# Content + Context Networks for User Classification in Twitter *

**W. M. Campbell**
Human Language Technology Group
MIT Lincoln Laboratory
Lexington, MA 01740
wcampbell@ll.mit.edu

**E. Baseman**
School of Computer Science
University of Massachusetts Amherst
Amherst, MA 01003
ebaseman@cs.umass.edu

**K. Greenfield**
Human Language Technology Group
MIT Lincoln Laboratory
Lexington, MA 01740
kara.greenfield@ll.mit.edu

## Abstract

Twitter is a massive platform for open communication between diverse groups of people. While traditional media segregates the world's population on lines of language, age, physical location, social status, and many other characteristics, Twitter cuts through these divides. The result is an extremely diverse social network. In this work, we combine features of this network structure with content analytics on the tweets in order to create a content + context network, capturing the relations not only between people, but also between people and content and between content and content. This rich structure allows deep analysis into many aspects of communication over Twitter. We focus on predicting user classifications by using relational probability trees with features from content + context networks. Experiments demonstrate that these features are salient and complementary for user classification.

## 1 Introduction

In recent years, Twitter has become an extremely prolific social media engine, attracting an extremely diverse user base, ranging from teenagers discussing the latest in pop culture, to businesses looking for free advertising space, to the president of the United States trying to reach a broader audience than traditional media will allow. Twitter is the place to say whatever you want to whoever you want..., as long as it is less than 140 characters. This conciseness constraint forced the Twitter user base to develop innovative way of maximizing the information content of each letter. As such, the resulting tweets constitute a vast data set of rich textual content. Additionally, these tweets traverse through and define a social network comprised of all kinds of people tweeting to people about people.

In this work, we try to identify who some of these people are by performing user classification. In [14], Teng and Chen performed a similar study on bloggers in order to classify them by interest type. Twitter, however, provides a much richer feature set due to the denser network structure and nearly ubiquitous adoption of user profiles. Romero et al. first defined a social graph structure for Twitter

*This work was sponsored by the Defense Advanced Research Projects Agency under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Government.

data in [11]. While this was the first paper of its kind to explore the benefits of extracting a network structure from textual data, they only considered a limited graph comprising the retweet structure. [10, 8] expanded the notion of a Twitter graph to more broadly encompass social communication and used this jointly with content features to predict some attributes of Twitter users. Finally, a general top-level approach to content + context is presented in [2].

We develop a content + context network representation of Twitter data that combines the traditionally exploited social network features with user tweeting behaviors (hashtag usage). This allows us to exploit the context of the tweet content and enables us to accurately classify users from the diverse Twitter population. By combining features derived from tweet content and social network structure, we are able to develop a deep understanding of how what you say and who you say it to predicts who you are.

The outline of this paper is as follows. In Sections 2 and 3, we describe context and content feature extraction. We describe our user classification techniques and present our experiments with those techniques in Section 4.

## 2 Content + Context Networks

### 2.1 Corpus Collection

Over the 8 month period from September 2012 to March 2013, we used Twitter's streaming API to collect a $< 1\%$ sample of the entire Twitter feed, totaling approximately 686 million tweets, which Myers et al. showed to be a representative sample of the entire Twitter space [7]. These tweets contain basic information—tweet id, date and time, user id, user location, and tweet text. The tweets are representative of the Twitter population and display a wide variety of user accounts, topics, languages, and locations.

### 2.2 Network Construction

In order to construct an analytics platform, we applied a mapping that converts a corpus of tweets into a content + context network. There are several methods of representing Twitter data as a graph, which capture diferent levels of data richness at inversely proportional computational expense (both in processing and storage requirements). Our content + context network is a graph with multityped nodes in one-to-one corespondence with the union of the set of user profiles (e.g., `@blueman`) and the set of hashtags (e.g., `#yankees`) that occur in one or more of the collected tweets. We chose not to include individual tweets as nodes in the graph in order to maintain tractability. In order to do this without burdening a large data loss, we propagated salient tweet features into the user nodes of the tweet authors, users mentioned or retweeted, and hashtags used. We describe this process in Section 3.

