Trend Visualization on Twitter: What's Hot and What's Not?

Sandjai Bhulai, Peter Kampstra, Lidewij Kooiman, and Ger Koole
Faculty of Sciences
VU University Amsterdam
Amsterdam, The Netherlands
{*s.bhulai, p.kampstra, ger.koole*}*@vu.nl, l.e.kooiman@gmail.com*

Marijn Deurloo and Bert Kok
CCinq
Amsterdam, The Netherlands
{*marijn, bert*}*@ccinq.com*

***Abstract*—Twitter is a social networking service in which users can create short messages related to a wide variety of subjects. Certain subjects are highlighted by Twitter as the most popular subjects and are known as trending topics. In this paper, we study the visual representation of these trending topics to maximize the information toward the users in the most effective way. For this purpose, we present a new visual representation of the trending topics based on dynamic squarified treemaps. In order to use this visual representation, one needs to determine (preferably forecast) the speed at which tweets on a particular subject are posted and one needs to detect acceleration. Moreover, one needs efficient ways to relate topics to each other when necessary, so that clusters of related trending topics are formed to be more informative about a particular subject. We will outline the methodologies for determining the speed and acceleration, and for clustering. We show that the visualization using dynamic squarified treemaps has many benefits over other visualization techniques.**

***Keywords*-microblogging; Twitter; trend detection; clustering; visualization; dynamic squarified treemaps.**

I. Introduction

Twitter, a popular microblogging service, has seen a lot of growth since it launched in 2006 and commands more than 140 active million users with 340 million messages (tweets) per day as of March 2012 [1]. Twitter users write tweets about any topic within the 140-character limit and follow others to receive their tweets. An important characteristic of Twitter is its real-time nature. For example, when a major event occurs, people disseminate tweets over the network related to the event, which enables detection of the event promptly by observing the tweets. The popular events and subjects are also known as trending topics, and their detection helps us to better understand what is happening in the world.

The visualization of trending topics is an important research question, since the representation of the trending topics has a significant impact on the interpretation of the topics by the user. This visualization can be done simply by providing a list of topics, as Twitter does (see [2] and Figure 1). However, this representation suffers from a number of drawbacks that prevent the user in assessing the importance of the topic correctly. First, although the list is ordered from the most popular topic to the least popular topic, one cannot infer the importance of each topic relative to the other topics. Second, a list also does not convey the dynamics in the trend, e.g., is the topic still trending to become more popular or is a different topic growing more popular? Third, it could very well be that several topics on the list are related to each other and should be grouped into a coherent set of topics. For example, it is not clear on the outset that topics 3 and 9 in Figure 1 are related to each other. This group of topics could provide more semantics to users than a single topic alone.

1) #PrayforMexico
2) #SocialMovies
3) #temblor
4) Sismo de 7.8
5) Earthquake in Mexico
6) John Elway
7) Pat Bowlen
8) Marcelo Lagos
9) Azcapotzalco
10) Niñas de 13 y 14

Figure 1. Trending topics on Twitter, recorded on 20 March 2012.

A popular method to visualize trending topics is a tag cloud (see Figure 2). However, the research on the effectiveness of this visualization technique is not conclusive. Sometimes, a simple list ordered by frequency may work better in practice than fancy sequential or spatial tag clouds [3]. In other research (e.g., [4]) an alphabetically ordered list performed best with variations in font size (a bigger font for more important topics worked better). Some results show that font size and font weight have stronger visual effects than intensity, number of characters, or tag area. However, when several visual properties are manipulated at once, there is no single property that stands out above the others according to [5]. Hearst and Rosner [6] even argues that "the limited research on the usefulness of tag clouds for understanding information and for other information processing tasks suggests that they are (unsurprisingly) inferior to a more standard alphabetical listing."

