Keyword Extraction for Social Snippets*

Zhenhui Li
UIUC
zli28@uiuc.edu

Ding Zhou
Facebook Inc.
dzhou@facebook.com

Yun-Fang Juan
Facebook Inc.
yunfang@facebook.com

Jiawei Han
UIUC
hanj@cs.uiuc.edu

ABSTRACT

Today, a huge amount of text is being generated for social purposes on social networking services on the Web. Unlike traditional documents, such text is usually extremely short and tends to be informal. Analysis of such text benefit many applications such as advertising, search, and content filtering. In this work, we study one traditional text mining task on such new form of text, that is extraction of meaningful keywords. We propose several intuitive yet useful features and experiment with various classification models. Evaluation is conducted on Facebook data. Performances of various features and models are reported and compared.

Categories and Subject Descriptors: H.3.1 [Content Analysis and Indexing]: Abstracting methods, H.4.m [Information Systems]: Miscellaneous.

General Terms: Algorithms, Experimentation, Performance.

Keywords: keyword extraction, social snippet, online advertising.

1. INTRODUCTION

Social networking services, such as Facebook, Twitter, and MSN Messenger, are developing at an amazing speed, leading to production of a special kind of user-generated text for social purposes. Usually, users generate such text to broadcast to friends or followers their current status, recent news, or interesting events. We name such text *social snippets*. Status updates on Facebook and Twitter messages are representative examples of social snippets.

Social snippets are valuable media to mine users' recent interest. For example, if someone publishes his/her social snippet saying "is excited to receive my Wii today". It is a sign that this person is interested in Wii games. One way to discover such interest is to extract meaningful keywords from social snippets. Extracted keywords can be used for many applications. They can be treated as personal social tags to better retrieve user-generated information. They can facilitate social recommendations. Most importantly, they are essential for advertisement targeting. For example, advertisers can target the users who might be interested in video games using the keyword "Wii".

In this work, we seek to develop the effective keywords extraction methods for social snippets in the form of a classification task. There have been many related works on keyword extraction from documents [4, 1] and web pages [3]. However, the special characteristics of social snippets make the keyword extraction a new and even more challenging problem. It is statistically shown that social snippets are extremely short and tend to be more informal. Accordingly, insights from related works on traditional data no longer hold true. In the following, we report a set of features to measure the importance of keywords and compare the performances among various classification models. We compare our approach with several other keyword extraction systems, such as KEA [1] and Yahoo! keyword extraction system. All the experiments are conducted on a sample of Facebook data.

*The work was supported in part by NSF IIS-0905215 and the AFOSR MURI award FA9550-08-1-0265.

WWW 2010, April 26–30, 2010, Raleigh, North Carolina, USA.
ACM 978-1-60558-799-8/10/04.

2. KEYWORD EXTRACTION ALGORITHM

For each social snippet, we first tokenize it and generate unigrams and bigrams to be considered as keyword candidates. Then, a set of features are calculated to represent each candidate. We train a classification model based on the labeled keywords of social snippets. And finally the keyword candidates with highest scores through classification model are returned.

Here are the features:

***TFIDF*: information retrieval feature.** *TFIDF* is a commonly used feature in information retrieval. Intuitively, the word that appears more often in a document but not very often in the corpus is more likely to be a keyword. However, in our problem, this feature could be less useful. Because in short social snippets, the TF for most keyword candidates is 1. Frequency does not help to distinguish real keywords from others.

***lin*: linguistic feature.** In most cases, nouns have higher probability to be keywords. For a word, feature *lin* is the number of its noun POS tags divided by the total number of its POS tags. This feature is not affected by the "short" and "informal" properties of social snippets so we expect this feature to be an important one for classification.

***pos*: relative position.** *pos* is defined as the relative position within a social snippet. For the bigrams, we consider the position of the first word.

***lenText*: length of social snippet.** *lenText* of a social snippet is calculated by the number of words (including stopwords) in it. For all the keyword candidates in one social snippet, they are assigned with the same value.

***DF*: document frequency.** A hot and trendy topic is more likely to be a keyword in social snippets. Feature *DF* reflects the popularity of a keyword.

***capital*: capitalization.** This feature has a binary value showing whether there is a upper case letter in a keyword candidate.

After applying the classification model on testing data, we can get a relevance score for each keyword candidate. Finally, we need to return the "top" candidates with highest score. The most common way to select the "top" ones is to choose top-k candidates. However, we observe from our experiment dataset that about 25%

social snippets have no keywords. Because social snippets are usually very short and some actually do not contain any valuable information. If we force to return top-k candidates for each social snippet, it could hurt the classification precision a lot. In this situation, we propose to use the minimum score in the top-$p\%$ candidates in the training data as the threshold. And the candidates below this threshold are not considered as real keywords. This way, those social snippets having no valuable keywords are fairly treated.

3. EXPERIMENT

3.1 Experiment Setup

The training/testing data is made up of 1830 Facebook status updates. Annotators manually extract keywords for each status update. To compare our data with random web pages, we sample 2000 web pages from WebBase project[1]. The average words in our social snippets is 21.45 whereas there are 1075.75 words per web page. Furthermore, the percentage of words that can be found in Brown corpus is 86.18% for social snippets and 90.84% for web pages. From the statistics, we observe that social snippets are very short and tend to be more noisy.

