Ontology- and Sentiment-aware Review Summarization

Nhat X.T. Le
nle020@ucr.edu
University of California, Riverside
Computer Science and Engineering

Vagelis Hristidis
vagelis@cs.ucr.edu
University of California, Riverside
Computer Science and Engineering

Neal Young
neal.young@ucr.edu
University of California, Riverside
Computer Science and Engineering

ABSTRACT

In this Web 2.0 era, there is an ever increasing number of product or service reviews, which must be summarized to help consumers effortlessly make informed decisions. Previous work on reviews summarization has simplified the problem by assuming that features (e.g., "display") are independent of each other and that the opinion for each feature in a review is Boolean: positive or negative. However, in reality features may be interrelated – e.g., "display" and "display color" – and the sentiment takes values in a continuous range – e.g., somewhat vs very positive.

We present a novel review summarization framework that advances the state-of-the-art by leveraging a domain hierarchy of concepts to handle the semantic overlap among the features, and by accounting for different sentiment levels. We show that the problem is NP-hard and present bounded approximate algorithms to compute the most representative set of sentences or reviews, based on a principled opinion coverage framework. We experimentally evaluate the proposed algorithms on real datasets in terms of their efficiency and effectiveness compared to the optimal algorithms. Further, the quality of the summaries is evaluated using both intuitive coverage measure, and a user study. As a side contribution, we compare various review sentence sentiment estimation methods.

CCS CONCEPTS

•**Information systems** →**Summarization;** *Sentiment analysis;*

KEYWORDS

review summarization; sentiment analysis

ACM Reference format:
Nhat X.T. Le, Vagelis Hristidis, and Neal Young. 2017. Ontology- and Sentiment-aware Review Summarization. In *Proceedings of SIGKDD, Halifax, Nova Scotia - Canada, August 2017 (KDD 2017),* 12 pages.
DOI: 10.475/123_4

1 INTRODUCTION

Online users are increasingly relying on user reviews to make decisions on shopping (e.g., Amazon, Newegg), finding venues (e.g., Yelp, Foursquare), seeking doctors (e.g., Vitals.com, zocdoc.com) and many others. However, as the number of reviews per item grows, especially for popular products, it is infeasible for customers to read all of them, and discern the useful information from them. Therefore, many methods have been proposed to summarize customer opinions from the reviews [5, 8, 12, 17]. They generally either adapt multi-document summarization techniques to choose important text segments [5], or they extract product concepts (also referred as features or attributes in other works), such as "display" of a phone, and customer's opinion (positive or negative) and aggregate them [8, 12, 17].

However, neither of these approaches takes into account the relationship among product's concepts. For example, assuming that we need the opinion summary of a smartphone, showing that the opinions for both *display* and *display color* are very positive is redundant, especially given that we would have to hide other concepts' opinion (e.g., "battery"), given the limited summary size. What makes the problem more challenging is that the opinion of a user for a concept is not Boolean (positive or negative) but can take values from a linear scale, e.g., "very positive", "positive", "somewhat positive", "neutral", and so on. Hence, if "display" has a positive opinion, but "display color" has neutral, the one does not subsume the other, and both should be part of the summary. Further, a more general concept may cover a more specific but not vice versa.

In Section 4 we prove that the problem of selecting the best concepts and opinions to display such all opinions are covered as much as possible is NP-hard even when the relationships among the concepts are represented by a Directed-Acyclic-Graph (DAG). Therefore we proposed bounded approximation algorithms to solve it.

We experimentally evaluated our method on real collections of online patient reviews about doctors. The reason we chose this dataset is that rich concept hierarchies exist for the medical domain [4], and effective tools exist to extract medical concepts from free text [2, 23].

To summarize, the review summarization framework consists of the following tasks:

(1) *Concept Extraction*: extract interesting medical concepts from reviews.
(2) *Concept's Sentiment Estimation*: model the context around the extracted concepts and estimate its sentiment (opinion polarity on a linear scale).
(3) *Select k representatives*: depending on the application, a representative can be a concept-sentiment pair (e.g., "display"=0.3) or a sentence from a review (e.g., "this phone has pretty sharp display") or a whole review. Our proposed selection algorithms can be used to select representatives at any of these granularities.

As secondary contributions, we first show how to filter our concepts with little interest to the users of patient reviews, e.g., the concept "dog" is not interesting in the context of doctor reviews.

We also evaluate several methods to estimate the sentiment of a review sentence, including both dictionary-based methods and neural network approaches.

Our contributions can be summarized as below:

- We propose a fresh perspective for the review summarization problem that exploits available concept hierarchies and a novel opinion coverage definition. We model the problem as a coverage optimization problem (Section 2).
- We show how to select interesting concepts for the doctor review domain and how to best estimate the sentiment of sentences (Section 3).
- We prove that the problem is NP-hard when the concept ontology is a DAG (Section 4), then propose several efficient approximation algorithms that have guaranteed bounds (Section 5).
- Finally we conducted extensive experiments on every task of our framework. We compare two sentiment estimation methods with a labeled dataset that is collected from a user study that we conducted. Then, we carried out a complete evaluation on the cost and time of our three proposed algorithms (Section 6.1).
- We conducted a second user study to compare the quality of the generated review summaries compared to reasonable baselines. We also quantitatively evaluate the quality using several coverage definitions. From this series of experiment we found that our Greedy algorithm performs best, as it outperform the baseline methods in both qualitative and quantitative evaluation (Section 6.2).

The remainder of the paper is organized as follows: we present the related work in Section 7, and the conclusion and future work in Section 8.

2 PROBLEM DEFINITIONS

Define an item (for example, a doctor) d as a set of reviews, where each review is a set of *concept-sentiment* pairs $\{(c_1, s_1), (c_2, s_2), \ldots, (c_n, s_n)\}$, and $s_j \in \mathbb{R}$ is the sentiment toward concept c_j in a review. Section 3 shows how the concept-sentiment pairs are extracted from the text of the reviews. The set of concepts (the *ontology*) depends on the application. We assume concepts are hierarchical, which is common in many domains (ConceptNet for example defines a general purpose concept hierarchy [16]). Further, previous work has studied how features hierarchies can be extracted from product reviews [26]. For the health-related content in our experiments, SNOMED CT [1] is a typical ontology with such a hierarchy (Figure 1).

Define the (directed) *distance* $d(p_1, p_2)$ between two concept-sentiment pairs $p_1 = (c_1, s_1)$ and $p_2 = (c_2, s_2)$, based on the concepts' relationship in the hierarchy, as follows.

Definition 2.1. First, define the distance between two concepts $d(c_1, c_2)$ to be the shortest-path length from c_1 to c_2 in the hierarchy. Let r be the root of the hierarchy. Let $\epsilon > 0$ be a pre-defined (sentiment) threshold. The distance $d(p_1, p_2)$ is:

$$d(p_1, p_2) = \begin{cases} d(r, c_2) & \text{if } c_1 \text{ is the root } r \text{, or} \\ d(c_1, c_2) & \text{if } c_1 \text{ is the ancestor of } c_2 \\ & \text{ and } |s_1 - s_2| \le \epsilon \text{, or} \\ \infty & \text{otherwise} \end{cases}$$

If pair p_1 has finite distance to p_2, say p_1 *covers* p_2. Pair p_1 covers p_2 iff p_1's concept c_1 is an ancestor of p_2's concept c_2, and either c_1 is the root concept or the sentiments of p_1 and p_2 differ by at most ϵ. Figure 2 shows an example of how the concept-sentiment pairs of an item's reviews are mapped on the concept hierarchy of Figure 1, where the dashed line is the path from the root, and concept c_6 doesn't have any pairs. For instance, pair $(c_1, 0.7)$ represents an occurrence of concept c_1 in a review with sentiment 0.7. The same pair is also represented by the circled 0.7 value inside the c_1 tree node.

Given a set $P = \{p_1, p_2, \ldots, p_q\}$ of concept-sentiment pairs for the reviews of an item, and an integer k, our goal is to compute a set $F = \{f_1, f_2, \ldots, f_k\} \subseteq P$ of k pairs that best summarize P. To measure the quality of such a summary F, we define its cost $C(F, P)$ as the distance from F to P, defined as follows.

Definition 2.2. Define the distance from F to a pair p to be the distance of the closest pair in $F \bigcup \{r\}$ to p: $d(F, p) = \min_{f \in F \bigcup \{r\}} d(f, p)$. Define the cost of F to be the sum of its distances to pairs in P: $C(F, P) = \sum_{p \in P} d(F, p)$.