In addition to the two classes of nodes previously described, we considered five classes of edges, see Figure 1. There are three classes of edges that connect two user profile nodes. The first is a directed, weighted edge corresponding to number times one user at-mentions another user (e.g., `@blueman` writes a tweet containing `@greenman`). The second type of user to user edge is a directed, weighted edge corresponding the number of times one user retweets another user (e.g., `@blueman` writes a tweet containing `RT @greenman`). Unlike the at-mentions and retweets edges, the third type of user to user edge doesn't map to communication between users; rather this edge classification refers to an undirected, weighted count of the number of times at-mentions of two users co-occur in the same tweet (e.g., `@redman` writes a tweet containing `@greenman` and `@blueman`). Similarly to the user to user co-occurence edge classification, there is an undirected, weighted edge classification corresponding to the co-occurrence of two hashtags. The final class of edges are weighted, directed edges corresponding to the number of times a given user tweets a particular hashtag.

In order to increase the completeness of the data we were able to gather from the $< 1\%$ sample, we gave special treatment to tweets containing nested retweets. If `@redman` retweets `@blueman` who retweeted `@greenman`, then we construct the Twitter graph with edges (`@redman`, `@blueman`) and (`@blueman`,`@greenman`). It is important to note that while this procedure improves data completeness, it may lead to some tweets being inadvertantly double counted. Since our sampling of tweets from the full Twitter collection is small (less than $< 1\%$), this will be a rare occurrence.

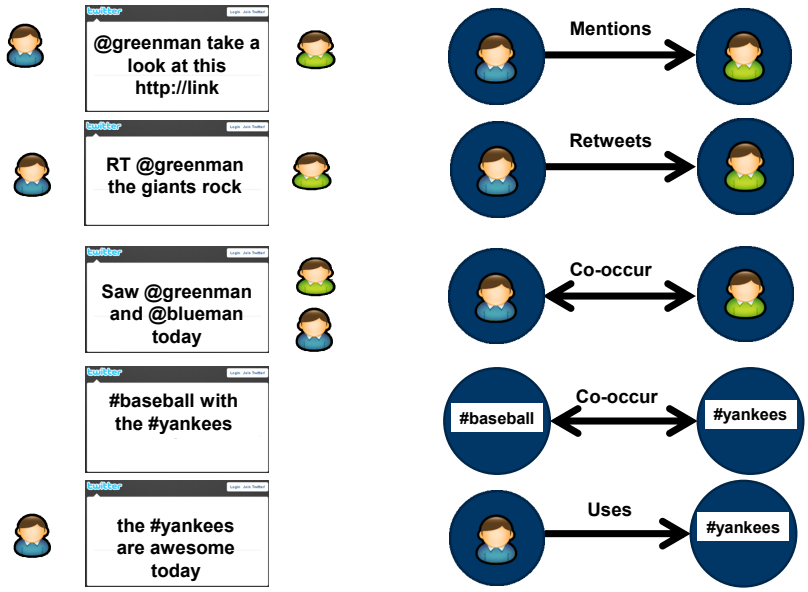


Figure 1: Edge types in the Twitter graph

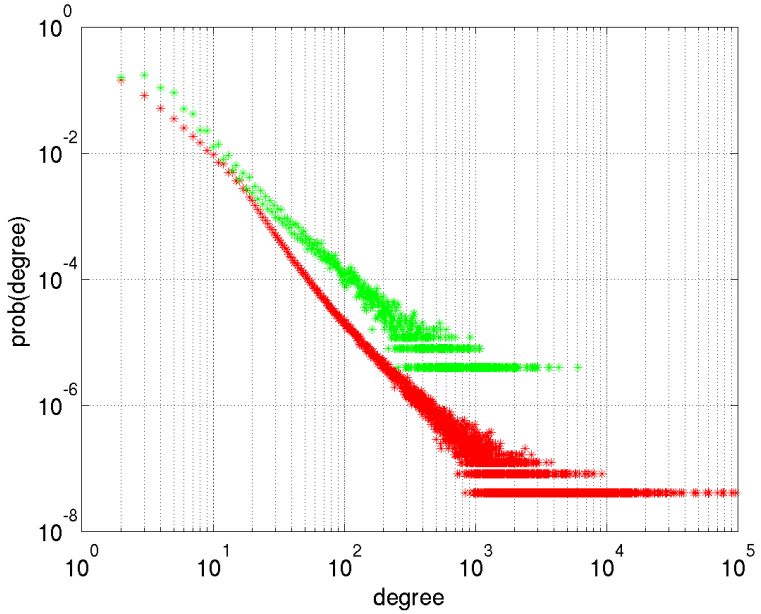


Figure 2: Degree distribution for different subsets of Twitter data—green (4%), red (20%)

