Comparison of Clustering Algorithms for the Identification of Topics on *Twitter*

Marjori N. M. Klinczak and Celso A. A. Kaestner

***Abstract*—Topic Identification in Social Networks has become an important task when dealing with event detection, particularly when global communities are affected. In order to attack this problem, text processing techniques and machine learning algorithms have been extensively used. In this paper we compare four clustering algorithms – k-means, k-*medoids*, DBSCAN and NMF (Non-negative Matrix Factorization) – in order to detect topics related to textual messages obtained from *Twitter*. The algorithms were applied to a database initially composed by *tweets* having hashtags related to the recent Nepal earthquake as initial context. Obtained results suggest that the NMF clustering algorithm presents superior results, providing simpler clusters that are also easier to interpret.**

***Index Terms*—text processing; clustering algorithms; NMF algorithm; Twitter topics identification.**

I. Introduction

Social Networks are naively defined as a way for one person to meet up with other people on the Net. They constitute a global phenomenon, and are employed for several activities, such as work, entertainment and personal use [24]. They are also a huge source of information, reflecting peoples' opinions and desires, and serve as an almost instantaneous channel for communication and spreading news [23].

However, to extract useful information from the texts that appear in the social networks is not an easy task, due to the huge size of the data involved and the speed of their creation [25]. The problem is only recently being attacked, employing automatic procedures whose fundamentals include Text Processing (TP) [1, 2] and Machine Learning (ML) [3] techniques.

The *Twitter* is a microblog created in 2006 and widely used over the world. Nowadays it contains more than 465 million of accounts, and its messages form a textual database where discussions and opinions of several matters can be found [17]. Also, it contains information about on-time events of many types and scales [5]. The high connectivity and the almost instantaneous responses entails that this social network is the one where the information travels faster [23].

Therefore, the *Twitter* can be considered as a real-time source of information [6]; this is especially true in the case of global, catastrophic and/or big media events. In this paper we discuss the automatic extraction of topics - a set of cohesive terms related to a specific subject – that appear in the *tweets* obtained from a broad initial context given by a list of hashtags.

The topics are obtained as a result of a clustering procedure as follows:

- initially the tweets are converted in plain text (some metadata are also stored for additional use);
- the obtained texts are preprocessed using classical techniques such as case conversion, stop-words removal and stemming; *urls*, *retweets* and profile information are also removed, this steps are showed at figure 1.
- a (*tweet* x term) matrix – which corresponds to the (document x term) matrix in the Information Retrieval area – is obtained according to the well-known Vector Space Model [1];
- the clustering algorithms are applied to this matrix; each obtained cluster is associated to a topic;
- the quality of the obtained clusters is considered in two ways: using the intra / inter cluster measures and using a word cloud associated to each cluster.

The overall clustering procedures were tested using *Twitter* data related to the recent Nepal's Earthquake.

The rest of this paper is organized as follows: section 2 presents similar works that deal with event extraction from social networks; section 3 describes the text preprocessing techniques used, the obtained the (*tweet* x term) matrix, and the employed clustering algorithms; section 4 presents the testing cases and discusses the obtained results; finally, section 5 presents the conclusions and future work.

II. Similar Works and Related Research

In the literature, there are several works dealing with the identification of topics that appear on Social Networks. Related applications range from real time event detection, the impact of natural disasters, opinion mining and the identification of diseases for public health actions [6]. Some of these works are briefly summarized in the following.

Shamma, Kennedy and Churchill [7] use as research scenario the debate between Barack Obama e John McCain that occurs in September 26th, 2008, during the national campaign for the USA presidency. They investigate the practice of sharing short messages (microblogging) around live media events. A

Marjori N. M. Klinczak is with the Mosaic Web Company, and is currently a student at the Graduate Program in Applied Computer Science at the Federal University of Technology of Paraná (UTFPR), Curitiba, Paraná, Brazil (e-mail: mnmk.lvseg@gmail.com).