We introduce two summarization problems on a set of concept-sentiment pairs:

(1) ***k-Pairs Coverage:*** given a set P of concept-sentiment pairs (coming from a given set of reviews for an item) and integer $k \le |P|$, find a subset $F \subseteq P$ with $|F| = k$ that summarizes P with minimum cost:
$$\min_{F \subseteq P, |F| = k} C(F, P)$$
(2) ***k-Reviews/Sentences Coverage:*** given a set R of reviews (or sentences) and integer $k \le |R|$, find a subset $X \subseteq R$ with $|X| = k$ that summarizes R with minimum cost:
$$\min_{X \subseteq R, |X| = k} C(P(X), P(R)),$$
where $P(R)$ is the set of concept-sentiment pairs derived from the set R of reviews/sentences, and $P(X)$ is the set of concept-sentiment pairs derived from the subset X of R.

Intuitively, the first problem is appropriate when the summaries consist of concise concept-sentiment pairs, e.g. "good Heart Disease management", extracted from the reviews, and may be more suitable for mobile phone-sized screens. The second problem is appropriate if the summaries consist of whole sentences of reviews, which better preserves the meaning of the review, but may require more space to display.

The k-Pairs Coverage problem can be viewed as a special case of the k-Reviews/Sentences Coverage problem, when each review/sentence has just one pair. For presentation simplicity, we first present algorithms for k-Pairs Coverage in Section 5, and describe how they can be applied to the k-Reviews/Sentences Coverage in Section 5.5.

3 REVIEWS TRANSFORMATION

In this section we describe how we process reviews to transform them into the (concept, sentiment) pairs required by the problem definitions in Section 2. We focus on doctor reviews to make the discussion and solutions concrete. First, we extract "interesting" medical concepts (Section 3.1), and then we compute the sentiment around that concept in the review (Section 3.2). Note that, this

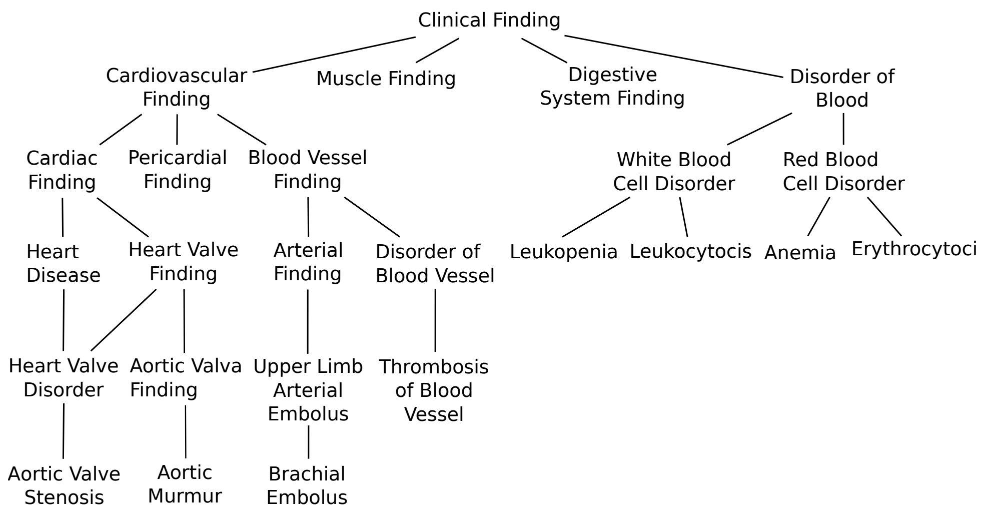


Figure 1: SNOMED CT's concept hierarchy sample

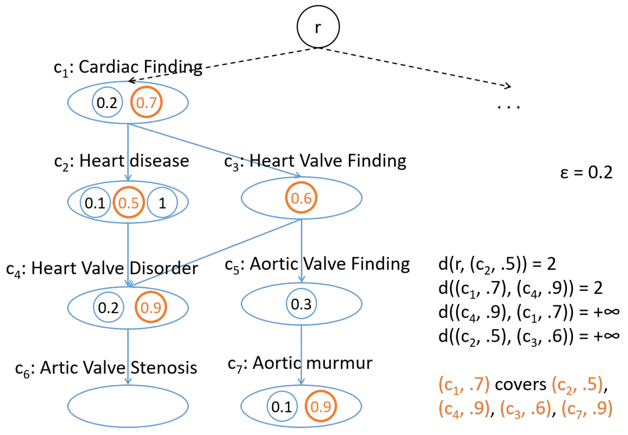


Figure 2: Representation of concept-sentiment pairs on the concept hierarchy DAG of Figure 1.

transformation can be pre-computed and save into disk as the input for later usages in Section 5.

3.1 Concepts and sentences extraction

To extract medical concepts we use MetaMap [2], which is an automated tool for mapping biomedical text to medical concepts from Unified Medical Language System (UMLS) Metathesaurus [4]. UMLS contains multiple medical ontologies; we choose SNOMED CT [1], which has more than 300,000 concepts and is suitable for our problem given its focus on describing medical conditions.

We adapt MetaMap for the needs of our problem as follows. First, given that MetaMap was built for medical records authored by healthcare professional, it makes several mistakes when analyzing patient-generated reviews. For that, we have manually generated a set of rules that are based on [29]. For example, the pronoun "I" is mapped to concept "Iodine," which has Concept Unique Identifier (CUI) C0021968. Similarly, "OMG" and "omg" were mapped to "OMG gene" (C1417949), and "said" was mapped to "Simian Acquired Immunodeficiency Syndrome" (C0080151).

Second, not all concepts are interesting in the context of provider reviews. For that, we manually selected a set of semantic types, and only consider concepts of these types. Specifically, we chose Disease or Syndrome (e.g., asthma), Sign or Symptom (e.g., pain), Occupational Activity (e.g., ...), ... We removed concepts of other types, including Geographic Area, Animal, Food, ... Finally, although selecting concepts by semantic type is effective, we defined a few exceptions to include some concepts from removed semantic types; specifically, we included "Exam Room" (C1547117), Time of visit (C1320304), Friendly (C2700214), ...

3.2 Sentiment Computation

To compute the sentiment around a concept, we first need to define the context, i.e. the *containing sentence*, which we compute using the PTBTokenizer library in Stanford POS tagger [27], and generally splits text based on punctuation marks.

Methods We compare two methods for estimating the sentiment of a containing sentence: (a) using a sentiment dictionary and (b) employing learning method based on vector representation of sentences

In the former, which has been the dominant in the last decade [8], a concept's sentiment is estimated by averaging the sentiment values of surrounding words within a containing sentence, normalized by the distance between the concept and the surrounding words. Given a sentence s containing concept c and a set of opinion terms $\{w_1 \dots w_n\}$, the opinion orientation (sentiment) for c is determined by the following formula:

$$score(c, s) = \sum_{w_j \in s} \frac{sentiment(w_j)}{dist(w_j, c)}$$

where $dist(w_j, c)$ is the distance in number of words between c and opinion term w_j in s. $sentiment(w_j)$ is the sentiment score of w_i in a sentiment dictionary. In this estimation, the further an opinion

Table 1: Comparison of different sentiment estimation methods using continuous values (Lower is better)

Error Metric	Baseline: predict as 0 (most common)	Dictionary	**Ridge**	Lasso	Linear SVR	Dict+Ridge	Dict+Lasso	Dict+Linear SVR
Absolute Error Mean	0.202	0.191	**0.150**	0.203	0.151	0.158	0.195	0.157
Standard Deviation	0.139	0.122	0.106	0.39	**0.104**	**0.104**	0.124	**0.104**

Table 2: Comparison of different sentiment estimation methods using class discretization (Higher is better)

Accuracy	Baseline: predict as the most common	Dictionary	**Ridge**	Lasso	Linear SVR	Dict+Ridge	Dict+Lasso	Dict+Linear SVR
3-class accuracy	0.389	0.402	**0.570**	0.389	0.557	0.500	0.98	0.512
5-class accuracy	0.303	0.316	0.398	0.303	**0.406**	0.361	0.299	0.361

word to the concept, the lesser effect it has. We use a popular sentiment dictionary named SentiWordNet [3]. Specifically, we first apply the Stanford POS Tagger [27] to find the Part-of-Speech (POS) of each word, then look up the word with corresponding POS in SentiWordNet dictionary.

