Topical Keyphrase Extraction from Twitter

Wayne Xin Zhao[†] Jing Jiang[‡] Jing He[†] Yang Song[†] Palakorn Achananuparp[‡]
Ee-Peng Lim[‡] Xiaoming Li[†]
[†]School of Electronics Engineering and Computer Science, Peking University
[‡]School of Information Systems, Singapore Management University
{batmanfly,peaceful.he,songyangmagic}@gmail.com,
{jingjiang,eplim,palakorna}@smu.edu.sg, lxm@pku.edu.cn

Abstract

Summarizing and analyzing Twitter content is an important and challenging task. In this paper, we propose to extract topical keyphrases as one way to summarize Twitter. We propose a context-sensitive topical PageRank method for keyword ranking and a probabilistic scoring function that considers both relevance and interestingness of keyphrases for keyphrase ranking. We evaluate our proposed methods on a large Twitter data set. Experiments show that these methods are very effective for topical keyphrase extraction.

1 Introduction

Twitter, a new microblogging website, has attracted hundreds of millions of users who publish short messages (a.k.a. *tweets*) on it. They either publish original tweets or *retweet* (i.e. forward) others' tweets if they find them interesting. Twitter has been shown to be useful in a number of applications, including tweets as social sensors of real-time events (Sakaki et al., 2010), the sentiment prediction power of Twitter (Tumasjan et al., 2010), etc. However, current explorations are still in an early stage and our understanding of Twitter content still remains limited. How to automatically understand, extract and summarize useful Twitter content has therefore become an important and emergent research topic.

In this paper, we propose to extract keyphrases as a way to summarize Twitter content. Traditionally, keyphrases are defined as a short list of terms to summarize the topics of a document (Turney, 2000). It can be used for various tasks such as document summarization (Litvak and Last, 2008) and indexing (Li et al., 2004). While it appears natural to use keyphrases to summarize Twitter content, compared with traditional text collections, keyphrase extraction from Twitter is more challenging in at least two aspects: 1) Tweets are much shorter than traditional articles and not all tweets contain useful information; 2) Topics tend to be more diverse in Twitter than in formal articles such as news reports.

So far there is little work on keyword or keyphrase extraction from Twitter. Wu et al. (2010) proposed to automatically generate personalized tags for Twitter users. However, user-level tags may not be suitable to summarize the overall Twitter content within a certain period and/or from a certain group of people such as people in the same region. Existing work on keyphrase extraction identifies keyphrases from either individual documents or an entire text collection (Turney, 2000; Tomokiyo and Hurst, 2003). These approaches are not immediately applicable to Twitter because it does not make sense to extract keyphrases from a single tweet, and if we extract keyphrases from a whole tweet collection we will mix a diverse range of topics together, which makes it difficult for users to follow the extracted keyphrases.

Therefore, in this paper, we propose to study the novel problem of extracting *topical keyphrases* for summarizing and analyzing Twitter content. In other words, we extract and organize keyphrases by topics learnt from Twitter. In our work, we follow the standard three steps of keyphrase extraction, namely, keyword ranking, candidate keyphrase generation and keyphrase ranking. For keyword ranking, we modify the Topical PageRank method proposed by Liu et al. (2010) by introducing topic-sensitive score propagation. We find that topic-sensitive propagation can largely help boost the performance. For keyphrase ranking, we propose a principled probabilistic phrase ranking method, which can be flexibly combined with any keyword ranking method and candidate keyphrase generation method. Experiments on a large Twitter data set show that our proposed methods are very effective in topical keyphrase extraction from Twitter. Interestingly, our proposed keyphrase ranking method can incorporate users' interests by modeling the *retweet* behavior. We further examine what topics are suitable for incorporating users' interests for topical keyphrase extraction.

To the best of our knowledge, our work is the first to study how to extract keyphrases from microblogs. We perform a thorough analysis of the proposed methods, which can be useful for future work in this direction.

2 Related Work

Our work is related to unsupervised keyphrase extraction. Graph-based ranking methods are the state of the art in unsupervised keyphrase extraction. Mihalcea and Tarau (2004) proposed to use TextRank, a modified PageRank algorithm to extract keyphrases. Based on the study by Mihalcea and Tarau (2004), Liu et al. (2010) proposed to decompose a traditional random walk into multiple random walks specific to various topics. Language modeling methods (Tomokiyo and Hurst, 2003) and natural language processing techniques (Barker and Cornacchia, 2000) have also been used for unsupervised keyphrase extraction. Our keyword extraction method is mainly based on the study by Liu et al. (2010). The difference is that we model the score propagation with topic context, which can lower the effect of noise, especially in microblogs.

Our work is also related to automatic topic labeling (Mei et al., 2007). We focus on extracting topical keyphrases in microblogs, which has its own challenges. Our method can also be used to label topics in other text collections.

Another line of relevant research is Twitter-related text mining. The most relevant work is by Wu et al. (2010), who directly applied TextRank (Mihalcea and Tarau, 2004) to extract keywords from tweets to tag users. Topic discovery from Twitter is also related to our work (Ramage et al., 2010), but we further extract keyphrases from each topic for summarizing and analyzing Twitter content.

