ISSN 2395-8618

Hybrid Entity Driven News Detection on Twitter

Linn Vikre, Henning M. Wold, Özlem Özgöbek, and Jon Atle Gulla

***Abstract*—In recent years, Twitter has become one of the most popular microblogging services on the Internet. People sharing their thoughts and feelings as well as the events happening around them, makes Twitter a promising source of the most recent news received directly from the observers. But detecting the newsworthy tweets is a challenging task. In this paper we propose a new hybrid method for detecting real-time news on Twitter using locality-sensitive hashing (LSH) and named-entity recognition (NER). The method is tested on 72,000 tweets from the San Fransisco area and yields a precision of 0.917.**

I. Introduction

Twitter[1] is a popular social media platform where users can share their opinions and publish facts about everything from news to more personal matters. For example in an emergency situation, people can provide information about the situation as observers, or give relevant knowledge about the situation obtained by other sources and then share it through Twitter [1]. On the one hand, this gives Twitter great potential in being able to discover breaking news and follow all angles surrounding a novelty. On the other hand, it is easy to spread a rumor or unreliable facts using Twitter.

Focusing on news, there exists a plethora of different ways to convey the news depending on your location, your political standing, and so on. Therefore there are several factors that may influence how a tweet is formulated, and how it is interpreted. As Twitter is a fast growing social media platform with 302 million monthly active users and around 500 million tweets sent per day[2], it seems possible to discover information which is not yet published news around the world. Previous research [2], [3], [4] shows that it is possible to discover news on Twitter. It has, however, proven to be difficult to detect these news at early stages, namely before they are put on the newswire. The difficulty lies in the shortness of text allowed in tweets. A user on Twitter can only express themselves using 140 characters, which provokes the use of abbreviations, incorrect sentence structure, and language.

This paper explores how data pre-processing, named-entity recognition and locality-sensitive hashing (LSH) can be used to achieve better results when detecting breaking news. Our system is based on clustering tweets with similar content on the work of [5]. Their approach enables us to detect the topics that are being discussed on Twitter in a given period of time without using any Twitter-specific services, such as trending Topics[3]. Our base assumption is that if an event occurs that can be considered as a breaking news, that will lead to a "burst of activity" of people tweeting about it as described for general document streams in [6]. This will lead to a new, high activity, cluster appearing that contains tweets about the event. Keeping this in mind, we explore how the various parameters in the LSH implementation can be changed to increase the efficacy of our approach.

Central to this work is the ability to identify breaking news from a live feed of real-time tweets. The main motivation for this is that finding breaking news is of little value if they are detected too long after the news event in question happened, as this makes the likelihood of it already having been picked up by the newswire all but certain.

In this paper, we address the following issues:

- Evaluate to what extent LSH clustering can be applied for live news detection on Twitter.
- Cluster tweets based on their contents using our chosen locality-sensitive hashing (LSH) algorithm implementation, and see how it performs on random tweets.
- Evaluate if named-entity recognition (NER) can be used as a filtering method to detect news.

The structure of this paper is as follows. In Section II we present the previous research in the field. Section III describes the different methods used to conduct the evaluations. In Section IV we explain the details of our experiments like the data sets and tools we used. In Section V we explain the results of our experiments and finally in Section VI we discuss our results and give the conclusions.

II. Related work

In [7] it is stated that using traditional NER systems on tweets were insufficient for news detection. Thus in this work it is developed and trained a classifier based on annotated tweets to recognize named entities. The classifier trained handles *Location*, *Organization* and *Person* classes. Later in [8] it is performed a research using the same three classes, in addition to a *Product* class. In this approach it is used k-nearest neighbors as a supervised method.

Chiticariu et. al. [9] presents the language NERL (NER rule language) which is a high-level language to customize,

Manuscript received on August 21, 2016, accepted for publication on December 18, 2016, published on June 20, 2017.

The authors are with the Department of Computer and Information Science, NTNU, Trondheim, Norway (e-mail: {vikre, henninwo}@stud.ntnu.no, {ozlemo, jag}@idi.ntnu.no).

[1]https://twitter.com/

[2]https://about.twitter.com/company

[3]https://twitter.com/trendingtopics

ISSN 2395-8618

understand, and build rule-based named entity annotators across different domains. This work does not particularly focuses on Twitter but it is important to understand the difficulties of named-entity recognition.