Celso A. A. Kaestner is a senior professor at the Graduate Program in Applied Computer Science at the Federal University of Technology of Paraná (UTFPR), Curitiba, Paraná, Brazil (e-mail: celsokaestner@utfpr.edu.br).

reactions' database was obtained from *Twitter*, considered as the act of live annotation of a broadcast media event. *Tweets* that contain the hashtags #*current*, #*tweetdebate* and #*debate08* were recorded, generating a database with 3,238 *tweets* from 1,160 different users. The traffic volume per minute was computed, as well as a network graph of all users and their tag relations as seen when clustered by tags; half of them used the #*current* tag when discussing the debates during air time. The authors hypothesized that frequent terms from *Twitter* traffic would reflect the topics being discussed; this was tested by breaking the debate into nine pieces and by computing the corresponding topic segments. They also show that the usage of twitter was not one of summarizing or even discussion about the debate on hand. Finally, they conclude that *Twitter* traffic can provide insights into segmentation and entity detection, however, the correlation between content leaves further questions to be investigated.

Sakaki, Okazaki and Matsuo [8] investigate if a real-time event can be detected only by monitoring *Twitter* activity. For example, when an earthquake occurs, there are many *tweets* related to this event, which enables its occurrence promptly, simply by observing these *tweets*. They use data extracted in Japan from this social media using the terms *earthquake*, *shaking* and *typhoon*. Then they use the size of the *tweet*, textual attributes given by a set of keywords and their context as attributes, and a Support Vector Machine (SVM) classifier. Subsequently, they produce a probabilistic spatiotemporal model for the center of the target event, by using the geographic location of the emitted tweets. In summary, they consider Twitter as a sensor network, and use *Kalman* filtering and particle filtering to provide location estimation, as widely employed in ubiquitous/pervasive computing. The paper contains detailed experimental results proving that their approach is feasible. They also propose a system that can detect an earthquake with high probability – 96% of earthquakes of Japan Meteorological Agency (JMA) seismic intensity scale 3 or more are detected – merely by monitoring tweets. This system also sends e-mails to registered users, and this notification is delivered much faster than the announcements that are broadcast by the JMA.

Becker et al. [5] argue that microblogging on social media have emerged as a powerful real-time mechanism to detect events. Due to its nature – short messages almost instantaneously propagated in the web – *Twitter* is particularly well suited as a source of theses contents. The authors focus their work in analyzing the stream of *tweets* to distinguish between messages about real world events and non-event messages. To do so they use an incremental, online clustering algorithm in order to effectively cluster a stream messages in real time. Employed features include temporal (e.g. traffic-volume), social (e.g. *retweets*), topical (e.g. term cohesion) and *Twitter* centered features (e.g. tag usage). They test a variety of classifier using the *Weka* platform[1] and the SVM (Support Vector Machines) algorithm. The conducted evaluation uses the macro-averaged F1 metric and precision for k clusters [9], obtaining and F1 score of 0.837 in the test set, and a precision superior of 80 % in the best clustering case (K=5).

Gupta and Kumaraguru [6] study the credibility of the information that is found in the *Twitter* messages. They use data related to 14 high impact global events of 2011, including for example the UK Riots, the Libya crisis, an earthquake in Virginia and the hurricane Irene. From the analyzed data, on average only 30% of the posted *tweets* related to an event really contain situational information about the event, while 14 % were merely spam. In addition, only 17 % of the total *tweets* posted contain situational awareness information that is credible. The authors use regression analysis to identify the important content and source-based features, in order to predict the credibility of a *tweet*. Employed features include the number of unique characters, swear words, pronouns, and emoticons in the text, and user based features like the number of followers and length of the username. A supervised machine learning procedure and the relevance feedback approach were used to rank *tweets* according to their credibility score. This performance evaluation has proved to significantly enhanced the results, allowing the automatic extraction of credible information from *Twitter*.

Godfrey [10] analyze the *Twitter* data during the FIFA World Cup. They employ cluster analysis and text-mining to extract underlying patterns from a database composed by large collections of text messages. A collection of about 30,000 *tweets* were extracted just before the 2014 World Cup started. To eliminate spurious tweets, unrelated to the main theme, they use an algorithm that combined the DBSCAN algorithm and a consensus matrix. Then the authors perform cluster analysis using k-means [3] and the Non-Negative Matrix Factorization (NMF) algorithm. Obtained results were very similar but, according to the authors, the NMF proved to be faster and provided results that are more easily to interpret. Result comparison in the paper is subjective, using graphics and figures from two visualization tools, *Gephi* [2] and *Wordle* [3].

