

Identifying Microblogs for Targeted Contextual Advertising

Kushal Dave
IIIT-Hyderabad
Hyderabad, India
kushal.dave@research.iiit.ac.in

Vasudeva Varma
IIIT-Hyderabad
Hyderabad, India
vv@iiit.ac.in

Abstract

Micro-blogging sites such as Facebook, Twitter, Google+ present a nice opportunity for targeting advertisements that are contextually related to the microblog content. By virtue of the sparse and noisy text makes identifying the microblogs suitable for advertising a very hard problem.

In this work, we approach the problem of identifying the microblogs that could be targeted for advertisements as a two-step classification approach. In the first pass, microblogs suitable for advertising are identified. Next, in the second pass, we build a model to find the sentiment of the advertisable microblog. The systems use features derived from the Part-of-speech tags, the tweet content and uses external resources such as query logs and n-gram dictionaries from previously labeled data. This work aims at providing a thorough insight into the problem and analyzing various features to assess which features contribute the most towards identifying the tweets that can be targeted for advertisements.

Introduction

Typical contextual advertising systems take the web content and fetch ads relevant to the content. Not all of the microblogs are suitable for contextual advertising, and choosing the wrong text fragments for contextual advertising might lead to bad user experience and may even irritate the user. Also, usually the content on microblog expresses opinions, hence it is imperative to only target the microblogs expressing positive or neutral sentiment. We propose to decompose the process of identifying such microblogs as a two-step classification task. First task is to identify the microblogs that are suitable for advertising and then from these set of microblogs suitable for ads, find the microblogs that have positive or neutral sentiment. Microblogs being short and noisy in nature, the classification tasks described here turn out to be *hard* classification problems.

We use twitter data for our experiments. Hence, in the rest of the paper, we refer to the microblogs as tweets. Also, tweets that are suitable for content targeted advertising are referred to as *advertisable tweets*.

Pre-processors

We take approximately 4 million tweets from twitter starting from the 11th Dec. 2010 to 12th Dec. 2010. Since we were looking only at the english data, we removed all the tweets

that had any foreign language content, which left us with 3.3 million tweets. These remaining tweets were first cleaned. for emoticons and repeated punctuations. After cleaning we also performed part-of-speech tagging of these tweets using Stanford NLP Tagger.

Information Sources

Twitter user statistics: We use a week' tweets for generating various information about users. We choose the week immediately before the train/test data collection date. From these (10 million) tweets we gather information like – # tweets by a user, # retweets by the user, # mentions of a user in tweets, # hashtags used by the user in their tweets, and # of URLs used by the user in their tweets. We also generate the term frequency of all the words occurring in the tweet.

Query Logs: Query logs are an important source of information, that tell what users usually look for. We use AOL query log which contains around 20 million queries from the AOL search engine. We also had the a sample of the query log from Yahoo!. This data is available as a part of Yahoo! webscope data sharing program (<http://webscope.sandbox.yahoo.com>).

Feature Set

User Information Set (UI): This set contains various counters about the user's tweet behavior. It contains features like the user's influence score. We used Klout API to get the influence scores of the twitter users. In addition, this set contains features described in the earlier user statistics section. The motivation behind keeping this set of tweets was to profile the user's tweet behavior. We wanted to see if tweets from influential person fall in the category of advertisable tweets and to assess the correlation of user's social behavior with the advertisable tweet class. In addition to these numeric features, we also used $\log(f + 1)$ as features in these set, where f are the statistics of the users.

Query Log Set (QL): Information from the query log is used to create this set of features. We split the tweet into various n-gram phrases and each n-gram is checked for occurrence in the query frequency table. For example, for the tweet "xbox 360 is mind blowing". Some of the n-grams are – 'xbox', '360', 'is', 'mind blowing', 'xbox 360' '360 is mind blowing' etc.

This category contains following features – if any n-gram phrase occurs in the query log, # of n-gram occurring in the

query log, max frequency of any of the n-gram occurring in the query log.

