Exploring the Meaning behind Twitter Hashtags through Clustering

Cristina Ioana Muntean, Gabriela Andreea Morar, and Darie Moldovan

Faculty of Economics and Business Administration, Babes-Bolyai University,
Cluj-Napoca, Romania
{cristina.muntean,gabriela.morar,darie.moldovan}@econ.ubbcluj.ro

Abstract. Social networks are generators of large amount of data produced by users, who are not limited with respect to the content of the information they exchange. The data generated can be a good indicator of trends and topic preferences among users. In our paper we focus on analyzing and representing hashtags by the corpus in which they appear. We cluster a large set of hashtags using K-means on map reduce in order to process data in a distributed manner. Our intention is to retrieve connections that might exist between different hashtags and their textual representation, and grasp their semantics through the main topics they occur with.

Keywords: k-means, clustering, hashtag, twitter.

1 Introduction

Twitter has been a prolific environment for analysis allowing research to dive into real worldwide large-scale phenomena. There have been various studies on its content and structure. Twitter is a micro-blogging platform that allows users to make *tweets*, messages no longer that 140 characters, resembling SMS (Short Message Service). Tweets are synthetic messages containing different kinds of information: links, media attachments, mentions (@) and hashtags (#).

The user has no limitations regarding the content of the text they can write in a tweet. This freedom and lack of formalism generates issues when it comes to analyzing the text and classic NLP tools seem almost powerless. The text can hold acronyms like "tfb", concatenated phrases like "ilikeitwhen" or it can contain spelling mistakes. Due to Twitter slang particularities, even the most popular terms can be cryptic to users, and even more so to automatic text processing applications. In our research[1] we attempt to make a first step towards finding structure and meaning in hashtags. In this preliminary study we wish to cluster hashtags in order to decipher their meaning, with the help of their

unique content, and group them into semantically interconnected groups. This can be a very useful task for disambiguating the meaning of hashtags such as "1thingiwant4christmas" without the need for a human assessor. The complex way of creating tags on Twitter can be revealed, and they can be structured according to the degree of granularity desired.

On the other hand one can find significant utility in hashtags. They can help Twitter users to go beyond the friendship level and follow topics, discussions or tweets that might not appear in their timeline, but are of interest to them. Just like following certain users [1] can help increase information gain, a similar effect can be achieved with following hashtags.

Application of machine learning techniques on Twitter, like classification, clustering or recommendation, can become useful for better fitting the interest of the user. In [2, 3] the authors study the applications of topic modeling algorithms, including LDA. The clustering task can be seen as a preliminary task to several more complex tasks like recommendations, filtering or ranking [4]. Clustering tasks on Twitter include tweet clustering or user clustering, but to the best of our knowledge hashtags clustering has not been thoroughly studied yet.

In our analysis we assume that each hashtag has a unique representation in our dataset, composed of the concatenation of all tweets which include it. It is quite improbable for two different hashtags to have the same so-called virtual document. This would mean they would have to co-occur each time either of them is mentioned. We present the proposed hashtag representation model, which forms the basis of our experiments, namely the dataset, and the results obtained after various clustering options.

In our experiments we cluster approximately 280.000 distinct hashtags from approximately 900.000 daily tweets per dataset, using K-means while varying the number of clusters k. According to the granularity of k we can obtain a more general grouping, if k is large, e.g. 500, it means we have 500 categories covering different major topics, the hashtags are more segregated and thus refer to more specific pieces of information. By lowering k, the groups become more generic. We carry out tests with different values of k, on datasets from three consecutive days.

The paper is structured as follows. In Section 2 we present a variety of studies conducted on Twitter, studies similar to ours regarding clustering, and some references to running large-scale data experiments in a distributed way using map-reduce. The following Section 3 presents a detailed analysis of our dataset. In Section 4 we describe our experiments, from preprocessing to algorithm tuning for the results shown and explained in Section 5. In the final part of the paper we present our conclusions, future work and possible applications.