Another study involving FIFA was done by Klinczak and Kaestner [12]; this study is related to the recent corruption scandal in the FIFA federation. Differently from other works, they are compared directly the performance of several clustering algorithms (k-means, k-*medoids* and NMF) in the same data obtained from *Twitter*, using the hashtags #*fifa* and #*fifagate* as initial context; after the text preprocessing the dataset has 2,460 *tweets*. The employed algorithms present similar results, but the NMF algorithm presents the best results in most of the cases. This can be partially explained because in the NMF algorithm the same term can be appear in many clusters with different weights.

The above research works make clear the importance of social networks – and particularly micro-bloggers as *Twitter* – in the detection and analysis of real-time events. Cluster analysis is frequently employed, but a direct comparison of the available clustering algorithms cannot be easily found. Also, only the last two works employ the NMF algorithm, considered

[1] http://www.cs.waikato.ac.nz/ml/weka/
[2] https://gephi.org/
[3] http://www.wordle.net/

nowadays the best technique to perform clustering in text applications. In the paper we address this research task.

III. Employed Techniques: Text Preprocessing and Clustering

To use *Twitter* data for Information Extraction purposes a series of preprocessing steps must be followed. Initially, the *tweets* are recorded using a specific API using the R language[4]; it includes several filters, such as the presence of specific hashtags, the language employed in the message, geographic location restrictions, information about *retweets*, etc. Besides the text message itself, the obtained record includes meta-attributes such as the user-id, time/date of the *tweet*, geographical location of the emitter, and some metadata like links and images, that can be used for specific purposes.

A. Preprocessing and Text Model

Text preprocessing is a very important step to obtain the semantic elements related to the message. The use of techniques originally employed in Information Retrieval (IR) [1, 2] is convenient for this task.

In the case of *Twitter* messages additional elements appear: due to the small size of the message users extensively use abbreviations and emoticons, introducing some noise in the pure text model.

The classical text preprocessing steps are (se also Figure 1):

- text unit identification: in this case the *tweet* textual information is considered the basic unit;
- case-folding: to standardize the extracted characters; it can include additional conversions because many tweets have strange characters;
- stop-words removal: stop-words are very frequent textual elements that carry almost no semantics and can be eliminated; a stop-word list includes articles, prepositions and conjunctions; in some applications, like that, numbers are also eliminated. Also is eliminated the initial hashtags and the noise like the emoticons and abbreviations.
- stemming: is a procedure that aims to connect textual elements of similar semantics, by obtaining their *root*; suffixes and prefixes are eliminated, plurals and verbal variations of the same term are reduced to a unique form.

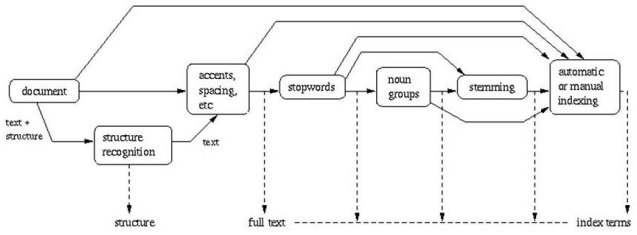

Fig. 1. *Text Preprocessing Steps [1]*

As result of the preprocessing step, each *tweet* is now a series of text elements, usually called indexing terms in the IR terminology. By computing the union of these terms, we have – after defining an order – a global list of terms of the database.

Then a text model must be used: the most employed model nowadays is the Vector Space Model (VSM) [1] where each term corresponds to a dimension in a huge *NT*-dimensional vector space, *NT* being the number of terms. Obviously, each tweet will contain only few terms: the message collection is therefore very sparse in this space. It is usual to view the text collection as a huge matrix *D* = (text unit X term), or *D* = (*tweet* x term) in this case.

Given a pair (*tweet*, term) the corresponding *D = (d, t)* entry is the weight of the term *t* in the *tweet d*. Several weighting schemes can be employed: the simplest is the Boolean model, where 0 is used for absence and 1 for presence of the term *t* in the *tweet d*; more employed schemes include the frequency model, where the weight if the frequency *tf* of the term *t* in the tweet *d*, or the *tf-idf* (term frequency - inverse document frequency) model, where the weight of the term *t* in the *tweet d* is given by:

$$tf\text{-}idf\,(d, t) = tf\,(d, t) * log(||ND|| / df\,(t)) \quad (1)$$

where $||ND||$ is the total number of documents, *tf* is the frequency of the term *t* in the *tweet d* and *df* stands for the number of documents in which the term *t* appears.

