Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

Efficiently Mining High Quality Phrases from Texts

Bing Li,[§†] Xiaochun Yang,[§†] Bin Wang,[§†] Wei Cui[‡]
§Key Laboratory of Medical Image Computing of Northeastern University, Ministry of Education
†School of Computer Science and Engineering and ‡College of Information Science and Engineering
Northeastern University, Shenyang 110819, China
{yangxc, binwang}@mail.neu.edu.cn {libing, cuiwei}@stumail.neu.edu.cn

Abstract

Phrase mining is a key research problem for semantic analysis and text-based information retrieval. The existing approaches based on NLP, frequency, and statistics cannot extract high quality phrases and the processing is also time consuming, which are not suitable for dynamic on-line applications. In this paper, we propose an *efficient high-quality phrase mining approach* (EQPM). To the best of our knowledge, our work is the first effort that considers both intra-cohesion and inter-isolation in mining phrases, which is able to guarantee appropriateness. We also propose a strategy to eliminate order sensitiveness, and ensure the completeness of phrases. We further design efficient algorithms to make the proposed model and strategy feasible. The empirical evaluations on four real data sets demonstrate that our approach achieved a considerable quality improvement and the processing time was $2.3\times \sim 29\times$ faster than the state-of-the-art works.

Introduction

With the explosive growth of the information, people are overwhelmed by a large number of unstructured text data. It is of high value to enhance the power and efficiency to facilitate human manipulating and understanding unstructured text data. Phrase mining could transform text document from word granularity to phrase granularity by automatically extracting semantically meaningful phrase. Particularly, phrase mining is an essential step for further semantic analysis or text-based retrieval in established fields of information retrieval and natural language processing (NLP). Moreover, phrase mining is also critical to various tasks in emerging applications. Examples of such applications include topic detection and tracking (Leskovec, Backstrom, and Kleinberg 2009), social event discovery (Li et al. 2016b), and document summarization (He et al. 2012).

The study of phrase mining originates from the natural language processing (NLP) community, which utilizes a set of language rules (Witten et al. 1999; Abney 1991; Clahsen et al. 2006) to derive phrases. Such rule-based approaches are rigorous and not suitable for the emerging applications, such as scientific papers, twitters, and query logs.

Therefore, there have been developed many data-driven approaches in this area. The raw frequency-based approaches regard each mined phrase as a frequent pattern (Ahonen 1999; Simitsis et al. 2008), where a phrase is extracted if it has longest consecutive words and its frequency is larger than a given threshold. However, ranking the word sequences according to the frequency will generate many false phrases. Recently, a variety of frequency statistical approaches (O'Neil and Sangiovanni-Vincentelli 2014; El-Kishky et al. 2014; Li et al. 2016a) are developed to estimate phrase quality and rank candidate phrases. Liu *et al.* (2015) consider integrating phrasal segmentation with phrase quality estimation to further rectify the inaccurate phrase quality initially estimated, based on local occurrence context.

These frequency statistical approaches mine quality phrases from a large collection of documents or a corpus. A phrase is a sequence of words that appear contiguously in the text, and serves as a whole (non-composible) semantic unit in certain context of the given documents (Liu et al. 2015). Generally, a high quality phrase should have the following criteria.

- *Frequency*. This criterion is based on the observation that a non-frequent phrase is likely to be not important (El-Kishky et al. 2014).
- *Phraseness*. If adjacent phrases co-occur more significant than expected under a given statistical significant level, these phrases should be concatenated into a longer phrase (Wang et al. 2013).
- *Completeness*. If long frequent phrases satisfy the above criteria, then their subsets also satisfy these criteria since the subsets will satisfy the criteria of frequency and phraseness, yet is clearly a subset of a larger and more intuitive phrase (El-Kishky et al. 2014).
- *Appropriateness*. If complete phrases are overlapping, an appropriate segmentation should ensure the extracted phrases are disjoint and each of them is semantically independent.

The existing approaches may generate low quality phrases that do not conform to the above criterions as demonstrated below.

(i) *Order sensitive processing causes incomplete phrases*. The existing frequency statistical approaches could not guarantee to extract complete phrases since they heuristically concatenate those words into one phrase who has

large statistic score, i.e., the quality of the extract phrase highly depends on the concatenating order among words. For example, consider four adjacent words w_1=Gaussian, w_2=Mixture, w_3=Model, and w_4 =Selection in a corpus. Assume Gaussian Mixture Model is a high quality phrase. We use $t(p)$ to represent the t-statistic score of a phrase p (i.e. the ratio of its actual frequency to its expected occurrence). Let $t(w_1w_2) = 6391.62$ and $t(w_2w_3) = 23.96$. The approaches heuristically concatenate Gaussian and Mixture firstly, and then determine if Gaussian Mixture can be concatenated with Model. By using this concatenating order, the complete phrase Gaussian Mixture Model cannot be extracted since $t(w_1w_2w_3) = 15.75$ is small (e.g., less than a given threshold 16). On the contrary, it can be extracted if Mixture and Model are concatenated in first step.

(ii) *Overlapping sequences causes inappropriate segmentation*. For aforementioned example Gaussian Mixture Model Selection, two sequences $w_1w_2w_3$ and w_3w_4 are overlapping if they both have high statistic scores. In the scenario of word segmentation, w_3 can only be set to one of these two sequences, i.e., $w_1w_2|w_3w_4$ or $w_1w_2w_3|w_4$. The approach in (Liu et al. 2015) prefers to choose $w_1w_2|w_3w_4$ based on a probability $p(\cdot)$ since $p(w_1w_2|w_3w_4) = 0.07$, which is greater than $p(w_1w_2w_3|w_4) = 0.039$. However, Gaussian Mixture Model should be a high quality phrase since it is an attributive noun that functions as an adjective, whereas Model Selection is rare to mention as a semantically independent phase.