2   Related Work

The overall popularity of Twitter has created various research themes. Several studies with regard to the content, the dynamics and the structural characteristics of Twitter have appeared in recent years. There are papers that concentrate

their attention on information diffusion throughout the network, on the discovering of communities, and on the analysis of user intents [5, 6, 7]. The detection of spam is also an important topic and has been studied in [8, 9]. Papers that characterize Twitter as a news media offer solutions to recommendation tasks like news and contents [10, 3, 11] or users [1]. In [12] the authors make a state-of-the-art survey on research on Twitter and try to define possible topics and open problems regarding the matter.

Several papers on machine learning techniques applied to Twitter tackle subjects like summarization and topic detection (LDA) [13], clustering [14] and disambiguation of topics or classification [15, 16]. Most studies on clustering regarding Twitter include topic modeling algorithms. In [15] the authors use LDA in order to classify short and sparse text using hidden topics from large-scale data. Recommendation systems use clustering as a prior step to offering suggestions. In [11] the authors suggest tweets based on a user's history and topic model. They transform text according to VSM and assign TFIDF weights to vectors. Similarly, TwitterRank [4] is based on tweet topics and the authors' attempt to find influential users. They use LDA to build topic models for each users according to their tweets.

In [2] the authors tackle the Twitter dataset from a NLP point of view and observe the esoteric nature of language and grammar, the fact that short text contains less stop words and word redundancy. In their analysis of disaster-related Twitter data they use probabilistic topic models and treat data as bags of words – due to the lack of fluency withing tweets – while also inferring latent relationships between data.

A comparison between K-means, SVD and affinity propagation, a graph based approach, has been made in [17], in which authors test various clustering techniques on short text documents, namely tweets. As in the case of our study the biggest challenge in handling short text is the problem of sparsity. TF will be very small in most cases so the vectors are basically represented by the IDF. In order to tackle this problem, they propose building the vectors with value 1 when the word occurs and 0 for the contrary case. They use two distance measures, one based on Jaccard coefficient and the other Cosine. Their experiments are run on a set of 661 tweets with a vocabulary consisting of 1678 distinct words. For evaluation they use cluster density techniques.

The above cited papers rely heavily on annotated data and small sample datasets on predefined topics. They mostly focus on clustering tweets, while in our case we are more preoccupied with clustering hashtags on a wide data sample in which also the rare hashtags are considered.

In what regards large scale data processing, experiments can be optimized by taking advantage of distributed computing. For our experiments we use the MapReduce paradigm [18], which is designed to simplify the concepts around large scale distributed computing and allows dealing with large datasets. It is divided into two steps: map and reduce. The map function takes a single instance of the type key/value pair as an input. The output of the function are key/value pairs that are grouped by key and are used as an input for the reduce function.

Based on the key value and the list of values outputted by the map function, the reduce function performs some computations over that list and outputs key/value pairs.

Map Reduce has been implemented by projects like Hadoop [19] and Disco [20]. Hadoop is an open-source implementation of MapReduce and it was chosen because it is currently the most feature-complete system and widely used in industry.

The Hadoop framework is composed of the MapReduce functionality and a distributed file system (HDFS) [21]. The distributed file system has the role of distributing input data across the cluster. Hadoop tries to allocate map tasks based on the physical location of each piece of data on HDFS. There are several distributed machine learning libraries that use MapReduce. The most famous ones are Apache Mahout [22] and Weka [23]. Apache Mahout has implemented algorithms described by Chu et al. [24].

The methods mentioned above are ideal for simple distribution, as partitioning the datasets used across multiple machines will not change the end result.

3   Dataset

For our experiments we use various datasets collected through the Twitter Streaming API for a period of three days, starting 14.12.2011 until 16.12.2011 The resulting dataset represents a random sample of 10% of the entire daily activity, thanks to the Gargenhose API account. The data used for the conducted experiments was retrieved from non-protected public accounts. The public accounts post public statuses as candidates for the streaming API, thus we do no need to tackle privacy issues.