B. Clustering algorithms

After text processing the obtained (*tweets* x terms) *D* matrix form the base for cluster computations, following the classical scenario employed in ML. So, several clustering algorithms can be readily employed.

a. k-means

An oldest option is to use the well-known k-means algorithm [13]. Briefly, it works as follows: (a) a series of k initial points are randomly generated; (b) these points are considered as cluster centers (or means); (c) each text instance is used as input: it will be assigned to the cluster with closest center; (d) the value of the cluster mean is updated to consider this new cluster element; (e) steps (c) to (d) are repeated until no changes occur in the instance cluster labels (the cluster assigned to it) [13].

The employed metric for distance is very important: in the case of text documents and following the VSM, it is common to use one of two metrics: the classical Euclidian distance or the cosine similarity measure, given, for documents d_1 and d_2, by:

$$dist\,(d_1, d_2) = < d_1, d_2 > / || d_1 ||.|| d_2 || \quad (2)$$

where <, > stands for the dot vector product, and $||d||$ is the norm of the document *d*.

b. k-medoids

The k-*medoids* algorithm is similar to the k-means, the only difference being the fact that in this case the mean of each cluster is replaced by its "*medoid*", the most central existing data point. That is, in the k-means centers of the clusters are the means of theirs points, not necessary a data point, whereas in

[4] https://www.r-project.org/

the k-*medoids* algorithm these values are chosen to be existing data-points [13].

c. Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

Another widely used cluster algorithm that uses a different principle is DBSCAN [13]. Its basic technique is to connect all the high density regions of the underlying space, the low density regions being the inter-cluster space.

In summary, DBSCAN works as follows: (a) a user-defined parameter $\varepsilon > 0$ is used to specify the radius of a neighborhood for every object; the ε-neighborhood of an object *O* is the space within a radius centered at *O*; (b) the density of a neighborhood can be measured simply by the number of objects in the neighborhood; to determine whether a neighborhood is dense or not, DBSCAN uses another parameter *MinPts*, which is a density threshold; (c) a core object is one where its ε-neighborhood contains at least *MinPts* objects; they are the pillars of dense regions; (d) after computing core objects, the clustering task is reduced to the use of core objects and their neighborhoods to form dense regions which are the clusters; (e) for a core object *O* and an object *P*, we say that *P* is directly density-reachable from *O* if it is within the ε-neighborhood of *O*; (f) using the directly density-reachable transitive relation, a core object can connect objects to form a dense region.

Here again, the metric employed to evaluate the distance between instances is crucial, the cosine distance being the most employed one for text applications.

d. Non-negative Matrix Factorization (NMF)

The NMF method was initially proposed by Lee e Seung [14] as an alternative for the Principal Component Analysis (PCA) method, which is classically used in matrix decompositions.

To remember, PCA is an orthogonal linear operator that transforms the data to new coordinates, such that the greatest variance by some projection of the data lies on the first coordinate (the first principal component), the second greatest variance lies on the second coordinate, and so on. That is, given an ($m \times n$) matrix *M*, PCA computes $M = W \cdot \Sigma \cdot W^T$, where Σ is a diagonal matrix of the principal components (sorted by magnitude), and *W* is formed by the eigenvectors' coordinates. To perform data reduction the sub-matrix of size ($k \times k$) of Σ, usually noted as Σ_k , is commonly employed [15].

PCA has been successfully used in text applications, but negative values that appear in the decomposition are difficult to interpret and sometimes contradict the reality. In NMF decomposition, on the other hand, non-negativity is preserved, making the resulting matrices easier to inspect, especially in applications such as text processing, where the non-negativity is inherent to the data being considered.

In NMF the original ($m \times n$) matrix *D* is decomposed as $D \approx W \cdot H$, where *W* and *H* have dimensions ($m \times k$) and ($k \times n$), respectively, and *k* is a user-defined parameter that depends on the application; in our case, it is associated to the number of considered *tweet* topics.

The decomposition $D \approx W \cdot H$ for given *k* is not an exactly solvable problem in general, so it is commonly approximated numerically. Given the ($m \times n$) matrix *D* and a positive integer $p < min$ (m, n), find two non-negative matrices *H* and *W* that minimize the functional:

$$f(W, H) = (1/2) \cdot ||D - W \cdot H||^2 \quad (3)$$

where ||_|| is a matrix Frobenius norm, and all the elements of *W* and *H* must be positive or zero, that is, w_{ij} , $h_{ij} \geq 0$.

