domains ensures the availability of sizable review corpora. Other than that, our methods do not benefit from this selection. For the BOOK dataset, we purposefully avoided introducing any further constraints on the genre or type of the book. Our goal was to evaluate our algorithms in the context of a very large dataset of ambiguous items that typically encourage subjective reviews.

For our experiments, we extracted the opinions expressed in a review using the method proposed by Ding *et al.* [2]. Recall that an opinion is defined as the mapping of a feature (e.g. the lens of a digital camera) to a positive or negative polarity. While the method of Ding *et al.* worked well in practice, our framework is compatible with any method for opinion extraction.

5.2 Convergence of opinion vectors

The Crs problem asks for a set for a set of reviews that respect the opinion vector $\pi(\mathcal{R})$ of the underlying corpus. Such a set would only be informative and depictive of the reviewed item's true qualities, if the target opinion vector has converged to a stable state. On the other hand, if the arrival of new reviews causes big variations to the opinion vector, then the resulting set can be misleading.The goal of this section is to experimentally explore whether such variations appear in practice.

In order to study this for a corpus $\mathcal{R}$ on a particular item, we proceed as follows. First, we sort the reviews in ascending order of their submission date and set a checkpoint every 10 reviews. Using a checkpoint attached to a calendar unit (e.g. monthly or weekly) was not desirable since, for most corpora, the number of included reviews varies greatly across units. As we saw in our experiments, setting checkpoints every 10 reviews results in batches that include opinions on at least 90% of the item's features. Further, 10 is the number of reviews that is typically included in a single webpage of a review-hosting site (e.g. Amazon.com), since it is big enough to provided sufficient information, and small enough to respect the users' attention span.

Therefore, for a corpus with T checkpoints and $t \in \{1, \ldots, T\}$ we use $\mathcal{R}_t$ to denote the set of the $10t$ oldest reviews in the corpus. We measure the variation between the opinion vectors of $\mathcal{R}_{t-1}$ and $\mathcal{R}_t$ by measuring

$$D_t = D\left(\pi(\mathcal{R}_{t-1}), \pi(\mathcal{R}_t)\right).$$

Given a sequence of T such measurements and a threshold d, we say that an item is *converged* with respect to d, if there exists a t such that $D_t \leq d$. If, in addition to the above, we also observe that for every $t' \geq t$ $D'_t \leq d$, then we say that the item is *strongly converged*. Therefore, the opinion vector of strongly-converged items shows little or no variation after a particular checkpoint. Finally, we refer to items that are converged but not strongly converged as *relapsed* items.

For a given item with corpus $\mathcal{R}$, and T measurements $D_1, \ldots, D_T$, we study the following questions: (a) Do the D_t values decrease as t increases? (b) How long does it take for a corpus converge? (c) How often do items relapse? Next, we address these questions and present the corresponding experimental findings.

Figure 2(a) shows the average D_t values across all items as a function of the checkpoint t. The figure validates our intuition that as the size of a corpus increases, the D_t values decrease. For the plot, we use $T = 25$ for all items, since the D_t values for $t > 25$ are almost zero. The plot demonstrates that the opinion vectors of all corpora show little variation as time progresses. Starting from a maximum value around 0.3, the average D_t value drops fast with t and for $t = 25$, it becomes almost zero.

To gain additional insight, we study the percentage of items that have relapsed at least once. The results for different values of the threshold $d \in \{0.025, 0.05, 0.1, 0.2, 0.4\}$ are shown in in Figure 2(b). In order to properly interpret these results, we need to know the percentage of converged items per threshold; this percentage is shown in Figure 2(c). Figure 2(b) clearly demonstrates that only a minuscule percentage of items relapse. Therefore, most of the items are actually strongly converged. The significance of this finding is enhanced even further by its applicability across all datasets, despite the fact that they correspond to different types of products. Figure 2(c) shows the percentage of converged items for different values of d. For threshold values $d = 0.025$ and $d = 0.05$, the percentage of converged items ranges from 1% (CAM-0.025) to 33% (MP3-0.05). This variance demonstrates that, for small values of d, the convergence results are domain-specific. However, as d increases (e.g., $d = 0.4$), the percentage of converged items is equal or very close to 100% for all datasets.

Overall, our analysis demonstrates that the opinion vectors of review corpora converge as new reviews are added. In addition, we discovered that the vast majority of the items in our collections are strongly converged.

5.3 Comparison of the selection algorithms

In this section we evaluate the solutions reported by the Greedy, Integer-Regression and Iterative-Random algorithms for the Crs problem.