After parsing the datastream obtained through the API, we succeed in building dataset as shown in Table 1. The raw datasets adds up to almost 20 million tweets per day and around 60 million tweets for the entire dataset. In order to be able to process the data, we first clean the dataset and kept just the English

Table 1. Data description

Datasets		
	Tweets	20.184.280
Dataset14	Tweets with hashtags	947.815
	Hashtags	1.293.470
	Hashtags distinct	287.091
	Tweets	18.543.703
Dataset15	Tweets with hashtags	877.760
	Hashtags	1.195.910
	Hashtags distinct	267.680
	Tweets	20.928.904
Dataset16	Tweets with hashtags	1.011.717
	Hashtags	1.385.771
	Hashtags distinct	288.542

tweets containing hashtags. Moreover we remove tweets that have been retweeted or which consist of dialogues.

Since our intention is to cluster hashtags we take a look at the structure of hashtags and try to identify certain patterns. According to the presented hashtags we can see that Twitter slang is quite poignant. We can see a top 10 of most frequent hashtags per dataset and their corresponding frequency.

- **Dataset14:** teamfollowback (18910), oomf (16966), np (16938), nowplaying (12303), 2011regrets (11274), idislike (10329), mygoalfor2012 (8837), thingsthataredead (8203), myfavoritethings (7568), jobs (6827).
- **Dataset15:** thingsweallhate (27415), teamfollowback (17091), np (15784), oomf (15265), thingsthatpissmeoff (11451), nowplaying (11350), 1thingiwant4christmas (8476), jobs (6012), mylastwordswillbe(5689), nf(5244).
- **Dataset16:** ff (61469), teamfollowback (20031), np (17202), oomf (15334), nowplaying (12266), ilikeitwhen (10083), jobs (6787), answer (6493), onlyifyouknew (6130), nf (5544).

Almost none of the hashtags follow the classical pattern of tagging with terms. We can see some represent abbreviations like "oomf", while others, entire phrases of concatenated words "mylastwordswillbe".

There are several challenges that we need to tackle in order to analyze properly such a distinct dataset. We first observe how the hashtags are structured.
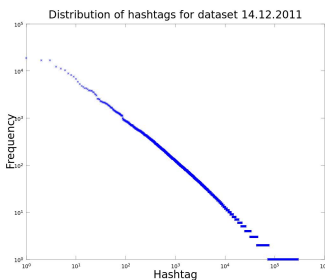

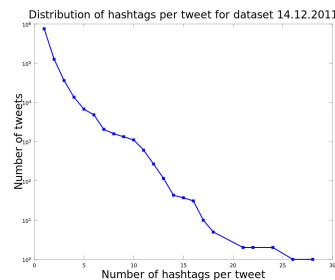**Fig. 1.** The distribution of hashtags

Fig. 2. Hashtags per tweet

Figure 1 represents the distribution of hashtags in the collection. The type of distribution is maintained throughout all daily datasets. We can observe that hashtags follow a *power law*, while a few popular hashtags repeat themselves in the collection a great number of times, a large number of hashatgs have a small frequency. Translated in our dataset, popular hashtags are represented as rich documents, while very less frequent ones have poor documents. The fact that many hashtags have small documents, creates a very sparse vector representation of the dataset.

Figure 2 represents the distribution of hashtags per tweet. We can see that most tweets have a small number of hashtags, and just a few, a large number. After manually inspecting those tweets we come to the conclusion that these last type of tweets represent spam. Users put together several popular hashtags and a shortened URL in order to drive traffic to a web page.

We follow several preprocessing steps. Starting from the JSON set, we parse the tweet text and hashtags within that tweet. We clean the text from: mentions and urls, while also trying to segment hashtags. The resulting dataset is then processed in order to build the inversed associations, namely from hashtag to tweet. For each hashtag we have build a virtual document, consisting of the concatenation of all the tweets in which it was mentioned. This task is solved using the Cascading [25] library over Hadoop. We also assume that each hashtag has an unique representation through its virtual document.