3 Method

3.1 Preliminaries

Let $\mathcal{U}$ be a set of Twitter users. Let $\mathcal{C} = \{\{d_{u,m}\}_{m=1}^{M_u}\}_{u\in\mathcal{U}}$ be a collection of tweets generated by $\mathcal{U}$, where M_u is the total number of tweets generated by user u and $d_{u,m}$ is the m-th tweet of u. Let $\mathcal{V}$ be the vocabulary. $d_{u,m}$ consists of a sequence of words $(w_{u,m,1}, w_{u,m,2}, \ldots, w_{u,m,N_{u,m}})$ where $N_{u,m}$ is the number of words in $d_{u,m}$ and $w_{u,m,n} \in \mathcal{V}$ $(1 \leq n \leq N_{u,m})$. We also assume that there is a set of topics $\mathcal{T}$ over the collection $\mathcal{C}$.

Given $\mathcal{T}$ and $\mathcal{C}$, topical keyphrase extraction is to discover a list of keyphrases for each topic $t \in \mathcal{T}$. Here each keyphrase is a sequence of words.

To extract keyphrases, we first identify topics from the Twitter collection using topic models (Section 3.2). Next for each topic, we run a topical PageRank algorithm to rank keywords and then generate candidate keyphrases using the top ranked keywords (Section 3.3). Finally, we use a probabilistic model to rank the candidate keyphrases (Section 3.4).

3.2 Topic discovery

We first describe how we discover the set of topics $\mathcal{T}$. Author-topic models have been shown to be effective for topic modeling of microblogs (Weng et al., 2010; Hong and Davison, 2010). In Twitter, we observe an important characteristic of tweets: tweets are short and a single tweet tends to be about a single topic. So we apply a modified author-topic model called Twitter-LDA introduced by Zhao et al. (2011), which assumes a single topic assignment for an entire tweet.

The model is based on the following assumptions. There is a set of topics $\mathcal{T}$ in Twitter, each represented by a word distribution. Each user has her topic interests modeled by a distribution over the topics. When a user wants to write a tweet, she first chooses a topic based on her topic distribution. Then she chooses a

```
1. Draw $\phi^{\mathcal{B}} \sim \mathrm{Dir}(\beta), \pi \sim \mathrm{Dir}(\gamma)$
2. For each topic $t \in \mathcal{T}$,
   (a) draw $\phi^t \sim \mathrm{Dir}(\beta)$
3. For each user $u \in \mathcal{U}$,
   (a) draw $\theta^u \sim \mathrm{Dir}(\alpha)$
   (b) for each tweet $d_{u,m}$
       i. draw $z_{u,m} \sim \mathrm{Multi}(\theta^u)$
       ii. for each word $w_{u,m,n}$
           A. draw $y_{u,m,n} \sim \mathrm{Bernoulli}(\pi)$
           B. draw $w_{u,m,n} \sim \mathrm{Multi}(\phi^{\mathcal{B}})$ if $y_{u,m,n} = 0$ and $w_{u,m,n} \sim \mathrm{Multi}(\phi^{z_{u,m}})$ if $y_{u,m,n} = 1$
```

Figure 1: The generation process of tweets.

bag of words one by one based on the chosen topic. However, not all words in a tweet are closely related to the topic of that tweet; some are background words commonly used in tweets on different topics. Therefore, for each word in a tweet, the user first decides whether it is a background word or a topic word and then chooses the word from its respective word distribution.

Formally, let ϕ^t denote the word distribution for topic t and $\phi^{\mathcal{B}}$ the word distribution for background words. Let θ^u denote the topic distribution of user u. Let π denote a Bernoulli distribution that governs the choice between background words and topic words. The generation process of tweets is described in Figure 1. Each multinomial distribution is governed by some symmetric Dirichlet distribution parameterized by α, β or γ.

3.3 Topical PageRank for Keyword Ranking

Topical PageRank was introduced by Liu et al. (2010) to identify keywords for future keyphrase extraction. It runs topic-biased PageRank for each topic separately and boosts those words with high relevance to the corresponding topic. Formally, the topic-specific PageRank scores can be defined as follows:

$$R_t(w_i) = \lambda \sum_{j:w_j \to w_i} \frac{e(w_j, w_i)}{O(w_j)} R_t(w_j) + (1-\lambda) P_t(w_i), \tag{1}$$

where $R_t(w)$ is the topic-specific PageRank score of word w in topic t, $e(w_j, w_i)$ is the weight for the edge $(w_j \to w_i)$, $O(w_j) = \sum_{w'} e(w_j, w')$ and λ is a damping factor ranging from 0 to 1. The topic-specific preference value $P_t(w)$ for each word w is its random jumping probability with the constraint that $\sum_{w \in \mathcal{V}} P_t(w) = 1$ given topic t. A large $R_t(\cdot)$ indicates a word is a good candidate keyword in topic t. We denote this original version of the Topical PageRank as *TPR*.

However, the original *TPR* ignores the topic context when setting the edge weights; the edge weight is set by counting the number of co-occurrences of the two words within a certain window size. Taking the topic of "electronic products" as an example, the word "juice" may co-occur frequently with a good keyword "apple" for this topic because of Apple electronic products, so "juice" may be ranked high by this context-free co-occurrence edge weight although it is not related to electronic products. In other words, context-free propagation may cause the scores to be off-topic.