We encountered multiple practical issues in constructing the Twitter graph—quantitative details will be presented in Section 4. In general, we took several steps to lower computational complexity. First, we applied language identification and discarded all non-english tweets. Second, we randomly sampled a subset of the original $< 1\%$ Twitter stream. Figure 2 shows the degree distributions for two random subsets, $4\%$ and $20\%$, of the data. Both are approximately power-law. The degree distributions vary by sampling as expected [13].

The number of user nodes was substantially larger than the number of hashtags, with the difference growing substantially as more tweets were added to the sample. The split was approximately $75\%$

(users) and $25\%$ (hashtags) in our case for the $4\%$ Twitter graph, and $93.5\%$ (users) and $6.5\%$ (hashtags) for the $20\%$ Twitter graph. Additionally, many of the user nodes had low degree. Therefore, we pruned low degree user nodes in order to increase the overall saliency of the graph. This strategy was reasonable for our experiments, since we were concerned with the impact of additional network features on classifier performance and since low-weight edges are likely spurious—the probability of a low weight edge in a graph derived from a small percentage of the Twitter stream having a high weight in the graph generated from the full Twitter stream is low.

### 2.3 Graph Features

We augmented the features that were extracted from the Twitter stream with graph features derived from community detection and centrality techniques.

Because community membership is a symmetric property, we removed directionality from all directed edges. We then proceeded to calculate the infomap communities. Community detection for infomap requires calculating a two-pass Huffman code over the node sets. First, an infinite random walk is considered over the weighted, undirected graph. Each node is then associated with the probability of the random walk traversing that node. A standard Huffman code is then applied to the node set, with high frequency nodes being treated as high frequency words are in traditional Huffman coding. In order to extract communities, we further compress the random walk encoding by identifying the sets of nodes which when given a shared prefix, result in an information-theoretically optimal encoding.

This can be quantitatively computed as the following optimization problem. Optimize,

$$L = q_{\curvearrowright} H(\mathcal{Q}) + \sum_i p^i_{\circlearrowleft} H(\mathcal{P}^i). \tag{1}$$

The two terms in (1) from left-to-right measure the across-community and within-community entropy. In the equation, $q_{\curvearrowright}$ is the probability of switching modules, and $H(\mathcal{Q})$ is the entropy of the community code. In the second term, $H(\mathcal{P}^i)$ is the within-community entropy, and $p^i_{\circlearrowleft}$ is the fraction of within-community movements that occur in community $i$.

We partitioned the node set into commuties by leveraging the infomap approach [12]. We used Pagerank to calculate the centrality of each node and we define community centrality as the sum of the Pageranks for all nodes in the community. The community "Pagerank" allows us to rank communities by centrality. After computing both Pagerank and communities, we added these node features to the original (directed) graph.

Optimizing the communities in the content + context network yields communities with both user and hashtag nodes. Nodes with high-pagerank in a community serve as community summarizations.

In Figure 3, we visualize the communites by combining nodes into super-nodes and plotting their relation [12]. Although this gives some insight into the relations between communities, setting a threshold for which edges are displayed is somewhat arbitrary.

## 3 Content Analysis

Before covering the details of content analysis, we describe the general framework for incorporating content features into our classification framework. We assume that content analysis produces a vector of posterior probabilities,

$$\mathbf{c}_j = [p(\omega_i|\text{tweet}_j)] \tag{2}$$

where $\omega_i$ is the indicator for the class label $i$. In general, including all tweets as nodes in a graph for classification leads to a graph of prohibitive size. Instead, we propagate the tweets to neighboring nodes and compute the expected posterior probability; i.e., we average all of the vectors propagated to a certain node.

