Content Coverage Maximization on Word Networks for Hierarchical Topic Summarization

Chi Wang
Department of Computer Science
University of Illinois at Urbana-Champaign
Champaign, IL, USA
chiwang1@illinois.edu

Xiao Yu
Department of Computer Science
University of Illinois at Urbana-Champaign
Champaign, IL, USA
xiaoyu1@illinois.edu

Yanen Li
Department of Computer Science
University of Illinois at Urbana-Champaign
Champaign, IL, USA
yanenli2@illinois.edu

Chengxiang Zhai
Department of Computer Science
University of Illinois at Urbana-Champaign
Champaign, IL, USA
czhai@illinois.edu

Jiawei Han
Department of Computer Science
University of Illinois at Urbana-Champaign
Champaign, IL, USA
hanj@illinois.edu

ABSTRACT

This paper studies text summarization by extracting hierarchical topics from a given collection of documents. We propose a new approach of text modeling via network analysis. We convert documents into a word influence network, and find the words summarizing the major topics with an efficient influence maximization algorithm. Besides, the influence capability of the topic words on other words in the network reveal the relations among the topic words. Then we cluster the words and build hierarchies for the topics. Experiments on large collections of Web documents show that a simple method based on the influence analysis is effective, compared with existing generative topic modeling and random walk based ranking.

Categories and Subject Descriptors

I.7 [**Computing Methodologies**]: Document and Text Processing; H.2.8 [**Database Applications**]: Data Mining

General Terms

Algorithms, Experimentation

Keywords

Information Coverage, Topic Hierarchy, Keyword Extraction, Text Summarization

CIKM'13, Oct. 27–Nov. 1, 2013, San Francisco, CA, USA.

ACM 978-1-4503-2263-8/13/10
http://dx.doi.org/10.1145/2505515.2505585 ...$15.00.

1. INTRODUCTION

Text summarization encompasses an important category of text mining problems. It refers to the automatic creation of a compressed version of a given text that provides useful information for the user. Most work in this area focuses on creating summary for a single document or a set of documents with the same topic. We study the summarization task for a collection of documents with arbitrary topics. On contrary to the sentence-level summarization on single document or multiple related documents, we summarize the collection with keywords, and further study their relatedness to discover the hierarchical topics of the given text.

Topic modeling is another important category of text mining problems. Most of existing topic modeling techniques, originating from Probabilistic Latent Semantic Indexing [10] and Latent Dirichlet Allocation [1], are based on the probabilistic generative process of text. They associate a document with soft clusters according to their topics, and find topic words that best represent each topic. In our work, we propose an alternative framework of discovering topics from text. We first extract keywords towards the goal of text summarization, and then construct hierarchical topics from these keywords based on their role and relation in the summary. While traditional topic modeling does not have the explicit goal of optimizing the conciseness and content coverage, we target using a small number of words to cover the major topics — as required by the summarization task.

Motivated by the interestingness of topic modeling and the need of concise summarization, this paper explores a new problem lying in the intersection of text summarization and topic modeling — we try to summarize a collection of documents with hierarchical topics, which comprise a small number of keywords covering the major content of the collection. Solving this problem allows us to have a general idea of the major topics and minor topics in this collection, summarize them in a taxonomy, and further categorize the documents and index them. This also benefits a lot of applications such as automatic construction of concept hierarchy for domain knowledge, with huge available text collections on the Web such as scientific papers and news reports. For documents with mixed topics, this allows hierarchical browsing of the major topics; for

documents with a relatively pure topic, we can find the most representative words and explore their semantic relations within this topic.

Recently, robust information network analysis techniques have been shown appealing to research in text summarization, *e.g.*, in multi-document extractive generic text summarization [7], single document summarization and keyphrase extraction [28]. Words and sentences can be modeled as nodes linked by their co-occurrence or content similarity. And then a variety of network analysis approaches can be applied. For text summarization, representative words and sentences can be found with certain assumptions about their properties in the network. In our case, by compressing text information into the word network, we do not need to scan every original single document. The complexity of mining the word network only depends on the scale of the vocabulary which can be small with the help of term filtering. That enables handling larger scale of document collections than mining directly from the documents, and provides a practical motivation for doing the summarization-oriented topic modeling. However, most network-based studies employ the idea of random walk to do prestige ranking as PageRank [2] does. Though such a ranking model captures some dependency between the nodes, it does not model the objective of maximizing the coverage on the content with a small number of words or sentences. Some issues such as redundancy removal are not inherently captured by importance ranking. That motivates us to find better methods.

We adopt the idea of network-based analysis, but propose a new summarization framework for extracting keywords from a document collection and organizing them in meaningful topic hierarchies. Rather than heuristically assessing word centrality, we explicitly model the information coverage with an influence propagation model. The text collection is modeled as a network with nodes representing words and links representing relatedness between two words. Influence is propagated in the network according to a stochastic cascade model. Under the influence cascade model, the expected number of nodes influenced by a set of k nodes (referred to as *seeds*) indicates the content coverage of the k seeds. We want to find a small set of seeds that maximize the content coverage, and build the relationship among the seeds by analyzing the words influenced by them.

The research questions we would like to study are: 1) Can we use the coverage maximization model to find summary words; 2) What is a good influence propagation model and how does it compare to other network analysis model such as random walk and 3) How to discover hierarchical topics from the extracted summary words, and how is it compared to a generative topic modeling method.

Our contribution. We first propose the idea of using an influence propagation model to model the content coverage. Then we define a meaningful word network where the relation of word A influencing word B corresponds to meaningful semantic relations of A and B. Based on an approximation algorithm with theoretical error bound, we develop a new algorithm that can efficiently discover the summary words and build topic hierarchy through influence analysis. The input and output are illustrated in Figure 1. We compare our approach with both the PageRank-alike approach and statistical topic modeling. The experiment results show that the algorithm is not only scalable but also achieved higher accuracy in both summary words selection and topic hierarchy construction.

The rest of this paper is organized as follows: We first give a review of the related literature in Section 2, followed by some background introduction of our techniques in Section 3. Then we present our framework and methodology in Section 4 and Section 5, about influence network modeling of text and the topic organization respectively. Section 6 demonstrates our performance on two collections of documents, and Section 7 concludes.

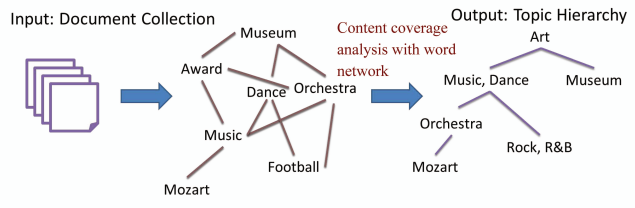


Figure 1: An illustration of our input and output

2. RELATED WORK

Erkan and Radev [7] propose LexRank, a variation of RageRank [2], to solve extractive generic multiple document summarization, and show it achieves better performance than centroid-based method [21]. Mihalcea and Tarau [16] and Wan *et al.* [29] also use PageRank to do single document summarization and keyword extraction. Their work shows how to choose the nodes and links, and assign transition probability along each link. Different choices are adopted for different applications. A common strategy is to rank the nodes, either sentences or words, and select them greedily with some reranking technique such as Maximal Marginal Relevance (MMR) [3], or Cross-Sentence Information Subsumption [20].