A dynamic tag cloud addresses the first two of the three shortcomings of lists to some extent. The importance of each

ISBN: 978-1-61208-242-4

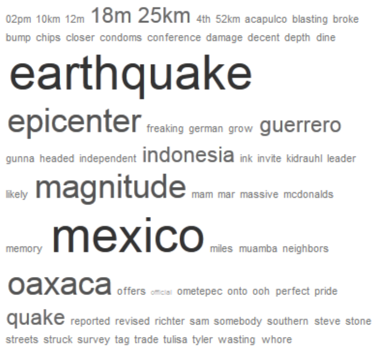


Figure 2. Twitscoop dynamic tag cloud.

topic is displayed by the font size in the tag cloud. The dynamics of the trend can be implemented by a dynamic tag cloud in which the text size grows or shrinks. However, the last shortcoming for addressing topics that are related to each other is more difficult. In this case, one needs to cluster trending topics into coherent groups and visualize them, e.g., through semantics [7], [8]. In order to visualize these clusters, one could use a Treemap [9] or a Squarified Treemap [10], [11]. A treemap displays hierarchical data as a set of nested rectangles.

In this paper, we propose a Dynamic Squarified Treemap (see Figures 6 and 7) to overcome all three aforementioned shortcomings. The importance of a topic can now be correlated to the size of a rectangle. The color of the rectangle can be used to identify if the topic is trending upwards, downwards or remains at its popularity. The rectangle itself can harbor multiple topics so that clusters can be visually represented in an appealing manner. In order to use this visual representation, we need to define how to choose the importance (which is directly related to the number of tweets per second on the topic) and how to choose the color (which is directly related to the acceleration or deceleration of the number of tweets per second).

Our contribution in this paper is threefold. First, we have a different perspective than most other works (e.g., as compared to [11], which is the only paper related to our work). We are focused on upcoming topics that will become a trend instead of a complete online overview of topics. The visualization of these topics is performed dynamically in which color, size, and animation carry additional information. Second, we develop algoritms to quickly determine the importance of topics using new smoothing methods based on little input data. Third, we show that for our purposes simple online clustering techniques perform sufficiently well.

The rest of the paper is structured as follows. In Section II, we outline the methodology to determine the input parameters for the dynamic squarified treemaps. In Section III, we explain how the dynamic aspect of squarified treemaps is more informative than other visualization methods. We conclude the paper with some additional remarks in Section IV.

II. Methodologies

In this section, we outline the methodology to determine the speed of tweets and the acceleration. These two parameters will serve as input parameters for the dynamic squarified treemap to generate a visualization of the trending topics. We first start with the twitter speed of a specific topic. For this purpose, we use the trending topics as posted by Twitter on 20 March 2012; see Figure 1. To illustrate our techniques, we focus on the tweets in hashtag #PrayforMexico. This hashtag was a trending topic at that time as a result of an earthquake in Mexico. The data derived from this hashtag consists of tweets with a time stamp (with seconds as accuracy). Based on this data, the absolute number of tweets over the day is given in Figure 3. One can see that around 7.30pm the number of tweets rapidly increases due to the earthquake.

A. *Speed of Tweets*

Let us for ease of notation focus on a stream of tweets on a particular subject for which the twitter speed needs to be determined. Let us denote by t_i the time stamp of the i-th tweet with $t_1 \leq t_2 \leq \cdots$. The speed can in principle be determined by a simple moving average, e.g., when tweet i arrives, the speed v_i can be determined by $k/(t_i - t_{i-k})$ for some k that determines how much history is included. There are two significant drawbacks to such a method. First, for high volume tweets (in particular, for popular topics), many tweets have the same time stamp. Thus, it could be that $t_i = \cdots = t_{i-k}$ so that v_i is not well-defined due to division by zero. Second, such an approach looks back at the history and has little predictive power.

To alleviate the drawback of the moving average, we first determine the interarrival times $a_i = t_i - t_{i-1}$. When tweet i is recorded, it could be that there are already several tweets that have the same time stamp (this is the case when $a_i = 0$). This number is given by $z_i = |\{k \,|\, t_i = t_k\}|$. Hence, we adjust the time stamp of the tweets by spreading them uniformly over the past second. Thus, we transform a_i to a'_i by

$$a'_i = \begin{cases} a_i - \left(1 - \frac{1}{z_i+1} - \frac{1}{z_{i-1}+1}\right), & a_i > 0, \\ \frac{1}{z_i+1}, & a_i = 0. \end{cases}$$