So in this paper, we propose to use a topic context sensitive PageRank method. Formally, we have

$$R_t(w_i) = \lambda \sum_{j:w_j \to w_i} \frac{e_t(w_j, w_i)}{O_t(w_j)} R_t(w_j) + (1-\lambda) P_t(w_i). \tag{2}$$

Here we compute the propagation from w_j to w_i in the context of topic t, namely, the edge weight from w_j to w_i is parameterized by t. In this paper, we compute edge weight $e_t(w_j, w_i)$ between two words by counting the number of co-occurrences of these two words in tweets *assigned to topic* t. We denote this context-sensitive topical PageRank as *cTPR*.

After keyword ranking using *cTPR* or any other method, we adopt a common candidate keyphrase generation method proposed by Mihalcea and Tarau (2004) as follows. We first select the top S keywords for each topic, and then look for combinations of these keywords that occur as frequent phrases in the text collection. More details are given in Section 4.

3.4 Probabilistic Models for Topical Keyphrase Ranking

With the candidate keyphrases, our next step is to rank them. While a standard method is to simply aggregate the scores of keywords inside a candidate keyphrase as the score for the keyphrase, here we propose a different probabilistic scoring function. Our method is based on the following hypotheses about good keyphrases given a topic:

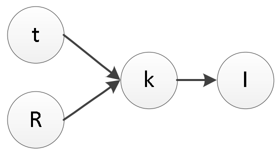

Figure 2: Assumptions of variable dependencies.

Relevance: A good keyphrase should be closely related to the given topic and also discriminative. For example, for the topic "news," "*president obama*" is a good keyphrase while "*math class*" is not.
Interestingness: A good keyphrase should be interesting and can attract users' attention. For example, for the topic "music," "*justin bieber*" is more interesting than "*song player*."

Sometimes, there is a trade-off between these two properties and a good keyphrase has to balance both.

Let R be a binary variable to denote relevance where 1 is relevant and 0 is irrelevant. Let I be another binary variable to denote interestingness where 1 is interesting and 0 is non-interesting. Let k denote a candidate keyphrase. Following the probabilistic relevance models in information retrieval (Lafferty and Zhai, 2003), we propose to use $P(R = 1, I = 1|t, k)$ to rank candidate keyphrases for topic t. We have

$$\begin{aligned}
& P(R=1, I=1|t,k) \\
= \; & P(R=1|t,k)P(I=1|t,k,R=1) \\
= \; & P(I=1|t,k,R=1)P(R=1|t,k) \\
= \; & P(I=1|k)P(R=1|t,k) \\
= \; & P(I=1|k) \times \frac{P(R=1|t,k)}{P(R=1|t,k)+P(R=0|t,k)} \\
= \; & P(I=1|k) \times \frac{1}{1+\frac{P(R=0|t,k)}{P(R=1|t,k)}} \\
= \; & P(I=1|k) \times \frac{1}{1+\frac{P(R=0,k|t)}{P(R=1,k|t)}} \\
= \; & P(I=1|k) \times \frac{1}{1+\frac{P(R=0|t)}{P(R=1|t)} \times \frac{P(k|t,R=0)}{P(k|t,R=1)}} \\
= \; & P(I=1|k) \times \frac{1}{1+\frac{P(R=0)}{P(R=1)} \times \frac{P(k|t,R=0)}{P(k|t,R=1)}}.
\end{aligned}$$

Here we have assumed that I is independent of t and R given k, i.e. the interestingness of a keyphrase is independent of the topic or whether the keyphrase is relevant to the topic. We have also assumed that R is independent of t when k is unknown, i.e. without knowing the keyphrase, the relevance is independent of the topic. Our assumptions can be depicted by Figure 2.

We further define $\delta = \frac{P(R=0)}{P(R=1)}$. In general we can assume that $P(R = 0) \gg P(R = 1)$ because there are much more non-relevant keyphrases than relevant ones, that is, $\delta \gg 1$. In this case, we have

$$\begin{aligned}
& \log P(R=1, I=1|t,k) \qquad (3) \\
= \; & \log \left(P(I=1|k) \times \frac{1}{1+\delta \times \frac{P(k|t,R=0)}{P(k|t,R=1)}} \right) \\
\approx \; & \log \left(P(I=1|k) \times \frac{P(k|t,R=1)}{P(k|t,R=0)} \times \frac{1}{\delta} \right) \\
= \; & \log P(I=1|k) + \log \frac{P(k|t,R=1)}{P(k|t,R=0)} - \log \delta.
\end{aligned}$$

We can see that the ranking score $\log P(R = 1, I = 1|t, k)$ can be decomposed into two components, a relevance score $\log \frac{P(k|t,R=1)}{P(k|t,R=0)}$ and an interestingness score $\log P(I = 1|k)$. The last term $\log \delta$ is a constant and thus not relevant.