In these graph-based ranking methods, the ranking and the redundancy removal are done separately, and the objective is not maximizing the information coverage. Influence maximization, while having not been applied as widely as PageRank, captures both the centrality and the coverage and better fits the text summarization need. Domingos and Richardson [6, 22] are the first to study influence maximization as an algorithmic problem. Their methods are probabilistic. Kempe, Kleinberg, and Tardos [11] are the first to formulate the problem as a discrete optimization problem, and they present a greedy approximation algorithm with theoretical guarantee. A major drawback of their work is the scalability of their greedy algorithm. Several recent studies aim at addressing this issue[12, 13, 4, 30]. It is not quite clear what is a good assumption on the influence probability as well as the network structure, though there is some work trying to infer those from observations [9, 24], and some other work trying to analyze it by inducting from intuitive features and constraints [25]. To the best of our knowledge, it is not explored how to apply influence maximization in other data mining problems, and how it is different with Random Walk model in such applications.

There are some studies using existing taxonomy to classify documents, or organize information into an existing taxonomy such as Open Directory Project (ODP, www.dmoz.org), WordNet and Wikipedia [8, 23, 19]. Our methodology is different as we do not refer to any existing, manually edited taxonomies. We attempt to automatically find the specific topic hierarchy for given document collections instead.

To find relations among summary words from a set of documents and form them in a topic hierarchy, stochastic topic modeling can be applied. Latent Dirichlet allocation (LDA) [1] is a widely-used topic model, and the basis for many variants. Teh *et al.* [26] proposes hierarchical Dirichlet processes (HDP) to model topics that have a hierarchical structure. Pachinko allocation models [14, 17] improve it by allowing children topics shared by different parent topics. However, the summarization need of finding a small set of keywords in these studies is not considered.

Recently, Wang *et al.* [31] propose to construct topic hierarchy from titles, not generic documents. Chuang and Chien [5] and Liu *et al.* [15] generate taxonomy of given keyword phrases by hierarchical clustering techniques, with the help of knowledge bases and search engine. These studies do not consider the summarization need. We are the first to study the topical structure discovery towards the summarization goal, *i.e.*, take a set of documents as input, summarize them with a small number of words, find relations among summary words and organize them in a topic hierarchy.

3. FRAMEWORK AND PREREQUISITES

Our summarization framework has two stages. The first stage extracts keywords from the document collection, and the second stage studies the relation among the keywords and organize them in hierarchical topics. We propose to use network analysis techniques for these two stages. Therefore, we briefly introduce two relevant network analysis techniques in this section.

3.1 Graph-based ranking for text data

Graph-based ranking methods have been proposed for document summarization and keyword extraction tasks [7, 16, 29]. These methods are inspired by PageRank [2], originally used for web page ranking based on their link structure. The assumption is that an important page is linked to many other important pages. In analogy, researchers construct word network (sentence network, resp.) for text data and assume the more important words (sentences, resp.) are linked to many other important words (sentences, resp.). The common abstract model for these studies is random Walk on weighted graphs: an imaginary walker starts from a random node, chooses a neighbor with a probability proportional to the link weight, and then walks from that neighbor. This random process eventually produces the probability of arriving at each node on the graph. The probability of arriving at each node is then used as an indication of the popularity, centrality, or importance of that node. This is the spirit of most graph-based ranking methods for text data. The difference is whether they use a node to represent a word or a sentence, and how they define the transition probability from one node to another, according to various text features like similarity and cooccurrences.

After computing the ranking score, one can retrieve the top ranked words or sentences as the output. However, many researchers have found that this naive strategy may result in redundant information because the top ranked words or sentences can be similar and have overlapped meaning. They applied reranking techniques such as maximal marginal relevance (MMR) principle to ameliorate that problem [3]. For example, suppose the ranking score is r_i for node i, one can perform the following reranking strategy: the top ranked node is retrieved one by one, and once a node r_j is retrieved, the ranking score of all the other nodes is discounted to be $r'_i = r_i - w_{ij}r_j$, where w_{ij} is the transition probability from node i to j. The reranked result may contain less redundancy than original results. Although the reranking technique provides a compromise between the random walk-based ranking and the information redundancy, it is not a principled solution to the content coverage maximization— the content coverage is not modeled in the random walk process, and the reranking does not guarantee the coverage is maximized either.

3.2 Influence maximization

Due the discussed issue above, we resort to another network analysis technique in social network study, namely influence maximization, as we find its objective is more analogous to ours. In this section, we briefly introduce the influence maximization problem in social network analysis. In next section we propose our model for the text summarization.

A social network is modeled as a graph $G = (V, E)$ with node sets V representing individuals and edge sets E representing connections or relationship between two individuals. Influence is propagated in the network according to a stochastic cascade model. One popular model is independent cascade (IC) model: Each edge (u, v) in the graph is associated with a *propagation probability* $pp(u, v)$, which is the probability that node u independently activates (a.k.a. influences) node v at step $t + 1$ if u is activated at step t. Given a seed set $S \subseteq V$, the independent cascade (IC) model works as follows. Let $S_t \subseteq V$ be the set of nodes that are activated at step $t \geq 0$, with $S_0 = S$. At step $t + 1$, every node $u \in S_t$ may activate its out-neighbors $v \in V \setminus \cup_{0 \leq i \leq t} S_i$ with an independent probability of $pp(u, v)$. The process ends at a step t with $S_t = \emptyset$. Note that each activated node only has one chance to activate its out-neighbors at the step right after itself is activated, and each node stays as an activated node after it is activated. The *influence spread* of S, which is the expected number of activated nodes given seed set S, is denoted as $\sigma_I(S)$.

Given an input k, the influence maximization problem under an influence propagation model is to find a subset $S^* \subseteq V$ such that $|S^*| = k$ and $\sigma_I(S^*) = \max\{\sigma_I(S) \mid |S| = k, S \subseteq V\}$. It is shown in [11] that for IC model, this problem is NP-hard, but a constant-ratio approximation algorithm is available. One important issue, however, is that there is no efficient way to compute $\sigma_I(S)$ given a set S. Wang *et al.* [30] proves that the computation is actually #P-hard. It follows that an efficient approximation algorithm is open to question, according to [27].

4. MODELING TEXT WITH INFLUENCE NETWORK

We choose to adopt the idea of influence maximization in our task because its principle suits our need of summarizing text with good coverage better than the random walk idea. The goal of social influence maximization is analogous to content coverage maximization, as we show in the following model.

Given a set of documents, after removing stopwords, we construct a network for the whole collection, where each node represents one word, and each link connects two words that ever cooccur in some document. The propagation probability from word x to word y can be explained as if we see word x, how likely we receive the meaning of y without seeing y. In this sense once a word is 'activated', its meaning is regarded to be covered by the initial set of seeds which correspond to the summary words we select. To obtain a good summary, we want to find a small set of words that cover the largest number of words in expectation.

