Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)

Identifying Search Keywords for Finding Relevant Social Media Posts

Shuai Wang[†], Zhiyuan Chen[†], Bing Liu[†] and Sherry Emery[‡]
[†] Department of Computer Science
[‡] Institute for Health Research and Policy
University of Illinois at Chicago
Chicago, Illinois, USA
{shuaiwanghk, czyuanacm}@gmail.com, liub@cs.uic.edu, slemery@uic.edu

Abstract

In almost any application of social media analysis, the user is interested in studying a particular topic or research question. Collecting posts or messages relevant to the topic from a social media source is a necessary step. Due to the huge size of social media sources (e.g., Twitter and Facebook), one has to use some topic keywords to search for possibly relevant posts. However, gathering a good set of keywords is a very tedious and time-consuming task. It often involves a lengthy iterative process of searching and manual reading. In this paper, we propose a novel technique to help the user identify topical search keywords. Our experiments are carried out on identifying such keywords for five (5) real-life application topics to be used for searching relevant tweets from the Twitter API. The results show that the proposed method is highly effective.

Introduction

In the past few years, we witnessed a rapidly growing trend of using social media posts to mine useful information for businesses and consumers, and to investigate social, economic, health, and political research questions. These applications are not only done by computer scientists, but are increasingly also done by researchers in management science, health science, and social sciences due to the rich social media content and its importance to these disciplines.

To perform an analysis, the user (often a researcher) first forms a research question, e.g., *what do people talk about electronic cigarettes on Twitter?* He/she then collects social media posts or data (e.g., tweets) that are relevant to the target topic. Due to the fact that social media data sources such as Twitter and Facebook are often huge, the user has to employ some keywords to search for possibly relevant posts. However, gathering a representative set of search keywords with a good coverage is a very tedious and time-consuming task. It often involves lengthy and iterative processes of searching and manual reading. Typically, the user first comes up with one or more keywords to search for posts from a social media data source. He/she then reads some returned posts and in the process identifies some additional keywords. He/she may search again using the new keywords and read more. We call this problem *search keyword mining*. In this paper, we study this problem and propose a novel method to help users identify search keywords to reduce the tedious manual effort.

Although related, our task is different from the classic task of keyword extraction from documents (e.g., Turney 2000, Zhao et al. 2011, El-Kishky et al. 2014) because we also need to ensure that the keywords are on target (topic) and not too general for use in search. These are not concerns of traditional keyword extraction. Additionally, we need an iterative process to find more and more suitable search keywords. Our task is also very different from term and query recommendation and query expansion in Web search and search advertising (e.g., Carpineto and Romano 2012). We will discuss these prior works in the related work section.

Note that finding a set of keywords to search for possibly relevant posts from a social media data source is just the first step. The resulting set of posts from the search can still be quite noisy because many posts containing the keywords may not be relevant to the user topic. A supervised classification step, which involves manual labeling of relevant and irrelevant posts and machine learning, is typically needed to identify the final set of relevant posts for analysis. This paper does not study this classification step as there are already many papers studying social media text classification.

However, one may ask why not skip the keyword finding/search step and directly go to the classification step because classification is needed anyway. Unfortunately, this approach is very hard, if not impossible, in practice. Let us use Twitter as a social media data source to explain.

1. Twitter is huge and people on it talk about everything under the sun. The proportion of relevant tweets (which are posts on Twitter) to a particular topic (e.g., electronic cigarettes or e-cig) is often very tiny. Thus, randomly sampling a set of tweets and labeling them as relevant or irrelevant tweets to create a training set for learning is almost impossible because the sampled tweets may contain few or no relevant tweets. Keyword search is thus necessary. A reasonable sized set of keywords is also necessary because the user needs to find tweets with a good coverage, rather than a biased coverage, of the topic, which may miss out tweets of many sub-topics of the research topic.

2. Even if by some means one can find a set of labeled relevant and irrelevant tweets to build a supervised classifier, it is almost impossible to apply it to classify all tweets on Twitter due to the computational difficulties.

This paper focuses only on helping the user identify a set of search keywords. Since our work and experiments use Twitter as the social media data source, we will describe our approach using tweets. The proposed method is called *Double Ranking* (DR). The basic idea is to first ask the user to provide one or more search keywords that he/she can think of. The system then uses them to search for a set of possibly relevant tweets from Twitter. It then ranks all the words in the returned tweets based on how likely each word is a valid keyword. A two-step ranking method is proposed. The first ranking produces an initial rank. A set of top ranked words is then re-ranked using the second ranking method. Both these ranking methods are based on the observation that words correlated with existing known keywords are also likely to be valid keywords. The initial ranking, which is less accurate but very efficient, is to identify a large shortlist of likely keywords, or in other words, to remove those unlikely keywords. Re-ranking refines the ranking of the shortlisted words. We use the shortlisted words in re-ranking because the re-ranking method (more effective) is not so efficient and it is thus hard to be applied to all words in the returned tweets. As we will see in the next section, this process can be applied iteratively to produce more and more keywords.