The latter method is a very recent approach [15], which represents words, sentences, and generally text by fixed-size vectors. Specifically, we first use an unsupervised method, based on a neural network, to learn the vector representation of all sentences, then feed the vectors to a simple trained regression model to predict their sentiment. We follow the experimental protocol in [15] and input a dataset of 200,000 reviews into a neural network to learn reviews' representative vectors. These reviews have uniform distribution of user star rating (1 to 4 stars).

Specifically, first, for each review we generate two 200-dimensional vectors, using two training methods: Distributed Memory with Concatenation and Distributed Bag of Word. Then, we concatenate them into 400-dimensional vectors. After that, we feed these vectors of the 200,000 reviews and their user star rating (normalized into $-1, +1$ range) to train a simple regression model. To predict the sentiment of new sentences, we first learn their representative vectors by extending the above neural network with the new sentences but not modifying the vectors of the 200,000 training reviews (and hence not changing the regression model). The reason we are using whole reviews and not sentences for training is that ratings are available for the whole reviews. Then, the sentences' sentiments are predicted using their vectors as the input features using the trained regression model. For these tasks we utilize Python packages: *Gensim* [22] for vector learning task and *scikit-learn* [19] for regression.

Evaluation To compare the accuracy of two methods, we created a sample set of 244 sentences extracted from 100 reviews (25 reviews for each star rating). Note that these 100 reviews are different than the training reviews. We then conducted a user study with four people (2 professors, 2 graduate students), where we asked each subject to select a sentiment polarity for each sentence: very negative, negative, neutral, positive and very positive, corresponding to values -1, -0.5, 0, 0.5, 1. The reliability of the user study is tested using Krippendorff's alpha coefficient [13] that estimates the degree of agreement between the users. Specifically, we consider our sentiment (from -1 to +1) as polar data type in Krippendorff's alpha formula, and our alpha is 0.735, which is considered reliable enough ($\alpha \geq 0.667$). We then computed the average of the subjects' judgment for each sentence and use it as the ground truth.

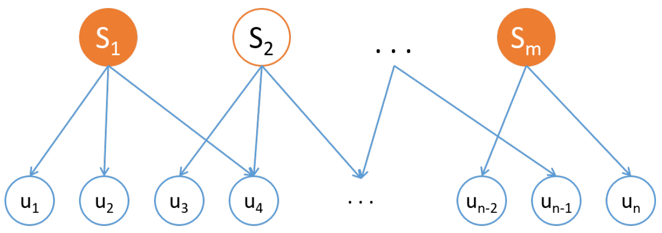
Our experimental result for the two approaches are presented in Table 1, 2. For the vector representation based method, we tried Ridge [11], Lasso [25] and Linear Support Vector [24] regression for predicting continuous sentiment values. We also considered combining the two methods (dictionary-based and vector-based) by averaging their estimation. For sanity, we also include a baseline method which assigns the same constant sentiment – the most common one – to every sentence to see if it can achieve a better error metric. Table 1 presents the mean and standard deviation of absolute errors normalized by dividing by 2 (due to the length 2 of range [-1, 1]). For this task, Ridge and Linear Support Vector Regression (SVR) perform the best. Table 2 reports the accuracy of methods when we discretize the predicted continuous sentiments into separate classes such as negative, neutral and positive. The discretization uses equal-length ranges of continuous sentiment for each class. That is, 5-class discretization means that sentiment range of $[-1, +1]$ is broken down to 5 equal ranges: very negative [-1, -0.6), negative [-0.6, 0.2), neutral [-0.2, 0.2), positive [0.2, 0.6) and very positive [0.6, 1]. Note that it is not surprising that the accuracy of this task is relatively low, as it is a hard problem; the state-of-the-art sentiment classification research achieves similar accuracies [15]. We can observe that Ridge regression not only performs the best in both tasks but also can be trained quickly (2.79 seconds, comparing to 92.63 seconds of Linear SVR). Therefore, we decided to use the vector representation-based method with Ridge regression for sentiment estimation in the rest of the paper's experiments.

4 BOTH PROBLEMS ARE NP-HARD

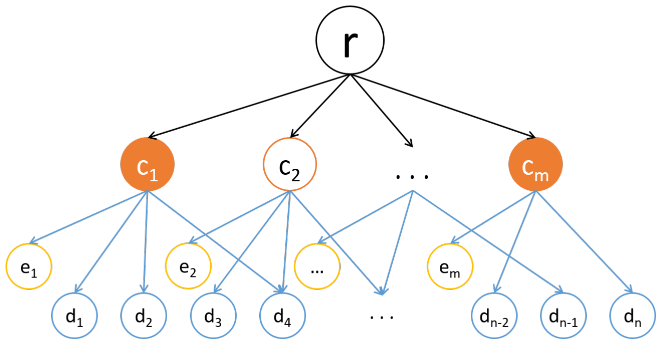
This section proves both proposed problems NP-hard.

THEOREM 4.1. *The k-Pairs Coverage problem is NP-hard.*

PROOF. The decision problem is, given a set P of concept-sentiment pairs, an integer $k \leq |P|$, and a target $t \geq 0$, to determine whether there exists a subset $F \subseteq P$ of size k with cost $C(F, P)$ at most t. We reduce Set Cover to it. Fix any Set-Cover instance (S, U, k) where U is the universe $\{u_1, u_2, \ldots, u_n\}$, and $S = \{S_1, S_2, \ldots, S_m\}$ is a collection of subsets of U, and $k \leq |S|$. Given (S, U, k), first construct a concept-hierarchy (DAG) with root r, concepts c_i and e_i for each subset S_i, and a concept d_j for each element u_j. For each set S_i, make c_i a child of r and e_i a child of c_i. For each element u_j,



(a) Set Cover



(b) Corresponding instance of k-Pairs Coverage

Figure 3: Reduction from Set Cover

make d_j a child of c_i for each set S_i containing u_j. (See Fig. 3.) Next, construct $2m + n$ concept-sentiment pairs $P = \{p_1, \ldots, p_{2m+n}\}$, one containing each node in the DAG other than the root r, and all with the same sentiment, say 0. Take target $t = 3m + n - 2k$. This completes the reduction. It is clearly polynomial time. Next we verify that it is correct. For brevity, identify each pair with its node.

Suppose S has a set cover of size k. For the summary $F \subseteq P$ of size k, take the k concepts in P that correspond to the sets in the cover. Then each d_i has distance 1 to F, contributing n to the cost. For each set in the cover, the corresponding c_i and e_i have distance 0 and 1 to F, contributing k to the cost. For each set not in the cover, the corresponding c_i and e_i have distance 1 and 2 to F, contributing $3(m - k)$ to the cost, for a total cost of $n + 3m - 2k = t$.

Conversely, suppose P has a summary of size k and cost $t = n + 3m - 2k$. Among size-k summaries of cost at most t, let F be one with a maximum number of c_i nodes. We show that the sets corresponding to the (at most k) c_i nodes in F form a set cover. Assume some $c_{i'}$ is missing from F (otherwise $k \geq m$ so we are done). For every e_i in F, its parent c_i is also in F. (Otherwise adding c_i to F and removing e_i would give a better summary F', i.e., a size-k summary of cost at most t, but with more c_i nodes than F, contradicting the choice of F). No e_i is in F (otherwise removing e_i and adding the missing node $c_{i'}$ would give a better summary F'). No d_j is in F (otherwise, since neither $e_{i'}$ nor $c_{i'}$ are in F, removing d_j from F and adding $c_{i'}$ would give a better summary F'). Since no e_i or d_j is in F, only c_i nodes are in F. Since the cost is at most $t = n + 3m - 2k$, by calculation as in the preceding paragraph, the sets S_i corresponding to the nodes c_i in F must form a set cover. □

When we already have k-Pairs Coverage as a NP-hard problem, it's natural to prove the following theorem.

Theorem 4.2. *The k-Reviews/Sentences Coverage problem is NP-hard.*

Proof. K-Reviews/Sentences Coverage is a generalization of k-Pairs Coverage, so the theorem follows from the previous theorem. □

5 ALGORITHMS

We implement three algorithms for k-Pair Coverage. The first, which is the only one generates an optimal solution, solves the standard integer-linear program (ILP) for the problem, as a special case of the well-known k-Medians problem. The second randomly solves the linear program (LP), then randomly rounds the fractional solution achieving a bounded approximation error. The third is a greedy bounded approximation algorithm. The three algorithms share a common initialization phase that we describe first.