We define $T = \{t_1, t_2, ...t_n\}$ as the set where each t_i is a tweet document and $H = \{h_1, h_2, ...h_m\}$ as the set of all the hashtags in the dataset. A virtual document for one hashtag h_j is a concatenation of tweets as follows:

$$d_j = \sum_{i \in [1,n],\, h_j \in t_i} t_i \quad , \qquad \forall j \in [1, m] \tag{1}$$

4 Experiments

4.1 Preprocessing

For running our experiments we structure our data into files named after each hashtag h_j and containing the text of the corresponding virtual document d_j, according to the definition in (1). We order the files into directories (10.000 files per directory). Dataset14 for example has 287.091 files, namely hashtags represented as virtual documents. In order to proceed to the clustering step we need to represent the text documents as vectors. A common way to do this is to use the vectors space model VSM. The vectorized representation of text in the case of hashtag virtual documents is sparse, mainly in the case of rarer hashtags.

In order to unify hashtags written in different ways, e.g "christmasgift" or "ChristmasGift", they are lowercased and then considered the same hashtag. In the case of "christmas" and "xmas", we expect their virtual documents to be quite similar and the two hashtags grouped together in the same cluster, as variations of the same concept.

For building the vectors, we make feature selection and build a dictionary that better allows us to represent significant words into vectors. We eliminate stop words and some spelling mistakes. Into the dictionary, we put tokens with a minimum document frequency value of 10, which represents the minimum number of documents the term should appear in. These parameter setting helps us prune the dataset from spelling mistakes and very rare words.

Other several preprocessing tasks include tokenization, stop words removal and stemming for the words building the vocabulary. The tasks are accomplished by customizing the Lucene Analyzer class [26]. Using the analyzer we succeed to

remove stop words and words that contain certain patterns, for example three identical characters in a row within a token, such as "aaa", which we treat as spelling mistakes. All tokens passing the filters are lowercased and stemmed according the Porter Stem algorithm [27]. The preprocessing phase is important because it helps to reduce sparsity. If in the beginning without any preprocessing our dictionary had around 500.000 terms, after preprocessing the dictionary for Dataset14 had around 60.000 terms.

The virtual documents corresponding to hashtags are transformed into weighted vectors. We use TFIDF for weighting the terms and 2-norm form normalizing vectors. Thus d_j becomes $\overrightarrow{d_j}$.

4.2   K-Means

K-means [28] is a rather simple but well known unsupervised learning algorithm for clustering. Given a dataset, the algorithm partitions data into a number of clusters. This number of clusters, k, is fixed a priori. The algorithm is divided into the following steps:

1. initialize k points, also known as centroids, randomly chosen from the dataset;
2. assign each virtual document $\overrightarrow{d_j}$ to the cluster having the closest centroid;
3. after all vectors have been assigned, recalculate the position of the centroids as the mean of the points in the relative cluster;
4. repeat steps 2 and 3 until a stop condition is reached or the centroids no longer change.

K-means algorithm minimizes an objective function, we use Jaccard distance measure, which is found to be a suited measure for text in [17]. The Jaccard distance is calculated as follows:

$$DIST_J(\overrightarrow{d_a}, \overrightarrow{d_b}) = 1 - \frac{\overrightarrow{d_a} \cdot \overrightarrow{d_b}}{|\overrightarrow{d_a}|^2 + |\overrightarrow{d_b}|^2 - \overrightarrow{d_a} \cdot \overrightarrow{d_b}}. \tag{2}$$

We also vary the number of clusters, so for each dataset we experiment with k equal to 20, 100 and 500.

The experiments were conducted using the Mahout library over a Hadoop single node cluster installation. This setup allows K-means to run 4 tasks in parallel, 2 map and 2 reduce jobs. Execution time is highly influenced by the number of clusters we wish to produce and can vary from tens of minutes to several hours. We wish to tackle efficiency and speed up problems on multi-node cluster architecture in future work.

A method for determining the best number of k for K-means clustering is by using Canopy clustering [29]. On the other hand, in order to construct canopies we need to set up a minimum and a maximum threshold, values that can themselves become a shortcoming if not set properly. Considering the sparsity of the data and the fact that clusters may not have clear boundaries, there is no need to divide the data in a precise number of clusters. K can be regarded as a measure of granularity of the clustered results, according to which we group hashtags into more specific or more general collections.