Next, we apply exponential smoothing with parameter $0 \leq \alpha \leq 1$ on the new interarrival times to derive a new time series b_i given by

$$b_i = \alpha b_{i-1} + (1-\alpha) a'_i,$$

starting with $b_1 = a'_1$. Since the resulting time series can still be too volatile, we apply a double smoothing by taking the average over the past k values of the time series b_i. Thus, the speed v_i (in tweets per second) is then given by

$$v_i = \frac{k}{\sum_{j=i-k+1}^{i} b_j}.$$

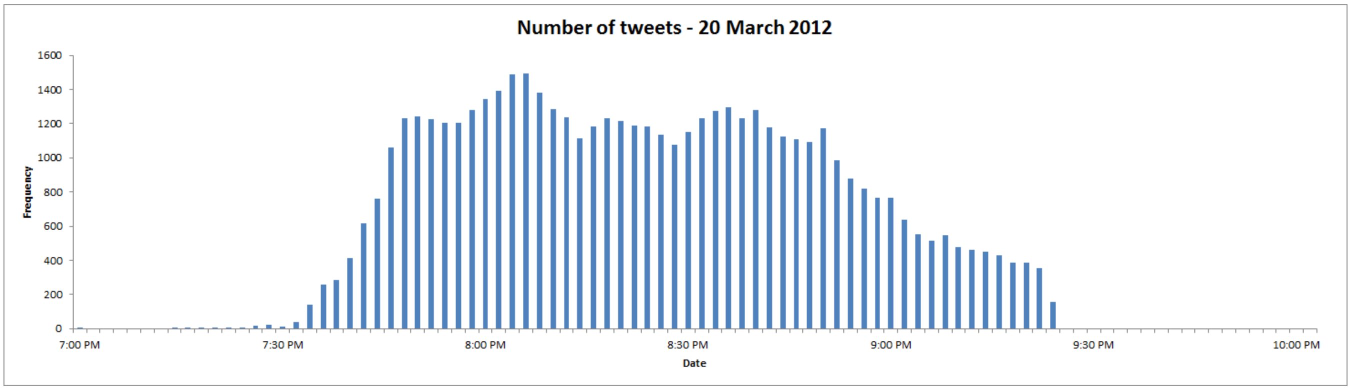


Figure 3. Absolute number of tweets for #PrayforMexico on 20 March 2012.

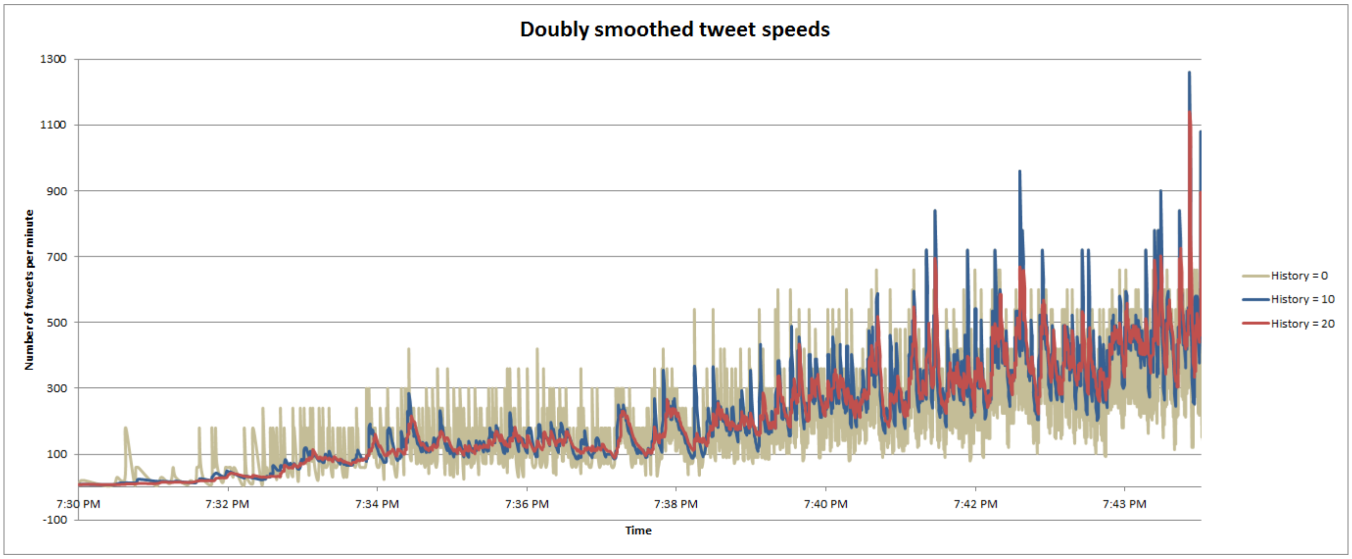


Figure 4. The number of tweets per second for different values of k (history) for #PrayforMexico on 20 March 2012.