Estimating the relevance score

Let a keyphrase candidate k be a sequence of words $(w_1, w_2, \ldots, w_N)$. Based on an independent assumption of words given R and t, we have

$$\begin{aligned}
\log \frac{P(k|t,R=1)}{P(k|t,R=0)} &= \log \frac{P(w_1 w_2 \ldots w_N|t,R=1)}{P(w_1 w_2 \ldots w_N|t,R=0)} \\
&= \sum_{n=1}^{N} \log \frac{P(w_n|t,R=1)}{P(w_n|t,R=0)}. \qquad (4)
\end{aligned}$$

Given the topic model ϕ^t previously learned for topic t, we can set $P(w|t, R = 1)$ to ϕ^t_w, i.e. the probability of w under ϕ^t. Following Griffiths and Steyvers (2004), we estimate ϕ^t_w as

$$\phi^t_w = \frac{\#(\mathcal{C}_t, w) + \beta}{\#(\mathcal{C}_t, \cdot) + \beta|\mathcal{V}|}. \qquad (5)$$

Here $\mathcal{C}_t$ denotes the collection of tweets assigned to topic t, $\#(\mathcal{C}_t, w)$ is the number of times w appears in $\mathcal{C}_t$, and $\#(\mathcal{C}_t, \cdot)$ is the total number of words in $\mathcal{C}_t$.

$P(w|t, R = 0)$ can be estimated using a smoothed background model.

$$P(w|R=0,t) = \frac{\#(\mathcal{C}, w) + \mu}{\#(\mathcal{C}, \cdot) + \mu|\mathcal{V}|}. \qquad (6)$$

Here $\#(\mathcal{C}, \cdot)$ denotes the number of words in the whole collection $\mathcal{C}$, and $\#(\mathcal{C}, w)$ denotes the number of times w appears in the whole collection.

After plugging Equation (5) and Equation (6) into Equation (4), we get the following formula for the relevance score:

$$
\begin{aligned}
&\log \frac{P(k|t, R=1)}{P(k|t, R=0)} \\
= & \sum_{w \in k} \Big(\log \frac{\#(\mathcal{C}_t, w) + \beta}{\#(\mathcal{C}, w) + \mu} + \log \frac{\#(\mathcal{C}, \cdot) + \mu|\mathcal{V}|}{\#(\mathcal{C}_t, \cdot) + \beta|\mathcal{V}|} \Big) \\
= & \Big(\sum_{w \in k} \log \frac{\#(\mathcal{C}_t, w) + \beta}{\#(\mathcal{C}, w) + \mu} \Big) + |k|\eta, \qquad (7)
\end{aligned}
$$

where $\eta = \frac{\#(\mathcal{C}, \cdot) + \mu|\mathcal{V}|}{\#(\mathcal{C}_t, \cdot) + \beta|\mathcal{V}|}$ and $|k|$ denotes the number of words in k.

Estimating the interestingness score

To capture the interestingness of keyphrases, we make use of the retweeting behavior in Twitter. We use string matching with `RT` to determine whether a tweet is an original posting or a retweet. If a tweet is interesting, it tends to get retweeted multiple times. Retweeting is therefore a stronger indicator of user interests than tweeting. We use retweet ratio $\frac{|\text{ReTweets}_k|}{|\text{Tweets}_k|}$ to estimate $P(I = 1|k)$. To prevent zero frequency, we use a modified add-one smoothing method. Finally, we get

$$
\log P(I = 1|k) = \log \frac{|\text{ReTweets}_k| + 1.0}{|\text{Tweets}_k| + l_{avg}}. \qquad (8)
$$

Here $|\text{ReTweets}_k|$ and $|\text{Tweets}_k|$ denote the numbers of retweets and tweets containing the keyphrase k, respectively, and l_{avg} is the average number of tweets that a candidate keyphrase appears in.

Finally, we can plug Equation (7) and Equation (8) into Equation (3) and obtain the following scoring function for ranking:

$$
\begin{aligned}
\text{Score}_t(k) = & \log \frac{|\text{ReTweets}_k| + 1.0}{|\text{Tweets}_k| + l_{avg}} \qquad (9) \\
& + \left(\sum_{w \in k} \log \frac{\#(\mathcal{C}_t, w) + \beta}{\#(\mathcal{C}, w) + \mu} \right) + |k|\eta.
\end{aligned}
$$

#user	#tweet	#term	#token
13,307	1,300,300	50,506	11,868,910

Table 1: Some statistics of the data set.

Incorporating length preference

Our preliminary experiments with Equation (9) show that this scoring function usually ranks longer keyphrases higher than shorter ones. However, because our candidate keyphrase are extracted without using any linguistic knowledge such as noun phrase boundaries, longer candidate keyphrases tend to be less meaningful as a phrase. Moreover, for our task of using keyphrases to summarize Twitter, we hypothesize that shorter keyphrases are preferred by users as they are more compact. We would therefore like to incorporate some length preference.