5   Results

The results of the clustering show that it is possible to identify semantically related hashtags. For each cluster we extract the top terms, i.e. the most frequent terms in the virtual documents of the cluster. These top terms are the most representative for the cluster, and fulfill their role as explanatory terms. We also extract top hashtags within a cluster, they are obtained by ranking all the hashtags in the cluster by an importance score. This score is computed multiplying the centrality of the hashtag, i.e. the distance from the centroid, by the dimension of its virtual document, that is proportional to the popularity of that hashtag.

For exemplification purposes we will show partial results from Dataset15. We show some sample clusters, for each cluster the top terms and the top hashtags are visualized:

Table 2. Cluster example for Dataset15 with $k = 100$

top terms	december, weather, light, red, degree, middle, warm, blue, green, rain.
top hashtags	buylightmeup, globalwarming, wdisplay, december, wiki, earthquake, climatechange, wheresthesnow, iwantsnow, die.

In the example in Table 2 hashtags and terms are mainly about the weather in middle December 2011, which resulted to be quite warm. We can notice some noise (like "die" or "buylightmeup"), this is due to the small number of clusters, with respect to the high number of topics in the dataset. This is even more notable for $k = 20$, when a large number of different topics are aggregated in a few unique clusters.

Table 3. Cluster example for Dataset15 with $k = 500$

top terms	occupy, ows, wall, street, protest, ndaa, movement, afghanistan, noccupy, st.
top hashtags	ndaa, ows, occupy, occupywallstreet, china, peace, yyc, economy, kpop, washington.

In Table 3 we can see that the tokens of which the top hashtags are composed are often present in the top terms. Some hashtags with ambiguous meaning can be understood reading the relative top terms. For example, in this cluster, clearly related to the popular Wall Street protest, "ows" can be explained with the top terms "occupy", "wall", "street". It is often evident that top hashtags are understandable themselves, this is because they are the most popular hashtags, the easiest to compose or read.

Figure 3 shows evaluation results for clustering Dataset14 using different values for k. The evaluation measures are the average *Inter-cluster* distance and the

average *Intra-cluster* distance, calculated according to Jaccard distance measure presented in (2). The first measure describes how well data is separated, computing the average of the distances between the centroids, higher values mean better separation. The second measure defines the average distance between points in the same cluster, lower values mean higher density in the cluster and a better separation. We can see that both measures tend to stabilize themselves for $k > 100$ at an average of 0.95 for Inter-cluster distance and 0.75 for Intra-cluster distance, and maintain this trend up until $k = 1000$. For $k = 20$, $k = 40$ and $k = 60$, the Inter-cluster distance has lower values: 0.600, 0.621 respectively 0.716, meaning clusters are not very well defined, but it increases as the number of clusters grows. The Intra-cluster distance rises slowly from 0.673 to a maximum of 0.757, meaning the sparsity of cluster points within each cluster has little variation.
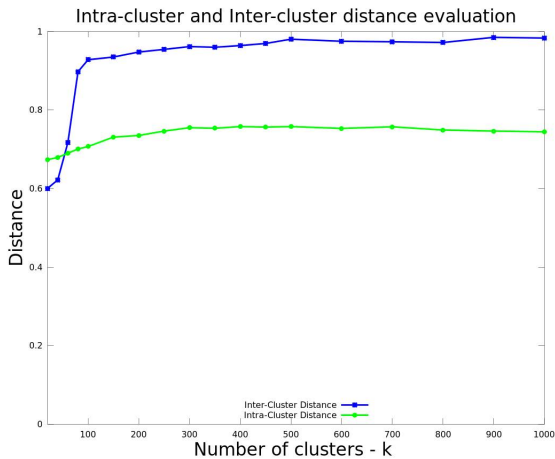

Fig. 3. Evaluation of K-means for Dataset14 by varying k

For a small k, i.e. $k = 20$, the size of the clusters varies quite a lot, some are larger while others are smaller, whereas for a larger number of k, i.e $k = 500$ or $k = 1000$, we notice a uniform distribution of cluster sizes.