Our algorithm thus has two parameters that can be chosen freely. We have the first smoothing parameter α that is used in the exponential smoothing, and we have the second smoothing parameter k that uses k tweets from history. In Figure 4, we can see the graph of the tweets from #PrayforMexico for various values of k. The parameter α is set to 0.8, which seems to work best for various examples in our setting. We can see that a value of $k = 0$, the case in which no history is taken into account, is rather volatile and does not produce stable results. The values of $k = 10$ and $k = 20$ provide more stable results and are much smoother than the graph for $k = 0$.

B. Acceleration of Tweets

The acceleration of tweets is basically a derivative of the speed of the tweets. We calculate the acceleration of the tweets for each minute. Let t be the start of a minute and $t{+}1$ the start of the next minute. Let $\overline{z}_t$ be the index of the last tweet before the end of the minute, thus $\overline{z}_t = \max\{i \,|\, t_i < t{+}1\}$. Denote by $\underline{z}_t$ the first tweet in that minute, or if there are not any, the one before that. Thus $\underline{z}_t = \max\{\min\{i \,|\, t \leq t_i < t_{\overline{z}_t}\}, \overline{z}_t - 1\}$. The acceleration w_t is then computed by

$$w_t = \frac{v_{\overline{z}_t} - v_{\underline{z}_t}}{t_{\overline{z}_t} - t_{\underline{z}_t}}.$$

Note that the definition closely reflects the regular definition of a derivative. However, we account for the fact that there can be no tweets in a particular minute. This is taken care by the way the variables $\overline{z}_t$ and $\underline{z}_t$ are defined. Furthermore, we also account for the fact that all tweets in the minute can have the same time stamp. Therefore, we use the adjusted timestamps a'_i instead of a_i.

In Figure 5, we can see the graph of the acceleration of the number of tweets per second for different values of k (the history that is used to determine v_i). As in the case of the calculation of the speed, we conclude that $k = 0$ (not using any history at all) results in volatile accelerations that are not preferred. Since the graph of speeds when using $k = 10$ is still very bursty, the acceleration shows large fluctuations that are not in accordance with ones intuition