Our experiments are conducted using five (5) real-life application topics. Three of them are from the health science applications that are currently studied by our collaborating health science researcher. The other two are studied by us based on the need of a social media monitoring company to analyze Twitter discussions about two TV shows. For evaluation, we also compare the proposed DR method with several strong baselines. The results show that the proposed method outperforms these baselines by a large margin.

Proposed Double Ranking Approach

The proposed approach consists of 6 steps, which works iteratively to help the user find more and more search keywords. The system architecture is given in Figure 1.

Step 1. The user first provides one or more seed keywords K that he/she can think of. This is needed because otherwise the system will not know what the user is interested in. This step is also not hard because the user should have some knowledge of the topic domain, although giving a reasonably complete set of keywords is very difficult.

Step 2. The system uses the current set of known keywords, called the *current keywords set* (CK), to search for tweets containing one or more of the keywords using the Twitter API. The set of returned tweets is called the *current tweets set* (CT). In the first iteration, the current keywords set CK is the seed keywords set K. For the rest of the iterations, it is the union of the seed set K and the set of all discovered keywords so far.

Step 3. The system uses the proposed *initial ranking* algorithm to rank all the words in the tweets of the current tweets set CT based on how likely each of them is a valid keyword. The main goal is to shortlist a (large) set of top ranked words (called *shortlisted candidate keywords* (SK)) for re-ranking in Step 4 in order to produce a better ranking. In other words, this step tries to remove those unlikely keywords from consideration. Note that those current keywords in CK and words in the *unsuitable words set* (UW) should be removed from consideration because they have been identified by the user (see the interactive Steps 5 and 6).

Step 4. The proposed *re-ranking* algorithm is used to re-rank the set of shortlisted candidate keywords SK produced in Step 3 by searching the Twitter again using each of the candidate keywords in SK and re-evaluating its likelihood of being a valid keyword by estimating the proportion of the returned tweets that are likely to be relevant to the target topic. The details will be presented in the second subsection below, which gives the key idea of the proposed technique. The re-ranked candidate keywords in SK, called the *new rank* (NR), are presented to the user.

As we indicated in the introduction section, two rankings (Step 3 and Step 4) are used because the re-ranking method is more effective but less efficient as it needs to use each word to search the Twitter API.

Step 5. The user inspects a subset of the re-ranked list NR from the top to judge which words are suitable keywords and which are not based on his/her knowledge and some sample tweets containing the keywords. The suitable keywords are added to the *current keywords set* CK, and those unsuitable words are added to the *unsuitable words set* UW.

Step 6. If the user is satisfied with the keywords gathered so far, exit; otherwise, go to Step 2 (starting a new iteration).

Note that the proposed technique is not completely automatic as the user is involved in the selection of correct keywords after each iteration. We take this approach as it is more effective. It is also not hard for human users to identify correct keywords given a shortlist of ranked keywords. Automated methods are inferior in practice because they inevitably produce errors, ranking some wrong words at the top. Then the subsequent iterations will result in severe topic drifts and produce many irrelevant keywords. In the end, the user still has to select. Shifting the manual selection to the discovering process requires less manual effort.

The steps 1, 2, 5, and 6 are straightforward and we will not discuss them further. In the following two subsections, we focus on the initial ranking algorithm used in Step 3 and the re-ranking algorithm used in Step 4 respectively.

Initial Ranking Algorithm

The algorithm for initial ranking is given in Algorithm 1. As mentioned above, the main input for this step is the set of tweets CT returned from searching Twitter using the current set of keywords CK. The objective is to rank those words in CT based on how correlated they are with the search keywords in CK. A naive way is to just count how many tweets each word w in CT occurs (called *tweet frequency of word w* and denoted by $f(w)$) since every tweet already contains one or more keywords in CK. However, this is not a good solution because many common words will be ranked high. To solve this problem, we use a large random (or reference) set of tweets RT and employ Entropy from information theory to measure the correlation of word w, which should give low ranking positions to those common words that also appear frequently in the random reference set RT. Entropy is defined in Equation 1, where $s \in S = \{CT, RT\}$ and λ and $|S|$ are used for smoothing.