Part-of-Speech set (PS): This set of features represent the linguistic information about the tweets. It contains features like – if the tweet contains any noun tags (like NN, NNS, NNP, NNPS), verb tags (VB) and adjective tags (like JJ, JJB), the count of the noun verb and adjective tags.

Tweet Information (TI): In this set, we try to capture the relation of various tweet attributes with the advertisability of a tweet. This feature set contains following features – The length of the tweet, # of words in it (excluding user mentions and ‘RT’), if it is a retweet, if there is a mention of any user, if it has any hashtag, # of hash tags, if it contain a url, maximum term frequency of a noun in the tweet, # of the capitalized words in the tweet, if the tweet contains the original URL’s (non-shortened URL’s).

Dataset

The dataset is built from the set of 3.3 million English tweets. In order to train the classifier and to evaluate it, labeled instances need to be presented to it. Typically, in such classification problems this task is done by manual annotation of the instances. Manual annotation is costly and can give limited annotations. With any web experiment, having a larger dataset for the experiments always helps. Hence, we resorted to web search engines for the automatic labeling.

One good way to assess advertisability of a tweet is to query it and see if the search engines fetch any ad. If yes, then the query is advertisable else it is not. However, with so much noise in the tweets, it is not a good idea to use the complete tweet as a query. Instead, the queries were made by using only the adjectives and nouns in a tweet. This helps in bringing out better results from the search engine and removes noise to some extent. We gathered 466,500 such labels by querying a web search engine which comprise out dataset.

For each tweet, when queried, we received 0 or more ads in return from the search engine. Next task was to convert the number of ads into a binary label of tweet being advertisable or not. One way is to put a threshold on the number of ads. That is, if the number of ads that are fetched are more than or equal to the threshold τ , the tweet is advertisable, else not.

Identifying Advertisable tweets

In this section, we discuss our experiments and present the corresponding results and findings for the first classification problem. We first start with choosing classifiers for our experiments. Next we compare these classifiers for various values of threshold τ . For all our experiments in this paper, we chose to do a 10-fold cross-validation. In the following experiments τ was set to 2. It should be noted that the results presented in this section are for the advertisable class. We chose not to average it with the non-advertisable tweet class, as we are more interested in the advertisable class.

Feature Performance

In this section, we try to find out which features are more important and which subset of features contribute the most.

Feature	Naïve Bayes		Bagging (REP Tree)	
	Precision	Recall	Precision	Recall
UI	0.134	0.09	0.35	0.001
QL	0.051	0.100	0.053	0.081
TI	0.257	0.074	0.599	0.042
PS	0.36	0.259	0.507	0.078
TI+PS	0.319	0.409	0.514	0.176
QL+PS	0.341	0.309	0.505	0.174
UI+QL+TI	0.237	0.135	0.547	0.051
UI+QL+PS	0.337	0.273	0.485	0.146
QL+TI+PS	0.29	0.456	0.514	0.177
UI+TI+PS	0.319	0.355	0.503	0.173
All	0.313	0.482	0.594	0.344

Table 1: Improvement for various features

We start by comparing the performance of various classifiers like Naïve Bayes, Bagging, SVM, RBF Network etc. on our dataset. Naïve Bayes performs the best on recall, while Bagging does better than other classifiers on precision. Hence for further experiments we only choose these two classifiers.

Feature Set Performance We try various combinations of features for both the Naïve Bayes and Bagging classifiers. Table 1 shows the various combinations of features that were tried and the performance of each subset of features. As shown, we try various combinations of feature sets. Individually, QL and TI performed better than UI and PS.

Next, we try combining the TI+PS and QL+PS sets, as these feature sets did well individually, for both the classifiers (row 5-6). TI+PS improves the performance over PS by 54.44% for NB and 125% for BG classifier. QL+PS improve the performance in terms of precision over PS, but there is a drop in recall for both classifiers. In the combination of three, QL+TI+PS gave the best recall while maintaining a good precision amongst all the other 3 set combinations. QL+TI+PS showed better recall for both the classifiers compared to the single and double combination of feature sets. AL+TI+PS achieved high precision and recall for naïve Bayes classifier. This combination gave improvement of 28.01% (in terms of recall) for NB classifier and 20.54% for BG classifier over the second best (UI+TI+PS) in the three set combination.