Several procedures have been used to solve this optimization problem, such as multiplicative update algorithms, gradient descent algorithms and alternate least square algorithms (ALS). These algorithms are summarized by Berry et al. [16], that also deal with algorithm performance in large datasets.

The NMF clustering algorithm has been employed successfully, mainly because it can be adapted to specific applications [11, 16].

IV. Clustering Comparison

We did some experiments to compare the performance of the described clustering algorithms: k-means, k-*medoids*, DBSCAN and NMF. The employed database use *Twitter* data obtained in May 19, 2015, using the two hashtags "*#NepalEarthquake*" and "*#NepalQuake*" as initial context. We use the Twitter API [4] from its R Language interface, initially obtaining 10,000 tweets, restricted to the English language. Some of them were discarded because they were not in English or due to the presence of unknown characters; also, due to performance issues, we restrict the current analysis to dataset of 500 *tweets*.

Text preprocessing follow the steps described in the previous Section. Our basic text unit is the message part of each *tweet*; we perform case folding and characters standardization, and use the stop-words list obtained from the work of [18], with some additional element such as "*RT*". For stemming we use the well-known Porter's stemming algorithm [19], and we only considered terms with more than two characters. The final (*tweet* x term) matrix was constructed using term frequency; in our experiments its dimension is (500 x 1203).

All clustering algorithms were executed using the R Language version 3.0.2. The employed metric was the Euclidian distance.

In order to compare the clustering algorithms, we use the same value of k for the k-means and for the k dimension of the NMF. We present the results for k = 3, 5 and 7. DBSCAN results are not directly comparable, but are given for reference; it obtains 4 clusters.

Obtained results are summarized in the following. To compare the results, we use the clustering measures separation (BSS) and cohesion (WSS), given by the formulas [20]:

$$BSS = \Sigma_i \, ||C_i|| \cdot (m - m_i)^2 \quad (4) \text{ and}$$

$$WSS = \Sigma_i \; \Sigma_{x \, \epsilon \, Ci} (x - m_i)^2 \quad (5)$$

where $||C_i||$ and m_i stands for the size and the mean of the cluster *i*, and *m* is the global mean of the dataset and *x* is a data point that belongs to the cluster C_i.

Table I presents the results for the case of the k-means algorithm: the first column indicates the number of clusters; the second column gives the intra-cluster measure or cohesion WSS; the third one indicates the inter-cluster measure or separation BSS; and the last column gives the number of instances that occur in each cluster [13, 22].

Similarly, Table II presents the results obtained with the k-*medoids* algorithm.

Results for the DBSCAN algorithm are summarized in Table III. As it is a density-based algorithm, its performance cannot be adequately measured using intra and inter-cluster measures. So, we indicate in Table III the employed parameters and obtained cluster characteristics. The first column indicates the neighbor proximity parameter; the second column shows the number of obtained clusters; third and fourth ones present the number of seed and border points for each cluster respectively. In all the experiments the minimum number of points (*MinPts*) is set to 0.2.

TABLE I. RESULTS FOR K-MEANS ALGORITHM

K: #of clusters	*WSS*	*BSS*	$\|C_i\|$
3	487.25 3,109.20 675.48	731.34	8 1,172 23
5	2,851.20 487.25 291.44 21.17 337.64	1,014.59	1,162 8 16 6 11
7	184.00 0.00 2,420.00 256.35 272.85 131.14 340.73	1,397.99	6 1 1,126 17 20 7 26

TABLE II. RESULTS FOR K-*MEDOIDS* ALGORITHM

K: #of clusters	*WSS*	*BSS*	$\|C_i\|$
3	4,545.73 2,445.67 660.34	702.00	1,183 12 8
5	4,529.16 3,038.71 62.42 63.10 2,539.19	1,129.00	1,162 8 16 6 11
7	2,532.10 3,030.00 63.15 3,401.13 2,529.70 0.00	2,592.92	1,149 11 12 7 113 10 1

In the case of the NMF algorithm, we use the default multiplicative update algorithm; this is not a deterministic algorithm, so different executions can provide different results. We recall that the original (*tweet* x term) matrix *D* is decomposed in the matrices (tweet x topic) W and (topic x term) *H*. So, we can analyze clusters related to *tweets* and related to terms. We compute cohesion and separation for both options considering that each one of the *k* lines of the *H* matrix is the "center" of a cluster; similarly, each one of the columns of the *W* matrix is considered also the "center" of a cluster.