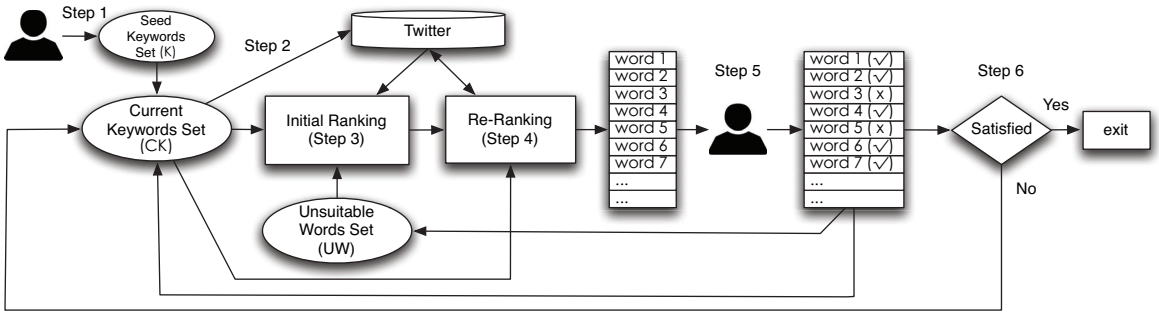

Figure 1: The System Architecture

Algorithm 1 Initial Ranking

Input: $\boldsymbol{CT}$ - The returned tweets set by searching Twitter using the current set of keywords CK
$\boldsymbol{RT}$ - A random (or reference) tweets set
$\boldsymbol{CK}$ - Current keywords set
$\boldsymbol{UW}$ - Unsuitable words set
Output: $\boldsymbol{SK}$ - Shortlisted candidate keywords set

1: $WE \leftarrow \{\}$
2: $V \leftarrow$ GetVocabulary(CT)
3: **for** each word w in V **do**
4: **if** $w \notin CK$ *and* $w \notin UW$ **then**
5: **if** $f_{CT}(w) + f_{RT}(w) > Min\_Freq$ **then**
6: **if** $f_{CT}(w) > f_{RT}(w)$ **then**
7: $e_w \leftarrow$ Entropy$(f_{CT}(w), f_{RT}(w))$
8: $WE \leftarrow WE \cup \{(w, e_w)\}$
9: **end if**
10: **end if**
11: **end if**
12: **end for**
13: $SK \leftarrow$ RankAndSelect(WE, N)

In Algorithm 1, the input parameter CK is the set of current keywords that have already been identified and UW is the set of words that the user believes to be unsuitable as keywords (see Step 5). SK is the output, a set of shortlisted candidate keywords produced from the initial ranking. Variable WE stores every word and its entropy value used in ranking (line 13). Line 2 gets all the words (vocabulary V) in CT. Line 3 - Line 12 computes the entropy for each word w in V. Line 4 removes those words in CK or UW that have already been judged by the user. Line 5 just ensures that w has a reasonable frequency. Min_Freq is a threshold. Line 6 ensures that w's frequency in CT is higher than in RT. Line 7 computes the entropy for word w. Line 13 first sorts the words based on their entropy values, and then selects the top ranked N words to be re-ranked in Step 4 (see below).

$$e_w = -\sum_s \frac{f_s(w) + \lambda}{\sum_s f_s(w) + |S|\lambda} log_2 \frac{f_s(w) + \lambda}{\sum_s f_s(w) + |S|\lambda} \quad (1)$$

Note that the word frequency could be normalized when the size of RT is larger than CT. However, the results do not vary much because although RT is larger, it covers a larger number of topics so the non-function words would still have low frequency in nature.

Re-Ranking Algorithm

The re-ranking algorithm of Step 4 is given in Algorithm 2. The innovation of this algorithm is as follows: The initial ranking in the previous step is based on only one direction correlation of how each word in CT is with the current set of keywords CK. However, this is not ideal because it does not tell us what happens if we search each word on Twitter, i.e., what percentage of the returned tweets is actually relevant, which is a better measure of the keyword quality. The re-ranking algorithm tries to do that. However, without manual labeling, we cannot get this percentage. Thus we have to approximate. One way to approximate is to actually search each word in SK on Twitter and then find how many tweets contain at least one keyword in the current keywords set CK. This is reasonable (at least for ranking) because if the word is a good keyword, it should have many co-occurrences with the existing keywords.

Algorithm 2 implements this idea. The inputs to the algorithm are as follows: SK is the shortlisted candidate keywords set produced from the initial ranking in the previous step. CK is the current keywords set that has been identified.