Finally, we compare the system with all the features (ALL) with other combinations discussed above. As shown, for both the classifiers, the best recall was obtained when all the features were included in the model. Compared to (QL+TI+PS), which is the second best in terms of recall, ALL features gave an improvement of 5.7% for NB and 94.35% for BG. The system with all features had an extra feature set UI compared to (QL+TI+PS) combination, this shows that even though UI did not perform well individually, it helps in slightly improving the recall and the precision of the system when combined with other feature sets. However, surprisingly, adding UI to QL+PS dipped the performance slightly.

Individual Feature Performance In addition to evaluating the performance of the feature sets, it is also worth noting the contribution of individual features in our data set. We evaluate the contribution of features based on Information gain. It was found that features from QL set dominate the top ranking positions. Features from TI and PS set also

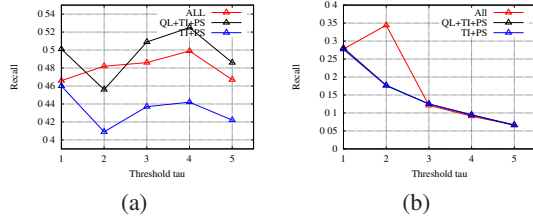


Figure 1: Recall for naïve Bayes (a) & Bagging(b)

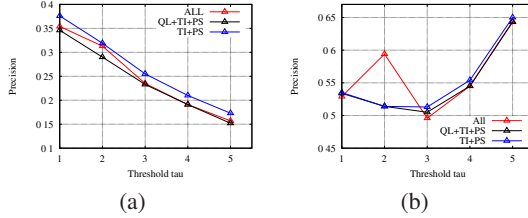


Figure 2: Precision for naïve Bayes (a) & Bagging(b)

have features in the top listings. Features like—position of first noun in the tweet, count of n-grams that occur in the query-log, whether n-gram from the tweet appears in the frequent query log, maximum term frequency of a word in the tweet featured in the top ten list.

Varying Tau

We vary $\tau = (1 \text{ to } 5)$ and discuss the results with regards to precision and recall. With higher τ values the classification becomes more difficult as there are fewer advertisable tweets in the dataset. We consider the top three performing feature sets, i.e. , All features, QL+TI+PS and TI+PS.

As shown in Figure 1, for naïve Bayes, QL+TI+PS surprisingly performs better than *All* features for all other τ values except 2. A similar trend is observed with bagging where *All* features perform well only for $\tau = 2$. Also, as was in Table 1, NB showed higher recall and lower precision compared to BG in Figure 1, and 2. However, with an increasing τ there was a dip in the precision for NB while, for BG the precision increased with higher values of τ . From figure 2, 1, it can be said that NB does well when there is loose bound on the definition of the advertisable tweets ($\tau = 1, 2$), while BG does well when there is a tighter bound on the advertisable tweet label.

Remarks on Features Some interesting conclusions can be drawn from the experiments and results on the feature contribution and importance. The system QL+TI+PS performs better than the system with *All* features for most of the τ values. Hence it can be concluded that UI does not add much value in making a decision on whether a tweet is advertisable or not. It was our intuition that some users, say news agencies usually tweet content that has a higher chance than a random user. Also, the influence of the user on the microblog network (klout score) also could not help much in the classification process.

TI and PS are very useful when used individually and in combination. This shows that the tweet content plays an important role in deciding whether a tweet is advertisable or not. QL wasn't much useful when used individually as a set,

but gave very good improvements when used in combination with other feature sets. Also, QL proved to be an important resource in this classification process. (Yih, Goodman, and Carvalho 2006) worked on the problem of advertisement keyword extraction from web pages and had a similar finding. Also, a feature from the term frequency resource (maxTF), featured in the top 10 features, hence it can be said that external resources like query logs and term frequency prove to be handy in such classification problems.