Another conclusion that can be drawn from our experiments is that, for such a large dataset, results improve as the k increases, i.e. $k = 100$ or $k = 500$. The precision of semantic association between top hashtags and top terms increases because clusters becomes more specialized. Exploring and tuning the k parameter will bring to a better comprehension of the hashtags distribution.

6   Conclusions and Future Work

In this paper we experiment with clustering and Twitter hashtags. We describe a hashtag as the concatenation of tweets in which it appears. Working on large

datasets and with distributed clustering algorithms we have obtained interesting results about the semantic association of hashtags. We can see a clear connection between hashtags and the top terms of the cluster. These results can bring future improvements and applications of our idea, refining the tuning of the algorithm and experimenting with even more options and number of clusters. We are also planning to use a larger dataset, that we have already extracted, which consist of one week of tweets.

By clustering hashtags we succeed in making an unsupervised classification into flexible groups that are not constrained by a target class. In the authors' opinion a traditional classification task into predetermined topics would not suffice in respect with the variety of constantly changing daily chatter. Topics in Twitter are divided into long term topics and short term topics, usually generated by important news. A topic can be described with the help of several hashtags and viral terms grouped together according to similarity.

The main applications of the obtained research results are hashtag prediction and recommendation tasks. Clustering is useful for restricting the search base for the recommendation candidates. Searching through all the dataset can be a time consuming task, while searching within clusters can reduce this workload. If we wish to have a small group of candidates to suggest from, the granularity of the clustering task must be high (large k). If we wish to recommend more general hashtags, we can cluster the dataset into somewhat bigger groups (small k). Hashtags in clusters are ranked according to their frequency.

The results presented above can be later refined through NLP techniques in order to discover synonyms, antonyms etc. As previously mentioned, the smaller the cluster size, the more concise and specific the topic, the stronger the connection between hashtags. Clustering captures the co-occurrence of terms and hashtags. Once the grouping is accomplished, one can apply named entity recognition models in order to discover related entities or sentiment analysis techniques for finding opinions.

In future works we wish to perform hierarchical clustering on hashtags with the purpose of bringing them closer to a taxonomy. Several algorithms and machine learning techniques, like classification or recommendation of hashtags, could be applied to hashtag virtual documents in order to extract or suggest useful information. One possible application could be a tool for generating a human readable explanation of the meaning of a hashtag, using the top terms in the clusters. Comparing hashtags can be useful for suggesting the most popular hashtags to a user, in order to help increase the popularity of his tweets.

References

1. Hannon, J., Bennett, M., Smyth, B.: Recommending twitter users to follow using content and collaborative filtering approaches. In: Proceedings of the Fourth ACM Conference on Recommender Systems, RecSys 2010, pp. 199–206. ACM, New York (2010)
2. Kireyev, K., Palen, L., Anderson, K.: Applications of Topics Models to Analysis of Disaster-Related Twitter Data (December 2009)