5.1 Initialization

The transformation in Section 3 gives a set P of concept-sentiment pairs with concepts from the ontology hierarchy (DAG). The initialization phase computes the underlying edge-weighted bipartite graph $G = (U, W, E)$ where vertex sets U and W are the concept-sentiment pairs in the given set P, edge set E is $\{(p, p') \in U \times W : d(p, p') < \infty\}$, and edge (p, p') has weight equal to the pair distance $d(p, p')$. The initialization phase builds G in two passes over P. The first pass puts the pairs $p = (c, s)$ into buckets by category c. The second pass, for each pair $p = (c, s)$, iterates over the ancestors of c in the DAG (using depth-first-search from c). For each ancestor c', it checks the pairs $p' = (c', s')$ in the bucket for c'. For those with finite distance $d(p, p')$, it adds the corresponding edge to G.

For our problems, the time for the initialization phase and the size of the resulting graph G are roughly linear in $|P|$, because the average number of ancestors for each node in the DAG is small.

5.2 ILP for optimal solution

Given the graph $G = (U, W, E)$, Fig. 4 shows the standard k-Medians ILP adapted for our non-standard cost function. Our first algorithm solves the ILP using the Gurobi solver. Of course, no worst-case polynomial-time bounds are known for solving this NP-hard ILP, but on our instances the algorithm finishes in reasonable time. (Details are in Section 6.)

5.3 Randomized rounding

The second algorithm computes an optimal fractional solution (x, y) to the LP relaxation of the ILP (using Gurobi, details in in Section 6), then randomly rounds it as shown in algorithm 1: it chooses the summary F by sampling k pairs p at random from the distribution $x/\|x\|_1$.

No good worst-case bounds are known on the time to solve the LP, but on our instances the solver solves it in reasonable time. The randomized-rounding phase can easily be implemented to run in linear time, $O(n)$ where $n = |P|$.

This randomized-rounding algorithm is due to [31] (see also [6]). The following worst-case approximation guarantee holds for this

$$
\begin{array}{lrl}
\text{minimize} & \displaystyle\sum_{(p,q)\in E} y_{pq} \times d(p,q) & \\
\text{subject to} & \displaystyle\sum_{p\in P\setminus\{r\}} x_p & = k \\
 & x_r & = 1 \\
(\forall q \in W) & \displaystyle\sum_{p:(p,q)\in E} y_{pq} & = 1 \\
(\forall (p,q) \in E) & 0 \le y_{pq} & \le x_p \\
(\forall p \in U) & x_p & \in \{0,1\}
\end{array}
$$

Figure 4: k-Medians ILP

algorithm, as a direct corollary of the analysis in [6]. Let $\text{OPT}_k(P)$ denote the minimum cost of any size-k summary of P.

Theorem 5.1. *The expected cost of the size-k summary returned by the randomized-rounding algorithm is $O(\text{OPT}_{k'}(P))$ for some $k' = O(k/\log n)$.*

In our experiments it gives near-optimal summary costs.

Algorithm 1 Randomized Rounding Algorithm

```
Input: fractional solution $x, y$
Output: summary $F$
1: procedure RANDOMIZED ROUNDING
2:     Define probability distribution $q$ on $P' = P \setminus \{r\}$
3: such that $q(p) = x_p / \sum_{p\in P'} x_p$.
4:     $F = \emptyset$
5:     while $|F| < k$ do
6:         Sample one pair $p$ without replacement from $q$.
7:         Add $p$ to $F$.
8:     Return $F$.
```

5.4 Greedy algorithm

The greedy algorithm is Algorithm 2. It starts with a set $F = \{r\}$ containing just the root. It then iterates k times, in each iteration adding a pair $p \in P$ to F chosen to minimize the resulting cost $C(F \cup \{p\}, P)$. Finally, it returns summary $F \setminus \{r\}$. This is essentially a standard greedy algorithm for k-medians. Since the cost is a submodular function of P, the algorithm is a special case of Wolsey's generalization of the greedy set-cover algorithm [30].

After the initialization phase, which computes the graph $G = (U, W, E)$, the algorithm further initializes a max-heap for selecting p in each iteration. The max-heap stores each pair p, keyed by $\delta(p, F) = C(F \cup \{p\}, P) - C(F, P)$. The max-heap is initialized naively, in time $O(m + n \log n)$ (where $m = |E|$, $n = |P|$). (This could be reduced to $O(m + n)$ with the linear-time build-heap operation.) Each iteration deletes the pair p with maximum key from the heap (in $O(\log n)$ time), adds p to F, and then updates the changed keys. The pairs q whose keys change are those that are neighbors of neighbors of p in G. The number of these updates is typically $O(d^2)$, where d is the typical degree of a node in G. The cost of each update is $O(\log n)$ time. After initialization, the algorithm typically takes $O(kd^2 \log n)$ time. In our experiments, our graphs are sparse (a typical node p has only hundreds of such pairs q), and k is a small constant, so the time after initialization is dominated by the time for initialization.

The following worst-case approximation guarantee is a direct corollary of Wolsey's analysis [30]. Let $H(i) = 1 + 1/2 + \cdots + 1/i \approx 1 + \log i$ be the ith harmonic number. Let Δ be the maximum depth of the concept DAG.

Theorem 5.2. *The greedy algorithm produces a size-k summary of cost at most $\text{OPT}_{k'}(P)$, where $k' = \lfloor k/H(\Delta n)\rfloor$.*

In our experiments, the algorithm returns near-optimal size-k summaries.

Algorithm 2 Greedy Algorithm

```
Input: $G = (U, W, E)$ from initialization, computed from $P$.
Output: Size-$k$ summary $F$.
1: procedure GREEDY
2:     Define $\delta(p, F) = C(F \cup \{p\}, P) - C(F, P)$.
3:     Let $F = \{r\}$.
4:     Initialize max-heap holding $p \in U$ keyed by $\delta(p, F)$.
5:     while $|F| < k + 1$ do
6:         Delete $p$ with highest key from max-heap.
7:         Add $p$ to $F$.
8:         for $w$ such that $(p, w) \in E$ do
9:             for $q$ such that $(q, w) \in E$ do
10:                Update max-heap key $\delta(q, F)$ for $q$.
11:    return $F \setminus \{r\}$
```

5.5 Adaptation for k-Reviews/Sentences Coverage problem

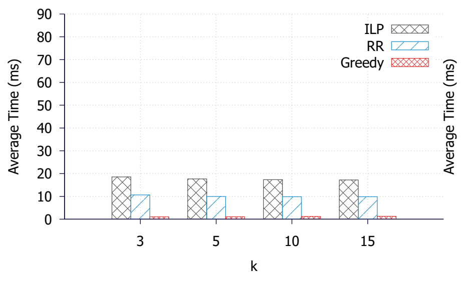
When whole reviews or sentences (each containing a set of concept-sentiment pairs) must be selected, the above algorithms can still be applied with only a modification of the initialization stage of Section 5.1. In particular, we modify the construction of bipartite graph $G = (U, W, E)$, so instead of having both U and W be concept-sentiment pairs in P, U represents the set of candidate reviews or sentences R, and W represents concept-sentiment pairs as before. Therefore the edge set E becomes $\{(r, p) \in U \times W : d(r, p) < \infty\}$, and edge (r, p) has weight equal to the distance $d(r, p)$ from review/sentence r to pair p. After this initialization, the algorithms work as usual.

6 EXPERIMENTAL EVALUATION

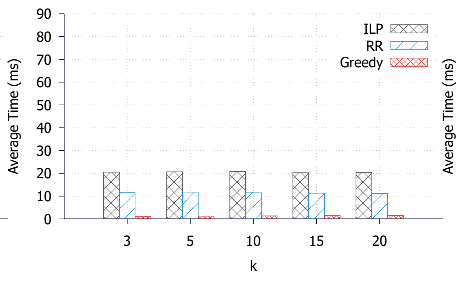
In this section we conduct both quantitative and qualitative evaluations. The quantitative one measures the time and accuracy tradeoffs of the proposed summarization algorithms. The qualitative evaluation measures the quality of the proposed methods compared to baseline state-of-the-art summarization methods in two ways: using a user study and using intuitive coverage measures.