The output of the algorithm is a new ranking (NR) of the words in SK. WR is created in line 1 to store each word in SK with its rank score. Lines 2 - 12 compute the rank score for each word w in SK (line 10). Line 3 uses word w to search Twitter to get a set of returned tweets $T_{(w)}$. Line 4 initializes the variable $H_{(w)}$ to record the number of tweets in $T_{(w)}$ that contain some words in current keywords set CK, which is computed in lines 5 to 9. The rank score is simply the proportion of tweets containing one or more keywords in CK (line 10) and is computed as follows:

$$RankScore(H_{(w)}, T_{(w)}) = \frac{H_{(w)}}{|T_{(w)}|} \quad (2)$$

WR, which just stores all words and their corresponding rank scores, is used in the final ranking in line 13. The $rank$ function simply sorts all words based on their rank scores.

Algorithm 2 Re-Ranking

Input: $\boldsymbol{SK}$ - Shortlisted candidate keywords
$\boldsymbol{CK}$ - Current keywords set
Output: $\boldsymbol{NR}$ - The new rank of $\boldsymbol{SK}$

$WR \leftarrow \{\}$
for each $w \in SK$ **do**
  $T_{(w)} \leftarrow$ TwitterSearch(w)
  $H_{(w)} = 0$ // Number of tweets in $T_{(w)}$ that contain one or more keywords in CK
  for each tweet $t \in T_{(w)}$ **do**
    if $(CK \cap t)$ is not empty **then**
      $H_{(w)} = H_{(w)} + 1$
    end if
  end for
  $r_w \leftarrow$ RankScore($H_{(w)}, T_{(w)}$)
  $WR \leftarrow WR \cup \{(w, r_w)\}$
end for
$NR \leftarrow$ Rank(WR)

Experimental Evaluation

Topics and Experimental Data

Five real-life application topics are used in our experiments. Their tasks aim to use tweets from Twitter to study research questions related to the topics: *E-Cigarette* (or *E-Cig*), *Cigar*, *Blunt*, *Grimm* and *The Whispers* (or *Whispers*). *E-Cig* and *Cigar* represent two types of general tabacco-related products. *Blunt* is a specific type of cigars that are filled with marijuana. These three topics are currently studied by the team of our health science collaborator. *Grimm* and *The Whispers* are TV shows, interested by a social media company. When our experiments were conducted, the TV shows *The Whispers* was still on the air, but *Grimm* has ended. More information about the 5 topics is given in Table 1, which also includes the initial seed keywords set (K) suggested by the user, the number of tweets (the size of CT) collected by searching Twitter using the seed keywords for each topic, and the number of unique words in each tweets set. These tweets are used in the initial ranking.

Note that @ and # tagged terms or words are used as seed keywords to extract the initial tweets sets for the TV shows because these tags are the usual ways that people use to refer to a TV show in their posts on Twitter. For instance, to target at *The Whispers* they use #thewhispers or @ABCTheWhispers in their tweets. Using these keywords, a clean initial tweets set about a TV show can be collected.

Experimental Settings

We use 20000 random tweets as the reference set RT for entropy computation in initial ranking. We experimented with various numbers of tweets. Too few (less than 5000) do not capture common words well and thus give poorer results. More tweets give similar results. Following the common practice, the smoothing parameter λ in the entropy Equation 1 is set to 1. *MIN_Freq* is set to 5. For efficiency and without being blocked by Twitter, only the top 100 words (SK) are picked up from the initial ranking (Step 3) and passed to re-ranking (Step 4). We use the Twitter-API[1] for Twitter search. In re-ranking, we set the number of returned tweets in $T_{(w)}$ from Twitter for each keyword w in SK to 300. This number is used just to be uniform for all topics because for some keywords Twitter returns a large number of tweets, while for some very specific keywords, the set of returned tweets is very small. We also want the system to be efficient. Even with the small number of 300 the proposed technique produces very good results. We also experimented with more returned tweets for those keywords that can extract large sets of tweets and the results are similar or sometimes slightly better. For the second iteration, similar numbers of tweets are used for CT to those in Table 1.