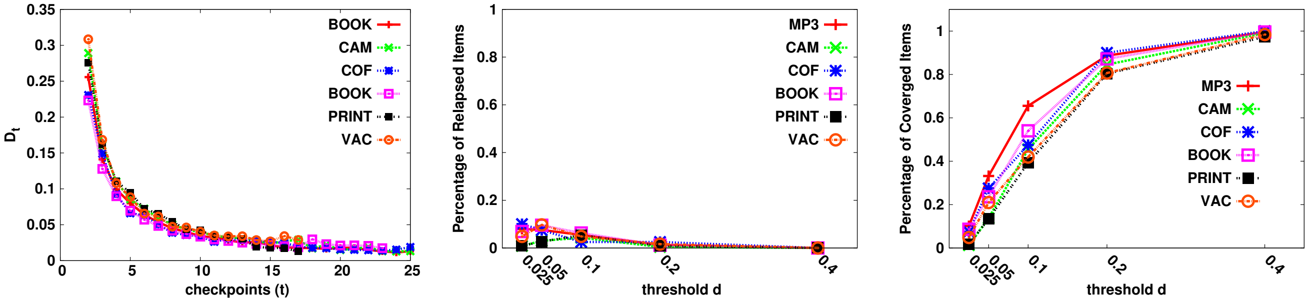
Randomization testing: We start our evaluation by performing the following randomization experiment. For each corpus $\mathcal{R}$ in our collection we use algorithm Alg[1] to extract a set of k characteristic reviews $\mathcal{S}_{\texttt{Alg}}$. Then, we evaluate the error of this set, i.e., $D_{\texttt{Alg}} = D(\pi(\mathcal{S}_{\texttt{Alg}}), \pi(\mathcal{R}))$. We compare the value $D_{\texttt{Alg}}$ with the corresponding error of a random set of k reviews $D_{\texttt{R}}$. After considering $N = 1000$ such random sets, we report the empirical p-value of algorithm Alg to be the fraction of times for which the random set exhibited a smaller error than the set reported by Alg. Formally, such an empirical p-value is computed as follows:

$$p_{\texttt{Alg}} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(D_{\texttt{Alg}} < D_i).$$

In the above equation, D_i corresponds to the error of the i-th random sample and $\mathbb{I}(D_{\texttt{Alg}}, D_i)$ is an indicator variable that takes value 1 if the error of the set reported by Alg is less than the error reported by the error of the i-th random sample of k reviews. Small values of $p_{\texttt{Alg}}$ are desirable, as they indicate that Alg has a very low probability of outperformed by a random set of k reviews from $\mathcal{R}$. The empirical p-values for the Greedy, Integer-Regression and Iterative-Random algorithms as a function of k are shown in Table 1. The values reported in the table are averages over all the corpora of every dataset.

We observe that the empirical p-values obtained for all our algorithms are either zero or close to zero. Therefore,

[1] Alg can be any of the three Iterative-Random, Greedy or Integer-Regression algorithms.



(a) Average D_t values across items for values of t in $\{1, \ldots, 25\}$.

(b) Percentage of relapsed items for different thresholds d.

(c) Percentage of converged items for different threshold values d.

Figure 2: Convergence experiments.

Table 1: Average empirical p-values for the solutions reported by `Greedy`, `Integer-Regression` and `Iterative-Random` for all datasets; average is taken over all items of a dataset.

	$k = 5$	$k = 10$	$k = 15$	$k = 20$
`MP3`				
`Greedy`	0.0	0.0	0.0	0.0
`Integer-Regression`	0.04	0.01	0.01	0.0
`Iterative-Random`	0.03	0.0	0.0	0.0
`CAM`				
`Greedy`	0.1	0.01	0.0	0.0
`Integer-Regression`	0.09	0.01	0.0	0.0
`Iterative-Random`	0.03	0.0	0.0	0.0
`COF`				
`Greedy`	0.14	0.0	0.0	0.0
`Integer-Regression`	0.9	0.0	0.0	0.0
`Iterative-Random`	0.06	0.0	0.0	0.0
`BOOK`				
`Greedy`	0.3	0.03	0.01	0.0
`Integer-Regression`	0.27	0.05	0.02	0.0
`Iterative-Random`	0.3	0.05	0.013	0.0
`PRINT`				
`Greedy`	0.13	0.02	0.01	0.0
`Integer-Regression`	0.07	0.0	0.0	0.0
`Iterative-Random`	0.05	0.0	0.0	0.0
`VAC`				
`Greedy`	0.07	0.014	0.0	0.0
`Integer-Regression`	0.8	0.01	0.0	0.0
`Iterative-Random`	0.05	0.0	0.0	0.0