6.1 Quantitative Evaluation

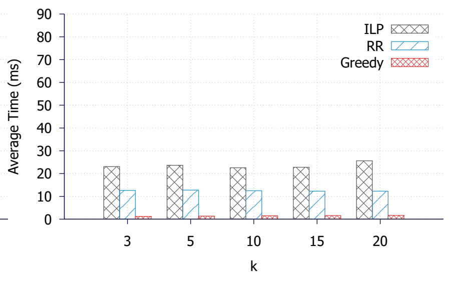
Setup and Datasets We evaluate the three algorithms in Section 5: Integer Linear Programming - ILP, Randomized Rounding - RR, and Greedy algorithm. For ILP and RR, we use the Gurobi optimization



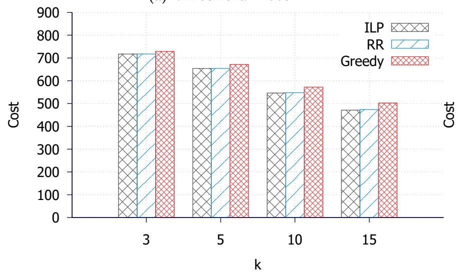
(a) threshold = 0.3



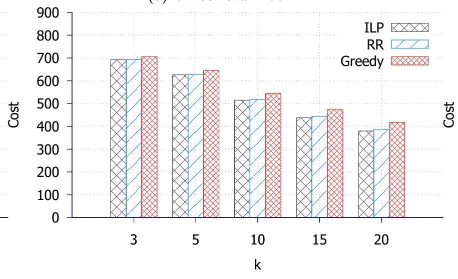
(b) threshold = 0.4



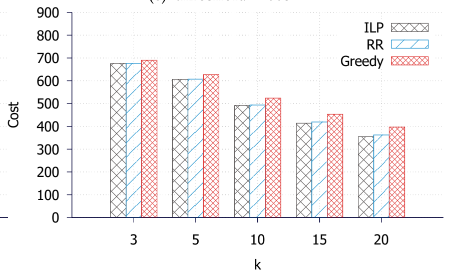
(c) threshold = 0.5



(d) threshold = 0.3

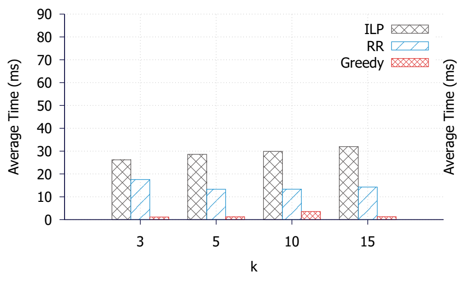


(e) threshold = 0.4

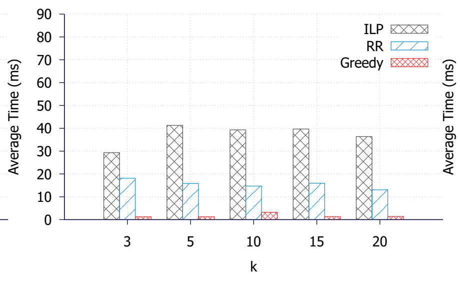


(f) threshold = 0.5

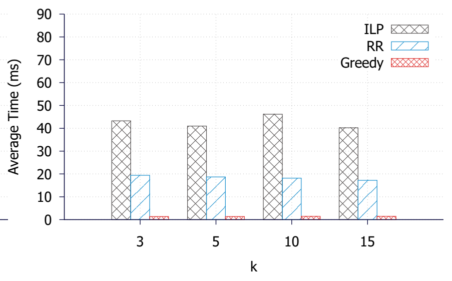
Figure 5: Evaluation of Top Pairs



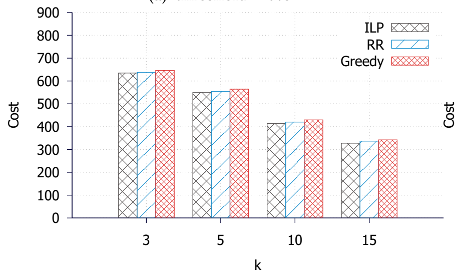
(a) threshold = 0.3



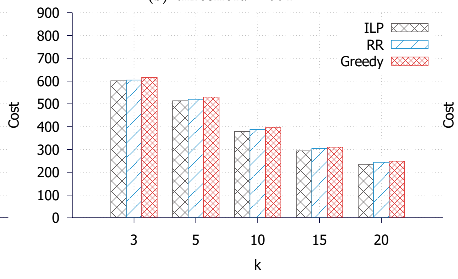
(b) threshold = 0.4



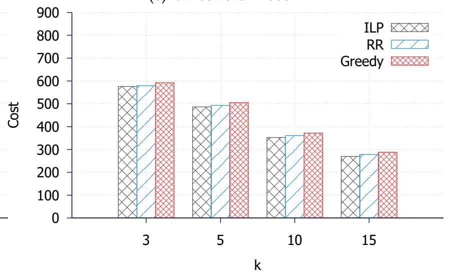
(c) threshold = 0.5



(d) threshold = 0.3



(e) threshold = 0.4

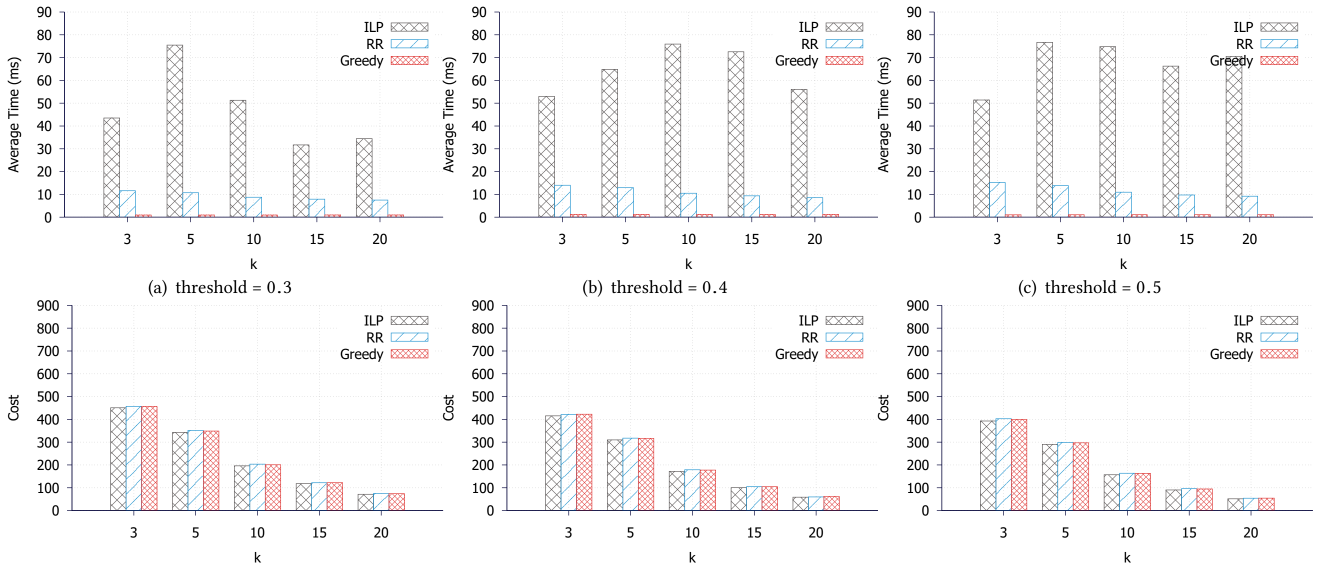


(f) threshold = 0.5

Figure 6: Evaluation of Top Sentences

library version 6.0.5 [10] with Dual-Simplex as the default method. This method is chosen because it shows the best performance in our case after experimental trials on different options available in Gurobi (primal simplex, barrier, auto-switching between methods, concurrent). All experiments were executed on a single machine with Intel Core i7-4790 3.60 GHz, 16 GB RAM, Windows 10 professional 64 bits. Our code was written in Java using the official Oracle Java version 8 update 45.

Our dataset consists of 68,686 patient reviews of the 1000 most reviewed doctors from *vitals.com*, which is a popular doctor rating website. The most commented doctor has 354 reviews while the least one has 43 reviews. On average, each doctor has 68 reviews which together contain 331 sentences. We ignore sentences and reviews with no extracted concepts (using MetaMap), which leaves an average of 44 reviews, 110.6 sentences, and 160.7 concept-sentiment pairs per doctor. The highest number of sentences per doctor that contain medical concepts is 1047 while the highest number of concept-sentiment pairs per doctor is 1793.