Li et. al. [10] presents a novel 2-step unsupervised method for automatic discovery of emerging named entities, which could potentially be linked to news events such as crises. Their system, called TwiNER, leverages global context that are obtained from Wikipedia [4] and the Web N-Gram corpus, and ranks segments that are potentially true named entities. The system currently does not categorize the detected named entities. However, the approach only works with targeted Twitter streams, which means it is not usable with our approach as the tweets come from a variety of users.

As Ritter et. al. [11] addresses the issue of the noisy nature and incorrectness of language in tweets by re-building the NLP-pipeline consists of part-of-speech tagging (POS-tagging), chunking, and finally named-entity recognition. Their study shows an increase in accuracy and obtained a large (41%) reduction in error compared with the Stanford POS tagger. In their work it is also shown that the features generated by POS tagging and chunking gives a benefit to the segmenting of named entities. In our work, we utilize the system they created, though only the part that classifies named entities. We use this only as an additional filtering step in our system to detect breaking news.

Petrovic et. al. [5] presents a locality sensitive hashing (LSH) algorithm for "First story detection" on Twitter. According to this work, LSH is able to overcome some of the limitations of the traditional approaches, e.g. centroids and vectors in term space weighted with idf (inverse document frequency). Their method drastically reduces the number of comparisons a tweet needs to have in order to find the N nearest neighbors. This is crucial in a system that should work as a real-time service. Their work also shows that their system can find news events after analyzing an entire corpus. We utilize their approach in our system also, but we focus more on the activity in tweet clusters in smaller time periods, to detect if something out of the ordinary is happening. By doing this, we are able to detect news events in real-time.

Agarwal [12] presents an approach that takes a continuous stream of Twitter data, processes them to filter out tweets characterized as "noise" and get the informative ones. After this, they use the filtered tweets to detect and predict trending topics at an early stage. We use some of their findings when pre-process to disqualify tweets unlikely to be about a news event.

III. Methodology

In this section, we introduce the different methods we used to conduct the experiments where the details are explained in the next section.

[4]http://wikipedia.com

A. Locality sensitive hashing

Locality sensitive hashing (LSH) [13] is a technique for finding the nearest neighbor document in vector space utilizing vectors of random values and representing hyperplanes to generate hashes. This approach reduces the time and space complexity when finding the nearest neighbor.

LSH hashes the input items using k hyperplanes. Increasing the value of k decreases the probability of collision between non-similar items, while at the same time decreasing the probability of collision between nearest neighbors. To alleviate this, the implementation contains L different hash tables, each with independently chosen random hyperplanes. These hash tables are also called "buckets". This increases the probability of the item colliding with its nearest neighbor in at least one of the L hash tables. In other words there is a high probability that the item's nearest neighbor is contained in one of the buckets the item gets assigned to. LSH thus seek to achieve a maximum probability for collision on similar items. To be able to perform LSH on tweets, each tweet is converted to vector space, using tf-idf combined with Euclidean normalization.

There are several parameters that can be adjusted to change the results of LSH. In this paper we seek to find optimal values for these parameters with respect to detecting news, and filtering out as many of the non-relevant tweets as possible.

Our approach for LSH is based on the one described by [5]. We use the cosine similarity between document vectors $\vec{u}$ and $\vec{v}$ to decide the nearest neighbor:

$$cos(\theta) = \frac{\vec{u} \cdot \vec{v}}{||\vec{u}|| \cdot ||\vec{v}||}$$

Specifics of the implementation is elaborated on in Section IV-B.

B. Entropy

After a the tweets are clustered, it is possible to measure its Shannon entropy [14], which also is called information entropy. This is done by concatenating the text of all the tweets in a cluster into a single document. By doing this, the Shannon entropy is given by

$$H(X) = -\sum_i P(x_i) \log_2 P(x_i)$$

where $P(X_i) = \frac{n_i}{N}$ is a tokens's probability of occurring in the document (found by dividing the number of times the token, n_i, appears in the document with how many tokens are in the document, N, in total).

Shannon entropy is a measure of how much information is contained within a document. Petrovic et. al. [5] suggests that by ignoring clusters with low entropy, it is possible to filter out many clusters that are filled with spam as they tend to have very low entropy.