The rules for propagating $\mathbf{c}_j$ for at-mentions are as follows. If user `@blueman` tweets with no recipient or multiple recipients, then $\mathbf{c}_j$ is propagated to the `@blueman` node. If user `@blueman` retweets from user `@greenman`, then $\mathbf{c}_j$ is propagated to both `@blueman` and `@greenman`. Similarly for hashtags nodes, we propagate the $\mathbf{c}_j$ vector to all hashtags used in $\text{tweet}_j$.

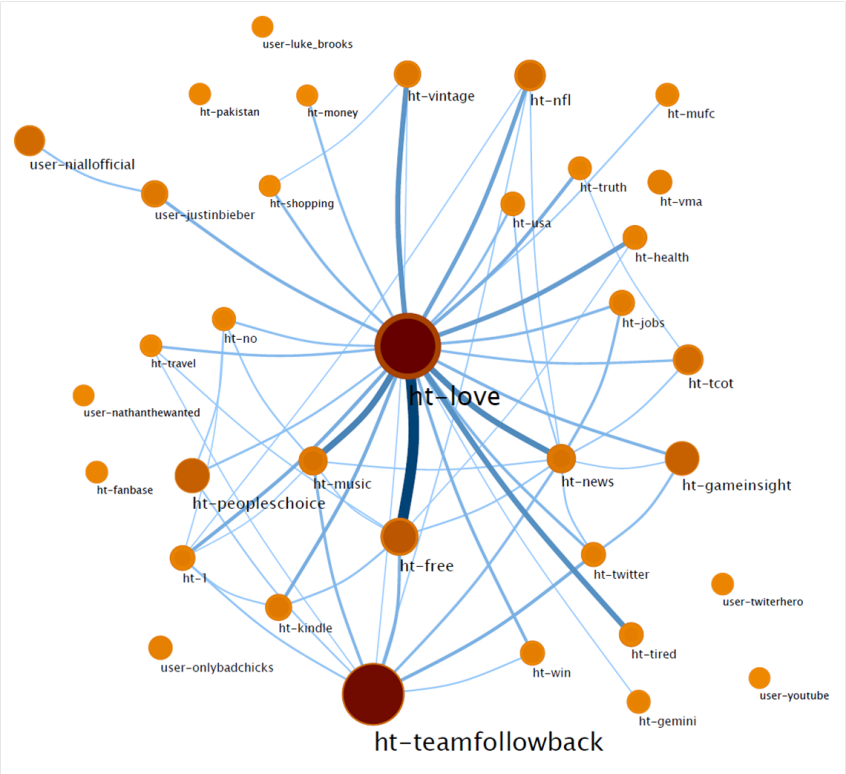


Figure 3: Supergraph of communities in Twitter Graph

Averaging the content vectors that were propagated to user and hashtag nodes defines representative content vectors for those nodes. A drawback of this aggregation strategy is that the average estimator can have a variable variance proportionate to the number of vectors propagated to a node, which can introduce noise into the classification process.

### 3.1 Text Normalization

We text normalize our entire corpus in order to enable further content analytics by providing a canonical form of the text that is relatively free of noise and spurious features.

First, we performed language identification via the Liu and Baldwin algorithm [6] and filtered the corpus to only English tweets. Next, we eliminated non-sentential punctuation. Then, we converted raw UTF-8 text to ascii in order to eliminate encoding variation.

Next, we removed Twitter meta-data including any URLs, retweet tags (RT), hashtags, and at-mentions. Note that this processing pipeline was designed to reduce some of the dependency between content and network analysis. For example, leaving hashtags in the tweet may produce better defined topic clusters, but could also create unusual dependencies between content vectors and hashtag nodes in the Twitter graph.

Finally, all words in the tweet were lower-cased and tweets were mapped to their representative word count vectors.

### 3.2 Topic Modeling

Our topic modeling is based upon probabilistic latent semantic analysis (PLSA). PLSA models the joint probability between documents and words by introducing a latent variable $z$ representing possible topics in a document. Specifically, the joint density between a document, $d_j$, and a word,

$w_i$, is given by

$$\begin{aligned} p(w_i, d_j) &= p(d_j)p(w_i|d_j) \\ &= p(d_j)\sum_z p(w_i, z|d_j) \\ &= p(d_j)\sum_z p(w_i|z)p(z|d_j). \end{aligned} \tag{3}$$

If we consider the vector $[p(w_i|d_j)]$ as $w_i$ ranges over all possible words, then the representation in (3) is a mixture model. The mixture weights are given by $p(z|d_j)$, and each mixture component (or latent topic) is represented by the vector $[p(w_i|z)]$.