Recall that Equation (9) is derived from $P(R = 1, I = 1|t, k)$, but this probability does not allow us to directly incorporate any length preference. We further observe that Equation (9) tends to give longer keyphrases higher scores mainly due to the term $|k|\eta$. So here we heuristically incorporate our length preference by removing $|k|\eta$ from Equation (9), resulting in the following final scoring function:

$$
\begin{aligned}
\text{Score}_t(k) = & \log \frac{|\text{ReTweets}_k| + 1.0}{|\text{Tweets}_k| + l_{avg}} \qquad (10) \\
& + \left(\sum_{w \in k} \log \frac{\#(\mathcal{C}_t, w) + \beta}{\#(\mathcal{C}, w) + \mu} \right).
\end{aligned}
$$

4 Experiments

4.1 Data Set and Preprocessing

We use a Twitter data set collected from Singapore users for evaluation. We used *Twitter REST API*[1] to facilitate the data collection. The majority of the tweets collected were published in a 20-week period from December 1, 2009 through April 18, 2010. We removed common stopwords and words which appeared in fewer than 10 tweets. We also removed all users who had fewer than 5 tweets. Some statistics of this data set after cleaning are shown in Table 1.

We ran Twitter-LDA with 500 iterations of Gibbs sampling. After trying a few different numbers of

[1]http://apiwiki.twitter.com/w/page/22554663/REST-API-Documentation

topics, we empirically set the number of topics to 30. We set α to $50.0/|\mathcal{T}|$ as Griffiths and Steyvers (2004) suggested, but set β to a smaller value of 0.01 and γ to 20. We chose these parameter settings because they generally gave coherent and meaningful topics for our data set. We selected 10 topics that cover a diverse range of content in Twitter for evaluation of topical keyphrase extraction. The top 10 words of these topics are shown in Table 2.

We also tried the standard LDA model and the author-topic model on our data set and found that our proposed topic model was better or at least comparable in terms of finding meaningful topics. In addition to generating meaningful topics, Twitter-LDA is much more convenient in supporting the computation of tweet-level statistics (e.g. the number of co-occurrences of two words in a specific topic) than the standard LDA or the author-topic model because Twitter-LDA assumes a single topic assignment for an entire tweet.

4.2 Methods for Comparison

As we have described in Section 3.1, there are three steps to generate keyphrases, namely, keyword ranking, candidate keyphrase generation, and keyphrase ranking. We have proposed a context-sensitive topical PageRank method (*cTPR*) for the first step of keyword ranking, and a probabilistic scoring function for the third step of keyphrase ranking. We now describe the baseline methods we use to compare with our proposed methods.

Keyword Ranking

We compare our *cTPR* method with the original topical PageRank method (Equation (1)), which represents the state of the art. We refer to this baseline as *TPR*.

For both *TPR* and *cTPR*, the damping factor is empirically set to 0.1, which always gives the best performance based on our preliminary experiments. We use normalized $P(t|w)$ to set $P_t(w)$ because our preliminary experiments showed that this was the best among the three choices discussed by Liu et al. (2010). This finding is also consistent with what Liu et al. (2010) found.

In addition, we also use two other baselines for comparison: (1) *kwBL1*: ranking by $P(w|t) = \phi^t_w$. (2) *kwBL2*: ranking by $P(t|w) = \frac{P(t)\phi^t_w}{\sum_{t'} P(t')\phi^{t'}_w}$.

Keyphrase Ranking

We use *kpRelInt* to denote our relevance and interestingness based keyphrase ranking function $P(R = 1, I = 1|t, k)$, i.e. Equation (10). β and μ are empirically set to 0.01 and 500. Usually μ can be set to zero, but in our experiments we find that our ranking method needs a more uniform estimation of the background model. We use the following ranking functions for comparison:

- *kpBL1*: Similar to what is used by Liu et al. (2010), we can rank candidate keyphrases by $\sum_{w \in k} f(w)$, where $f(w)$ is the score assigned to word w by a keyword ranking method.
- *kpBL2*: We consider another baseline ranking method by $\sum_{w \in k} \log f(w)$.
- *kpRel*: If we consider only relevance but not interestingness, we can rank candidate keyphrases by $\sum_{w \in k} \log \frac{\#(\mathcal{C}_t, w)+\beta}{\#(\mathcal{C}, w)+\mu}$.

4.3 Gold Standard Generation

Since there is no existing test collection for topical keyphrase extraction from Twitter, we manually constructed our test collection. For each of the 10 selected topics, we ran all the methods to rank keywords. For each method we selected the top 3000 keywords and searched all the combinations of these words as phrases which have a frequency larger than 30. In order to achieve high phraseness, we first computed the minimum value of pointwise mutual information for all bigrams in one combination, and we removed combinations having a value below a threshold, which was empirically set to 2.135. Then we merged all these candidate phrases. We did not consider single-word phrases because we found that it would include too many frequent words that might not be useful for summaries.