This paper takes on the challenge of designing a phrasal segmentation model to improve the quality of extracted phrases. We propose a novel phrasal segmentation model by making the first effort to consider both intra-cohesion and inter-isolation in mining phrases, which could guarantee appropriateness. We then propose a complete phrase mining strategy to eliminate order sensitive and guarantee to avoid incomplete phrases. Both the new phrasal segmentation model and phrase mining strategy are time consuming, therefore, the second challenge of this paper is to design efficient algorithms to make the proposed model and strategy feasible. We propose a dynamic programming approach to reduce the inference cost of updating the probability in our model and a parameter estimation with a strict error bound to avoid cost of learning parameters. Moreover, we propose two efficient algorithms to improve the efficiency of complete phrase mining. The experiments on real data sets demonstrate that we can efficiently get phrases with high quality compared with the state-of-the-art methods.

Related Work

Phrase mining originates from the NLP shallow parsing (also known as chunking) problem. As the origin, shallow parsing methods (Witten et al. 1999; Abney 1991; Clahsen et al. 2006) mostly rely on part-of-speech (POS) tagging techniques and use predefined NLP rules to group noun phrases. Obviously, these rule-based methods lack enough flexibility to handle various languages and heterogeneous corpora. Thus, other NLP methods have been proposed to enhance the accuracy by introducing supervised learning models (Punyakanok and Roth 2001; Brill 2002; Kudoh and Matsumoto 2002; Mcdonald, Crammer, and Pereira 2005) or stochastic models (Church 1989; Shen and Sarkar 2005; Sha and Pereira 2003; Vishwanathan et al. 2006; Sun et al. 2008; Huang, Xu, and Yu 2015). Supervised shallow parsing methods take a number of annotated texts as training data, and learn classification rules based on POS features. Supervised methods are barely able to overcome the high annotation cost. Stochastic shallow parsing methods use stochastic model to parse noun phrases, where Sha and Pereira (2003), Vishwanathan *et al.* (2006), and Sun *et al.* (2008) adopt CRF model, and Shen and Sarkar (2005) adopt HMM as the stochastic model. However, these methods show low scalability to a new language or a new domain. These shortages hinder their applications in domain-specific, dynamic, and emerging applications.

Recent efforts derived statistics of data distribution from a large corpus to further improve the accuracy of phrase quality estimation. Based on the distributional features of a web-scale corpus, Pitler *et al.* (2010) used a statistical measure PMI to mine n-grams; Parameswaranc *et al.* (2010) extracted n-grams using several indicators. Deane (2005) proposed a statistical measure based on Zipfan ranking to measure lexical association in a phrase. El-Kishky *et al.* (2014) uses t-statistic to filter and rank candidate phrases. These statistical measure based methods do not rely on language-specific linguistic feature, and can thus achieve greater scalability compared with the aforementioned methods.

Word sequence segmentation is another strategy of phrase mining which partitions a word sequence into disjoint subsequences, like query segmentation (Tan and Peng 2008; Li et al. 2011), or chunking (Tjong K. S. and Buchholz 2000; Blackwood, Gispert, and Byrne 2008; Echizen-ya and Araki 2010). A recent work is phrasal segmentation (Liu et al. 2015). The existing models only consider intra-cohesion of phrases such as the number of words in the phrase and tokens, while ignore the inter-isolation between phrases.

Quality Phrase Mining

We propose a novel phrasal segmentation model to mine phrases and ensure the appropriateness requirement. For completeness requirement, we propose a complete phrase mining approach to eliminate incomplete phrases.

Phrasal Segmentation Model

In order to solve the problem of inappropriate segmentation, we propose a more comprehensive and effective phrasal segmentation model by considering inter-isolation which is formally defined as follow:

Given a sequence S with n words $w_1 \dots w_n$, we want to find a set of positions $P = \{b_1, \dots, b_m\}$ to split S into $m-1$ disjoint subsequences $s_i, \dots, s_m$, where $b_1 = 1, b_m = n+1, b_1 \leq b_j \leq b_m$ $(1 \leq j \leq m)$, and we use $S(b_i, b_{i+1}-1)$ to represent each subsequence $s_i = w_{b_i} \dots w_{b_{i+1}-1}$ $(1 \leq i \leq m-1)$. We use $|S|$ to denote the sequence length, i.e. the number of words in S. Let $P^* = \{{b_1}^*, {b_2}^*, \dots, {b_m}^*\}$ be a set of optimal split positions that can maximize the follow-

ing joint probability:

$$p(P^*, S) = \prod_{i=1}^{m-1} p(s_i, b_{i+1}|b_i) \times p({b_i}^*|s_i, s_{i-1}), \quad (1)$$

where $p(s_i, b_{i+1}|b_i)$ denotes the conditional probability of observing a subsequence s_i as the i-th phrase, which reflects the intra-cohesion of phrases. $p({b_i}^*|s_i, s_{i-1})$ is an inter-isolation indicator of i-th split position b_i when subsequences s_{i-1} and s_i are given. Notice that $p(b_1^*|\cdot) = 1$.