Tweets are limited to 140 characters, and as Shannon entropy effectively use the length of the text as a factor (in N), longer tweets will be given a higher entropy than shorter

ISSN 2395-8618

tweets by default. For example, if a cluster contains five completely equal tweets of 50 characters each, and another cluster contains five completely equal tweets of 100 characters each, the latter will always have a higher entropy. For this reason, it would be unwise to set the entropy threshold too high (we could miss interesting stories that are simply too short), neither would setting it too low be wise (many stories containing spam and chatter would not be filtered out). The parameter governing the lowest entropy allowed in a cluster is *MIN_ENTROPY*.

C. Named-entity recognition (NER)

To detect breaking news, in addition to LSH, we examined if named-entity recognition can improve the results that we get by using LSH to filter tweets.

Named-entity recognition (NER) is a type of natural language processing (NLP). NLP refers to research that explore how computers can be utilized to understand, interpret, and manipulate natural languages, such as text and speech [15]. NER is a powerful tool for analyzing the deeper meaning behind sentences or longer sequences of words [8]. Thus NER is commonly understood as the task of identifying so-called named-entities in a text, such as organizations, products, locations, and persons.

A few different methods exist for performing NER on text documents: The rule-based method, the use of machine learning, and a hybrid between the two. The rule-based method has two extensive drawbacks; it lacks both portability and robustness. For this reason, machine learning has emerged as the better choice for coping with the problem [16], and is the method we have decided to utilize for our research.

D. Online processing of data

Since we are interested in detecting breaking news, we have to process the real-time data in a quick way. So we have to set up our system in such a way that it is quick enough to get fed data from the Twitter streaming API. We examine the clusters in set windows of time that has a length of *WINDOW_TIME_IN_SECONDS* seconds. We take the difference between the timestamp of the tweet currently being processed and the last tweet processed before the previous output. If this difference is greater than the duration of our window of time, we output information about the clusters matching our set parameters. As keeping every tweet previously processed in memory would quickly exhaust available resources, we only keep the message of the original tweet for a cluster in memory, in addition to the tweets added to it during the current window. A downside of this is that it also limits the amount of information available to decide whether a given cluster is news relevant or not.

IV. Experiment

In this section, we present the details of the experiment we have devised.

A. Data sets

Our data set consist of 72,000 tweets collected from the San Fransisco area using the Twitter streaming API[5]. The data was collected over a period of 30 hours from May 11 to May 12, 2015. The API provides extensive metadata for the tweets, most of which are not of interest to us in this experiment. We thus, to conserve storage space, strip most of it away, only keeping the tweet text itself, the tweet id, the user id of the poster, and the timestamp the tweet was posted. As previously mentioned, our system needs to work for tweets arriving in real-time. Despite this, we still chose to have a static data set for the experiments so that the same data would be used in each test.

B. Tools

Before hashing, the tweets were run through the NER engine. This put an extra field into the underlying JSON structure of the tweets where entities were detected called "entities". This field contained a list of all the entities detected in the tweets.

To perform the experiment, we require an implementation of the LSH algorithm. We have based our implementation on the ones outlined in [5] and [17]. Our implementation consists of two modules. The first module is the main LSH module and takes in a stream of tweets and outputs a stream of tweets augmented with information about their nearest neighbor (the id of the nearest neighbor, and its cosine similarity to that neighbor). The second module takes in a stream of tweets augmented with information about their nearest neighbor and outputs clusters of related tweets based on parameters discussed below.

We use the following static parameters in our LSH implementation:

- 13 bit hash values (k)
- 36 hash tables (L)
- 20 collisions per hash value
- 2000 previous tweets comparison

These values remain unchanged between experiments. In addition to these, we have parameters we adjust between experiments to find ideal values. These parameters, along with their initial values, can be found in table I.

The first step is tokenizing the tweets by splitting the text into tokens while removing punctuation. Additionally, tokens identified URLs, mentions or hashtags are not included in the list of tokens. If the tweet contains any non-ASCII tokens, the entire tweet is discarded. The entire tweet is also discarded if any of its tokens are included in the *IGNORE* list, or if the number of tokens left after tokenizing is $<$ *MIN_TOKENS*. After tokenizing, the nearest neighbor and the cosine similarity to this neighbor is found. Once this is done, the results are output and used in the next module responsible for creating clusters of related stories.