TABLE III. RESULTS FOR THE DBSCAN ALGORITHM

ε	*# of clusters*	*# of seed points*	*# of border points*
2	5	1,076 11 8 6 5	14 2 1 0 2
3	4	1,146 6 5 0	2 0 0 44
4	2	1,172 1	22 4

In the case of the NMF algorithm, we use the default multiplicative update algorithm; this is not a deterministic algorithm, so different executions can provide different results. We recall that the original (*tweet* x term) matrix *D* is decomposed in the matrices (tweet x topic) W and (topic x term) *H*. So, we can analyze clusters related to *tweets* and related to terms. We compute cohesion and separation for both options considering that each one of the *k* lines of the *H* matrix is the "center" of a cluster; similarly, each one of the columns of the *W* matrix is considered also the "center" of a cluster.

Tables IV and V indicate the values of cohesion, separation and cluster size for 3, 5 and 7 clusters. Results for NMF-*tweets* are comparable to the ones obtained for k-means and k-*medoids*; for NMF-terms they are difficult to interpret since several clusters are empty, or has few elements.

TABLE IV. RESULTS FOR THE NMF ALGORITHM FOR *TWEETS*

K: #of clusters	*WSS*	*BSS*	$\|C_i\|$
3	513.12 3,266.52 76.16	830.66	59 391 50
5	177.24 2,695.30 139.04 338.46 296.02	834.44	58 318 23 57 44
7	207.60 39.08 224.15 2,719.07 0.93 160.97 54.37	1,139.43	45 13 27 322 15 29 49

RESULTS FOR NMF ALGORITHM FOR TERMS

K: #of clusters	*WSS*	*BSS*	$\|C_i\|$
3	0.00 344.67 4,399.31	43.32	0 3 1,200
5	0.00 0.00 0.00 4,057.80 235.50	45,298.76	1 1 1 1,198 2

K: #of clusters	*WSS*	*BSS*	*$\|\|C_i\|\|$*
3	0.00 344.67 4,399.31	43.32	0 3 1,200
7	4,264.95 0.00 0.00 184.00 0.00 0.00 0.00	23.37	1,195 0 1 6 1 0 0

We also compute the word cloud of each cluster using Wordle for an empirical evaluation. Some of these results, for k = 3, are presented.

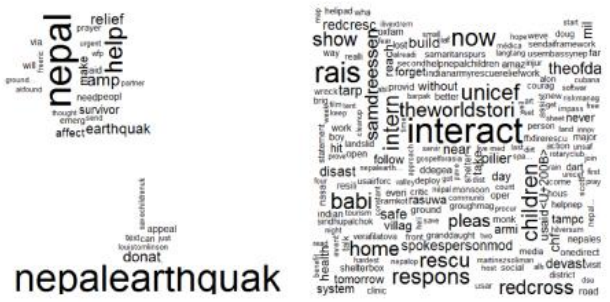

Fig 2. *Word clouds of terms for the clusters using k-means, k=3*

Figure 2 presents the terms – as word clouds – that appear in each of the clusters for the k-means algorithm. Figure 3 presents similar word clouds for the k-*medoids* algorithm. Figure 4 presents the word clouds obtained from DBSCAN with provides 4 clusters. Finally Figure 5 presents the word clouds of terms obtained from the NMF algorithm for 3 clusters.