Eq. 1 is derived from the following two-step generative model. The first step is to generate split position b_{i+1} with a probability of $p(L)$, where L is a random variable representing the number of words in a subsequence. The variable L is drawn from a Poisson distribution:

$$p(L) \sim \frac{\lambda^L e^{-\lambda}}{L}, \quad (2)$$

where λ can be estimated based on the distribution of phrase length (i.e. the number of words in the phrase). We count the number of words of all high quality phrases in a corpus, and use maximum likelihood estimation to estimate λ.

After generating b_{i+1}, we generate s_i according to a multinomial distribution over L such that L equals to the length of s_i, i.e. $L = l(s_i)$. Suppose we can get the frequency $f(s_i)$ of each s_i in a corpus, then the probability of generating s_i under condition L=$l(s_i)$ can be estimated as:

$$p(s_i|L = l(s_i)) = \frac{f(s_i)}{\sum_{\forall l(s_j)=l(s_i)} f(s_j)}. \quad (3)$$

Based on the above two prior probabilities, the conditional probability $p(s_i, b_{i+1}|b_i)$ in Eq. 1 can be derived via the following probabilistic factorization:

$$\begin{aligned} p(s_i, b_{i+1}|b_i) =& p(b_{i+1}|b_i) \cdot p(s_i|b_{i+1}, b_i) \\ =& p(L) \cdot p(s_i|L = l(s_i)). \end{aligned}$$

The second step is to determine whether b_i is a good split position to divide $S(b_{i-1}, b_{i+1} - 1)$ into two independent phrases s_{i-1} and s_i according to $p(b_i^*|s_{i-1}, s_i)$.

$$p(b_i^*|s_{i-1}, s_i) = \begin{cases} 1, & i = 0 \; or \; i = |S| \\ \frac{H(s_{i-1})+H(s_i)}{2 \times I(s_{i-1}; s_i)}, & otherwise \end{cases} \quad (4)$$

where $H(s_i) = p(s_i) \log p(s_i)$ and

$$I(s_{i-1}; s_i) = p(s_{i-1} \oplus s_i) \log \frac{p(s_{i-1} \oplus s_i)}{p(s_{i-1})p(s_i)}.$$

Here we use $s_{i-1} \oplus s_i$ to represent a concatenated phrase of consecutive phrases s_{i-1} and s_i. Let s be a phrase, then

$$p(s) = \frac{f(s)}{\sum_{s' \in U} f(s')},$$

where U is the collection of all computed phrases.

Since for every newly generated split position, we have to "look back" at its previous split position, our model could achieve a better appropriateness than existing methods. However, this "look back" feature also causes a large computation cost. A naive method is to check all possible segmentations, compute joint probabilities, and choose the best one. In the worst case, the naive method requires an $O(n^4)$ time in a single inference. Moreover, due to $p(s_i|L = l(s_i))$ is unknown, it will cause $O(ite \cdot n^4)$ learning cost, in which ite denotes the rounds of iterations.

Complete Phrase Mining

Recall that generating a complete phrase is sensitive to the concatenating order. To strictly guarantee the completeness requirements, we propose a complete phrase mining strategy to enumerate all possible concatenating orders and choose the best one. In this way, we could avoid the incompleteness. This naturally raises a straightforward algorithm, which firstly enumerates every possible subsequences, and secondly verifies whether each of them is a complete phrase. There are totally n^2 sequences for a sequence S with n words. Verifying a sequence needs n^2 time complexity in the worst case. Therefore, the total time complexity is $O(n^4)$.

In this paper, we adopt χ^2-test (F.R.S. 1900) as phraseness measurement. Notice that, we can also use other statistics-based measurements such as z-test and mutual information to replace χ^2-test in our framework. The focus of this work is to show an efficient algorithmic design of quality phrase mining, not to optimize a specific phraseness measurement.

Efficiency Improvement

In EQPM, we improve the efficiency from the following two aspects. (i) In order to make our phrasal segmentation model feasible, we adopt a dynamic programming based method to reduce the inference cost of finding the optimal segmentation. Meanwhile we use the result of complete phrase mining to estimate the unknown parameter to avoid learning cost. (ii) We propose two algorithmic designs – a dynamic programming approach and a seed extension based approach to improve the efficiency of complete phrase mining.

Reducing Inference Cost for Phrasal Segmentation

We adopt a dynamic programming strategy to reduce the inference cost of our phrasal segmentation model. Since the split positions in $S(1, i-1)$ is based on the split positions in $S(1, j-1)$ $(j < i)$, we construct a matrix $D_{(n+1)\times(n+1)}$, in which each cell $D(i, j)$ stores the optimal probability that j is the last split positions in $S(1, i-1)$. Initially, $D(0, j) = -\infty$, $D(1, 0) = 1$, $D(i, 0) = -\infty$ (if $i > 1$), $D(i, j) = -\infty$ (if $i = j$). The recursion function is given as follows:

$$D(i, j) = \max_{k \in [0, j-1]} \left\{ \begin{array}{l} D(j, k)p(S(j, i-1), i|j) \\ \times p(j^*|S(k, j-1), S(j, i-1)) \end{array} \right\}, \quad (5)$$

where $i \in [1, n+1]$ and $j \in [0, i-1]$.

Based on Eq. 5, we choose k such that $D(n+1, k)$ is the maximal. Such value equals to the maximum joint probability $p(P^*, S)$ (Eq. 1) for the given sequence $S(1, n)$. Then the optimal segmentation P^* can be easily fetched by backtracking the matrix from $D(n+1, k)$. Since we need to fill $D_{(n+1)\times(n+1)}$, and computing each cell needs $O(n+1)$ time cost, the total time complexity is $O(n^3)$.