Dataset	Seeds	Tweets	Terms
E-Cig	{e-cig(s),e-cigarette(s)}	4643	4087
Cigar	{cigar(s)}	4053	5537
Blunt	{blunt(s)}	4643	6551
Grimm	{#grimm, @NBCGrimm}	3081	4469
Whispers	{#thewhispers, @ABCTheWhispers}	4342	2938

Table 1: Five Topics, Seed Keywords and Initial Tweets

Baselines for Comparison

We compare our proposed Double-Ranking (DR) method with six (6) baselines that can be used to identify important words and rank them from a set of posts:

LDA-1: LDA is a classic topic model (Blei, Ng, and Jordan 2003) for finding topics and topical words in a corpus. It can naturally be used for our task. However, LDA produces multiple word rankings, one per topic. We use two methods to combine the rankings to produce one ranking for our use, which gives us baselines LDA-1 and LDA-2. LDA-1 ranks words based on their probabilities in the topic-word distribution, φ, i.e, $\hat{\varphi}_w = \max_{t \in \boldsymbol{T}} (\varphi_{t,w})$, where w and T denote a word and the set of topics respectively. We use 10 topics in modeling.

LDA-2: It ranks words using their average probabilities across topic-word distributions φ, i.e., $\overline{\varphi}_w = \frac{1}{|T_w|} \sum_{t_w}^{T_w} \varphi_{t,w}$, where T_w is the set of topics that contain word w in its top 20 positions.

Term Frequency (TF): This baseline simply ranks words based on their frequency in the tweets set.

Term Frequency-Inverse Document Frequency (TF-IDF): It is the same as above but uses the TF-IDF measure.

PageRank (PR): PageRank is a graph-based algorithm (Page et al. 1999) and was used for keyword extraction in (Mihalcea and Tarau 2004). In this and the LDA-1, LDA-2 and TF methods, the random tweets set RT is not used.

Entropy (ENT): This baseline is basically the initial ranking of the proposed technique with no re-ranking.

Experimental Results: Keyword Precision

This section presents the experimental results.

[1] https://dev.twitter.com/rest/public

Data Set	P@n	Baselines						DR	
		LDA1	LDA2	TF	TF-IDF	PR	ENT	Iter-1	Iter-2
E-Cig	P@5	0.00	0.00	0.00	0.00	0.00	0.00	0.20	**1.00**
	P@10	0.00	0.00	0.00	0.00	0.10	0.10	0.30	**0.90**
	P@20	0.05	0.00	0.00	0.00	0.10	0.05	0.25	**0.75**
Cigar	P@5	0.00	0.00	0.00	0.00	0.00	0.20	**1.00**	0.40
	P@10	0.10	0.00	0.00	0.00	0.00	0.10	**0.80**	0.40
	P@20	0.05	0.00	0.05	0.05	0.00	0.15	**0.40**	0.25
Blunt	P@5	0.00	0.00	0.00	0.00	0.00	0.20	0.40	**0.60**
	P@10	0.00	0.00	0.00	0.00	0.00	0.10	0.20	**0.60**
	P@20	0.05	0.00	0.05	0.00	0.00	0.05	0.25	**0.35**
Grimm	P@5	0.00	0.00	0.00	0.00	0.00	0.20	**0.80**	0.40
	P@10	0.00	0.10	0.00	0.00	0.00	0.20	**0.70**	0.20
	P@20	0.05	0.05	0.05	0.05	0.05	0.15	**0.45**	0.15
Whispers	P@5	0.00	0.00	0.20	0.00	0.00	0.20	**0.40**	0.20
	P@10	0.00	0.00	0.10	0.00	0.00	0.10	**0.20**	**0.20**
	P@20	0.05	0.05	0.10	0.05	0.05	0.05	**0.10**	**0.10**

Table 2: Keywords $Precision@n$ ($P@n$)

Data Set	P@n	Baselines						DR	
		LDA1	LDA2	TF	TF-IDF	PR	ENT	Iter-1	Iter-2
E-Cig	P@5	0.20	0.40	0.40	0.40	0.60	0.60	**1.00**	**1.00**
	P@10	0.20	0.20	0.40	0.40	0.40	0.50	**0.90**	**0.90**
	P@20	0.25	0.10	0.20	0.25	0.35	0.45	0.60	**0.75**
Cigar	P@5	0.40	0.40	0.40	0.40	0.40	0.40	**1.00**	0.60
	P@10	0.30	0.20	0.30	0.30	0.30	0.20	**0.90**	0.70
	P@20	0.15	0.15	0.20	0.20	0.15	0.30	**0.55**	**0.55**
Blunt	P@5	0.60	0.20	0.60	0.60	0.60	0.60	**0.80**	0.60
	P@10	0.50	0.20	0.50	0.50	0.50	0.70	**0.70**	**0.70**
	P@20	0.35	0.20	0.35	0.40	0.40	0.60	0.60	**0.65**
Grimm	P@5	0.20	0.20	0.40	0.40	0.20	0.60	**0.80**	0.40
	P@10	0.40	0.20	0.50	0.50	0.20	0.60	**0.70**	0.60
	P@20	0.30	0.10	0.30	0.40	0.25	0.55	**0.65**	0.55
Whispers	P@5	0.40	0.20	0.20	0.00	0.20	0.60	**0.80**	0.40
	P@10	0.50	0.20	0.30	0.30	0.30	0.30	**0.70**	0.50
	P@20	0.30	0.30	0.25	0.30	0.30	0.35	**0.65**	0.30

Table 3: Relevant word $Precision@n$ ($P@n$)

Since our goal is to identify search keywords for a given topic but the set of correct keywords is unknown due to the nature of our problem, a popular evaluation metric is to give the precision results at different rank positions n, called $Precision@n$ (or $P@n$ for short).