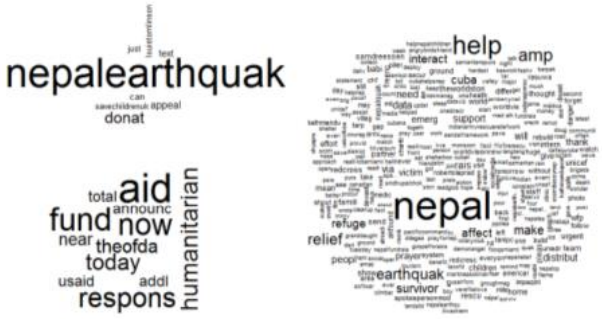

Fig. 3. *Word clouds of terms for the clusters using k-medoids algorithm, k=3*

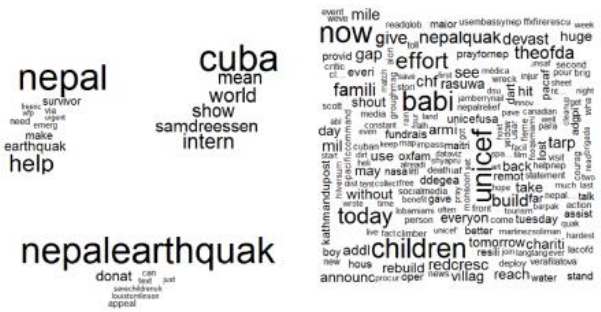

Fig. 4. *Word clouds of terms for the clusters using DBSCAN algorithm, ε=3*

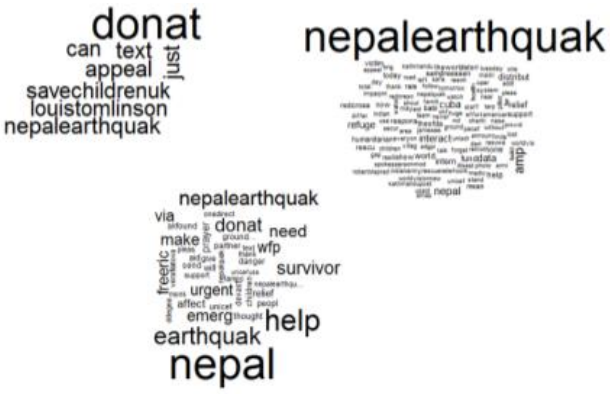

Fig. 5. *Word clouds of terms for the clusters using NMF, k=3*

V. Conclusions and Future Work

In this paper we analyze topics identification and clustering algorithms in the context of text messages obtained from the social microblog *Twitter*. To do so, we use text preprocessing techniques and Machine Learning clustering algorithms.

This study was tested using a dataset of *tweets* related to the recent Nepal's earthquake. Results show that this proposal is feasible for human analytical purposes: using word clouds is possible to obtain the main topics related to the given initial context.

Four different clustering algorithms are employed: k-means, k-*medoids*, DBSCAN and NMF. We numerically compare their results using the cohesion (WSS) and separation (BSS) measures. Results are presented on Tables I, II, III, IV and V; in general, these tables show similar results; in the case of the NMF algorithm, the clusters obtained from the (topic x term) matrix are difficult to compare to the others, since most if the terms are clustered together.

The word clouds associated to each cluster is used to show an empirical evaluation, in this case we argue that the NMF algorithm present the best results, since it seems that the associated topics are easy to understand; this result is also in accordance with the conclusion obtained by Godfrey [21].

We plan to extend this research work in several ways: (a) testing alternative text preprocessing techniques; (b) employing different term weighting schemes; (c) testing different

algorithmic options, particularly in the case of the NMF algorithm where several optimization options are available; and (d) testing human interpretations of the word clouds associated to each cluster.

We also plan to extend this work by using the geographic information – latitude and longitude of the *tweet* emitter – available on *Twitter*: using this information it will be possible to generate graphs associated to the emission and spreading of the specific topics that appear on this social microblog.

REFERENCES

[1] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. Modern Information Retrieval. ACM Press, New York. 1999.

[2] C. D. Manning, P. Raghavan and H. Schütze. Introduction to Information Retrieval, Cambridge University Press. 2008.

[3] T. M. Mitchell. 1997. Machine Learning, McGraw-Hill.

[4] Twitter Documentation. https://dev.twitter.com/rest/public. Accessed at May 29, 2015.

[5] Hila Becker, Mor Naaman and Luis Gravano. Beyond Trending Topics: Real-World Event Identification on Twitter. Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media, 438–441. 2011.

[6] Aditi Gupta and Ponnurangam Kumaraguru. Credibility Ranking of Tweets during High Impact Events. Proceedings of the 1st Workshop on Privacy and Security in Online Social Media, article 2. 2012.

[7] David A. Shamma, Lyndon Kennedy e Elizabeth Churchill, 2009. Tweet the Debates: Understanding Community Annotation of Uncollected Sources. *Proceedings of the first SIGMM Workshop on Social Media*, vol. 22(1), pp. 3-10.