4.1 Word influence network construction

One important question is how to define the influence probability to capture the meaningful semantic relations of two cooccured words x and y. In this paper we consider three alternative definitions for the influence probability from y to x:

i) the conditional probability of $x \in d$ given $y \in d$ in an arbitrary document d,

$$p(x \in d | y \in d)$$

ii) the conditional probability of $y \in d$ given $x \in d$,

$$p(y \in d | x \in d)$$

iii) the mutual information between $x \in d$ and $y \in d$,

$$I(x,y) = p(x \in d, y \in d) \log \frac{p(x \in d, y \in d)}{p(x \in d)p(y \in d)} + p(x \in d, y \notin d) \log \frac{p(x \in d, y \notin d)}{p(x \in d)p(y \notin d)} + p(x \notin d, y \in d) \log \frac{p(x \notin d, y \in d)}{p(x \notin d)p(y \in d)} + p(x \notin d, y \notin d) \log \frac{p(x \notin d, y \notin d)}{p(x \notin d)p(y \notin d)}$$

The first two definitions are natural candidates according to our influence propagation model. However, they have obvious biases: definition i) favors highly frequent words, or stopwords as seeds because these words cooccur with other words a lot. Likewise, definition 2) favors rare words. Mutual information is a better measure to characterize the mutual dependence of two words. However, our formulation requires that the influence probability in the range of [0,1]. Also, a node should have probability 1 of activating itself according the definition. Therefore, we normalize the mutual information as follows.

$$pp(x,y) = \frac{I(x,y)}{I(x,x)} \tag{4.1}$$

This normalization ensures that a node activates itself with probability 1. We use Eq. (4.1) as our influence probablity definition to construct the influence network.

4.2 Efficient coverage maximization

The scale of the word network can exceed what algorithms for IC model can efficiently handle. We resort to alternative models that approximate IC model. The basic *maximum influence arborescence* (MIA) model, and its variance *prefix excluding MIA* (PMIA) model in [30] are both shown to have scalable approximation algorithms and achieve a near-optimal solution for IC model as well. Here we take the basic MIA model as an example to illustrate.

The basic idea is to restrict the range and path of influence propagation to reduce the computation for the neglectable influence. For a path $P = \langle u = p_1, p_2, \ldots, p_m = v \rangle$, we define the *propagation probability* of the path, $pp(P)$, as

$$pp(P) = \prod_{i=1}^{m-1} pp(p_i, p_{i+1}).$$

Intuitively the probability that u activates v through path P is $pp(P)$, because it needs to activate all nodes along the path. To approximate the actual expected coverage within the word network, we can use the *maximum influence path* (MIP) to estimate the influence from one node to another. Let $\mathcal{P}(G,u,v)$ denote the set of all paths from u to v in a graph G. We define the maximum influence path $MIP_G(u,v)$ from u to v in G as

$$MIP_G(u,v) = \arg\max_P \{pp(P) \mid P \in \mathcal{P}(G,u,v)\}.$$

Note that for each edge (u,v) in the graph, if we translate the propagation probability $pp(u,v)$ to a distance weight $-\log pp(u,v)$ on the edge, then $MIP_G(u,v)$ is simply the shortest path from u to v in the weighted graph G. Therefore, the maximum influence paths directly correspond to shortest paths, and thus they permit efficient algorithms to compute them. For a given node v in the graph, the union of the maximum influence paths to v, form the *maximum influence in-arborescence.*

Algorithm 1: $ap(u,S,\theta)$

1: **if** $u \in S$ **then**
2: $\quad ap(u,S,\theta) = 1$
3: **else if** $N^{in}(u) = \emptyset$ **then**
4: $\quad ap(u,S,\theta) = 0$
5: **else**
6: $\quad ap(u,S,\theta) = 1 - \Pi_{w \in N^{in}(u)}(1 - ap(w,S,\theta) \cdot pp(w,u))$
7: **end if**

To further prune the neglectable influence, we use an *influence threshold* θ to eliminate MIPs that have too small propagation probabilities. The *activation probability* of any node v, denoted as $ap(v,S,\theta)$, is defined to be the probability that v is activated by seed set S via MIPs with propagation probabilities larger θ. $ap(u,S,\theta)$ can be computed efficiently when S and θ are given. First, we use Dijkstra algorithm to compute the *maximum influence in-arborescence* (*MIIA*) which is the union of all the maximum influence paths to v.

$$MIIA(v,\theta) = \cup_{u \in V, pp(MIP_G(u,v)) \geq \theta} MIP_G(u,v)$$

Intuitively, $MIIA(v,\theta)$ give the local influence regions of words that can cover the meaning of v, and different values of θ control the size of these local influence regions. Second, we recursively traverse the arborescence $MIIA(v,\theta)$ to compute the restrictive activation probability $ap(u,S,\theta)$ for every node u in it: $ap(u,S,\theta) = 1$ if $u \in S$, and $ap(u,S,\theta) = 1 - \Pi_{w \in N^{in}(u)}(1 - ap(w,S,\theta) \cdot pp(w,u))$ if $u \notin S$, where $N^{in}(u)$ is the set of in-neighbors of u in $MIIA(v,\theta)$. The algorithm is outlined in Algorithm 1.

Let $\sigma_\theta(S)$ denote the coverage of S in MIA model with influence threshold θ, then we have:

$$\sigma_\theta(S) = \sum_{v \in V} ap(v,S,\theta). \tag{4.2}$$

due to the linearity of the expectation over the sum of random variables.

We are interested in finding a set of seeds S of size k such that $\sigma_\theta(S)$ is maximized. It is not surprising that this optimization problem is NP-hard [30]. Furthermore, it is NP-hard to approximate within a factor of $1 - 1/e + \epsilon$ for any $\epsilon > 0$. However, function σ_θ is submodular and monotone and $\sigma_\theta(\emptyset) = 0$. We say that a non-negative real valued function f on subsets of V is *submodular* if $f(S \cup \{v\}) - f(S) \geq f(T \cup \{v\}) - f(T)$, for all $v \in V$ and all pairs of subsets S and T with $S \subseteq T \subseteq V$. Intuitively, this means that f has diminishing marginal return. Moreover, we say that f is *monotone* if $f(S) \leq f(T)$ for all $S \subseteq T$. By a theorem in [18], for any submodular and monotone function f with $f(\emptyset) = 0$, the problem of finding a set S of size k that maximizes $f(S)$ can be approximated by a simple greedy algorithm shown as Algorithm 2. The greedy strategy is iteratively selecting new seed u that maximizes the incremental change of f into the seed set S until k seeds are selected. It achieves $1 - 1/e$ approximation ratio for the coverage maximization problem in the MIA model.