the low-error solutions obtained by our algorithms cannot be attributed to randomness. For all datasets, the highest p-values are observed for $k = 5$. This trend is particularly pronounced in the `BOOK` dataset. This indicates that, for this dataset, it was more challenging for the algorithms to select a characteristic set with such a small size. This can be attributed to the ambiguity of the domain, which is less entertaining to objective judgments. Such ambiguity may lead to a more diverse opinion vector (i.e. with multiple positive and negative opinions per feature), and makes it more challenging to find a very small characteristic set of reviews. Nonetheless, the results show that the empirical p-values drop sharply as k increases. In fact, they reach zero or (near-zero) value for all datasets when $k \geq 15$.

Error values: The actual error values of the solutions reported by `Greedy`, `Integer-Regression` and `Iterative-Random` as a function of k are also shown in Table 2. Again the reported values are averages over all corpora in each dataset. The standard deviations within every dataset (not shown in the table) are values in the range $[0, 0.05]$ for all algorithms, datasets and values of k. The error values indicate all three algorithms achieve low errors that are reduced to near-zero as k is increased. Further, `Greedy` and `Integer-Regression` have a consistent advantage over `Iterative-Random`, in terms of both the absolute error values, and their ability to utilize larger values of k; observe that, for larger values of k, both `Greedy` and `Integer-Regression` achieve lower errors than `Iterative-Random`.

Error ratios: In order to further investigate the relationship between `Greedy`, `Integer-Regression` and `Iterative-Random`, we also conduct the following experiment. First, we collapse all the items from all six datasets into a single collection. For each value of k and each item, we compute $E_{\texttt{G}}$ (resp. $E_{\texttt{IR}}$) as the ratio of the error of the solution reported by `Greedy` (resp. `Integer-Regression`) to the error of the solution reported by `Iterative-Random`. We refer to both $E_{\texttt{G}}$ and $E_{\texttt{IR}}$ as the *error ratios*. Figure 3 shows the average values of these ratios for different values of k; the average is taken over all items. The results demonstrate that both ratios are consistently less than 1, indicating that both `Greedy` and `Integer-Regression` achieve less error than `Iterative-Random`. In fact, as the value of k increases the values of both ratios drop. This indicates that the advantage of `Greedy` and `Integer-Regression` becomes more pronounced as the value of k increases. Another observation is that `Integer-Regression` slightly outperforms `Greedy`, especially for larger values of k. This demonstrates that this algorithm is better at utilizing the flexibility of large values of k than `Greedy`, which simply extends the solution that it has obtained for smaller values of k.

5.4 User study

The goal of our user study is to demonstrate that the set of reviews reported by our methods is more appealing to the average user than the current state-of-the-art.

To achieve this, we we randomly select 10 items from the `MP3` dataset. For the 10 review corpora that correspond to these items, we select representative sets of reviews by us-

Table 2: Average error of the solutions reported by Greedy, Integer-Regression and Iterative-Random for all datasets; the average is taken over all items of a dataset.

	$k = 5$	$k = 10$	$k = 15$	$k = 20$
MP3				
Greedy	0.22	0.12	0.09	0.07
Integer-Regression	0.28	0.16	0.11	0.09
Iterative-Random	0.36	0.25	0.2	0.17
CAM				
Greedy	0.25	0.14	0.1	0.08
Integer-Regression	0.24	0.13	0.08	0.06
Iterative-Random	0.27	0.18	0.14	0.12
COF				
Greedy	0.22	0.13	0.1	0.07
Integer-Regression	0.23	0.12	0.08	0.06
Iterative-Random	0.23	0.15	0.12	0.11
BOOK				
Greedy	0.18	0.1	0.07	0.06
Integer-Regression	0.18	0.09	0.06	0.05
Iterative-Random	0.18	0.12	0.1	0.08
PRINT				
Greedy	0.24	0.14	0.1	0.09
Integer-Regression	0.23	0.13	0.09	0.07
Iterative-Random	0.25	0.17	0.14	0.12
VAC				
Greedy	0.28	0.16	0.12	0.1
Integer-Regression	0.26	0.14	0.09	0.07
Iterative-Random	0.29	0.2	0.16	0.13

ing three approaches: `Helpfulness`, `GroupCover`, and our own `Integer-Regression` algorithm. The first one ranks the review by the standard *helpfulness* measure that is implemented as a feature in major review hosting-sites such as `Yelp.com` or `Amazon.com`; we obtain the helpfulness scores of the reviews of our corpora while crawling. The second is the review-selection method proposed by Tsaparas *et al.* [16]. This method tries to maximize the number of features for which there is at least one positive *and* one negative opinion in the reported set. We use the `Integer-Regression` algorithm to represent our own selection paradigm, since it was shown to outperform the other algorithms in our other experiments, and is the one that gave the least-error solutions.