3. Phelan, O., McCarthy, K., Smyth, B.: Using twitter to recommend real-time topical news. In: Proceedings of the Third ACM Conference on Recommender Systems, RecSys 2009, pp. 385–388. ACM, New York (2009)
4. Weng, J., Lim, E.P., Jiang, J., He, Q.: Twitterrank: finding topic-sensitive influential twitterers. In: Proceedings of the Third ACM International Conference on Web Search and Data Mining, WSDM 2010, pp. 261–270. ACM, New York (2010)
5. Romero, D.M., Meeder, B., Kleinberg, J.: Differences in the mechanics of information diffusion across topics: idioms, political hashtags, and complex contagion on twitter. In: Proceedings of the 20th International Conference on World Wide Web, WWW 2011, pp. 695–704. ACM, New York (2011)
6. Gruhl, D., Guha, R., Liben-Nowell, D., Tomkins, A.: Information diffusion through blogspace. In: Proceedings of the 13th International Conference on World Wide Web, WWW 2004, pp. 491–501. ACM, New York (2004)
7. Java, A., Song, X., Finin, T., Tseng, B.: Why we twitter: understanding microblogging usage and communities. In: Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 Workshop on Web Mining and Social Network Analysis, WebKDD/SNA-KDD 2007, pp. 56–65. ACM, New York (2007)
8. Thomas, K., Grier, C., Song, D., Paxson, V.: Suspended accounts in retrospect: an analysis of twitter spam. In: Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference, IMC 2011, pp. 243–258. ACM, New York (2011)
9. Wang, A.H.: Dont't Follow me: Spam Detection in Twitter. In: Proceedings of the International Conference on Security and Cryptography (SECRYPT) (July 2010)
10. Kwak, H., Lee, C., Park, H., Moon, S.: What is twitter, a social network or a news media? In: Proceedings of the 19th International Conference on World Wide Web, pp. 591–600. ACM (2010)
11. Chen, J., Nairn, R., Nelson, L., Bernstein, M., Chi, E.: Short and tweet: experiments on recommending content from information streams. In: Proceedings of the 28th International Conference on Human Factors in Computing Systems, CHI 2010, pp. 1185–1194. ACM, New York (2010)
12. Ellen, J.: All about microtext - a working definition and a survey of current microtext research within artificial intelligence and natural language processing. In: ICAART (1) 2011, pp. 329–336 (2011)
13. O'Connor, B., Krieger, M., Ahn, D.: TweetMotif: Exploratory Search and Topic Summarization for Twitter. In: Cohen, W.W., Gosling, S., Cohen, W.W., Gosling, S. (eds.) ICWSM. The AAAI Press (2010)
14. Xu, T., Oard, D.W.: Wikipedia-based topic clustering for microblogs. Proceedings of the American Society for Information Science and Technology 48(1), 1–10 (2011)
15. Phan, X.H., Nguyen, L.M., Horiguchi, S.: Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In: WWW 2008: Proceeding of the 17th International Conference on World Wide Web, pp. 91–100. ACM, New York (2008)
16. Pennacchiotti, M., Popescu, A.M.: A machine learning approach to twitter user classification (2011)
17. Rangrej, A., Kulkarni, S., Tendulkar, A.V.: Comparative study of clustering techniques for short text documents. In: Proceedings of the 20th International Conference Companion on World Wide Web, WWW 2011, pp. 111–112. ACM, New York (2011)
18. Dean, J., Ghemawat, S.: Mapreduce: simplified data processing on large clusters. Commun. ACM 51(1), 107–113 (2008)

19. Hadoop, `http://hadoop.apache.org`
20. Papadimitriou, S., Sun, J.: Disco: Distributed co-clustering with map-reduce: A case study towards petabyte-scale end-to-end mining. In: Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, ICDM 2008, pp. 512–521. IEEE Computer Society, Washington, DC (2008)
21. Shvachko, K., Kuang, H., Radia, S., Chansler, R.: The hadoop distributed file system. In: IEEE/NASA Goddard Conference on Mass Storage Systems and Technologies, pp. 1–10 (2010)
22. Apache Mahout, `http://hadoop.apache.or`
23. Wegener, D., Mock, M., Adranale, D., Wrobel, S.: Toolkit-based high-performance data mining of large data on mapreduce clusters. In: Proceedings of the 2009 IEEE International Conference on Data Mining Workshops, ICDMW 2009, pp. 296–301. IEEE Computer Society, Washington, DC (2009)
24. Chu, C.T., Kim, S.K., Lin, Y.A., Yu, Y., Bradski, G.R., Ng, A.Y., Olukotun, K.: Map-reduce for machine learning on multicore. In: Schölkopf, B., Platt, J.C., Hoffman, T. (eds.) NIPS, pp. 281–288. MIT Press (2006)
25. Cascading, `http://www.cascading.org/`
26. Lucene, `http://lucene.apache.org/core/old_versioned_docs/versions/3_0_1/api/all/org/apache/lucene/analysis/Analyzer.html`
27. Willett, P.: The Porter Stemming Algorithm: Then and Now. Program: Electronic Library and Information Systems 40(3), 219–223 (2006)
28. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Cam, L.M.L., Neyman, J. (eds.) Proc. of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297. University of California Press (1967)
29. McCallum, A., Nigam, K., Ungar, L.: Efficient clustering of high-dimensional data sets with application to reference matching. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 169–178. ACM (2000)