Computing $\sigma_\theta(S)$ using Eq. (4.2) and Algorithm 1 is polynomial-time. Together with Algorithm 2, we already have a polynomial-time approximation algorithm. It can be optimized to near-linear. Consider the maximum influence in-arborescence $MIIA(v,\theta)$ of size s and a given seed set S. To select the next seed u, we need to compute the activation probability $ap(v, S \cup \{w\}, MIIA(v,\theta))$ for every $w \in MIIA(v,\theta)$, which takes $O(s^2)$ time if we simply use Algorithm 1 to compute every $ap(v, S \cup \{w\}, \theta)$. We now

Algorithm 2: Greedy(k, f)

1: initialize $S = \emptyset$
2: **for** $i = 1$ to k **do**
3: select $u = \arg\max_{w \in V \setminus S}(f(S \cup \{w\}) - f(S))$
4: $S = S \cup \{u\}$
5: **end for**
6: output S

show a batch update scheme such that we could compute $ap(v, S \cup \{w\}, \theta)$'s for all $w \in MIIA(v, \theta)$ in $O(s)$ time.

To do so, we utilize the linear relationship between $ap(u, S, \theta)$ and $ap(v, S, \theta)$ in $MIIA(v, \theta)$, as shown by the following lemma, which is not difficult to derive from line 6 of Algorithm 1.

LEMMA 1 (INFLUENCE LINEARITY). *Consider $MIIA(v, \theta)$ and a node u in it. If we treat the activation probability $ap(u, S, \theta)$ as an independent variable, $ap(v, S, \theta)$ as a dependent variable, and other $ap(w, S, \theta)$'s as constants for all w's not on the path from u to v in $MIIA(v, \theta)$, then $ap(v, S, \theta) = \alpha(v, u) \cdot ap(u, S, \theta) + \beta(v, u)$, where $\alpha(v, u), \beta(v, u)$ are constants independent of $ap(u, S, \theta)$.*

Algorithm 3: Compute $\alpha(v, u)$ given $MIIA(v, \theta)$ and S, after $ap(u, S, \theta)$ for all u in $MIIA(v, \theta)$ are known.

1: /* the following is computed recursively */
2: **if** $u = v$ **then**
3: $\alpha(v, u) = 1$
4: **else**
5: set w to be the out-neighbor of u
6: **if** $w \in S$ **then**
7: $\alpha(v, u) = 0$ /* u's influence to v is blocked by seed w */
8: **else**
9: $\alpha(v, u) = \alpha(v, w) \cdot pp(u, w) \cdot \Pi_{u' \in N^{in}(w) \setminus \{u\}}(1 - ap(u', S, \theta) \cdot pp(u', w))$
10: **end if**
11: **end if**

Based on the recursive computation of $ap(u, S, \theta)$ as shown in line 6 of Algorithm 1, it is straightforward to derive a recursive computation of $\alpha(v, u)$, as shown in Algorithm 3. To see the intuition behind the equations, we remind that $\alpha(v, u)$ is the increment of the activation probability of v caused by unit increment of u's activation probability. Therefore, the boundary of the recursive computation is natural: when $u = v$, unit increment of u's activation probability results in the same increment of v's activation probability; when u's out-neighbor w is a seed node, the increment of its activation probability does not induce any change of v's activation probability because w is already activated. In other cases, the effect of u on v is determined by the effect of its out-neighbor w on v and the chance that w is activated by its in-neighbor u exclusively. The product over w's in-neighbors except for u in line 9 corresponds to the probability that w is not activated by other in-neighbors. Note that Algorithm 3 can be transformed into an iterative form such that all $\alpha(v, u)$'s can be computed by one traverse of $MIIA(v, \theta)$ from the root to the leaves.

Computing the linear coefficients $\alpha(v, u)$ as defined in Lemma 1 is crucial in computing the incremental coverage of a node u. Let us consider again the maximum influence in-arborescence $MIIA(v, \theta)$ of size s and a given seed set S. For any $w \in MIIA(v, \theta)$, if we select w as the next seed, its $ap(w)$ increases from the current value

Algorithm 4: MIA(G, k, θ)

1: /* initialization */
2: set $S = \emptyset$
3: set $IncCov(v) = 0$ for each node $v \in V$
4: **for** each node $v \in V$ **do**
5: compute $MIIA(v, \theta)$
6: set $ap(u, S, \theta) = 0, \forall u \in MIIA(v, \theta)$ /* since $S = \emptyset$ */
7: compute $\alpha(v, u), \forall u \in MIIA(v, \theta)$ (Algorithm 3)
8: **for** each node $u \in MIIA(v, \theta)$ **do**
9: $IncCov(u) \mathrel{+}= \alpha(v, u) \cdot (1 - ap(u, S, \theta))$
10: **end for**
11: **end for**
12: /* main loop */
13: **for** $i = 1$ to k **do**
14: pick $u = \arg\max_{v \in V \setminus S}\{IncCov(v)\}$
15: /* update incremental content coverage*/
16: **for** $v \notin S$ such that $u \in MIIA(v, \theta)$ **do**
17: /* subtract previous incremental coverage */
18: **for** $w \in MIIA(v, \theta) \setminus S$ **do**
19: $IncCov(w) \mathrel{-}= \alpha(v, w) \cdot (1 - ap(w, S, \theta))$
20: **end for**
21: **end for**
22: $S = S \cup \{u\}$
23: **for** $v \in MIOA(u, \theta) \setminus S$ **do**
24: compute $ap(w, S, \theta), \forall w \in MIIA(v, \theta)$ (Algo. 1)
25: compute $\alpha(v, w), \forall w \in MIIA(v, \theta)$ (Algo. 3)
26: /* add new incremental coverage */
27: **for** $w \in MIIA(v, \theta) \setminus S$ **do**
28: $IncCov(w) \mathrel{+}= \alpha(v, w) \cdot (1 - ap(w, S, \theta))$
29: **end for**
30: **end for**
31: **end for**
32: **return** S

to 1. Since $ap(w)$ and $ap(v)$ has a linear relationship with the linear coefficient $\alpha(v, w)$, the incremental coverage of w on v is given by $\alpha(v, w) \cdot (1 - ap(w))$. Therefore, we only need one pass of $MIIA(v, \theta)$ to compute $ap(w)$'s for all $w \in MIIA(v, \theta)$, and a second pass of $MIIA(v, \theta)$ to compute $\alpha(v, w)$'s and $\alpha(v, w) \cdot (1 - ap(w))$'s for all $w \in MIIA(v, \theta)$. This reduces the running time of computing incremental coverage of all nodes in $MIIA(v, \theta)$ from $O(s^2)$ to $O(s)$.

The complete greedy algorithm for the basic MIA model is presented in Algorithm 4. Lines (2–11) evaluate the incremental content coverage $IncCov(u)$ for any node u when the current seed set is empty. The evaluation is exactly as we described above using the linear coefficients $\alpha(v, u)$. Lines (15–30) update the incremental coverage whenever a new seed is selected in line 14. Suppose u is selected as the new seed in an iteration. The influence of u in the MIA model only reaches nodes that contain u in their maximal influence in-arborescences. Thus the incremental influence spread $IncCov(w)$ for some w needs to be updated if and only if w is in $MIIA(v, \theta)$ for some v containing u in its MIIA. This means that the update process is relatively local to u.