For the evaluation of the first three topics, two experts (from our health science collaborator's team) specialized in tobacco-related research were asked to label the results for *E-Cig*, *Cigar* and *Blunt*. Given the ranked words from each method, they selected keywords that are suitable for search based on their domain knowledge. Likewise, two judges who are familiar with the two TV-shows were asked to label the results for *Grimm* and *The Whispers*. The Cohen's Kappa agreement scores are 0.862 and 0.813 respectively.

The keyword precision results at the rank positions of 5, 10 and 20 are reported in Table 2. For the proposed DR method, we show results of two iterations. In the first iteration (Iter-1), only the user provided seed keywords are used. In the second iteration (Iter-2), the newly discovered keywords from iteration 1 are also used to find more keywords. We did not use two iterations for the baseline methods because their first iterations are already much poorer. From the table, we make the following observations:

1. Our DR method achieves the highest precisions at all rank positions and outperforms all baselines by a large margin.
2. Compared with all baselines, DR in the 1st-iteration already produces dramatically better results. The entropy based baseline (ENT) also outperforms other baselines.
3. DR in the 2nd-iteration produces very good results too.
4. For *E-Cig* and *Blunt*, the 2nd-iteration of DR produces better results than the 1st-iteration. One main reason is that these two topics require more keywords than only the seed keywords for better description of the topic (indicated by the current keywords set CK). So with new keywords detected and appended to the CK after the 1st-iteration, more accurate keywords are distilled subsequently, resulting in the major improvements of the 2nd-iteration.
5. The precisions of the 2nd-iterations are weaker than the 1st-iteration for *Grimm* and *Cigar* because many good keywords have been obtained in their 1st-Iterations.
6. Interestingly, *The Whispers* has relatively low precisions compared to other four topics. Comparing with *Grimm* and through further analysis we found that different from *Grimm*, since *The Whispers* was still being aired, a lot of tweets extracted are related mainly to the latest episode. This leads to many words found are more coherent to that episode but may not suitable to describe the topic itself. In contrast, the words for describing *Grimm* on Twitter are more focused and stable as the TV-shows has ended. Although the precisions are not high, DR still achieves the best results.

In summary, we can conclude that DR outperforms all baselines in all the five topics, which demonstrates the high effectiveness of our proposed approach.

Relevant Word Precision: In our interaction with the experts, we found that apart from the identified keywords, our domain experts are also interested in many other words that are very $relevant$ to the target topic. That is, although these words are not suitable to be search keywords because they are usually too general, they are highly related to the target topic. They may allow the user to understand the topic better, or may even be combined with some other words to form search patterns or search keyphrases. For instance, the word "tobacco" found in topic *E-Cig* is unsuitable to be a search keyword for the topic as it is too general. Likewise, the word "rolling" identified in the *Blunt* topic is also not suitable to be a keyword as it could easily drift to other topics like *rolling stone* (a famous magazine) or *rolling ball* (a popular sport game). However, according to our experts, "rolling paper" is likely to indicate $Blunt$ because people use paper (often made with tobacco) to wrap marijuana for smoking. Note that our current algorithm does not find multi-word keyphrases or more complex patterns. We plan to study them in our future work.

Table 3 reports the precision results in this setting, which includes both the keywords and the additional relevant words. It is clear that our proposed DR method again achieves the best results.

Example Discovered Keywords

Here we mainly use the *E-Cig* topic and the discovered keywords as examples to give a flavor of the type of results produced by the proposed technique and the baseline methods. Due to space limitations, for the other topics, only the best two of the top five ranked words are shown.

Table 4 first lists the top 10 results (words) from each method for *E-Cig*. The discovered correct *keywords* are italicized and marked in red while the discovered *relevant words* are also italicized but marked in blue. We can see that all the baseline methods find almost no keyword in their top 10 results. The 1st-iteration of DR found three keywords, namely $\{vaping, vapor, vaporizer\}$. These three words are appended to the current keywords set CK and used in the 2nd-iteration, which produces superior results. For example, in the 2nd-iteration the keywords *coudtank*, *clearomizer* and *atomizer* are specific e-cigarette products.