(a) threshold = 0.3 (b) threshold = 0.4 (c) threshold = 0.5

(d) threshold = 0.3 (e) threshold = 0.4 (f) threshold = 0.5

Figure 7: Evaluation of Top Reviews

Average Coverage Cost and Time of Algorithms We compare the average coverage cost (defined in Definition 2.2) and time of our 3 algorithms. They are evaluated on the problems of finding top pairs, sentences and reviews, in Figures 5, 6 and 7 respectively. These figures show the elapsed time (ms) and cost of the algorithms when the threshold on sentiment coverage definition is varied from 0.3 to 0.5. A key observation from these experiments is that Greedy is always the fastest algorithm while maintaining reasonable costs compared to ILP and RR algorithms. Of course, ILP gives optimal solution and always offers the cheapest cost. The Greedy algorithm has the worst cost but never more than 8% higher than the optimal (within 5% most of time). In terms of time, the Greedy algorithm outperforms ILP by a factor up to 19x, 32x and 63x in the top pairs, top sentences and top reviews problems, respectively. Similarly, Greedy runs faster than RR, at most 14 times, and usually takes only 1–2 ms per doctor. RR algorithm often works similarly to ILP regarding cost, specifically, the difference is about 1-2 percent. The speedup of RR over ILP is usually about 2–5x. This is because RR only solves a Linear Program system and then randomizes the solution instead of finding an optimal integer solution.

We also notice that with the same threshold, the cost decreases from top pairs to top sentences, and then to top reviews problem. The reason is that a sentence or review can have multiple pairs, so they typically cover more pairs than a single pair can cover. Therefore, k sentences or reviews usually cover more pairs than k pairs can, which leads to smaller costs. Similarly, the elapsed time of all algorithms for top sentences/reviews problem are larger than for top pairs problem. It's because for top sentences/reviews, there are more connections (edges) between selecting candidates and pairs to consider.

In general, the results suggest that our problem has latent structures friendly to Greedy algorithm. Therefore, the optimal solution from ILP algorithm seem to be close to the one of Greedy algorithm which can be achieved much faster. Because of this reason, we choose Greedy algorithm for the next qualitative experiments.

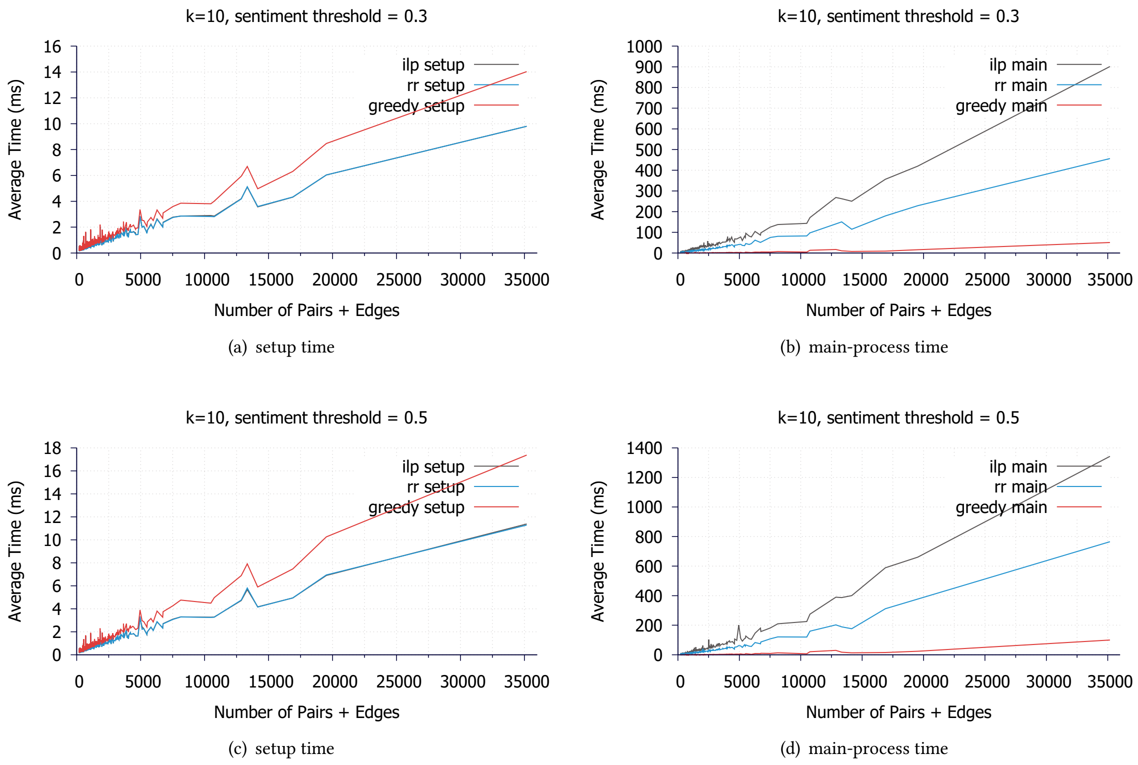
Algorithm's scalability In this experiment, we evaluate the algorithms' scalability over various problem sizes, that is, we vary the number of pairs plus the number of edges $(n + m)$. When there are multiple doctors with the same $(m + n)$, we report the average time of these doctors. Due to space limitation, we only present the results for the k-pairs coverage problem, and for the case of $k = 10, sentiment\_threshold(\epsilon) = 0.3$ or 0.5. It's also worth to note that for the other cases, algorithms follow similar trends.

In Section 5 we mentioned that the running time of the setup part is roughly linear on $|P| = n$ for our instances. This trend is observed in Figure 8(a), 8(c) since in our sparse graph instances the number of edges (m) is also roughly linear to the number of pairs (n). Note that for the Greedy algorithm we also count the time of initializing max-heap into setup time, thus it is a bit higher than the others. The main-process times are presented in Figure 8(b), 8(d). It's not surprising that Greedy always outperform the others significantly and ILP is the worst candidate. Even though we can get the worst case guarantee for ILP, LP solvers, ILP and Randomized rounding algorithms finish in reasonable time. For the Greedy algorithm the main process is dominated by the setup part. The results suggest that the algorithms can be applied to the real-life situations, especially the Greedy algorithm that not only have the poly-time upper bound in theory but also perform really well in practice.

6.2 Qualitative Evaluation

The goal of this section is to study the quality of the summarization achieved by the proposed algorithms, comparing them to a state-of-the-art baseline.

Baseline summarization method The baseline method being used to select top k sentences in this evaluation is adapted from [12].



(a) setup time (b) main-process time (c) setup time (d) main-process time

Figure 8: Scalability

The algorithm in [12] was designed to summarize customer reviews of online shopping products. It first extracts product features (attributes like "picture quality" for product "digital camera"), then classifies review sentences that mention these features as positive or negative, and finally sums up the number of positive and negative sentences for each feature. To have a fair comparison, we adapt their method to select top k sentences. We first count the number of pair (concept, positive) or (concept, negative), for example: feature "picture quality" with sentiment "positive" occurs in 200 sentences. Then, we select k most popular pairs and return one containing sentence for each selected pair. Note that the task of extracting product features is equivalent to identifying interesting medical concepts in our case, so their first two tasks are executed by re-using our work described in Section 3. From now on we refer to this adaptation as *baseline method*.

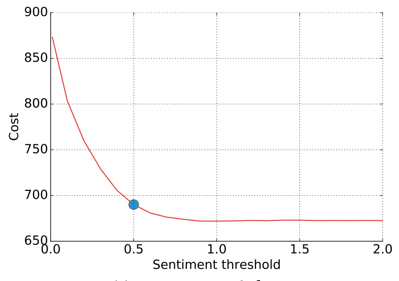
Selecting sentiment threshold ϵ Next, we discuss how to choose a good sentiment threshold ϵ, which determines when a sentence covers another sentence. Specifically, we study the threshold value ϵ for which the rate of covering sentences (or reviews) significantly drops if we further increase ϵ. That is, if we further increase the selected threshold, there is small impact on the coverage cost of the summarization algorithms. For that, we employ the elbow method.

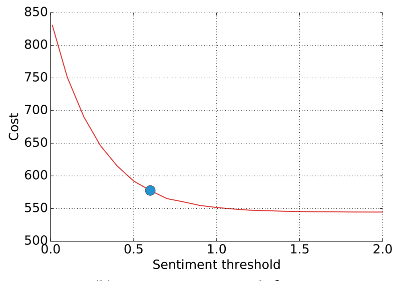
Clearly, there is a trade-off between the sentiment threshold and the coverage cost. A larger threshold means a bigger chance that a concept-sentiment pair can cover another, thus reducing the coverage cost (i.e. reducing the number of uncovered pairs). We empirically confirm this trade-off in Figure 9, which shows that there is an L-curve between sentiment threshold and cost. Hence, we can choose as sentiment threshold the elbow point of the curve.