Time and space complexity. Let $s_\theta = \max_{v \in V}\{|MIIA(v, \theta)|\}$. The total running time of the algorithm is $O(|V| s_\theta + k s_\theta^2 \log |V|))$. For every node $v \in V$, the algorithm stores $MIIA(v, \theta)$, and for every $u \in MIIA(v, \theta)$, $ap(u, S, \theta)$ and $\alpha(v, u)$ are stored (note that $ap(u, S, \theta)$ can reuse the same entry for different seed set S). We also need a max-heap to store and update $IncCov(v)$ for all $v \in V$. In total, the space complexity of the algorithm is $O(s_\theta |V|)$.

5. TOPIC ORGANIZATION

Based on the above influence network model, we can find the words that cover the major topics in a document collection. The next question is how to find the relation among these words and organize them into hierarchical topics. We still approach this problem with coverage analysis.

We reexamine the coverage maximization algorithm in Section 4.2. The only important step in the greedy algorithm is to select the next seed that gives the largest incremental coverage spread. Suppose the current seed set is S. To select the next seed u, we need to evaluate the incremental coverage of each $w \in MIIA(v, \theta)$ for every word u. As we mentioned in Section 4.2, there is a linear relationship between the activation probability $ap(v, S, \theta)$ and $ap(w, S, \theta)$ when the seed set is fixed, under MIA model. Assume the linear coefficient is $\alpha^t(v, w)$ after t seeds are selected. The incremental coverage of w on v after t seeds are selected can be computed as:

$$IncCov^t(w, v) = \alpha^t(v, w) \cdot (1 - ap(w, S^t, \theta))$$

where S^t is the set of first t seeds selected. $IncCov^t(w, v)$ predicts that given the current seed set S^t, if w is added as a seed, how much more coverage on node v we can gain. While the summation $\sum_v IncCov^t(w, v) = \sigma_\theta(S^t \cup \{w\}) - \sigma_\theta(S^t)$ is used to guide the seed selection, the individual incremental coverage $IncCov^t(w, v)$ has not been utilized.

We believe the *coverage vector* $IncCov^t(w, \cdot)$ for each individual w is an important feature for finding the word relations. The coverage vector refelects the capability of coverage from each word. Semantically related words should also have related coverage capability on other words.

Suppose u_x and u_y are two seeds selected at the x-th and y-th place, and u_x is selected before u_y (*i.e.*, $x < y$). We examine their coverage vectors to infer their relation.

First we define the dot product between their coverage vector:

$$dp(u_x, u_y) = \sum_v IncCov^x(u_x, v) IncCov^y(u_y, v)$$

If u_x and u_y are about the same topic, u_y should cover a similar set of words as u_x does. So we define a measure:

$$same(u_x, u_y) = \frac{dp(u_x, u_y)}{\sum_v IncCov(u_x, v)} \tag{5.3}$$

for the 'same topic' relations between the two nodes.

Second, if u_y is talking about a subtopic that is covered by the topic of u_x, the incremental coverage brought by u_y should be mostly covered by u_x. We have:

$$super(u_x, u_y) = \frac{dp(u_y, u_x)}{\sum_v IncCov(u_y, v)} \tag{5.4}$$

The nodes can be grouped to form topics. To compute the two measures in group level, we first compute the individual node-level measure according to Eq. (5.3) and (5.4), then take average over the first group S_1, and finally select median over the second group S_2, as follows.

$$same(S_1, S_2) = \underset{u_y \in S_2}{median} \underset{\substack{u_x \in S_1 \\ x<y}}{mean} same(u_x, u_y) \tag{5.5}$$

$$super(S_1, S_2) = \underset{u_y \in S_2}{median} \underset{\substack{u_x \in S_1 \\ x<y}}{mean} super(u_x, u_y) \tag{5.6}$$

We tried other operators like max, min and other combinations, and this choice gives the best results among them.

We use these measures *same* and *super* to organize the words into hierarchical topics. Our algorithm has two phases: grouping and splitting. In grouping phase the groups are built by merging seed words. We initialize each group as containing only one seed word. For any group, if the largest *same* measure with another groups is above a threshold t_1, we can merge them. And the merged groups can be further merged to form larger groups. In splitting phase, the relation between each group and its candidate parent are examined. If the *super* measure is above another threshold t_2, we assign the super-sub relationship between them; otherwise we assume the parent-child relation is weak and the two groups should represent independnt topics. The algorithm is outlined as follows.

• **Initialization.** Each word forms a single group, and the parent pointer is set to a word selected before it and having the largest *super* measure with it.

• **Grouping.** In grouping phase the groups are merged bottom-up iteratively. In each round two groups that are either parent-children or siblings, with largest *same* measure will be merged. The *same* and *super* measures are recomputed between the new group and existing groups. The parent pointer is updated according to the new *super* measure. This process lasts until the number of groups is small enough or the *same* measure of any two groups is smaller than a threshold t_1.

• **Splitting.** In splitting phase, we break the weak relation between a group and its parent group. We break one weakest relation with the smallest *super* measure each time. We repeat this breaking until the weakest relation has a larger *super* measure than a threshold t_2, or the number of root topics has been desirable.

This algorithm is simple and heuristic. It only relies on the incremental coverage $IncCov$ which can be naturally obtained from the efficient MIA algorithm. It is guaranteed to stop in finite steps because the maximum number of grouping and splitting is limited by the total number of seeds. One advantage of this algorithm is that it can generate flexible hierarchy with only two parameters. Depending on different use cases, one has the following choices of parameter specification.

• The number of total topics N and the number of root topics N_0.

• The threshold for grouping phase t_1 and the number of root topics N_0.

• The number of total topics N and the threshold for splitting phase t_2.

• The threshold for grouping phase t_1 and the threshold for splitting phase t_2.

Therefore, although there are 4 parameters which can control the shape of the hierarchy, one only needs to specify two of them. Larger t_1 leads to larger N, and larger t_2 leads to larger N_0. Generating hierarchies with different parameters is low-cost because in our 2-stage summarization framework, the coverage maximization, the word selection and the computation of two measures *same* and *super* are done before the topic construction. So when we change the parameters like the total number of desired topics and the number of top-level topics, we only need to run the heuristic algorithm again on the same small seed word graph. The simplicity in changing parameters allows fast generation of word clusters with controllable structures.

6. EXPERIMENT

We conduct experiments with our algorithms and other algorithms, based on network analysis and generative topic modeling respectively, on several text collections including TREC news and online discussions. Our experiments aim at evaluating the following aspects of our approach: a) its performance in picking up summary words; b) its ability in topic discovery and organization; c) the effect of parameters.

6.1 Experiment setup

Datasets. We use two document collections. One is TREC AP news collection, the other is 20 newsgroup.

Build Influence Network. We remove stopwords from a given list and build index with Lemur. Then we build the word influence network by counting co-occurrence of each pair of words with the help of inverted index. The mutual information can be computed based on the document frequency (df) of their occurrence and co-occurrence:

$$\begin{aligned} I(u,v) = & \frac{df(u,v)}{|D|}\log\frac{df(u,v)|D|}{df(u)df(v)} \\ & + \frac{df(u,\neg v)}{|D|}\log\frac{df(u,\neg v)|D|}{df(u)(|D|-df(v))} \\ & + \frac{df(\neg u,v)}{|D|}\log\frac{df(\neg u,v)|D|}{(|D|-df(u))df(v)} \\ & + \frac{df(\neg u,\neg v)}{|D|}\log\frac{df(\neg u,\neg v)|D|}{(|D|-df(u))(|D|-df(v))} \end{aligned}$$

where $|D|$ is the number of total documents. For each node in these networks, we remove the neighbors with influence probability smaller than a threshold $t_0 = 0.001$. And we only keep at most top $k = 100$ neighbors for each node. This reduces the density of the network.