3.2 Model Comparison

Model	top-1		top-3		top-5	
	prec	recall	prec	recall	prec	recall
GBM	**0.4084**	**0.2486**	**0.3003**	**0.4717**	**0.2551**	**0.5667**
DT	0.3814	0.2322	0.2922	0.4589	0.2453	0.5448
SVM	0.3303	0.2011	0.2654	0.4168	0.2288	0.5082
LR	0.3273	0.1993	0.2724	0.4278	0.2428	0.5393
TFIDF	0.3299	0.1738	0.2748	0.2888	0.2424	0.3824
Model	top-10%		top-20%		top-30%	
	prec	recall	prec	recall	prec	recall
GBM	**0.5025**	**0.2647**	**0.4427**	**0.4652**	0.3814	**0.6016**
DT	0.4444	0.2139	0.4342	0.4412	**0.3929**	0.5882
SVM	0.4264	0.2246	0.3791	0.3984	0.3119	0.4920
LR	0.4061	0.2139	0.3461	0.3636	0.3271	0.5160
TFIDF	0.3453	0.2102	0.2561	0.4022	0.2296	0.5101

Table 1: Precision and Recall by Different Threshold Cut

We select four models to compare: Gradient Boosting Machine (GBM), Decision Tree (DT), Support Vector Machine (SVM) and Logistic Regression (LR). The parameters of each model are finely tuned by cross validation process. Besides, to set a baseline, we directly use *TFIDF* value as the score for each keyword candidate. We randomly select 80% as training and 20% as testing. The result is shown in Table 1. GBM [2] performs the best among all. Compared with GBM, which combines linear distribution and decision tree, tree constructed by DT is simpler whereas LR and SVM require the data to satisfy a linear distribution. The results also show that top-$p\%$ is a better threshold selection method than top-k method.

3.3 Feature Comparison

In this section, we analyze the contribution of each feature in GBM model. The importance of features is first measured in terms of "relative influence". As described in [2], for tree based methods, the approximate relative influence of a variable x_j is $\hat{J}_j^2 = \sum_{\text{split on } x_j} I_t^2$, where I_t^2 is the empirical improvement by splitting on x_j at that point. From Figure 1, we can see that the most important feature is *TFIDF*. But different from its contribution in keyword extraction from web pages [3], *TFIDF* is not dominating the

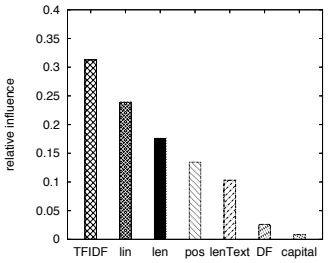

Figure 1: The relative influence of each feature in GBM

importance. Except *DF* and *capital* features, all other features also show fairy significant importance.

3.4 Performance Comparison

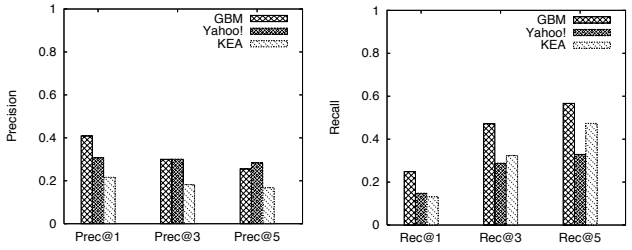

Figure 2: Comparison of different systems

We compare our performance with Yahoo! api for term extraction[2] and KEA [1]. Note that Yahoo! does not take any training data as input. KEA takes the same training data as our system. Figure 2 shows the performance of different methods. We use our best model GBM to compare with the other two. Generally, Yahoo! has considerably high precision but really low recall. KEA, on the contrary, tend to have high recall but low precision. Our method achieves a good balance between precision and recall. Even though at some points, its precision is a little lower than that of Yahoo!, but its recall is significantly higher.

4. CONCLUSION AND FUTURE WORK

Social snippet is a new form of text calling for re-examination of traditional text mining algorithms. In this poster, we focused on the keyword extraction problem. We tested a set of features and learning algorithms. The experimental results demonstrated effectiveness of our proposed features and gradient boosting machine. One nice property of social snippets we did not make use of is the underlying social networks, which we consider as a promising future direction.

5. REFERENCES

[1] E. Frank, G. W. Paynter, I. H. Witten, C. Gutwin, and C. G. Nevill-Manning. Domain-specific keyphrase extraction. In *IJCAI*, 1999.

[2] J. Friedman. Greedy function approximation: A gradient boosting machine. In *Annals of Statistics 29(5)*, 2001.

[3] W. tau Yih, J. Goodman, and V. R. Carvalho. Finding advertising keywords on web pages. In *WWW*, 2006.

[4] P. D. Turney. Learning algorithms for keyphrase extraction. *Information Retrieval*, 2000.

[1]http://diglib.stanford.edu:8091/ testbed/doc2/WebBase/

[2]http://developer.yahoo.com/search/content/V1/termExtraction.html