For each of the 10 items, we use the above three methods to select a set of 5 reviews. We chose to select no more than 5 reviews, to ensure that the human annotators would be able to complete the task in a reasonable amount of time. We asked the human annotators to rank the three sets from best (score 1) to worst (score 3). The criterion for the ranking was how well each set represents the entire corpus of reviews on the item. Clearly, it was impossible for the annotators to process the tens or even hundreds of reviews of the entire review corpus in order to make a decision. Thus, for each item, the annotator was shown the total number of positive and negative opinions expressed on each feature (*Price, Sound Quality, Battery Life, Connectivity, Design, Screen, Menu* and *Radio*) in the entire corpus. We also provided annotators with the original link to the review corpus on Amazon's website. In this way, we provided them with all the necessary information needed to determine their rankings. To conduct our survey, we created a HIT (Human Intelligence Task) on Amazon's Mechanical Turks platform. We hired 40 annotators to work on our HIT. The sets were shown in a randomized order to the annotators, who were thus oblivious to which set correspond to each approach.

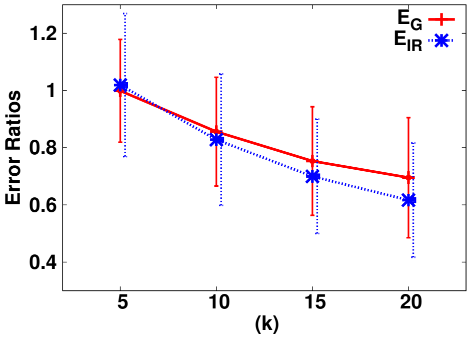


Figure 3: Average error ratios E_G and E_{IR} for $k \in \{5, 10, 15, 20\}$. The reported ratios are averages over all items in the union of all six datasets.

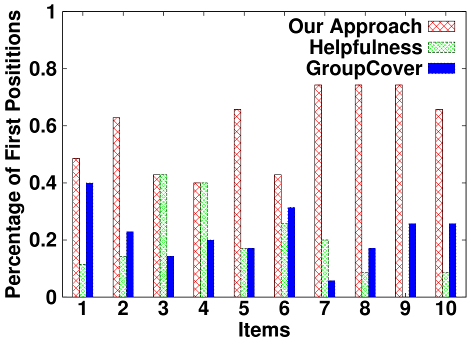


Figure 4: User study: percentage of rankings for which different methods were ranked first by the human annotators.

Figure 4 shows, for each of the 10 items, the percentage of workers that chose the set reported by each approach as the best one (i.e. ranked as 1). The results illustrate that the human subjects had a clear preference to the sets reported by our approach. A high percentage of the annotators consistently reported our approach as the best one, leading to percentages as high as 77%. In fact, our set was selected as the best for all items, with the exception of items 3 and 4 (for which our approach was tied with helpfulness).

Figure 5 shows the average rank assigned to each approach, for each of the 10 items. Recall that lower values are desirable (since smaller ranks indicate higher preference of users to the results of a method). This plot illustrates that our approach outperforms the two baselines. Items 3 and 4 are the only exceptions. For these, the reviews selected using the helpfulness scores were ranked slightly higher than ours. For all the other items, the average ranking achieved by our method was significantly lower than that of `Helpfulness` and `GroupCover`.

In conclusion, the results of the user study demonstrate the validity and superiority of our selection paradigm, when

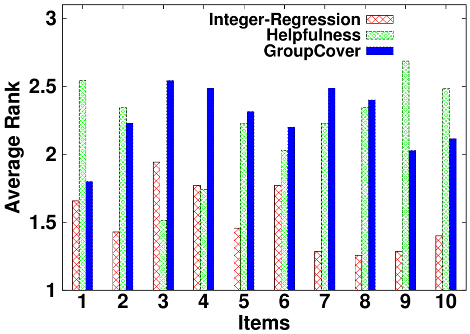


Figure 5: User study: average ranking of different methods as ranked by human annotators.

compared to existing ranking criteria and state-of-the-art review-selection methods.