Algorithms. We compare our algorithm with three algorithms based on three different modeling assumptions.

• Degree+MMC. The first algorithm is a simpler heuristic algorithm for the coverage maximization. It chooses the words with maximal weighted out-degree, to perform coverage maximization. The weighted degree of a node is the sum of propagation probabilities on all its outgoing edges. When each seed is selected, we will discount the weighted degree of each neighboring word by the influence probability from the neighbor to the seed, because the selected seed cannot contribute to the neighbor's marginal coverage any more. This is similar to Maximal Marginal Relevance but it is maximizing the marginal coverage, so we name it Degree+MMC. We chose this algorithm to see how much the performance relies on the accuracy of coverage maximization.

• PageRank+MMR. This second algorithm is based on random walk. It uses PageRank for word ranking, and MMR for reranking, like previous researchers [7, 28] did. We use the normalized influence probability $\frac{pp(u,v)}{\sum_w pp(u,w)}$ as the transition probability from u to v, and restart probability is set to be 0.85. We chose this algorithm to study the difference between content coverage maximization and the random walk model.

• PAM. The last algorithm we compare to is a stochastic topic model, Pachinko Allocation Model (PAM) [14], which is an extension of latent dirichlet allocation (LDA). It generates hierarchical topics according to user-specified number of topics in each level. It does not generate keyword summarization of the entire document collection, and we only compare the performance in topic construction. This algorithm is representative of the Bayesian topic modeling approach.

For the first two algorithms Degree+MMC and PageRank+MMR, we use them to select keywords and use the same topic organization algorithm in Section 5. For the thrid algorithm PAM, we need to do post processing to output comparable results because PAM generates word distribution only at the leaf level. The internal, non-leaf topics in PAM are represented as mixture of child topics. We generate the word distribution for non-leaf topics by mixing the children topics according to the mixture distribution.

Table 1: Summary words selected by different algorithms on a subset of 20 newsgroups: alt.atheism, comp.graphics, comp.os.ms-windows.misc, comp.sys.ibm.pc.hardware, comp.sys.mac.hardware

Rank	PR+MMR	Degree+MMC	Ours
1	challenge	**contrib**	**contrib**
2	keith	modified	modified
3	**ide**	**anonymous**	**god**
4	rushdie	**renderer**	**anonymous**
5	**monitor**	**tar**	sunview
6	**umd**	provides	provides
7	ac	**god**	**renderer**
8	seas	readme	say
9	gulf	imagery	**atheists**
10	**duo**	sunview	dwyer
11	**qwk**	**header**	**tar**
12	motss	**directory**	**directory**
13	**serial**	**ftp**	**sgi**
14	ca	lcs	**image**
15	austin	**export**	astronomical
16	gatech	manipulating	**gis**
17	cleveland	supported	transfer
18	de	**distributed**	**pbmplus**
19	au	**formats**	**amigas**
20	timmons	**sites**	following
Sum	6/2	11/3	12/4
# topic words / # covered topics			

6.2 Experiment results

6.2.1 *Summary words*

To evaluate summary words selected by different algorithms, we conduct both quantitative study and qualitative study.

For qualitative study, we intensionally use documents of known topics to feed the summarization algorithms, and examine the results. We use 20 newsgroups data as the base of documents. We order the 20 groups from group 1 to group 20, and add one group each a time to our document collection to summarize. In this way we obtain 20 sets of documents with different number of known topics: set 1 with group 1, set 2 with group 1 + group 2, *etc.*, and set 20 with everything. We examine the top 20 words as summary for each set to see whether they cover the known topics in each set. Table 1 shows an example of the results of PageRank(PR)+MMR, Degree+MMC and our algorithm on a subset of 5 groups. PAM is not relevant for this task.

We find that with the same number of summary words, our algorithm covers more topics, and generate more meaningful topic words in general. In the top 20 ranked words, there are 12 of them each of which covers one of the topics discussed in the newsgroups, and they in total cover 4 topics. We notice that there are some noises in the data and all these algorithms will report some non-relevant words. We also observe that Degree heuristic and our method tend to generate more similar words while the output of PageRank is quite different. This is because both Degree heuristic and our method have a goal of maximizing the content coverage while PageRank does not. And one interesting observation is that PageRank will select more entity words like ca, austin, gatech *etc.*, perhaps due to their broad connection with other words. These are true for other test sets too.

For quantitative study, we evaluate the correspondence between the summary words and the known categories. If the words cor-

Table 2: Mutual information of summary words selected by different algorithms on the whole 20 newsgroups data

k	PR+MMR	Degree+MMC	Ours
10	0.276	0.227	**0.281**
20	**0.606**	0.403	0.597
30	0.770	0.599	**0.849**
40	0.833	0.828	**1.076**
50	0.876	1.028	**1.308**

respond well to the known categories, and cover all the major categories, they form a good summary. We measure the correspondence between the words $W = \{w_1, \ldots, w_k\}$ and m categories $C = \{C_1, \ldots, C_m\}$ with mutual information (MI)

$$I(W;C) = \sum_{i,j} p(w_j, C_i) \log \frac{p(w_j, C_i)}{p(w_j)p(C_i)} \qquad (6.7)$$

Table 2 shows an example of the detailed results on the complete set of 20 newsgroup documents, including the word-category mutual information of top k words when k varies. We can see that our method generates a summary that better corresponds to the known categories, though we do not use any supervised information. The other two baselines have different properties. When k is small, PageRank + MMR performs better than Degree + MMC, because it ranks the words not only according to how many words they can influence directly but also which words they can influence. However, since it does not model the coverage, the highly ranked words do not cover different information in every category. This is implied from the slow increase of the mutual information from $k = 30$ to 50. On the other hand, although Degree + MMC does not select the most influential words at the beginning, it consistently choose words with large degree so these words have a good coverage on their neighbors at least. Our algorithm also maximizes the information coverage as Degree + MMC does, while it has the ability to utilize the structure of indirect neighbors as PageRank has. That is the reason it can select words with better quality than both.

Figure 2 shows the mutual information for different subsets, each of which contains all the documents with the same first-level topic such as Computer, Talk and Science. Our algorithm constantly generates good summary words, while PageRank and Degree have no clear winner between them. On the subset of topic Science and Talk, the Weighted Degree heuristic works well, but for Computer and the whole collection it is inferior to PageRank; PageRank + MMR produces similar mutual information for the three subsets. Again, the variation of their performance in different situation validates the premise that PageRank works well when the number of chosen words is smaller or close to the number of topics, while the coverage-based method exhibits potential when more words are allowed to be selected in the summary. The subtopics in Talk, politics about guns, politics about Mideast, and misc politics and religion, are easy to distinguish because they have quite different terminology. However, in Computer topic, it is not easy to separate some of them because of the commonly shared terms, and we see none of the three approaches achieve an average mutual information gain above 0.01.