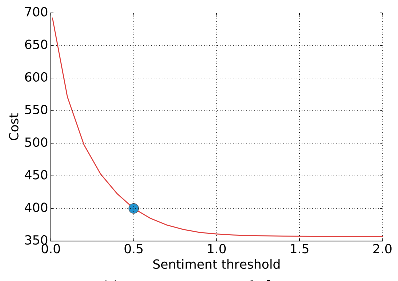
Figure 9 shows how the threshold is calculated using this method for our dataset, for $k = 3$ and 10. The sentiment threshold varies from 0 to 2 with step 0.1. Theoretically, for this kind of L-curve, the elbow point is where the second derivative of the curve's function is largest. However, since we don't know the curve's function, we use a common trick, where the elbow is the point on the curve with the largest distance to the line connecting the curve's start and end points. In our case, most of the time the elbow is close to sentiment threshold of 0.5. Intuitively, this sentiment threshold is also reasonable in the sense that a very positive sentiment of value 1.0 can cover a positive sentiment of value 0.5. Therefore, we choose sentiment threshold 0.5 in the user study.

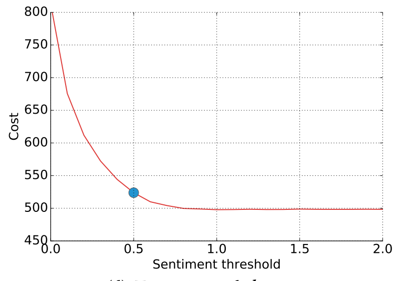
This threshold value also has another desirable property: it is larger than the average absolute error of our sentiment estimation (0.3 on sentiment range $[-1, 1]$, Section 3.2). Hence, it is less likely that the sentiments of two sentences cover each other merely due to the estimation error.

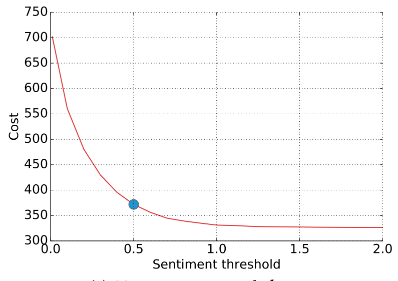
User Study During this user study, we ask users to indicate the semantic coverage between pairs of sentences found in doctor reviews. Using this information, we then measure the coverage cost of various summarization methods. The user study is taken by three graduate students.


(a) Top pairs with $k = 3$


(b) Top sentences with $k = 3$


(c) Top reviews with $k = 3$


(d) Top pairs with $k = 10$


(e) Top sentences with $k = 10$

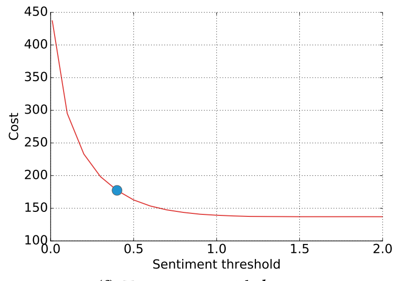

(f) Top reviews with $k = 10$

Figure 9: Cost – sentiment threshold trade-off. Elbow is the blue dot.

We ask each user to evaluate the sentences' coverage for the same 10 randomly selected doctors from our dataset, where for each doctor we select 3 random positive and 3 random negative reviews. These 6 reviews are broken down into sentences that are input to our Greedy algorithm (with sentiment threshold of 0.3) and the baseline method to select the top 3 sentences per method. We then take the union of the top-3 results of the two methods and ask the participants to judge if a selected sentence semantically covers another one from the full set of sentences. A typical task for a doctor is presented in Table 3, in which the first row shows 5 sentences selected by our method (Greedy) and the baseline (1 overlapping sentence). The "X"s are filled by the participant to express that the selected sentence of that column covers that row's sentence. As mentioned above, each row corresponds to one sentence of one of the doctor's reviews, and the columns are the sentences selected by one of the two summarization methods. Note that the sentence of the first column does not cover the sentence of second row because the former has very high sentiment and the latter is closer to neutral. The same explains why the fourth column review does not cover the first row review. The number of covered sentences of each method are averaged for all participants and is shown in Figure 10.

The result shows that our method outperforms the baseline in 8 cases, and is equivalent or worse in only 1 case. The number of covered sentences of our is 64% higher for doctor 784091. We further observe that for doctor 1088737 the number of covered sentences is much higher. The reason is that this doctor has longer reviews and hence more sentences (20 in total), compared to 1052723, which only has 10 sentences. The results for each participant are showed in Table 4.

Coverage Measures Besides the user survey, we also compare our method (greedy) with the baseline based on an intuitive coverage

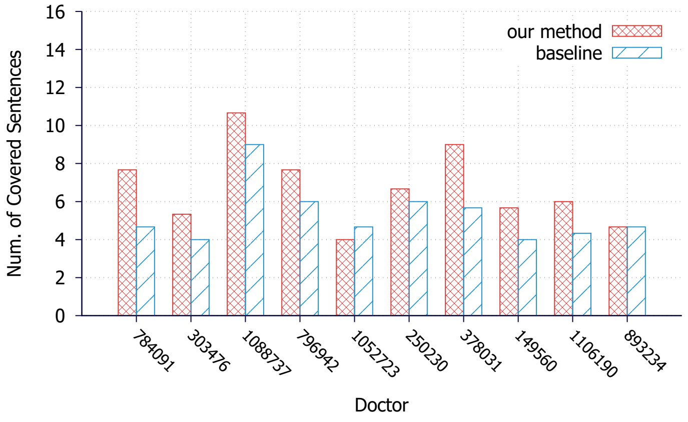


Figure 10: Comparison between greedy and baseline

measure that is different than the one proposed in Section 2, to avoid giving an unfair advantage to our method. Specifically, the measure is defined as the percentage of concept-sentiment pairs covered by that selected k sentences divided by the total number of pairs of a doctor. A sentence *covers* a pair p if the sentence contains at least one pair that covers p. A pair is said to cover another if their sentiment difference is less than or equal to a (sentiment) threshold and their dissimilarity is at most a (distant) threshold. In our experiment we use the path length between concepts of pairs in the ontology as dissimilarity measure. We evaluated this coverage measure for several distance and sentiment thresholds, but only show the results for distance threshold of 2, and sentiment threshold of 0.3, 0.4 and 0.5 due to the space limitation, in Figure 11. Consistently in both cases, our method outperforms the baseline about 10 − 30% per case. The coverage is slightly increased from

Table 3: A sample survey of a doctor. We ask participants to fill in cells with "X" to say that the column sentence can cover the row sentence. The medical concepts are underlined.

selected sentences / all sentences	He has showed our family what kind of great physician he is, and I have no worries of his capabilities to take care of my neck.	My back pain has now come back, so I called Dr XX because I wanted to see him, and he will not even see me.	He has completely gave me back the mobility of my legs when he corrected the Lumbar.	I am now having my neck fixed by him on Tues.	He removed 9 disc in a single operation.
I got to say when the Lord gave me someone to fix my neck, he gave me a wonderful Dr with his staff.	X				
I had the pleasure of meeting, Dr XX, when I was referred about my neck.				X	
…	…	…	…	…	

Table 4: Survey summary for each participant

Participant	our method is better	equivalent	worse
participant 1	7	2	1
participant 2	6	3	1
participant 3	7	1	2

Figure 11(a) to Figure 11(c) since a sentence can cover more pairs when we increase the sentiment threshold from 0.3 to 0.5.

7 RELATED WORK

Multi-document Summarization: this topic has been around for about two decades with the most well-known applications about summarizing online news, articles [7]. A typical technique is presented in [9], where the single-document summarization techniques are extended. A key difference is that there is more redundancy across documents of a similar topic than within a single document. This is an observation we also adopt in our work. In short, [9] constantly select a passage (normally sentence) that has the high "marginal relevance" score (combination of relevance and novelty) into the summary, and finally re-order candidates to maintain content cohesion. The other approaches based on cluster extraction such as MEAD summarizer [21] first extract common topics/clusters/centroids using respectively linguistic features or word statistics from input documents, then select one sentence per cluster into summary. However, none of above methods consider the sentiment sense in input documents. Our method is also different from them in the sense that we only care about interesting concepts by filtering their types (section 3.1).

Sentiment Analysis: this topic has been studied extensively in recent years, especially at the level of document . The methods fall into two categories, using unsupervised or supervised learning. The unsupervised methods such as [28] make use of linguistic rules to find opinion phrases containing adjectives or adverbs in document. [18] is a very first research to apply supervised learning methods to classify movie reviews as positive or negative based on vectors of reviews from Bag-Of-Word model. Recently, [15] use neural network model to extract the better review's vectors, thus get the better results on sentiment classification task. These two approaches are evaluated in our subsection 3.2.