On the right side of Table 4, we list the best two words from the five top ranked words for each topic produced by the baseline methods (BL) and our proposed method DR. For the *Cigar* and *Blunt* topics, *robusto* is a specific type of cigar while *cohiba* and *backwoods* are brands. For *Grimm*, the words *hexenbiest* and *grimmster* are the symbols of this TV-shows. For *The Whispers*, the words *minx* and *bennigan* are two representative and influential characters of the TV-shows. Clearly, the words from the baselines (BL) are not suitable to be search keywords.

Related Work

Our work is related to the classic keyword/keyphrase extraction from documents. The task has been studied extensively in the past (Witten et al. 1999, Turney 2000, Mihalcea and Tarau 2004, Yih, Goodman, and Carvalho 2006, Wan, Yang, and Xiao 2007, Liu et al. 2010, Zhao et al. 2011, Danilevsky et al. 2014, El-Kishky et al. 2014). Existing techniques range from using frequency statistics, machine learning and graph-based algorithms to topic modeling. As indicated in the introduction, our task needs keywords on a specific topic and also suitable for search, not too general.

(King, Lam, and Roberts 2014) presented a method to find topic keywords. In addition to the initial keywords and dataset (like our K and CT sets), they also use a search set S and then find keywords from S. S is retrieved using some broad keywords, which may not be easy to identify. Our problem setting is different. We do not use S because S needs keywords to find too. Also, S can be highly biased if its keywords do not cover a superset of those relevant tweets. Clearly, using the data source such as Twitter as S is not possible because Twitter is too huge to be used for finding keywords directly. Even if we can sample a set of tweets from Twitter as S, S may contain no relevant tweets at all.

Our work is also related to keyword recommendation in search advertising and query suggestion in search (Cucerzan and White 2007, Bhatia, Majumdar, and Mitra 2011, Jones et al. 2006, Cao et al. 2008, Lüke, Schaer, and Mayr 2013). In search advertising, systems suggest keywords based on relevance to the advertised item, query logs mining, and economic considerations (Zhang et al. 2014, Zhang et al. 2012). Search query suggestion mainly exploits query logs, which we do not have. Although there are works using a corpus or Web pages to help term/query suggestions (Bhatia, Majumdar, and Mitra 2011, Zhang et al. 2012), they are similar to the traditional keyword extraction above.

Finally, our work is related to query expansion (Xu and Croft 1996, Carpineto and Romano 2012, Hahm et al. 2014). Query expansion aims to expand the original query to improve the document ranking for search engines. Our goal is to identify keywords that can track relevant posts based on exact keyword match. We do not do post ranking. Thus, we are solving a different problem, though some query expansion methods can be useful to our task.

Conclusion

In this paper, we studied the problem of helping the user identify search keywords to find tweets on Twitter that are relevant to a particular topic. This is a very important problem because almost all applications and researches involving social media data have to use a set of keywords to search for topical tweets. This paper proposed a novel method to perform the task. Experimental results obtained from using five (5) real-life application topics showed that the proposed method outperforms the baselines by a large margin. In our future work, we plan to also identify multi-word search keyphrases and more complex patterns.

Acknowledgement

Shuai Wang and Sherry Emery's research was supported by the National Cancer Institute (NCI) of the National Institutes of Health (NIH) and FDA Center for Tobacco Products (CTP) under Award Number P50CA179546. Bing Liu's research was supported by the NSF grant IIS-1407927 and the NCI grant R01CA192240. The content of the paper is solely the responsibility of the authors and does not necessarily represent the official views of the NSF, NCI, NIH or the Food and Drug Administration (FDA).

References

Bhatia, S.; Majumdar, D.; and Mitra, P. 2011. Query suggestions in the absence of query logs. In *SIGIR*, 795–804.

Blei, D. M.; Ng, A. Y.; and Jordan, M. I. 2003. Latent dirichlet allocation. *the Journal of machine Learning research* 3:993–1022.

Cao, H.; Jiang, D.; Pei, J.; He, Q.; Liao, Z.; Chen, E.; and Li, H. 2008. Context-aware query suggestion by mining click-through and session data. In *KDD*, 875–883.

Carpineto, C., and Romano, G. 2012. A survey of automatic query expansion in information retrieval. *ACM Computing Surveys (CSUR)* 44(1):1.

Cucerzan, S., and White, R. W. 2007. Query suggestion based on user landing pages. In *SIGIR*, 875–876.