6.2.2 *Topic hierarchy*

We evaluate the topic hierarchy by user study and quantitative measure.

Figure 3 shows an example of the topic hierarchy constructed by different algorithms on TREC AP news data in the year 1998. Each node in the hierarchy represents a topic, and each topic can

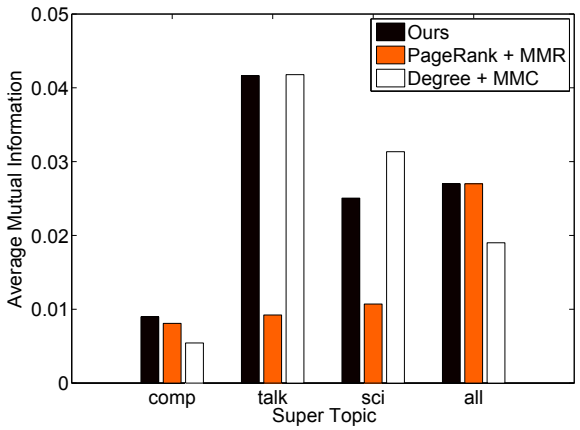


Figure 2: Word-category mutual information for the top 25 ranked words on different subsets defined by the first-level group (super topic)

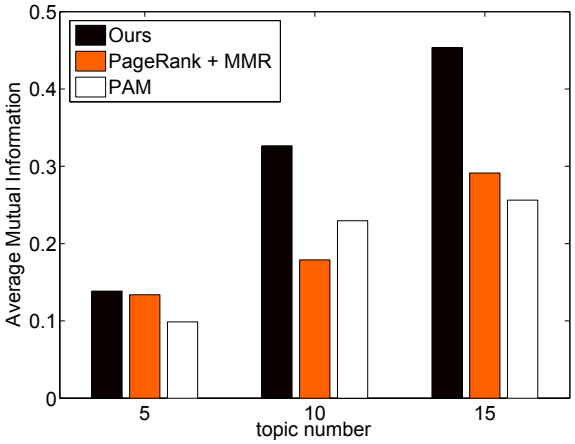
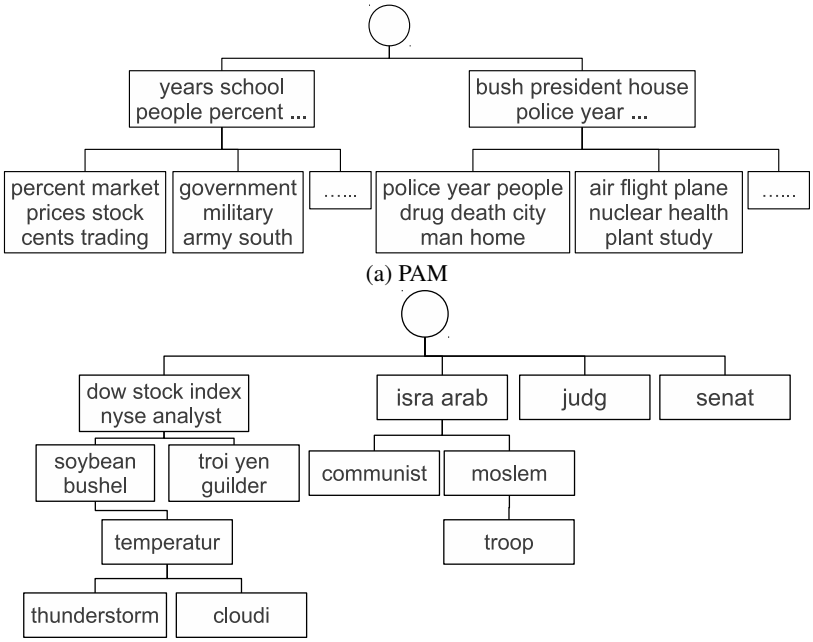


Figure 4: Topic-category mutual information for n topics, on 20 newsgroup data

have one or more subtopics that is covered by this topic. Note that in PAM the non-leaf topic is not a word cluster but a distribution of sub topics. We post-process the results to generate the word cluster from the topic distribution. Also we need to specify the number of levels and the number of topics on each level for PAM. Tuning the parameters is time consuming in PAM model, since each time the parameters is changed, we need to rerun the inference algorithm on the whole documents. Nevertheless, in our heuristic method the parameter adjustment is much easier, as we explained in Section 5.

For evaluation, we design a user study as follows. Since PAM gives different shapes of hierarchy as ours, the two hierarchies are not easily comparable. We try to evaluate a topic hierarchy by measuring whether two topics close to each other in the tree path distance are also regarded similar by human being. If so, the tree structure of the topics is consistent with human judgment. So we ask each evaluator two common sets of questions, each set corresponding to one topic hierarchy generated by either PAM or our method. The number of questions in both sets are equal. They are mixed and shuffled before presented to every evaluator, so one will not tell the source of a question. Each question requires an evaluator to look at a triple of topics (T_1, T_2, T_3), and answer whether T_1



(a) PAM

(b) Our algorithm (words are stemmed)

Figure 3: Examples of topic hierarchies on one year of TREC AP data set (1998)

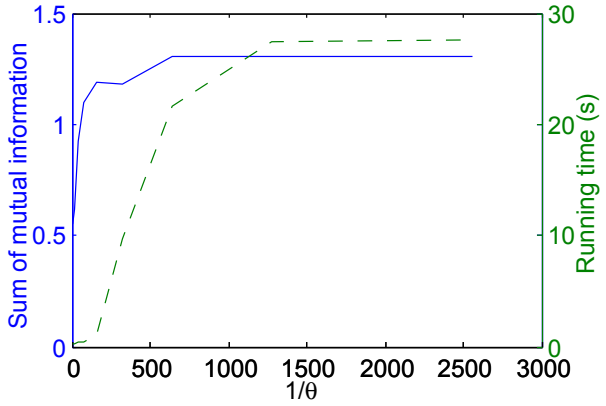


Figure 5: Word-category mutual information by 50 summary words and running time w.r.t. θ, for the TREC AP dataset in the MIA model. Blue line – mutual information; Green dashed line – running time

is conceptually closer to T_2, T_3, or hard to tell. Then we compare whether the answer matches the real tree path distance $d(T_1, T_2)$ and $d(T_1, T_3)$. For example, if $d(T_1, T_2) > d(T_1, T_3)$ and the answer is that T_1 is closer to T_3 than it is to T_2, we count it a match. A match rewards the evaluator 2 points and a 'hard-to-tell' answer gains 1 point. A non-match answer contributes nothing. In this way we obtain a score from each evaluator's answers to each set of questions. Thus the score can be used to judge which topic hierarchy has higher consistency with common sense.