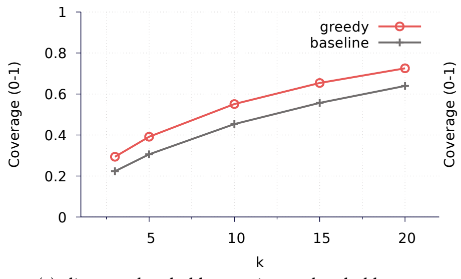
Feature Extraction: recent researches focus on online review analysis in along with the growth of the Internet, social media and online shopping. The common task is to extract the product features, attributes. [12] use association mining to find frequent features, then apply pruning rule to remove meaningless, redundant ones; later they also have a rule to discover additional infrequent features based on both frequent ones and opinion words. [20] propose a more advanced technique by estimating *Point-wise Mutual Information* (PMI) score between phrases using their Web search engine hit counts to find phrases relation. For example, they extract noun phrases in reviews, then evaluate each one by calculating their PMI score with product relation phrases such as "of scanner", "scanner has", "scanner comes with", …for Scanner class. In our method, this task is mainly based on Metamap [2].

Opinion Summarization: So far, the most popular approach is based on feature/aspect extraction. In [8, 12], authors extract product features from online customer reviews, then classify containing sentences as positive or negative, and report the number of positive/negative sentences for each feature. Different from this kind of statistical summaries, [14] formulates the problem as selecting k reviews that optimize the feature coverage while maintaining their proportion of opinions. Instead of sentences/reviews selection approaches, [5] propose another method for the reporting part by using language generated summary that use a set of templates varying on the feature's sentiments. Some publications also propose different ways of presenting opinion summaries such as showing aggregated rating accompanying with a representative phrase for each feature [17]. However, these projects do not consider the relationships between the features nor a continuous sentiment scale.

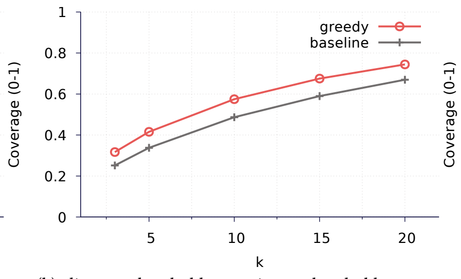
8 CONCLUSIONS AND FUTURE WORK

We introduced a novel review summarization problem that considers both the ontological relationships between the review concepts and their sentiments. We proposed a methods for extracting concepts and estimating their sentiment. We proved that the summarization problem is NP-hard even when the concept ontology is a DAG, and for that we presented efficient approximation algorithms We evaluated the proposed methods extensively with both quantitative and qualitative experiments. We found that the Greedy algorithm can achieve quality comparable to the optimal is much shorter time, comparing to other algorithms. Moreover, using a user study and a coverage measurement, we show that the Greedy outperforms a baseline method on selecting k sentences to summarize real reviews.

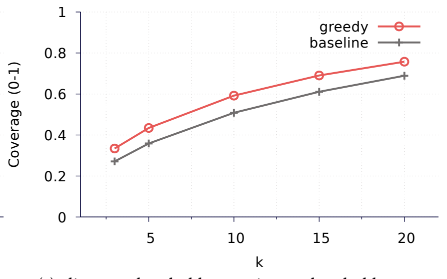
In the future, we will study more automated ways of mapping the reviews of any domain to concept-sentiment pairs. Further, we



(a) distance threshold 2, sentiment threshold 0.3



(b) distance threshold 2, sentiment threshold 0.4



(c) distance threshold 2, sentiment threshold 0.5

Figure 11: Comparison of Distance Threshold 2

will study what are the best user interfaces to present the generated summaries. For instance, should they be presented as a coherent text or as a table?

ACKNOWLEDGMENT

This project was partially supported by National Science Foundation (NSF) grants IIS-1216007 and IIS-1447826. We also thank Matthew Wiley for his help with the MetaMap setup.

REFERENCES

[1] 2016. SNOMED CT. (2016). https://www.nlm.nih.gov/research/umls/Snomed/snomed_main.html
[2] Alan R Aronson. 2001. Effective mapping of biomedical text to the UMLS Metathesaurus: the MetaMap program.. In *Proceedings of the AMIA Symposium*. American Medical Informatics Association, 17.
[3] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining.. In *LREC*, Vol. 10. 2200–2204.
[4] Olivier Bodenreider. 2004. The unified medical language system (UMLS): integrating biomedical terminology. *Nucleic acids research* 32, suppl 1 (2004), D267–D270.
[5] Giuseppe Carenini, Jackie Chi Kit Cheung, and Adam Pauls. 2013. MULTI-DOCUMENT SUMMARIZATION OF EVALUATIVE TEXT. *Computational Intelligence* 29, 4 (2013), 545–576.
[6] Marek Chrobak, Claire Kenyon, John Noga, and Neal E Young. 2006. Oblivious medians via online bidding. In *LATIN 2006: Theoretical Informatics*. Springer, 311–322.
[7] Dipanjan Das and André FT Martins. 2007. A survey on automatic text summarization. *Literature Survey for the Language and Statistics II course at CMU* 4 (2007), 192–195.
[8] Xiaowen Ding, Bing Liu, and Philip S Yu. 2008. A holistic lexicon-based approach to opinion mining. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*. ACM, 231–240.
[9] Jade Goldstein, Vibhu Mittal, Jaime Carbonell, and Mark Kantrowitz. 2000. Multi-document summarization by sentence extraction. In *Proceedings of the 2000 NAACL-ANLPWorkshop on Automatic summarization-Volume 4*. Association for Computational Linguistics, 40–48.
[10] Inc. Gurobi Optimization. 2015. Gurobi Optimizer Reference Manual. (2015). http://www.gurobi.com
[11] Arthur E Hoerl and Robert W Kennard. 1970. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics* 12, 1 (1970), 55–67.
[12] Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 168–177.
[13] Klaus Krippendorff. 2012. *Content analysis: An introduction to its methodology*. Sage.
[14] Theodoros Lappas, Mark Crovella, and Evimaria Terzi. 2012. Selecting a characteristic set of reviews. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 832–840.
[15] Quoc V Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents.. In *ICML*, Vol. 14. 1188–1196.
[16] Hugo Liu and Push Singh. 2004. ConceptNet - a practical commonsense reasoning tool-kit. *BT technology journal* 22, 4 (2004), 211–226.
[17] Yue Lu, ChengXiang Zhai, and Neel Sundaresan. 2009. Rated aspect summarization of short comments. In *Proceedings of the 18th international conference on World wide web*. ACM, 131–140.
[18] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*. Association for Computational Linguistics, 79–86.
[19] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.
[20] Ana-Maria Popescu and Orena Etzioni. 2007. Extracting product features and opinions from reviews. In *Natural language processing and text mining*. Springer, 9–28.
[21] Dragomir R Radev, Hongyan Jing, and Malgorzata Budzikowska. 2000. Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. In *Proceedings of the 2000 NAACL-ANLP Workshop on Automatic summarization*. Association for Computational Linguistics, 21–30.
[22] Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, 45–50. http://is.muni.cz/publication/884893/en
[23] Guergana K Savova, James J Masanz, Philip V Ogren, Jiaping Zheng, Sunghwan Sohn, Karin C Kipper-Schuler, and Christopher G Chute. 2010. Mayo clinical Text Analysis and Knowledge Extraction System (cTAKES): architecture, component evaluation and applications. *Journal of the American Medical Informatics Association* 17, 5 (2010), 507–513.
[24] Alex Smola and Vladimir Vapnik. 1997. Support vector regression machines. *Advances in neural information processing systems* 9 (1997), 155–161.
[25] Robert Tibshirani. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* (1996), 267–288.
[26] Ivan Titov and Ryan McDonald. 2008. Modeling online reviews with multi-grain topic models. In *Proceedings of the 17th international conference on World Wide Web*. ACM, 111–120.
[27] Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*. Association for Computational Linguistics, 173–180.
[28] Peter D Turney. 2002. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 417–424.
[29] Matthew T Wiley, Canghong Jin, Vagelis Hristidis, and Kevin M Esterling. 2014. Pharmaceutical drugs chatter on online social networks. *Journal of biomedical informatics* 49 (2014), 245–254.
[30] Laurence A Wolsey. 1982. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica* 2, 4 (1982), 385–393.
[31] Neal E Young. 2002. K-medians, facility location, and the Chernoff-Wald bound. *arXiv preprint cs/0205047* (2002).