We trained the PLSA model in (3) with an EM algorithm. The optimization criterion in this case is to maximize the log-likelihood of the data. We used a variation of this method called tempered EM which improves generalization of the model by introducing a parameter $\beta$, which affects the estimates of posterior probabilities in the EM iterations. We initialized the algorithm by using $k$-means to find estimates of the vectors $[p(w_i|z)]$. For more details, see [3] and [4].

### 3.3 Offensiveness

Another significant attribute of a tweet is the linguistic register—the variation due to the social setting. In attempt to capture some of the phenomena that occurs due to formality, familiarity, etc., we trained an offensiveness detector. The goal of building this detector is that variations in offensiveness might distinguish user type (e.g., politician versus generic user).

Offensiveness is a broad term and could be defined in many different ways. We define offensiveness using a pragmatic two-stage approach. First, we defined a set of offensive tweets by issuing queries via Lucene of offensive terms to obtain a set of offensive tweets. To obtain a set of putative non-offensive tweets, we took a random sample of the remaining tweets from a large pool, assuming that offensiveness has a lower prior. We then trained a classifier with the offensive and non-offensive data sets. The resulting output of the detector yields a consistent definition of offensiveness. Additionally, training a detector rather than using just a dictionary approaches captures some of the additional co-occurring terms and allows learning appropriate weightings of terms.

For training the offensiveness detector, we started with word counts, $c(w_i|\text{tweet}_j)$, for word, $w_i$ and tweet $j$ and normalize by the $\ell^1$-norm to obtain a probability of a word occurring in that tweet,

$$p(w_i|d_j) = \frac{c(w_i|d_j)}{\sum_k c(w_k|d_j)}. \tag{4}$$

Additionally, we computed the probability of a word occurring in all documents,

$$p(w_i|\text{bkg}) = \frac{c(w_i|d_j)}{\sum_k c(w_k|\text{bkg})}. \tag{5}$$

We then formed a sparse vector which has elements,

$$v_{i,j} = p(w_i|d_j)\log\left(\frac{1.0}{p(w_i|\text{bkg})}\right). \tag{6}$$

Note that for tweets, the term $p(w_i|d_j)$ will be 0 for most words, so the vector, $\mathbf{v_i} = [v_{i,j}]$ is very sparse. The log weighting in (6) is similar in performance to TFIDF but captures more detail in the DF term, see [1, 5].

We split the annotated data into distinct train and test sets for performance measurement and calibration of the detector. Approximately $14000$ tweets were in both sets. We trained an SVM by using the vectors in (6) and a linear kernel. The SVM regularization parameter was tuned for optimal performance to $c = 0.1$.

We needed two additional steps in order to apply the detector to all English tweets. First, we converted the output of the SVM to a posterior probability using the standard approach in Platt [9] and optimized by using a conjugate gradient method. Second, there were numerous cases in the data where unseen vocabulary in the training set resulted in the zero-vector for (6). In these cases, we used an imputed posterior of offensiveness by taking the average value across all nodes.

## 4 User Classification

Our goal in this work was to study the saliency of network and content features for classifying Twitter users. In order to do this, we used those features to develop classifiers to predict which users are verified.

### 4.1 Experimental Setup

From a 1% sample of raw tweets, we created a content + context network from content and network features. The raw profile features available on Twitter users include number of followers, number of tweets, location, time zone, and account duration, etc. Unfortunately, some fields such as location and time zone are free text and user-entered, and therefore unreliable. We did not use these unreliable features. We included network structure in the graph by letting edges indicate retweets, communication and co-occurance of users in tweets, or co-occurance of hashtags within tweets. In addition, we annotated each edge with counts for each interaction type. We added additional content features for topic and offensiveness as well as network features for cluster and cluster pagerank for each user and hashtag. Topic vectors were assigned using a PLSA approach, and offensiveness was determined using an SVM. Clustering and pagerank were achieved using the infomap approach. The resulting graph, after removing user nodes for which we could not obtain profile information, had 121,523 nodes (84,176 users and 37,347 hashtags), and 826,542 edges.