We asked two judges to judge the quality of the candidate keyphrases. The judges live in Singapore and had used Twitter before. For each topic, the judges were given the top topic words and a short topic description. Web search was also available. For each candidate keyphrase, we asked the judges to score it as follows: 2 (relevant, meaningful and informative), 1 (relevant but either too general or too specific, or informal) and 0 (irrelevant or meaningless). Here in addition to relevance, the other two criteria, namely, whether a phrase is meaningful and informative, were studied by Tomokiyo and Hurst

T_2	T_4	T_5	T_{10}	T_{12}	T_{13}	T_{18}	T_{20}	T_{23}	T_{25}
eat	twitter	love	singapore	singapore	hot	iphone	song	study	win
food	tweet	idol	road	#singapore	rain	google	video	school	game
dinner	blog	adam	mrt	#business	weather	social	youtube	time	team
lunch	facebook	watch	sgreinfo	#news	cold	media	love	homework	match
eating	internet	april	east	health	morning	ipad	songs	tomorrow	play
ice	tweets	hot	park	asia	sun	twitter	bieber	maths	chelsea
chicken	follow	lambert	room	market	good	free	music	class	world
cream	msn	awesome	sqft	world	night	app	justin	paper	united
tea	followers	girl	price	prices	raining	apple	feature	math	liverpool
hungry	time	american	built	bank	air	marketing	twitter	finish	arsenal

Table 2: Top 10 Words of Sample Topics on our Singapore Twitter Dateset.

(2003). We then averaged the scores of the two judges as the final scores. The Cohen's Kappa coefficients of the 10 topics range from 0.45 to 0.80, showing fair to good agreement[2]. We further discarded all candidates with an average score less than 1. The number of the remaining keyphrases for each topic ranges from 56 to 282.

4.4 Evaluation Metrics

Traditionally keyphrase extraction is evaluated using precision and recall on all the extracted keyphrases. We choose not to use these measures for the following reasons: (1) Traditional keyphrase extraction works on single documents while we study topical keyphrase extraction. The gold standard keyphrase list for a single document is usually short and clean, while for each Twitter topic there can be many keyphrases, some are more relevant and interesting than others. (2) Our extracted topical keyphrases are meant for summarizing Twitter content, and they are likely to be directly shown to the users. It is therefore more meaningful to focus on the quality of the top-ranked keyphrases.

Inspired by the popular *nDCG* metric in information retrieval (Järvelin and Kekäläinen, 2002), we define the following normalized keyphrase quality measure (*nKQM*) for a method $\mathcal{M}$:

$$\mathit{nKQM}@K = \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} \frac{\sum_{j=1}^{K} \frac{1}{\log_2(j+1)} \mathit{score}(\mathcal{M}_{t,j})}{\mathit{IdealScore}_{(K,t)}},$$

where $\mathcal{T}$ is the set of topics, $\mathcal{M}_{t,j}$ is the j-th keyphrase generated by method $\mathcal{M}$ for topic t, $\mathit{score}(\cdot)$ is the average score from the two human judges, and $\mathit{IdealScore}_{(K,t)}$ is the normalization factor—score of the top K keyphrases of topic t under the ideal ranking. Intuitively, if $\mathcal{M}$ returns more good keyphrases in top ranks, its *nKQM* value will be higher.

We also use mean average precision (MAP) to measure the overall performance of keyphrase ranking:

$$\mathit{MAP} = \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} \frac{1}{N_{\mathcal{M},t}} \sum_{j=1}^{|\mathcal{M}_t|} \frac{N_{\mathcal{M},t,j}}{j} \mathbb{1}(\mathit{score}(\mathcal{M}_{t,j}) \geq 1),$$

where $\mathbb{1}(\mathcal{S})$ is an indicator function which returns 1 when $\mathcal{S}$ is true and 0 otherwise, $N_{\mathcal{M},t,j}$ denotes the number of correct keyphrases among the top j keyphrases returned by $\mathcal{M}$ for topic t, and $N_{\mathcal{M},t}$ denotes the total number of correct keyphrases of topic t returned by $\mathcal{M}$.

4.5 Experiment Results

Evaluation of keyword ranking methods

Since keyword ranking is the first step for keyphrase extraction, we first compare our keyword ranking method *cTPR* with other methods. For each topic, we pooled the top 20 keywords ranked by all four methods. We manually examined whether a word is a good keyword or a noisy word based on topic context. Then we computed the average number of noisy words in the 10 topics for each method. As shown in Table 5, we can observe that *cTPR* performed the best among the four methods.

Since our final goal is to extract topical keyphrases, we further compare the performance of *cTPR* and *TPR* when they are combined with a keyphrase ranking algorithm. Here we use the two

[2]We find that judgments on topics related to social media (e.g. T_4) and daily life (e.g. T_{13}) tend to have a higher degree of disagreement.

	Method	nKQM@5	nKQM@10	nKQM@25	nKQM@50	MAP
kpBL1	*TPR*	0.5015	0.54331	0.5611	0.5715	0.5984
	kwBL1	0.6026	0.5683	0.5579	0.5254	0.5984
	kwBL2	0.5418	0.5652	0.6038	0.5896	0.6279
	cTPR	**0.6109**	**0.6218**	**0.6139**	**0.6062**	**0.6608**
kpBL2	*TPR*	0.7294	0.7172	0.6921	0.6433	0.6379
	kwBL1	0.7111	0.6614	0.6306	0.5829	0.5416
	kwBL2	0.5418	0.5652	0.6038	0.5896	0.6545
	cTPR	**0.7491**	**0.7429**	**0.6930**	**0.6519**	**0.6688**

Table 3: Comparisons of keyphrase extraction for *cTPR* and baselines.