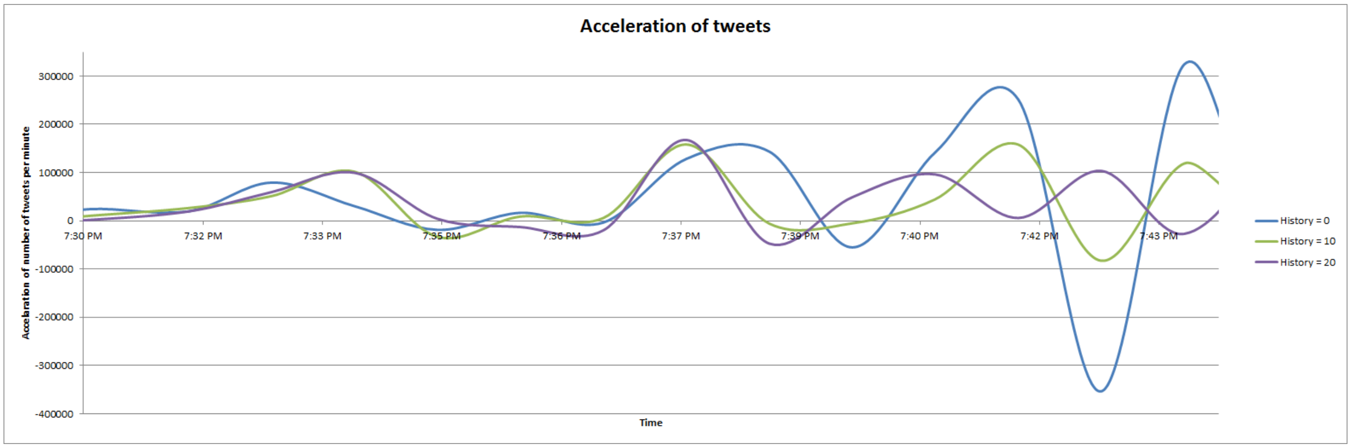
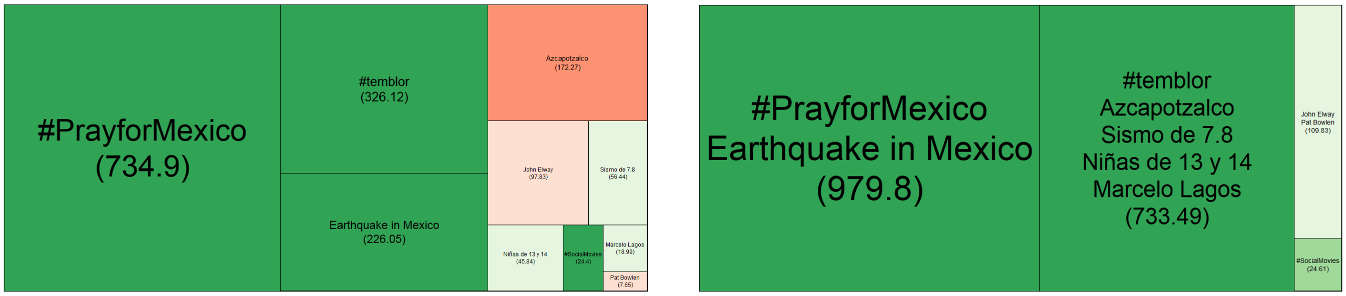


Figure 5. The acceleration of the number of tweets per second for different values of k (history) for #PrayforMexico on 20 March 2012.



(a) Unclustered

(b) Clustered

Figure 6. Numerical results.

(see, e.g., the timestamps around 7.42pm). The graph with $k = 20$, however, seems to perform well in this case, and in other cases as well. We can clearly see that the acceleration is picked up at 7.32pm, which corresponds to a real surge in the absolute number of tweets. Thus, this is precisely the moment at which one would like to detect this trend. Hence, in the rest of the paper, our algorithms run with $\alpha = 0.8$ and $k = 20$.

III. Dynamic Squarified Treemaps

In the previous section, we have identified the major ingredients for building a squarified treemap. First, we have determined the variable v_i, which represents the twitter speed in tweets per second on a particular topic. Second, we have identified the acceleration w_t of the number of tweets for the same topic. Based on this information, we build rectangles for each topic of which the relative areas correspond to the relative speeds of each topic. On top of that, each rectangle is color-coded from green to white to red, based on a positive to neutral to negative acceleration. This gives rise to a representation as depicted in Figure 6a. The numbers in parentheses represent v_i. This representation solves many of the issues tied to lists (see Figure 1) and tag clouds (see Figure 2). In this section, we improve the visual representation by clustering related topics.

A. *Clustering topics*

The clustering of tweets is not an easy process. Standard algorithms, such as K-means clustering [12], are slow. Therefore, most algorithms usually work iteratively. For speed, a single assignment is usually used in the literature (e.g., [13], [14]).

A simple way to cluster tweets is by using a cosine similarity as defined in [15]. In this algorithm, the term frequency and inverse document frequency (TF-IDF) [16] can be used as a weighing scheme. A more involved method to cluster tweets is the Latent Dirichlet Allocation (LDA) [17], which can be used to track topics over time [18]. The clustering that is obtained by this method is better than when using TF-IDF [19] (while a combination works best). However, LDA is not perfect for Twitter because tweets are limited in size [20]. Methods based on non-negative Matrix Factorization [21] could be an alternative to TF-IDF and LDA (from [22]). Some experimentation has already been performed in [23] on a small dataset. One can also think of mixture models [24], [25], which were developed for producing recommendations, for clustering tweets.

B. *Clustering based on tweet list comparison*

As a first clustering algorithm, we adopt a very simple but efficient clustering algorithm. For each topic a, at time

Table I
CLUSTERING BASED ON COMPARISON OF TWEET LISTS.