Avoiding Learning Cost by Unknown Parameter Estimation

Recall our phrasal segmentation model contains an unknown parameter $p(s_i|L = l(s_i))$. To learn this parameter, existing approaches usually employ an expectation-maximization (EM) based method (Liu et al. 2015), which keeps searching

for an optimal segmentation and updating unknown parameter until a stationary point has been reached. However, this iterative approach leads to an extremely heavy learning cost.

In order to avoid such learning cost, we could estimate $p(s_i|L = l(s_i))$ using those complete phrases and their frequencies derived by the complete phrase mining stage. Theorem 1 shows the relative error bound of our estimation. Let θ and θ' be the actual value and estimated value of $p(s_i|L = l(s_i))$, respectively.

Theorem 1. *The relative error bound ε of our parameter estimation is $\varepsilon(\rho, \epsilon) \leq \max\{\epsilon, \frac{\rho-\epsilon}{1-\rho}\}$, where $\epsilon = \frac{\tau}{\sum_{\forall l(s_j)=l(s_i)} f(s_j)}$, and $\rho = \frac{\tau}{f(s_i)}$, in which τ is the number of all overlapping phrases after complete phrase mining.*

Proof. The error is caused by the overlapping phrases, in extreme cases, the frequencies of overlapped parts are counted into only one phrase (e.g., left phrase or righ phrase). Therefore, we have

$$\min\{\frac{f(s_i)-\tau}{\Sigma-\tau}, \frac{f(s_i)}{\Sigma+\tau}\} \leq \theta \leq \frac{f(s_i)+\tau}{\Sigma+\tau}$$
$$\Rightarrow \min\{\frac{\theta'-\epsilon}{1-\epsilon}, \frac{\theta'}{1+\epsilon}\} \leq \theta \leq \frac{\theta'+\epsilon}{1+\epsilon}.$$

Thus, the approximation ratio

$$\begin{aligned}\eta &= \max\{\frac{\theta}{\theta'}, \frac{\theta'}{\theta}\} \\ &= max\{\frac{1-\epsilon}{1-\rho}, 1+\epsilon, \frac{1+\rho}{1+\epsilon}\} \\ &= \max\{1+\epsilon, \frac{1-\rho}{1-\epsilon}\ (0 \leq \rho \leq 1)\}.\end{aligned}$$

Utilizing the fact that $\varepsilon \leq \eta - 1$, we have $\varepsilon(\rho, \epsilon) \leq \max\{\epsilon, \frac{\rho-\epsilon}{1-\rho}\}$. Thus Theorem 1 holds. □

Dynamic Programming for Complete Phrase Mining

For complete phrase mining, since a corpus can be very large, the time cost of the aforementioned straightforward algorithm ($O(n^4)$) is prohibitively expensive. To address this issue, firstly, we propose a dynamic programming (DP) approach. This approach is based on the observation that if $S(i, j)$ is a phrase, there must exist an integer k ($i \leq k \leq j-1$), such that $S(i, k)$ and $S(k+1, j)$ are also phrases. Therefore, we set up a matrix M, in which a cell $M(i, j)$ stores a boolean value to denote whether $S(i, j)$ is a phrase or not. The recursion function is given as follows:

$$M(i,j) = \begin{cases} true, & if\ i = j \\ \bigvee_{k=i}^{j-1} \begin{pmatrix} M(i,k) \\ M(k+1,j) \\ \upsilon(S(i,k), S(k+1,j)) \end{pmatrix} & ozws., \end{cases} \quad (6)$$

where υ denotes a boolean function whose value is true if it satisfies our phraseness measurement, i.e. χ^2-test. In this way, each sub-problem needs to be solved only once, and the computation complexity reduces from $O(n^4)$ to $O(n^2)$.

Seed Extension for Complete Phrase Mining

We propose a seed extension based approach (SEBA) to further improve the efficiency. SEBA is based on the fact that phrase length (the number of words) follows a long-tailed distribution which means that the predominant majority of phrases have relatively fixed and short lengths. For any multi-token phrase (phrase length ≥ 2), it must contain at least one bi-gram phrase (phrase length $= 2$). We define these bi-gram phrases as seeds. Based on the above analysis, the main process of SEBA is that: (1) selecting bi-gram phrases as seeds; (2) extending each seed to subsequences bounded by a window with length w that centered on the seed, and computing their local solution using Eq. 6; and (3) checking whether a window needs to be extended. Given a window that contains words in $S(i, i+w)$, it needs to be extended if it satisfies $\upsilon(w_{i-2}, w_{i-1}) \vee \upsilon(w_{i-1}, w_i) \vee \upsilon(w_i, w_{i+1})$ or $\upsilon(w_{i+w-1}, w_{i+w}) \vee \upsilon(w_{i+w}, w_{i+w+1}) \vee \upsilon(w_{i+w+1}, w_{i+w+2})$. If a window needs to be extended, we extend the window to either its left or right until the new window does not satisfy the above conditions. Algorithm 1 describes our seed extension based approach.