We grouped our features into two categories: content and network. The content features were topic and offensiveness. We included the top three most likely topics for each user and hashtag in our feature set for these experiments. Our network features for users and hashtags were cluster and cluster pagerank.

To predict whether or not a user has a verified account, we ran a decision tree as implemented in Proximity, an open-source software environment developed by the Knowledge Discovery Laboratory at the University of Massachusetts Amherst[*]. We varied features that were included in the analysis. There were a total of 84,176 users with profile information in our graph, which gave too large a data set for Proximity to run its learning algorithms. We took a 10% sample of these users, giving us 8,519 user nodes.

For predicting account verification, we developed classifiers using only content features, and only network features, and the combined set of features. We ran 10 experiments for each feature set, randomly splitting the 8,519 user nodes into 2 equally sized training and test sets and setting the maximum decision tree depth to 3. For evaluation, we report the average AUC and its standard deviation for each set of experiments.

### 4.2 Experimental Results

Table 1 shows average AUC results for propositional prediction of account verification. We found that while content features alone provide enough information for the decision tree to do better than random, network features alone perform better than just content features, and combining network and content features give us the best performance, yielding an average AUC of 0.7557.

Table 1: Average AUC for Propositional Prediction of Account Verification

| Included Feature Sets | Average AUC | Standard Deviation |
|---|---|---|
| Content | 0.6682 | 0.0364 |
| Network | 0.6925 | 0.0183 |
| Content+Network | 0.7557 | 0.0192 |

## 5 Conclusions

Twitter contains a rich set of social network and content cues that give insight into user characteristics. In this paper, we explored features that captured these characteristics via topic analysis,

[*]Software and documentation is available at http://kdl.cs.umass.edu/proximity

offensiveness, and network analytics. We presented content + context networks as a unified platform for storing and processing these features. Our different approaches demonstrated that the different feature types are complementary and all indicative of user classification. Possible future work includes predicting other Twitter user characteristics and applying content + context networks to other data sets.

## References


[1] W. Campbell, J. Campbell, D. Reynolds, D. Jones, and T. Leek. High-level speaker verification with support vector machines. In *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on*, volume 1, pages I–73. IEEE, 2004.

[2] G. A. Coppersmith and C. E. Priebe. Vertex nomination via content and context. *arXiv preprint arXiv:1201.4118*, 2012.

[3] T. J. Hazen. Topic identification. In G. Tur and R. D. Mori, editors, *Spoken Language Understanding*. Wiley, 2011.

[4] T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999.

[5] T. Joachims. *Learning to Classify Text Using Support Vector Machines*. Kluwer Academic Publishers, 2002.

[6] M. Lui and T. Baldwin. langid. py: An off-the-shelf language identification tool. In *Proceedings of the ACL 2012 System Demonstrations*, pages 25–30. Association for Computational Linguistics, 2012.

[7] F. Morstatter, J. Pfeffer, H. Liu, and K. M. Carley. Is the sample good enough? comparing data from twitters streaming api with twitters firehose. *Proceedings of ICWSM*, 2013.

[8] M. Pennacchiotti and A.-M. Popescu. A machine learning approach to twitter user classification. In *ICWSM*, 2011.

[9] J. C. Platt. Probabilities for SV machines. In A. J. Smola, P. L. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 61–74. The MIT Press, 2000.

[10] D. Rao, D. Yarowsky, A. Shreevats, and M. Gupta. Classifying latent user attributes in twitter. In *Proceedings of the 2nd international workshop on Search and mining user-generated contents*, pages 37–44. ACM, 2010.

[11] D. M. Romero, W. Galuba, S. Asur, and B. A. Huberman. Influence and passivity in social media. In *Machine learning and knowledge discovery in databases*, pages 18–33. Springer, 2011.

[12] M. Rosvall and C. T. Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 2008.

[13] M. P. Stumpf and C. Wiuf. Sampling properties of random graphs: the degree distribution. *Physical Review E*, 72(3):036118, 2005.

[14] C.-Y. Teng and H.-H. Chen. Detection of bloggers' interests: using textual, temporal, and interactive features. In *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 366–369. IEEE Computer Society, 2006.