Sentiment Analysis of Advertisable tweets

Once the advertisable tweets are identified, next step is to predict the sentiment of the tweet. For the sentiment analysis experiments, we take all the advertisable tweets ($\tau = 4$) from the dataset described earlier. The choice of $\tau = 4$ was to keep a tight bound on the set of advertisable tweets used in this experiment. For this experiment we restored emoticons and slang words in the tweets. From the set of advertisable tweets, 700 tweets were tagged manually by the labelers from the lab. It was observed by the labelers that some hashtags give a very strong indication of the sentiment. For example, hashtags like '#nice', '#wow', '#awesome' indicated a +ve sentiment, while hashtags like '#fail', '#crap', '#sucks' indicated a -ve sentiment with a very high precision. We augment these advertisable tweets containing one or more of these hashtags into our labeled set. Eventually, the dataset contained 1,431 examples labeled with positive-sentiment and 1,690 with negative-sentiment.

Features for Sentiment Analysis

We used a total of 68 features divided into these sets –

N-Grams from the Product Review Dataset (PR) To generate this resource, we leveraged the product review dataset used in (Blitzer, Dredze, and Pereira 2007). We build a list of frequent N-grams (unigrams, bigrams and trigrams) for both the positive and negative sentimental reviews separately. We give a negative frequency to the n-grams occurring in negative reviews. For example, if a bigram 'not good' has a frequency of 542, we keep the frequency as -542. For each instance (tweet) in our dataset, we check if any of the unigrams, bigrams or trigrams from the tweet appear in any of the above frequent n-gram list. Also, the cumulative sum of the frequency for all n-grams is used as a feature.

Emotion Word List (EM) We used a word list, from the Sentistrength tool (<http://sentistrength.wlv.ac.uk>). It has a list of words with their prior polarity. It contained 900 words. The polarity ranges from +4 to -4 based on the sentiment of the word. This list of words were augmented by fetching a list of synonyms from the WordNet (Miller 1995). Finally, we had 4883 words in our emotion-word list. We count the number of +ve and -ve polarity words in a tweet and use them as features in our model. We also use the cumulative sum of positive, negative words as features.

POS Data (PS) We count the number of nouns, verbs, adjectives and adverbs in a tweet and consider them as features. Next, we keep the count of nouns, verbs, adverbs and adjectives for both the positive polarity words and negative polarity words. The polarity is taken from EM.

Model	Precision	Recall	FP Rate
<i>Unigram</i> ^{b1}	0.737 ^{b1}	0.730 ^{b1}	0.262^{b1}
<i>Bigram</i> ^{b2}	0.709 ^{b2}	0.707 ^{b2}	0.304 ^{b2}
<i>Unigram + Bigram</i> ^{b3}	0.729 ^{b3}	0.725 ^{b3}	0.262^{b3}
All	0.771	0.755	0.263

Table 2: Performance against baselines (p-value < 0.001)

Model	Precision	Recall	FP Rate
PR	0.615	0.617	0.398
EM	0.562	0.564	0.447
PS	0.653	0.654	0.362
BF	0.777	0.660	0.385
TW	0.641	0.642	0.371
All	0.771	0.755	0.263

Table 3: Performance of individual set of features

Boolean Features (BF) This set contains boolean features like, whether the tweet contains a question mark, whether it has any exclamation symbol, if it contains a wh-type question word in the tweet. We keep two boolean features to check if any of the +ve or -ve emoticon occurs in the tweet.

N-Grams from labeled Tweets (TW) We also had a manually labeled sentiment dataset of twitter which was used in (Agarwal et al. 2011). This dataset contained 5,128 tweets. We only take the positive and negative english tweets from the data set. It should be noted that Agrawal et. al generated this dataset for their own experiments and the fact that it did not contain many advertisable tweets made us to not use it as our dataset. we create a list of n-grams for both the positive and negative reviews in the same manner as with the PR set.