Algorithm 1: SEBA

Input: A corpus C, window length w;
Output: A set of high quality phrases R in C;

```
1  foreach word sequence S ∈ C do
2      Initialize matrix M ← φ;
3      Initialize SeedsList ← φ;
4      for i = 1 to i = |S| − 1 do
5          if υ(w_i, w_{i+1}) then
6              preSeed ← get the last seed in SeedsList;
7              if i − preSeed.end < w then
8                  preSeed.end ← i;
9              else
10                 SeedsList ← (i, i);
11     foreach seed d_i ∈ SeedsList do
12         Compute M(d_i.start − ⌊w/2⌋, d_i.end + ⌈w/2⌉);
13     foreach seed d_i ∈ SeedsList do
14         while d_i.start do not need extension do
15             d_i.start ← d_i.start − 1;
16             Compute M(d_i.start, d_i.end);
17         while d_i.end do not need extension do
18             d_i.end ← d_i.end + 1;
19             Compute M(d_i.start, d_i.end);
20     R ← Back tacking optimal phrases;
21 return R;
```

Algorithm 1 shows when the current seed and the previous seed are within a window with length w, these two seeds may belong to a same phrase. Therefore, we simply concatenate them into one seed (lines 6–8).

Theorem 2. *Given a word sequence S, a window length w, a phrase ratio r (i.e. the average number of phrases divided by total number of words in S), and an average*

phrase length l. Let N be the expected number of seeds, $O(\sigma(w))$ be the cost of checking a window whether it needs to be extended, and e be the expected number of extra verification. The expected running time of Algorithm 1 is $O(n + N \cdot w^2 + (2N + e)\sigma(w) + e \cdot l^2)$, where

$$N = \begin{cases} r \cdot n, & if\ l \leq w \\ r \cdot n\lceil \frac{l}{w} \rceil, & otherwise \end{cases}$$

and

$$e = \begin{cases} l - w, & if\ l > w \\ 0, & otherwise. \end{cases}$$

Proof. The time cost of seed generation is n. The time cost of step (2) is $O(N \cdot w^2)$ since the algorithm requires $O(w^2)$ time to calculate each seed. Notice that, if $l \leq w$, each phrase generates no more than one seed, so the number of seeds is $r \cdot n$; otherwise, $r \cdot n\lceil \frac{l}{w} \rceil$ seeds will be generated. Algorithm 1 then needs to check the two boundaries of each seed in $O(2N \cdot \sigma(w))$ time. Moreover, if $l > w$, it requires an extra time cost $O((l - w)l^2)$ as well as a verification cost $O((l - w)\sigma(w))$. □

From Theorem 2 we can see that the window length w is the key parameter to determine the performance of the algorithm SEBA. In practice, l and r are relatively fixed values, so they can be regarded as constant values and estimated by empirical statistics, and n is already known. Therefore we hope to find a "good" w. We can do so by minimizing the cost of the algorithm. Then the theoretically optimal parameter w can be easily estimated as

$$\arg\min_w f(w) = n + N \cdot w^2 + (2N + e)\sigma(w)) + e \cdot l^2.$$

Experimental Evaluation

Data sets. We test four real-world data sets as follows.

- *5Conf*[1] is a set of paper titles that were published in conferences on the areas of artificial intelligence, databases, data mining, information retrieval, machine learning, and natural language processing;
- *APNews*[2] contains 106K TREC AP news articles that were published in 1989;
- *Titles*[3] is a full collection of paper titles that were extracted from DBLP data set; and
- *Abstracts*[4] contains 529K abstracts of computer science papers that were downloaded from DBLP data set.

The detailed statistics are summarized in Table 1.

Compared Methods. To demonstrate the quality and efficiency of our framework, we compared our method with the following state-of-the-art methods:

[1]http://web.engr.illinois.edu/ elkishk2/

[2]http://www.ap.org/

[3]http://dblp.uni-trier.de/db/

[4]http://dblp.uni-trier.de/db/

Table 1: Statics on the four data sets

Data sets	*5Conf*	*APNews*	*Titles*	*Abstracts*
# of Documents	44K	106K	1555K	529K
# of Vocabularies	5K	170K	96K	135K
Data Size	2.8M	229M	182M	479M

- ToPMine (El-Kishky et al. 2014) is a topical phrase mining method which performs phrase mining and then infer topic modeling strategy. Since we only consider phrase mining in this paper, we used its phrase mining part for comparison.
- SegPhrase+ (Liu et al. 2015) is a phrase mining method, which utilizes phrasal segmentation to prune over-estimated phrases based on rectified frequency, and adds segmentation features to refine quality estimation.

Experimental Settings. In our experiments, we set significance level $\alpha = 0.05$ for all data sets. We used a frequency threshold f_t to specify that only those phrases whose frequencies are larger than f_t were regarded as candidate phrases. We set a wide rang of f_t from 2 to 150 to comprehensively evaluate the effectiveness.

In complete phrase mining stage, we used the theoretically optimal parameter w (see Theorem 2) as the window length w. We set average phrase length $l = 2.1$ and phrase ratio $r = 0.2$ based on our empirical statistics on data sets. For the other compared methods, we used their default settings or the setting reported in their papers.

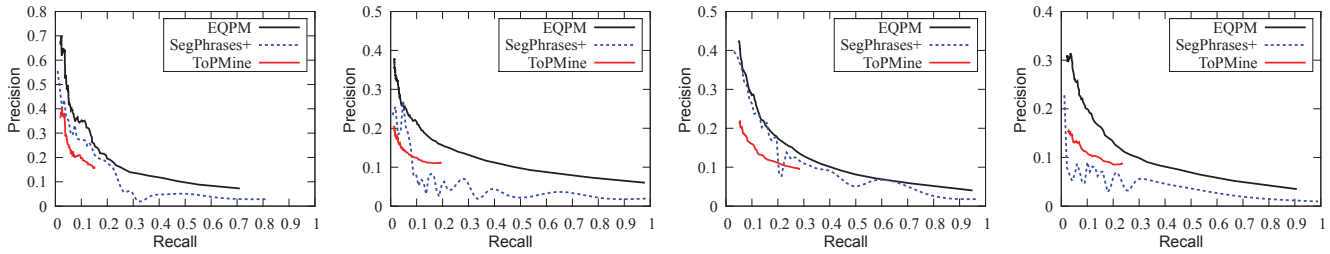
Our algorithms were implemented using Java SE Development Kit 8. The experiments were run on a PC with an Intel Xeon 3.3GHz 6-Cores CPU X5680 and 24GB memory with a 1TB disk, running Ubuntu (Linux) operating system.