Method	nKQM@5	nKQM@10	nKQM@25	nKQM@50	MAP
cTPR+kpBL1	0.61095	0.62182	0.61389	0.60618	0.6608
cTPR+kpBL2	0.74913	0.74294	0.69303	0.65194	0.6688
cTPR+kpRel	0.75361	0.74926	0.69645	0.65065	**0.6696**
cTPR+kpRelInt	**0.81061**	**0.75184**	**0.71422**	**0.66319**	0.6694

Table 4: Comparisons of keyphrase extraction for different keyphrase ranking methods.

kwBL1	*kwBL2*	*TPR*	*cTPR*
2	3	4.9	1.5

Table 5: Average number of noisy words among the top 20 keywords of the 10 topics.

baseline keyphrase ranking algorithms *kpBL1* and *kpBL2*. The comparison is shown in Table 3. We can see that *cTPR* is consistently better than the three other methods for both *kpBL1* and *kpBL2*.

Evaluation of keyphrase ranking methods

In this section we compare keypharse ranking methods. Previously we have shown that *cTPR* is better than *TPR*, *kwBL1* and *kwBL2* for keyword ranking. Therefore we use *cTPR* as the keyword ranking method and examine the keyphrase ranking method *kpRelInt* with *kpBL1*, *kpBL2* and *kpRel* when they are combined with *cTPR*. The results are shown in Table 4. From the results we can see the following: (1) Keyphrase ranking methods *kpRelInt* and *kpRel* are more effective than *kpBL1* and *kpBL2*, especially when using the *nKQM* metric. (2) *kpRelInt* is better than *kpRel*, especially for the *nKQM* metric. Interestingly, we also see that for the *nKQM* metric, *kpBL1*, which is the most commonly used keyphrase ranking method, did not perform as well as *kpBL2*, a modified version of *kpBL1*.

We also tested *kpRelInt* and *kpRel* on *TPR*, *kwBL1* and *kwBL2* and found that *kpRelInt* and *kpRel* are consistently better than *kpBL2* and *kpBL1*. Due to space limit, we do not report all the results here. These findings support our assumption that our proposed keyphrase ranking method is effective.

The comparison between *kpBL2* with *kpBL1* shows that taking the product of keyword scores is more effective than taking their sum. *kpRel* and *kpRelInt* also use the product of keyword scores. This may be because there is more noise in Twitter than traditional documents. Common words (e.g. "good") and domain background words (e.g. "Singapore") tend to gain higher weights during keyword ranking due to their high frequency, especially in graph-based method, but we do not want such words to contribute too much to keyphrase scores. Taking the product of keyword scores is therefore more suitable here than taking their sum.

Further analysis of interestingness

As shown in Table 4, *kpRelInt* performs better in terms of *nKQM* compared with *kpRel*. Here we study why it worked better for keyphrase ranking. The only difference between *kpRel* and *kpRelInt* is that *kpRelInt* includes the factor of user interests. By manually examining the top keyphrases, we find that the topics "Movie-TV" (T_5), "News" (T_{12}), "Music" (T_{20}) and "Sports" (T_{25}) particularly benefited from *kpRelInt* compared with other topics. We find that well-known named entities (e.g. celebrities, political leaders, football clubs and big companies) and significant events tend to be ranked higher by *kpRelInt* than *kpRel*.

We then counted the numbers of *entity and event keyphrases* for these four topics retrieved by different methods, shown in Table 6 . We can see that in these four topics, *kpRelInt* is consistently better than *kpRel* in terms of the number of *entity and event keyphrases* retrieved.

T_2	T_5	T_{10}	T_{12}	T_{20}	T_{25}
chicken rice	adam lambert	north east	president obama	justin bieber	manchester united
ice cream	jack neo	rent blk	magnitude earthquake	music video	champions league
fried chicken	american idol	east coast	volcanic ash	lady gaga	football match
curry rice	david archuleta	east plaza	prime minister	taylor swift	premier league
chicken porridge	robert pattinson	west coast	iceland volcano	demi lovato	f1 grand prix
curry chicken	alexander mcqueen	bukit timah	chile earthquake	youtube channel	tiger woods
beef noodles	april fools	street view	goldman sachs	miley cyrus	grand slam(tennis)
chocolate cake	harry potter	orchard road	coe prices	telephone video	liverpool fans
cheese fries	april fool	toa payoh	haiti earthquake	song lyrics	final score
instant noodles	andrew garcia	marina bay	#singapore #business	joe jonas	manchester derby

Table 7: Top 10 keyphrases of 6 topics from *cTPR+kpRelInt*.

Methods	T_5	T_{12}	T_{20}	T_{25}
cTPR+kpRel	8	9	16	11
cTPR+kpRelInt	10	12	17	14

Table 6: Numbers of entity and event keyphrases retrieved by different methods within top 20.

On the other hand, we also find that for some topics interestingness helped little or even hurt the performance a little, e.g. for the topics "Food" and "Traffic." We find that the keyphrases in these topics are stable and change less over time. This may suggest that we can modify our formula to handle different topics different. We will explore this direction in our future work.

Parameter settings

We also examine how the parameters in our model affect the performance.

λ: We performed a search from 0.1 to 0.9 with a step size of 0.1. We found $\lambda = 0.1$ was the optimal parameter for *cTPR* and *TPR*. However, *TPR* is more sensitive to λ. The performance went down quickly with λ increasing.