Comparison with N-Gram Model

For our model, various classifier were tried but bagging performed better than the other classifiers, hence we use bagging in our experiments. We compare the feature-based model against three bag-of-word baseline models– 1. unigram 2. bigram & 3. unigram+bigram model. Pak et. al (Pak and Paroubek 2010) show that n-gram models give good performance on the sentiment analysis task of twitter corpus. The baseline uses SVM with polynomial kernel as a classifier. Initially, we also tried naïve Bayes, but SVM outperformed naïve Bayes for the bag-of-words unigram model.

The comparison between the baselines and the feature-based model is as shown in Table 2. The feature based model does better than the n-gram models in terms of precision and recall. We note that the feature-based model improves by 4.40%, 8.74% and 5.76% over baselines 1, 2 and 3 in terms of precision, while in terms of recall the performance improvements were found to be 4.00%, 6.80% and 3.31% respectively.

Feature analysis

Table 3 shows the individual performance of each of the feature set. Feature set PS and BF gave good performance in terms of precision and recall, in fact, BF gave a better precision than the model with *All* set of features. However, BF could not do as well as the model with *All* features in terms of recall. Feature set EM and PR did not do as well as the others. As before, we assess the performance drop (or improvement) in a system when a feature set is removed from

the complete set. We found that the performance drops by 10.51% in terms of precision and 9.42% in terms of recall, when our top-performing set BF is dropped. PS set, when removed, also resulted into a 5.47% and 3.97% drop. For other sets, PR, EM and TW, a very small drop was observed.

From the above results on feature analysis, It can be said that BF is the most contributing set with the emoticons being a strong signal of the sentiment of the tweet. Pak et al. (Pak and Paroubek 2010) even used emoticons as a ‘noisy’ label in their experiments. Features from PS set, like count of verbs, count of nouns are also salient features. Resources like the n-grams from the product review dataset and labeled tweets are useful, as PR and TW gave a decent performance when used individually (Table 3).

Discussion

The fact that whether a microblog is suitable for targeting also depends, to some extent, on the set of ads available with the publisher. Hence, manual labeling can’t be reliable in such a case and using search engines for this task is a favorable choice. Broder et. al (Broder et al. 2008) worked on whether a set of ads should be displayed for a query or not. They leveraged ad and query features to decide whether to advertise or not. We did not have the luxury of an ad database, hence querying a web search engine allowed us to work on this problem, even without the ad dataset. However, as with Broder et. al. having ad features for the first classification task would have helped the classification process, hence the performances presented in the first experiment should be considered as lower bound to what can be achieved. This work is on similar lines to (Dave and Varma 2010; Yih, Goodman, and Carvalho 2006). However, in their work complete web page content was available to aid the classification task, where as with tweets its a lot harder as the get is very sparse and noisy. With both the identifying advertisable tweets task and finding sentiment task the PS features performed well. Also, external resources like query logs for the first classification task and n-grams from the product review and labeled tweet data proved to be useful for the classification. With microblogs such as tweets, there is very less information to perform classification, hence such external resources tend to be useful.

References

- Agarwal, A.; Xie, B.; Vovsha, I.; Rambow, O.; and Passonneau, R. 2011. Sentiment analysis of twitter data. In *Proceedings of the Workshop on Language in Social Media*.
- Blitzer, J.; Dredze, M.; and Pereira, F. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. 440–447. *ACL*.
- Broder, A.; Ciaramita, M.; Fontoura, M.; Gabrilovich, E.; Josifovski, V.; Metzler, D.; Murdock, V.; and Plachouras, V. 2008. To swing or not to swing: learning when (not) to advertise. *CIKM '08*, 1003–1012. *ACM*.
- Dave, K. S., and Varma, V. 2010. Pattern based keyword extraction for contextual advertising. *CIKM '10*. *ACM*.
- Miller, G. A. 1995. Wordnet: a lexical database for english. *Commun. ACM* 38:39–41.
- Pak, A., and Paroubek, P. 2010. Twitter as a corpus for sentiment analysis and opinion mining. In *LREC'10*.
- Yih, W.-t.; Goodman, J.; and Carvalho, V. R. 2006. Finding advertising keywords on web pages. *WWW '06*. *ACM*.