Phrases Quality Evaluation

We conducted a phrase quality evaluation on *Wiki phrases* benchmark (Liu et al. 2015) along with an expert evaluation.

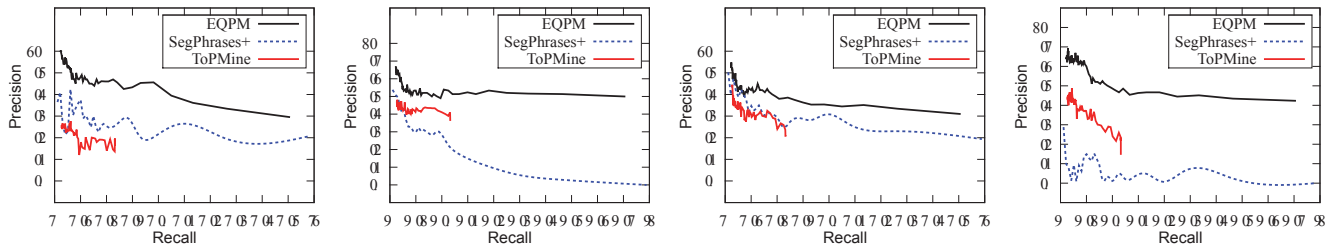
Wiki Phrases: We use *Wiki phrases* as ground truth labels, which were got from the authors of (Liu et al. 2015). *Wiki phrases* refer to popular mentions of entities by crawling intra-Wiki citations within Wiki content. A mined phrase is considered to be positive if it is the same with a *Wiki phrase*. Then precision is defined as the number of positive phrases to the number of mined phrases, and recall is the ratio of the number of positive phrases to the number of mined *Wiki phrases* returned by all comparison approaches and ours. Precision and recall are biased in this case because positive labels are restricted to *Wiki phrases*, however, they can still provide some insights regarding the performance between EQPM and baselines.

Fig. 1 shows the precision-recall curves (PR-curves) based on *Wiki phrases* evaluation. The curves are created by plotting the precision against recall at various frequency threshold settings. We can see that EQPM outperforms all baselines, and the trends on all data sets are similar. To be specific, EQPM could achieve a higher recall while its precision is maintained at a satisfactory level. Conversely, given



(a) *5Conf*. (b) *APNews*. (c) *Titles*. (d) *Abstracts*.

Figure 1: Precision-recall curves of four data sets evaluated by Wiki phrases benchmark.



(a) *5Conf*. (b) *APNews*. (c) *Titles*. (d) *Abstracts*.

Figure 2: Precision-recall curves of four data sets evaluated by experts.

a recall, the precision of our method is higher than other methods. It indicates that EQPM could find more quality phrases than baselines. Not surprisingly, our method could achieve a better performance than TopMine, indicating our complete phrase mining algorithm really beats the heuristic approach adopted by TopMine. Besides, the higher performance compared with SegPhrase+ demonstrates EQPM could effectively eliminate inappropriate segmentation. We also observed that, SegPhrase+ fluctuates more heavily than EQPM, which demonstrates our method is less sensitive to f_t than SegPhrase+.

Expert Evaluation: For each method, we randomly sampled 500 Wiki-uncovered phrases from the candidates to form a pool. If the number of Wiki-uncovered phrases was smaller than 500, all of them were put into the pool, in which each phrase was then evaluated by 5 reviewers (computer science Ph.D. candidates in year 2 or above). The metric was whether the phrases were natural, meaningful, and unambiguous. The reviewers independently evaluated the phrase quality, based on their background knowledge, and possibly with the help of search engine. We took the majority of opinions as results and accordingly evaluated the precision of phrases by the methods. The experiment result of expert evaluation is shown in Fig. 2.

Comparing Fig. 1 and Fig. 2, an interesting observation is that, the difference between EQPM and baselines is more significant on expert evaluation. This is because *Wiki phrases* is not a complete source of phrases, many phrases especially terminologies have not been covered, and this is also the reason why we need to conduct an expert evaluation. From Fig. 2, we can see that EQPM outperforms baselines significantly, it could achieve a higher recall with a fairly high precision. In practice, EQPM could mine 90% of those Wiki-uncovered phrases out and keep a very high precision level (nearly 70%). Therefore, the evaluation results suggest that our methods not only detect the well-known *Wiki phrases*, but also work properly for the long tail phrases which might occur not so frequently.