	#PrayforMexico	#temblor	Earthquake in Mexico	Azcapotzalco	John Elway	Sismo de 7.8	Niñas de 13 y 14	#SocialMovies	Marcelo Lagos	Pat Bowlen
#PrayforMexico	1	0.01	0.18	0.01	0.01	0.04	0	0.01	0.01	0
#temblor	0.01	1	0.02	0.02	0.01	0.01	0	0.01	0.01	0
Earthquake in Mexico	0.18	0.02	1	0.01	0.01	0.03	0	0.01	0.01	0
Azcapotzalco	0.01	0.02	0.01	1	0.01	0.01	0	0.01	0.01	0
John Elway	0.01	0.01	0.01	0.01	1	0.01	0	0.01	0.01	0.04
Sismo de 7.8	0.04	0.01	0.03	0.01	0.01	1	0	0.01	0.01	0
Niñas de 13 y 14	0	0	0	0	0	0	1	0	0	0
#SocialMovies	0.01	0.01	0.01	0.01	0.01	0.01	0	1	0.01	0
Marcelo Lagos	0.01	0.01	0.01	0.01	0.01	0.01	0	0.01	1	0
Pat Bowlen	0	0	0	0	0.04	0	0	0	0	1

Table II
CLUSTERING BASED ON THE COSINE SIMILARITY INDEX.

	#PrayforMexico	#temblor	Earthquake in Mexico	Azcapotzalco	John Elway	Sismo de 7.8	Niñas de 13 y 14	#SocialMovies	Marcelo Lagos	Pat Bowlen
#PrayforMexico	1	0.30	0.70	0.21	0.14	0.27	0.19	0.16	0.21	0.13
#temblor	0.30	1	0.20	0.47	0.11	0.54	0.41	0.12	0.42	0.10
Earthquake in Mexico	0.70	0.20	1	0.15	0.16	0.17	0.11	0.16	0.14	0.15
Azcapotzalco	0.21	0.47	0.15	1	0.08	0.48	0.33	0.09	0.33	0.08
John Elway	0.14	0.11	0.16	0.08	1	0.06	0.06	0.11	0.08	0.32
Sismo de 7.8	0.27	0.54	0.17	0.48	0.06	1	0.46	0.08	0.38	0.06
Niñas de 13 y 14	0.19	0.41	0.11	0.33	0.06	0.46	1	0.06	0.33	0.06
#SocialMovies	0.16	0.12	0.16	0.09	0.11	0.08	0.06	1	0.10	0.11
Marcelo Lagos	0.21	0.42	0.14	0.33	0.08	0.38	0.33	0.10	1	0.07
Pat Bowlen	0.13	0.10	0.15	0.08	0.32	0.06	0.06	0.11	0.07	1

t, we keep a list l_a of the last 100 tweets counting back from time t. Our similarity metric for topic a and topic b is defined as the number of times that both terms a and b appear in the lists l_a and l_b. If the similarity metric is above the threshold of 0.15, then the two topics are clustered, and clustering continues until no more tokens can be added to the cluster. Table I displays the results of this clustering. In the results we can see that '#PrayforMexico' and 'Earthquake in Mexico' are clustered.

C. Clustering based on the cosine similarity index

We also adopt the cosine similarity [26] to cluster the tweets. The cosine similarity of two topics a and b is a measure of similarity, defined by

$$\frac{< f_a, f_b >}{||f_a|| \cdot ||f_b||} = \frac{\sum_i f_a(i) f_b(i)}{\sqrt{\sum_i f_a(i)^2}\sqrt{\sum_i f_b(i)^2}},$$

where the vector f_a (and f_b) is the frequency list of terms that appear in the list l_a (and l_b). The cosine similarity is bounded between 0 and 1 since both f_a and f_b are non-negative. The name of the similarity index is derived from the interpretation of the cosine of the angle between the two vectors. Hence, similar vectors (with an angle close to zero) have a high cosine similarity, whereas vectors that are not similar (with an angle close to $\pi/2$) have a low cosine similarity. If the similarity metric is above the threshold of 0.30, then the two topics are clustered. Table II displays the results of this clustering. In the results we can see that '#PrayforMexico' and 'Earthquake in Mexico' are in one cluster. In addition, '#temblor', 'Azcapotzalco', 'Sismo de 7.8', 'Niñas de 13 y 14', and 'Marcelo Lagos' form one cluster, as well as 'John Elway' and 'Pat Bowlen'. Observe that the two largest clusters are actually about the same subject, but in two different languages. A human observer would either put these into one cluster or into two. In fact, our clustering algorithm almost puts these into one cluster, with a cosine similarity of 0.30.