We generate topic hierarchy with our algorithm and PAM for the same subset of TREC AP collection, and randomly generate questions so that in each triple (T_1, T_2, T_3), T_1 is either T_2's parent, child or sibling, while T_3 has none of those relationships with T_1. Thus T_2 is regarded to be the matching answer. Then we randomly shuffle some T_2 and T_3 to form our question sets. The evaluation on 6 users with different background knowledge shows that our topic hierarchy with 12 topics in total is more consistent than that of PAM with 2 levels, 2 super topics and 10 leaf topics. The average score for our hierarchy is 7.5, compared to PAM hierarchy of 3.3. Moreover, each user gives higher score on our hierarchy and the t-test has a p-value of 0.008. So the test result is statistically significant.

We also use topic-category mutual information to measure how well we can predict the category label of a document given that we know whether words of a topic occur in the document. Let $T = \{w_1, \ldots, w_p\}$ be a topic with p words, and $\mathcal{T} = \{T_1, \ldots, T_n\}$ be the set of topics. We can define the mutual information between topics and categories $C = \{C_1, \ldots, C_m\}$ as

$$I(\mathcal{T}; C) = \sum_{i,j} p(T_i, C_j) \log \frac{p(T_i, C_j)}{p(T_i)p(C_j)}$$

We generate different number N of topics with each algorithm, and draw a curve of the mutual information of N topics, as shown in Figure 4. Our algorithm constantly outperforms the other two baselines.

6.2.3 *Parameter study*

The threshold θ in the coverage maximization algorithm controls the degree of approximation. The smaller θ is, the more accurate the coverage maximization problem can be solved, but the running time is larger. We investigate the summary results by different θ. The positive correlation between the coverage maximization accuracy and the word-category mutual information confirms our presumptions. As expected, the result quality and the running time stops increasing when θ becomes small enough, which means the influence to the whole network has been taken into consideration. Even in that case, the running time is only around 30 seconds, which shows the superiority of the efficiency over most generative topic models.

7. CONCLUSION AND FUTURE WORK

We propose a summarization method based on the analysis of the word network constructed from an arbitrary set of documents. The advantage is we can compress the information in the network and study the relations among the words by simple but intuitive means. We explore the analogy of the word influence network to social influence network, and explore the idea of finding topics by studying the influence among the words and modeling it as an coverage maximization problem. The result of word summary and topic hierarchy shows the potential of this approach on both traditional and novel tasks in text summarization, and suggests promising new research direction.

One possible way to improve the results is to consider more fine granularity when creating the word network. For example, we may only link words that co-occur in a paragraph or a sentence, instead of in a document. This also helps reducing the density of the network. The idea of using influence network to model text can be further explored on sentence, document or heterogeneous networks with different type of nodes and links. It is also possible to solve different tasks by changing the definition of the influence probability. Most summarization tasks concern with not only the ranking but also the coverage. So all those tasks that are previously solved by PageRank-alike approach can now be reexamined in the coverage maximization framework.

Acknowledgments

This material is based upon work supported in part by the U.S. National Science Foundation grants CNS–0931975, IIS–1017362, IIS–132061, CNS–1027965, and U.S. Army Research Laboratory under Cooperative Agreement No. W911NF–09–2–0053 (NS–CTA). Chi Wang was supported by a Microsoft Research PhD Fellowship. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The authors would like to thank Wei Chen for his helpful discussion.

8. REFERENCES

[1] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet allocation. *J. Machine Learning Research*, 3:993–1022, 2003.

[2] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30(1-7):107–117, 1998.

[3] J. Carbonell and J. Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*, 1998.

[4] W. Chen, Y. Wang, and S. Yang. Efficient influence maximization in social networks. In *KDD*, 2009.

[5] S.-L. Chuang and L.-F. Chien. A practical web-based approach to generating topic hierarchy for text segments. In *CIKM*, 2004.

[6] P. Domingos and M. Richardson. Mining the network value of customers. In *KDD*, 2001.

[7] G. Erkan and D. R. Radev. Lexrank: graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res.*, 22:457–479, 2004.

[8] A. Fuxman, P. Tsaparas, K. Achan, and R. Agrawal. Using the wisdom of the crowds for keyword generation. In *WWW*, 2008.

[9] A. Goyal, F. Bonchi, and L. V. Lakshmanan. Learning influence probabilities in social networks. In *WSDM*, 2010.

[10] T. Hofmann. Probabilistic latent semantic indexing. In *SIGIR*, 1999.

[11] D. Kempe, J. M. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *KDD*, 2003.

[12] M. Kimura and K. Saito. Tractable models for information diffusion in social networks. In *PKDD*, 2006.

[13] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. S. Glance. Cost-effective outbreak detection in networks. In *KDD*, pages 420–429, 2007.

[14] W. Li and A. McCallum. Pachinko allocation: Dag-structured mixture models of topic correlations. In *ICML*, 2006.

[15] X. Liu, Y. Song, S. Liu, and H. Wang. Automatic taxonomy construction from keywords. In *KDD*, 2012.

[16] R. Mihalcea and P. Tarau. TextRank: Bringing Order into Texts. In *Conference on Empirical Methods in Natural Language Processing*, Barcelona, Spain, 2004.

[17] D. Mimno, W. Li, and A. McCallum. Mixtures of hierarchical topics with pachinko allocation. In *ICML*, 2007.

[18] G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming*, 14:265–294, 1978.

[19] S. P. Ponzetto and M. Strube. Deriving a large scale taxonomy from wikipedia. In *Proceedings of the 22nd national conference on Artificial intelligence - Volume 2*, pages 1440–1445, 2007.

[20] D. R. Radev. A common theory of information fusion from multiple text sources step one: cross-document structure. In *Proceedings of the 1st SIGdial workshop on Discourse and dialogue - Volume 10*, 2000.

[21] D. R. Radev, H. Jing, and M. Budzikowska. Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. In *NAACL-ANLP 2000 Workshop on Automatic summarization - Volume 4*, 2000.

[22] M. Richardson and P. Domingos. Mining knowledge-sharing sites for viral marketing. In *KDD*, 2002.

[23] M. d. B. Rodriguez, J. M. G. Hidalgo, and B. D. Agudo. Using wordnet to complement training information in text categorization. In *Proc. RANLP*, 1997.

[24] M. G. Rodriguez, J. Leskovec, and A. Krause. Inferring networks of diffusion and influence. In *KDD*, 2010.

[25] J. Tang, J. Sun, C. Wang, and Z. Yang. Social influence analysis in large-scale networks. In *KDD*, 2009.

[26] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101, 2004.

[27] V. V. Vazirani. *Approximation Algorithms*. Springer, 2004.

[28] X. Wan and J. Xiao. Exploiting neighborhood knowledge for single document summarization and keyphrase extraction. *ACM Trans. Inf. Syst.*, 28:1–34, 2010.

[29] X. Wan, J. Yang, and J. Xiao. Towards an iterative reinforcement approach for simultaneous document summarization and keyword extraction. In *ACL*, 2007.

[30] C. Wang, W. Chen, and Y. Wang. Scalable influence maximization for independent cascade model in large-scale social networks. *Data Mining and Knowledge Discovery*, 25(3):545–576, 2012.

[31] C. Wang, M. Danilevsky, N. Desai, Y. Zhang, P. Nguyen, T. Taula, and J. Han. A phrase mining framework for recursive construction of a topical hierarchy. In *KDD*, 2013.