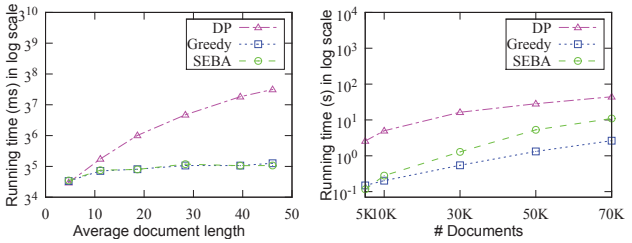
Table 2: Relative weight w.r.t. expert evaluation

Data sets	Precision		Recall	
	CPM	**PS**	**CPM**	**PS**
5Conf	97.46%	2.54%	95.76%	4.24%
APNews	96.35%	3.65%	95.41%	4.59%
Titles	98.02%	1.98%	96.64%	3.36%
Abstracts	96.76%	3.24%	95.79%	4.21%

We conduct that both phrasal segmentation (PS) and complete phrase mining (CPM) stage can improve precision and recall. Table 2 shows their relative weight using expert evaluation. For the four data sets, among all generated phrases CPM contributed above 96.35% on precision, and around 95.41% on recall, whereas PS contributed the remaining. This is because the number of overlapped phrases was much smaller than the whole data set. Whereas CPM focuses on order sensitive which was very common in data sets.

Efficiency Evaluation

We firstly verified the efficiency of the proposed two complete phrase mining algorithms DP and SEBA by comparing them with the Greedy algorithm in ToPMine. Fig. 3(a)



(a) Varying document length. (b) Varying # of documents.

Figure 3: Efficiency of complete phrase mining algorithms

shows the running time on the data sets that have different average document lengths but with a constant total data size (2.8MB). With the average document length increasing, the time cost of DP grows much faster than others, whereas Greedy and SEBA are less sensitive to the average document length. This is because DP conducts dynamic programming on whole word sequences, whereas SEBA extends each seed to a window and then conducts dynamic programming on word subsequence within the window. Not surprisingly, SEBA could achieve almost the same time cost as Greedy, even though the latter has a theoretical $O(n)$ time complexity. Fig. 3(b) shows running time varies with data size (with a 40 average document length). In this setting, SEBA is more efficient than DP. The results shown in Fig. 3 demonstrate that SEBA algorithm can greatly reduce time cost than DP, moreover, the efficiency of SEBA is competitive even to the greedy algorithm.

We then examined the efficiency of our method compared with the two state-of-the-art methods ToPMine and SegPhrase+. To make the comparison fair, in this evaluation, neither our method nor the baselines use parallelization. ToPMine does not discuss parallelism. Segphrase+ claims the penalty learning and parameter training could be parallelized. In our method, both complete phrase mining and overlapped phrases segmentation (which are also the most time consuming parts) can be easily parallelized, since our method could independently run on individual sentence and document.

Table 3 shows the running time of the whole quality phrase mining method compared with EQPM, ToPMine, and SegPhrase+ on different data sets. As expected, the utilization of efficient phrasal segmentation and efficient complete phrase mining methods account for EQPM's better efficiency. Unsurprisingly, our method has a huge advantage compared with SegPhrase+ ($13.5\times \sim 29\times$ faster), which has a huge learning cost. Comparing with ToPMine, EQPM is $2.3\times \sim 17.5\times$ faster.

Table 3: Running time

Methods \ Data sets	*5Conf*	*APNews*	*Titles*	*Abstracts*
EQPM	0.31s	34s	56s	240s
ToPMine	5.434s	4min25s	5min6s	9min34s
SegPhrases+	9.02s	19min14s	25min56s	54min4s

Table 4: Running time of different components in EQPM

Component \ Data sets	*5Conf*	*APNews*	*Titles*	*Abstracts*
Frequency Counting	0.101s	13.742s	34.315s	89.621s
Complete Phrase Mining (SEBA)	0.163s	16.637s	14.358s	101.139s
Phrasal Segmentation	0.046s	5.825s	7.495s	49.923s

Besides, Table 4 shows the time cost of different components of EQPM. We can see that the time cost of phrasal segmentation was less than the other two components, since the number of such phrases only takes a small portion (on average only 1% to 2%) among all the phrases.

Conclusion

In this paper, we propose an efficient integrated framework for high quality topical phrase mining, which adopts complete phrase mining to guarantee completeness, and utilizes a novel phrasal segmentation model to handle overlapping phrases. Moreover, by means of an accurate parameter estimation and two efficient algorithmic designs, the efficiency could be greatly improved. The experimental evaluation demonstrates that, compared with two state-of-the-art methods, our framework is of the highest quality and the highest efficiency as well.

Acknowledgments

This work is partially supported by the NSF of China for Outstanding Young Scholars under grant No. 61322208, the NSF of China for Key Program under grant No. 61532021, and the NSF of China under grant Nos. 61272178 and 61572122. Xiaochun Yang is the corresponding author of this work.

References

Abney, S. P. 1991. *Parsing By Chunks*. Kluwer Academic Publishers.

Ahonen, H. 1999. Knowledge discovery in documents by extracting frequent word sequences. *Library trends* 48(1):160–160.

Blackwood, G. W.; Gispert, A. D.; and Byrne, W. 2008. Phrasal segmentation models for statistical machine translation. In *The 22nd International Conference on Computational Linguistics, Manchester, UK*, 19–22.

Brill, E. 2002. A simple rule-based part of speech tagger. In *Conference on Applied Natural Language Processing*, 152–155.

Church, K. W. 1989. A stochastic parts program and noun phrase parser for unrestricted text. In *Conference on Applied Natural Language Processing*, 136–143.

Clahsen; Harald; Felser; and Claudia. 2006. Grammatical processing in language learners. *Applied Psycholinguistics* 27(1):3–41.

Deane, P. 2005. A nonparametric method for extraction of candidate phrasal terms. In *The 43rd Annual Meeting of Association for Computational Linguistics, University of Michigan, USA*, 605–613. Stroudsburg, PA, USA: ACL.