[8] Takeshi Sakaki, Makoto Okazaki and Yutaka Matsuo. Earthquake Shakes Twitter Users: Real-Time Event Detection By Social Sensors. Proceedings of the 19th International conference on Word Wide Web, 851–860. 2010.

[9] C. D. Manning, P. Raghavan e H. Schütze. *Introduction to Information Retrieval*, Cambridge University Press. 2008.

[10] Daniel Godfrey, Caley Johns, Carol Sadek, Carl Meyer e Shaina Race, 2014. *A Case Study in Text Mining: Interpreting Twitter Data From World Cup Tweets*. Cornell University Library, http://arxiv.org/abs/1408.5427, acessado em 20 de maio de 2015.

[11] Moody Chu and Robert Plemmons. Nonnegative matrix factorization and applications. Bulletin of the International Linear Algebra Society, 34, 2–7. 2005.

[12] Klinczak, Marjori N. M. and Kaestner, Celso A. A. Identification of Topics on Twitter: Comparison of Clustering Algorithms and Case Study. LA-CCI. 2015.

[13] J. Han and M. Kamber. Data Mining Concepts and Techniques, Morgan Kaufmann. 2001.

[14] D. D. Lee and H. S. Seung. Unsupervised learning by convex and conic coding. Advances in Neural Information Processing Systems, 9(1):515–521, MIT Press. 1997.

[15] I.T. Jolliffe. Principal Component Analysis, Springer-Verlag. 2002.

[16] Michael W. Berry, Murray Browne, Amy N. Langville, V. Paul Pauca and Robert J. Plemmons. Algorithms and applications for approximate nonnegative matrix factorization. Computational Statistics and Data Analysis, 52. 2006.

[17] Twitter Team. 2012. Twitter turns six. http://blog.twitter.com/2012/03/twitter-turns-six.html . accessed at July 08, 2014.

[18] David D. Lewis, Yiming Yang, Tony G. Rose and Fan Li. RCV1: A New Benchmark Collection for Text Categorization Research. Journal of Machine Learning Research, 5(1):361–397. 2004.

[19] Martin F. Porter. An Algorithm for Suffix Stripping. Program, 14(3): 130–137. 1980.

[20] Anil K. Jain and Richard C. Dubes. Algorithms for Clustering Data, Prentice Hall. 1988.

[21] Daniel Godfrey, Caley Johns, Carol Sadek, Carl Meyer and Shaina Race. 2014. A Case Study in Text Mining: Interpreting Twitter Data From World Cup Tweets. Cornell University Library, http://arxiv.org/abs/1408.5427 , accessed in May 20th, 2015.

[22] Anil K. Jain and Richard C. Dubes. Algorithms for Clustering Data, Prentice Hall. 1988.

[23] Naaman, Boase, and Lai. *Is it really about me? Message content in social awareness streams*. CSCW10. 2010.

[24] Gupta, Aditi & Kumaraguru, Ponnurangam. *Credibility Ranking of Tweets during High Impact Events*. Proceedings of the 1st Workshop on Privacy and Security in Online Social Media. 2012.

[25] Ryaboy, Dmitriy & Lin, Jimmy. *Scaling Big Data Mining Infrastructure: The Twitter Experience*. ACM SIGKDD. 2012.

Celso A. A. Kaestner, has a BSc in Mathematics from the Pontifical Catholic University of Paraná and in Civil Engineering from the Federal University of Paraná. He has a MSc– Control Systems – and a PhD – Information Systems – in Electrical Engineering at the Federal University of Santa Catarina. He had post-doctoral positions at the École de Technologie Supérière the University of Quebec in Montreal, Canada, and at York University in Toronto, Canada. He was a Full Professor at the Informatics Department, of the Federal Technological University of Paraná in Curitiba, Brazil until his retirement in June 2015. He is currently a senior professor at the Graduate Program in Applied Computing at the same University.

Marjori Naiele Mocelin Klinczak, has a degree in Internet Systems by the Faculty of Administration and Economy of Paraná (FAE), graduate in Software Development in International Markets at the Federal University of Paraná (UFPR) and is currently studying at the Graduate Program in Applied Computer Science at the Federal University of Technology – Paraná (UTFPR). She is also the CEO of the Mosaic Web company and works with web and mobile development.