Figure 6b shows the squarified treemap for the clustered topics. It is clear that this representation is even better than Figure 6a. From the clusters is becomes clear that 'Azcapotzalco' is related to the earthquake in Mexico, although this was not clear before. Figure 7 depicts the dynamic part of the squarified treemaps. Using jQuery [27], the tiles in the treemap transition to their new size and position based on the newly calculated speed and acceleration values. This dynamic part has the appealing feature that one can directly identify visually the emerging and receding topic. The dynamic clustered squarified treemap resolves the three issues that were mentioned as problems with lists and tag clouds. Experiments with test persons seem to suggest that the dynamic squarified treemap is an effective method for the display of dynamic data from Twitter.

IV. CONCLUSION AND FUTURE WORK

In this paper, we have discussed the dynamic squarified treemap for visually representing the trending topics on Twitter. The main ingredients for this graph are the speed of tweets and the acceleration of them. We have developed algorithms to calculate both of them. Moreover, we have discussed a simple clustering algorithm to deal with grouping related topics in online twitter streams. The final representation in a dynamic squarified treemap fills the gaps

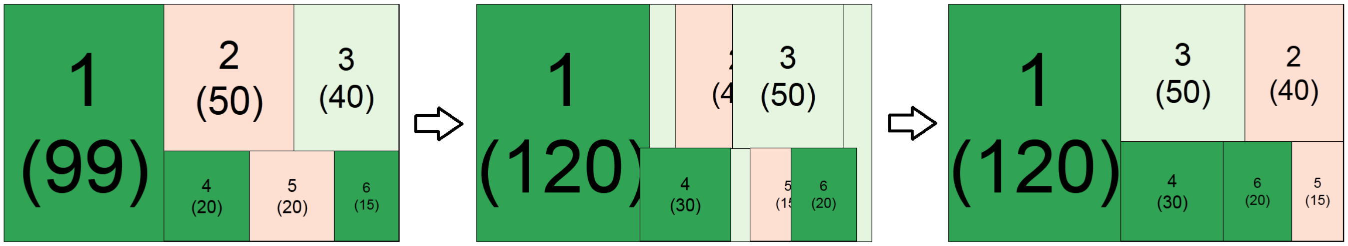


Figure 7. The transitions in the dynamic squarified treemap.

that are present in list and tag cloud representations. Hence, the dynamic squarified treemap forms a powerful visual tool to visualize trending topics.

The analysis in this paper has been done on the trending topics based on the list provided by Twitter. However, we are currently working on a system in which we monitor a sample of the twitter stream and detect trending topics ourselves. The system calculates the speed and acceleration every second and updates the screen accordingly. Based on the size and rate of growth of a cluster of words / topics the dynamic squarified treemap serves as an early warning system for trends.

REFERENCES

[1] Wikipedia, "Twitter," URL: en.wikipedia.org/wiki/Twitter.

[2] Twitter, URL: www.twitter.com.

[3] A. W. Rivadeneira, D. M. Gruen, M. J. Muller, and D. R. Millen, "Getting our head in the clouds: toward evaluation studies of tagclouds," in *Proc. of the SIGCHI Conf. on Human factors in computing systems*, 2007, pp. 995–998.

[4] M. Halvey and M. Keane, "An assessment of tag presentation techniques," in *Proc. of the 16th Intl. Conf. on World Wide Web*. ACM, 2007, pp. 1313–1314.

[5] S. Bateman, C. Gutwin, and M. Nacenta, "Seeing things in the clouds: the effect of visual features on tag cloud selections," in *Proc. of the 19th ACM Conf. on Hypertext and hypermedia*, New York, NY, USA, 2008, pp. 193–202.

[6] M. Hearst and D. Rosner, "Tag clouds: Data analysis tool or social signaller?" in *Hawaii Intl. Conf. on System Sciences, Proc. of the 41st Annual*. IEEE, 2008, pp. 160–160.