μ: We checked the overall performance with $\mu \in \{400, 450, 500, 550, 600\}$. We found that $\mu = 500 \approx 0.01|\mathcal{V}|$ gave the best performance generally for *cTPR*. The performance difference is not very significant between these different values of μ, which indicates that the our method is robust.

4.6 Qualitative evaluation of *cTPR+kpRelInt*

We show the top 10 keyphrases discovered by *cTPR+kRelInt* in Table 7. We can observe that these keyphrases are clear, interesting and informative for summarizing Twitter topics.

We hypothesize that the following applications can benefit from the extracted keyphrases:

Automatic generation of realtime trendy phrases: For exampoe, keyphrases in the topic "Food" (T_2) can be used to help online restaurant reviews.

Event detection and topic tracking: In the topic "News" top keyphrases can be used as candidate trendy topics for event detection and topic tracking.

Automatic discovery of important named entities: As discussed previously, our methods tend to rank important named entities such as celebrities in high ranks.

5 Conclusion

In this paper, we studied the novel problem of topical keyphrase extraction for summarizing and analyzing Twitter content. We proposed the context-sensitive topical PageRank (cTPR) method for keyword ranking. Experiments showed that *cTPR* is consistently better than the original *TPR* and other baseline methods in terms of top keyword and keyphrase extraction. For keyphrase ranking, we proposed a probabilistic ranking method, which models both relevance and interestingness of keyphrases. In our experiments, this method is shown to be very effective to boost the performance of keyphrase extraction for different kinds of keyword ranking methods. In the future, we may consider how to incorporate keyword scores into our keyphrase ranking method. Note that we propose to rank keyphrases by a general formula $P(R = 1, I = 1|t, k)$ and we have made some approximations based on reasonable assumptions. There should be other potential ways to estimate $P(R = 1, I = 1|t, k)$.

Acknowledgements

This work was done during Xin Zhao's visit to the Singapore Management University. Xin Zhao and Xiaoming Li are partially supported by NSFC under the grant No. 60933004, 61073082, 61050009 and HGJ Grant No. 2011ZX01042-001-001.

References

Ken Barker and Nadia Cornacchia. 2000. Using noun phrase heads to extract document keyphrases. In *Proceedings of the 13th Biennial Conference of the Canadian Society on Computational Studies of Intelligence: Advances in Artificial Intelligence*, pages 40–52.

Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National Academy of Sciences of the United States of America*, 101(Suppl. 1):5228–5235.

Liangjie Hong and Brian D. Davison. 2010. Empirical study of topic modeling in Twitter. In *Proceedings of the First Workshop on Social Media Analytics*.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems*, 20(4):422–446.

John Lafferty and Chengxiang Zhai. 2003. Probabilistic relevance models based on document and query generation. *Language Modeling and Information Retrieval*, 13.

Quanzhi Li, Yi-Fang Wu, Razvan Bot, and Xin Chen. 2004. Incorporating document keyphrases in search results. In *Proceedings of the 10th Americas Conference on Information Systems*.

Marina Litvak and Mark Last. 2008. Graph-based keyword extraction for single-document summarization. In *Proceedings of the Workshop on Multi-source Multilingual Information Extraction and Summarization*, pages 17–24.

Zhiyuan Liu, Wenyi Huang, Yabin Zheng, and Maosong Sun. 2010. Automatic keyphrase extraction via topic decomposition. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 366–376.

Qiaozhu Mei, Xuehua Shen, and ChengXiang Zhai. 2007. Automatic labeling of multinomial topic models. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 490–499.

R. Mihalcea and P. Tarau. 2004. TextRank: Bringing order into texts. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*.

Daniel Ramage, Susan Dumais, and Dan Liebling. 2010. Characterizing micorblogs with topic models. In *Proceedings of the 4th International Conference on Weblogs and Social Media*.

Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. 2010. Earthquake shakes Twitter users: real-time event detection by social sensors. In *Proceedings of the 19th International World Wide Web Conference*.

Takashi Tomokiyo and Matthew Hurst. 2003. A language model approach to keyphrase extraction. In *Proceedings of the ACL 2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment*, pages 33–40.

Andranik Tumasjan, Timm O. Sprenger, Philipp G. Sandner, and Isabell M. Welpe. 2010. Predicting elections with Twitter: What 140 characters reveal about political sentiment. In *Proceedings of the 4th International Conference on Weblogs and Social Media*.

Peter Turney. 2000. Learning algorithms for keyphrase extraction. *Information Retrieval*, (4):303–336.

Jianshu Weng, Ee-Peng Lim, Jing Jiang, and Qi He. 2010. TwitterRank: finding topic-sensitive influential twitterers. In *Proceedings of the third ACM International Conference on Web Search and Data Mining*.

Wei Wu, Bin Zhang, and Mari Ostendorf. 2010. Automatic generation of personalized annotation tags for twitter users. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 689–692.

Xin Zhao, Jing Jiang, Jianshu Weng, Jing He, Lim Ee-Peng, Hongfei Yan, and Xiaoming Li. 2011. Comparing Twitter and traditional media using topic models. In *Proceedings of the 33rd European Conference on Information Retrieval*.