Echizen-ya, H., and Araki, K. 2010. Automatic evaluation method for machine translation using noun-phrase chunking. In *The 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden*, 108–117.

El-Kishky, A.; Song, Y.; Wang, C.; Voss, C. R.; and Han, J. 2014. Scalable topical phrase mining from text corpora. *Proceedings of the 40th VLDB Endowment Conference, Hangzhou, China* 8(3):305–316.

F.R.S., K. P. 1900. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *Philosophical Magazine Series 5* 50(302):157–175.

He, Z.; Chen, C.; Bu, J.; Wang, C.; Zhang, L.; Cai, D.; and He, X. 2012. Document summarization based on data reconstruction. *The 26th AAAI Conference on Artificial Intelligence, Toronto, Canada* 620–626.

Huang, Z.; Xu, W.; and Yu, K. 2015. Bidirectional lstm-crf models for sequence tagging. *Computer Science*.

Kudoh, T., and Matsumoto, Y. 2002. Use of support vector learning for chunk identification. *The 2002 Conference on Computational Natural Language Learning, Taipei, Taiwan* 142–144.

Leskovec, J.; Backstrom, L.; and Kleinberg, J. M. 2009. Meme-tracking and the dynamics of the news cycle. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, Paris, France*, 497–506.

Li, Y.; Hsu, B.-J. P.; Zhai, C.; and Wang, K. 2011. Unsupervised query segmentation using clickthrough for information retrieval. The 34th Annual ACM SIGIR Conference, Beijing, China, 285–294. New York, NY, USA: ACM.

Li, B.; Wang, B.; Zhou, R.; Yang, X.; and Liu, C. 2016a. Citpm: A cluster-based iterative topical phrase mining framework. In *The 21st International Conference on Database Systems for Advanced Applications, Dallas, USA*, 197–213. Springer.

Li, M.; Wang, J.; Tong, W.; Yu, H.; Ma, X.; Chen, Y.; Cai, H.; and Han, J. 2016b. EKNOT: event knowledge from news and opinions in twitter. In *Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, Arizona, USA.*, 4367–4368.

Liu, J.; Shang, J.; Wang, C.; Ren, X.; and Han, J. 2015. Mining quality phrases from massive text corpora. In *The 36th ACM SIGMOD/PODS conference, Melbourne, Victoria, Australia*, 1729–1744.

Mcdonald, R.; Crammer, K.; and Pereira, F. 2005. Online large-margin training of dependency parsers. In *The 43rd Annual Meeting of Association for Computational Linguistics, University of Michigan, USA*, 91–98.

O'Neil, T., and Sangiovanni-Vincentelli, A. L. 2014. Automatic construction and ranking of topical keyphrases on collections of short documents. *2014 SIAM Annual Meeting, Chicago, USA*.

Parameswaran, A.; Garcia-Molina, H.; and Rajaraman, A. 2010. Towards the web of concepts: Extracting concepts from large datasets. *Proceedings of the 36th VLDB Endowment Conference, Singapore* 3(1-2):566–577.

Pitler, E.; Bergsma, S.; and Church, K. 2010. Using web-scale n-grams to improve base np parsing performance. In *The 1st International Conference on Computational Linguistics, Shanghai, China*, 886–894.

Punyakanok, V., and Roth, D. 2001. The Use of Classifiers in Sequential Inference. *Neural Information Processing Systems 2001* cs.LG:995–1001.

Sha, F., and Pereira, F. 2003. Shallow parsing with conditional random fields. NAACL 2003, 134–141. Stroudsburg, PA, USA: Association for Computational Linguistics.

Shen, H., and Sarkar, A. 2005. Voting between multiple data representations for text chunking. In *Advances in Artificial Intelligence: 18th Conference of the Canadian Society for Computational Studies of Intelligence, Victoria, Canada*, 389–400.

Simitsis, A.; Baid, A.; Sismanis, Y.; and Reinwald, B. 2008. Multidimensional content exploration. *Proceedings of the 34th VLDB Endowment Conference, Auckland, New Zealand* 1(1):660–671.

Sun, X.; Morency, L. P.; Okanohara, D.; and Tsujii, J. 2008. Modeling latent-dynamic in shallow parsing: a latent conditional model with improved inference. In *International Conference on Computational Linguistics*, 841–848.

Tan, B., and Peng, F. 2008. Unsupervised query segmentation using generative language models and wikipedia. In *International Conference on World Wide Web, Beijing, China*, 347–356.

Tjong K. S., E. F., and Buchholz, S. 2000. Introduction to the conll-2000 shared task: Chunking. The 2000 Conference on Computational Natural Language Learning, 127–132. Stroudsburg, PA, USA: ACL.

Vishwanathan, S. V. N.; Schraudolph, N. N.; Schmidt, M. W.; and Murphy, K. P. 2006. Accelerated training of conditional random fields with stochastic gradient methods. In *Proceedings of the 23rd international conference on Machine learning, Pittsburgh, USA*, 969–976.

Wang, C.; Danilevsky, M.; Desai, N.; Zhang, Y.; Nguyen, P.; Taula, T.; and Han, J. 2013. A phrase mining framework for recursive construction of a topical hierarchy. In *The 19th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Chicago, USA*, 437.

Witten, I. H.; Paynter, G. W.; Frank, E.; Gutwin, C.; and Nevill-Manning, C. G. 1999. Kea: practical automatic keyphrase extraction. In *ACM Conference on Digital Libraries*, 254–255.