[7] L. Di Caro, K. S. Candan, and M. L. Sapino, "Using tagflake for condensing navigable tag hierarchies from tag clouds," in *Proc. of the 14th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*. New York, NY, USA: ACM, 2008, pp. 1069–1072.

[8] C. Cattuto, D. Benz, A. Hotho, and G. Stumme, "Semantic grounding of tag relatedness in social bookmarking systems," in *Proc. of the 7th Intl. Conf. on The Semantic Web*, ser. ISWC '08. Berlin: Springer-Verlag, 2008, pp. 615–631.

[9] B. Shneiderman and M. Wattenberg, "Ordered treemap layouts," in *Proc. of the IEEE Symp. on Information Visualization 2001 (INFOVIS'01)*, Washington, DC, USA, 2001, pp. 73–.

[10] M. Bruls, K. Huizing, and J. Van Wijk, "Squarified treemaps," in *Proc. of the Joint Eurographics and IEEE TCVG Symp. on Visualization*. Citeseer, 2000, pp. 33–42.

[11] D. Archambault, D. Greene, P. Cunningham, and N. Hurley, "Themecrowds: multiresolution summaries of twitter usage," in *Proc. of the 3rd Intl. Workshop on Search and mining user-generated contents*. ACM, 2011, pp. 77–84.

[12] A. Karandikar, "Clustering short status messages: A topic model based approach," Master's thesis, Faculty of the Graduate School of the University of Maryland, 2010.

[13] H. Becker, M. Naaman, and L. Gravano, "Beyond trending topics: Real-world event identification on twitter," in *Proc. of the 5th Intl. AAAI Conf. on Weblogs and Social Media*, 2011.

[14] J. Sankaranarayanan, H. Samet, B. E. Teitler, M. D. Lieberman, and J. Sperling, "Twitterstand: news in tweets," in *Proc. of the 17th ACM SIGSPATIAL Intl. Conf. on Advances in Geographic Information Systems*, 2009, pp. 42–51.

[15] G. Kumaran and J. Allan, "Text classification and named entities for new event detection," in *Proc. of the 27th Annual Intl. ACM SIGIR Conf. on Research and development in information retrieval*, 2004, pp. 297–304.

[16] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Inf. Process. Mngmnt*, vol. 24, pp. 513–523, August 1988.

[17] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, 2003.

[18] D. Knights, M. C. Mozer, and N. Nicolov, "Detecting topic drift with compound topic models," in *Proc. of the Fourth Intl. AAAI Conf. on Weblogs and Social Media*, 2009.

[19] D. Ramage, S. Dumais, and D. Liebling, "Characterizing microblogs with topic models," in *Proc. of the Fourth Intl. AAAI Conf. on Weblogs and Social Media*, 2010.

[20] K. D. Rosa, R. Shah, B. Lin, A. Gershman, and R. Frederking, "Topical clustering of tweets," in *Proc. of the ACM SIGIR 3rd Workshop on Social Web Search and Mining*, 2011.

[21] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, Oct 1999.

[22] S. P. Kasiviswanathan, P. Melville, A. Banerjee, and V. Sindhwani, "Emerging topic detection using dictionary learning," in *20th ACM Conf. on Info. and Knowledge Mngmnt*, 2011.

[23] A. Saha and V. Sindhwani, "Learning evolving and emerging topics in social media: a dynamic NMF approach with temporal regularization," in *Proc. of the Fifth ACM Intl. Conf. on Web search and data mining*, 2012, pp. 693–702.

[24] J. Kleinberg and M. Sandler, "Using mixture models for collaborative filtering," in *Proc. of the Thirty-Sixth Annual ACM Symp. on Theory of computing*, 2004, pp. 569–578.

[25] S. Morinaga and K. Yamanishi, "Tracking dynamics of topic trends using a finite mixture model," in *Proc. of the Tenth ACM SIGKDD Intl. Conf. on Knowledge discovery and data mining*, New York, NY, USA, 2004, pp. 811–816.

[26] P.-N. Tan, M. Steinbach, and V. Kumar. "Introduction to Data Mining," Addison-Wesley, 2005, Chapter 8.

[27] jQuery, URL: jquery.com.