Danilevsky, M.; Wang, C.; Desai, N.; Ren, X.; Guo, J.; and Han, J. 2014. Automatic construction and ranking of topical keyphrases on collections of short documents. In *Proceedings of SDM*.

E-Cigarette								Others	
Baselines (BL)						Double-Ranking (DR)		Cigar (BL)	Blunt (BL)
LDA-1	LDA-2	TF	TF-IDF	PageRank	Entropy	1st Iteration	2nd Iteration	*smoke*	*smoke*
children	*carcinogens*	children	children	*smoking*	*carcinogens*	*smokeless*	*cloutank*	dead	*roll*
fears	times	times	times	*tobacco*	fears	*vaping*	*clearomizer*	**Cigar (DR)**	**Blunt (DR)**
groups	*tobacco*	fears	fears	*smoke*	*suns*	*nicotine*	*evod*	*robusto*	*backwoods*
carcinogens	regular	*carcinogens*	*carcinogens*	news	endorse	*smokers*	*shisha*	*cohiba*	*ashing*
times	big	*tobacco*	*tobacco*	people	*malware*	*smoking*	*arizer*	**Grimm (BL)**	**Whispers (BL)**
tobacco	study	*smoking*	*smoking*	times	chemicals	*tobacco*	*ejuice*	season	sacrifice
source	pharma	regular	groups	*vape*	*virus*	*carcinogens*	*ploom*	regarder	finale
nasty	finds	groups	regular	health	*vape*	*vapers*	vicks	**Grimm (DR)**	**Whispers (DR)**
regular	regulation	*virus*	big	research	pharma	bans	*atomizer*	*hexenbiest*	*minx*
fingered	create	big	*virus*	study	fingered	*vaporizer*	*innokin*	*grimmster*	*bennigan*

Table 4: Example results of ranked words. Keywords are italicized in red. Relevant words are italicized in blue.

El-Kishky, A.; Song, Y.; Wang, C.; Voss, C. R.; and Han, J. 2014. Scalable topical phrase mining from text corpora. *VLDB* 8(3):305–316.

Hahm, G. J.; Yi, M. Y.; Lee, J. H.; and Suh, H. W. 2014. A personalized query expansion approach for engineering document retrieval. *Advanced Engineering Informatics* 28(4):344–359.

Jones, R.; Rey, B.; Madani, O.; and Greiner, W. 2006. Generating query substitutions. In *WWW*, 387–396.

King, G.; Lam, P.; and Roberts, M. 2014. Computer-assisted keyword and document set discovery from unstructured text. *Copy at http://j.mp/1qdVqhx* 456.

Liu, Z.; Huang, W.; Zheng, Y.; and Sun, M. 2010. Automatic keyphrase extraction via topic decomposition. In *EMNLP*, 366–376.

Lüke, T.; Schaer, P.; and Mayr, P. 2013. A framework for specific term recommendation systems. In *SIGIR*, 1093–1094.

Mihalcea, R., and Tarau, P. 2004. Textrank: Bringing order into texts. ACL.

Page, L.; Brin, S.; Motwani, R.; and Winograd, T. 1999. The pagerank citation ranking: bringing order to the web.

Turney, P. D. 2000. Learning algorithms for keyphrase extraction. *Information Retrieval* 2(4):303–336.

Wan, X.; Yang, J.; and Xiao, J. 2007. Towards an iterative reinforcement approach for simultaneous document summarization and keyword extraction. In *ACL*, volume 45, 552.

Witten, I. H.; Paynter, G. W.; Frank, E.; Gutwin, C.; and Nevill-Manning, C. G. 1999. Kea: Practical automatic keyphrase extraction. In *JCDL*, 254–255.

Xu, J., and Croft, W. B. 1996. Query expansion using local and global document analysis. In *Proceedings of SIGIR*, 4–11. ACM.

Yih, W.-t.; Goodman, J.; and Carvalho, V. R. 2006. Finding advertising keywords on web pages. In *WWW*, 213–222.

Zhang, W.; Wang, D.; Xue, G.-R.; and Zha, H. 2012. Advertising keywords recommendation for short-text web pages using wikipedia. *TIST* 3(2):36.

Zhang, Y.; Zhang, W.; Gao, B.; Yuan, X.; and Liu, T.-Y. 2014. Bid keyword suggestion in sponsored search based on competitiveness and relevance. *Information Processing & Management* 50(4):508–523.

Zhao, W. X.; Jiang, J.; He, J.; Song, Y.; Achananuparp, P.; Lim, E.-P.; and Li, X. 2011. Topical keyphrase extraction from twitter. In *ACL*, 379–388.