6. CONCLUSIONS

In this paper we introduce the problem of selecting a characteristic set of reviews from a given review-corpus. Our selection paradigm is a significant improvement over previous relevant work, since it is the first to ask for a set that respects the *proportion* of opinions on each feature (both positive and negative), as observed in the underlying corpus. We formally define the CHARACTERISTIC-REVIEW SELECTION problem and prove that it is not only NP-hard, but also NP-hard to approximate. Nevertheless, we propose three heuristic algorithms for selecting a characteristic review set, which we evaluate via experiments on a wide range of datasets of real reviews from different domains. The results indicate that our algorithms are consistently able to find a small set of reviews that yield a highly accurate approximation of the set of opinions in a large review corpus. To demonstrate that our problem definition and solution is an important improvement over previous efforts, we perform a user study using the Amazon Mechanical Turks platform. The study reveals that human subjects consistently prefer the set of reviews selected by our methods, as opposed to the corresponding sets selected by previous state-of-the-art methods. From the practical point of view, we point out that our work can be easily incorporated into any review-hosting website, in order to provide users with a compact and informative set of real reviews, that accurately represents the opinions expressed in the entire corpus.

Acknowledgments

This research was supported by gifts from Microsoft, Yahoo! and Google.

7. REFERENCES

[1] S. Baccianella, A. Esuli, and F. Sebastiani. Multi-facet rating of product reviews. In *Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval*, ECIR '09, pages 461–472, Berlin, Heidelberg, 2009. Springer-Verlag.

[2] X. Ding, B. Liu, and P. S. Yu. A holistic lexicon-based approach to opinion mining. In *Proceedings of the international conference on Web search and web data mining*, WSDM '08, pages 231–240, New York, NY, USA, 2008. ACM.

[3] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. 1979.

[4] A. Ghose and P. G. Ipeirotis. Designing novel review ranking systems: predicting the usefulness and impact of reviews. In *ICEC '07*, pages 303–310. ACM, 2007.

[5] M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 168–177, 2004.

[6] N. Jindal and B. Liu. Opinion spam and analysis. In *In Proceedings of the International Conference on Web Search and Web Data Mining (WSDM*, pages 219–230. ACM Press, 2008.

[7] S.-M. Kim, P. Pantel, T. Chklovski, and M. Pennacchiotti. Automatically assessing review helpfulness. In *EMNLP*, pages 423–430, 2006.

[8] T. Lappas and D. Gunopulos. Efficient confident search in large review corpora. In *ECML/PKDD (2)*, pages 195–210, 2010.

[9] J. Liu, Y. Cao, C.-Y. Lin, Y. Huang, and M. Zhou. Low-quality product review detection in opinion summarization. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 334–342, 2007. Poster paper.

[10] Y. Liu, X. Huang, A. An, and X. Yu. Modeling and predicting the helpfulness of online reviews. In *ICDM*, pages 443–452, 2008.

[11] Y. Lu, P. Tsaparas, A. Ntoulas, and L. Polanyi. Exploiting social context for review quality prediction. In *WWW*, 2010.

[12] S. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, 1999.

[13] S. Mallat, G. Davis, and Z. Zhang. Adaptive time-frequency decompositions. *SPIE Journal of Optical Engineering*, pages 2183–2191, 1994.

[14] X. Meng and H. Wang. Mining user reviews: from specification to summarization. In *ACL/AFNLP (Short Papers)*, pages 177–180, 2009.

[15] R. Tadano, K. Shimada, and T. Endo. Multi-aspects review summarization with objective information. 2010.

[16] P. Tsaparas, A. Ntoulas, and E. Terzi. Selecting a comprehensive set of reviews. In *KDD*, 2011.

[17] O. Tsur and A. Rappoport. Revrank: a fully unsupervised algorithm for selecting the most helpful book reviews. In *ICWSM*, 2009.

[18] J. Zhan, H. T. Loh, and Y. Liu. Gather customer concerns from online product reviews: A text summarization approach. *Expert Systems with Applications*, 36(2, Part 1):2107 – 2115, 2009.

[19] Z. Zhang and B. Varadarajan. Utility scoring of product reviews. In *CIKM*, pages 51–57. ACM, 2006.

[20] L. Zhuang, F. Jing, X. Zhu, and L. Zhang. Movie review mining and summarization. In *Proceedings of the ACM SIGIR Conference on Information and Knowledge Management (CIKM)*, 2006.