[5]http://dev.twitter.com

ISSN 2395-8618

The next module transforms the stream output from the LSH module into clusters of tweets based on a few parameters. If the cosine similarity between a tweet and its nearest neighbor is at least *MIN_COSSIM_FOR_EXISTING*, it is added to the same cluster as its nearest neighbor. If not, a new cluster is created with the tweet as its first message. After *WINDOW_SIZE_IN_SECONDS* seconds have passed between the timestamp of the first tweet processed and the current tweet being processed, all stories matching certain parameters will be printed. A cluster must have had at least *MIN_TWEETS_IN_SLICE* tweets added to it during the current window to be printed. Additionally at least *MIN_UNIQUES* unique users must have had their tweets added to that cluster for it to be printed. The last parameter to be matched for a cluster to be printed is that its information entropy must be at least *MIN_ENTROPY*.

In addition to finding optimal LSH parameters for detecting breaking news on Twitter, we want to examine if utilizing named-entity recognition can further improve the results. To this end we have used the work of [11]. They achieved better results than the baseline presented in their work to customize the named-entity recognition engine to fit the nature of tweets. We have thus elected to use the implementation the authors created, which they have published open source on the Internet[6].

C. Conducting the experiment

When conducting the experiment, we first need to establish a baseline. For this baseline, we gave the parameters the values listed in table I and calculated the precision achieved using those values. We define precision here as the proportion of the returned clusters deemed as news. The values of *MIN_COSSIM_FOR_ EXISTING* and *MIN_ENTROPY* in the baseline were chosen based on the findings of [5], while the values of *MIN_TOKENS* and *IGNORE* were chosen based on the findings of [12].

The *MIN_TWEETS_IN_SLICE* parameter is initially set to 1 (that is any non-empty cluster may get through the clustering algorithm). The reasoning behind this is that we suspect the value to have an impact on the results that we want to test, but setting it to 1 for the baseline effectively switches it off as to not pollute the results.

Another parameter of interest to the baseline is *MIN_UNIQUES*, which dictates how many unique users must have tweets in a cluster for it be to qualified. If this value is set lower than 2, it opens the proverbial floodgates for various single-user spam accounts, irrelevant or out-of-context tweets and other similar phenomena. We thus elected to set this value to 2 in the baseline and only examine values higher than this. As a news event is highly unlikely to be of interest if only one person is writing about it, with no one retweeting them, we consider this a fair choice to make.

[6]https://github.com/aritter/twitter_nlp

The final tested parameter is *WINDOW_SIZE_IN_SECONDS*, which dictates the length of time between clusters being printed. We were quite unsure of what would be the ideal values for this parameter. A short windows would enable the system to more rapidly detect news events. On the other hand, it is possible that more time has to elapse for the news events to become properly identifiable in the clusters. We thus defined the values to test to be from 5 minutes (300 seconds) to 25 minutes (1500 seconds). We set the baseline to be the middle of the range; 15 minutes (900 seconds).

Though the precision values calculated are valuable, another interesting facet is how the number of relevant clusters and non-relevant clusters change between the tests. We want to maximize the number of relevant clusters while minimizing the number of not relevant tweets.

After recording the number of relevant and not relevant clusters in the baseline, we proceeded to change the parameters one at a time, leaving the others at their baseline value. This enabled us to see how changing a parameter changes results. An overview over the various tests we performed, as well as their results, can be found in table II.

Next we used the information from the NER engine to further filter the clusters. This was done by simply dropping those not contain the "entities" field from the output. We did not perform this additional step on all tests; the ones we did perform it for are marked with NER in table II.

Finally, we combined a few of the most promising candidates for news detection in a test. This test is marked FINAL in table II.

V. Results

Our results, which can fully be found in Table II, show that the parameters we picked out for our baseline had a fairly average precision of 0.472. By combining various values for the different parameters, in addition to including named-entity recognition, we were able to increase this precision to 0.917.

From the results we can see that there were three things in particular that largely affected the result: the minimum amount of entropy allowed in a cluster (*MIN_ENTROPY* - ME), whether or not named entities were used to filter the cluster (NER), and the minimum cosine similarity between two tweets for them to be put in the same bucket (*MIN_COSSIM_FOR_EXISTING* - MCFE). We had assumed in advance that using NER to filter the clusters would have a positive impact. That this turns out to be the case is thus not surprising. The MCFE parameter is still within the range suggested for it in [5] for the optimal values we found. The experiments show that when MCFE is set lower than its suggested values, the precision value plummets. We did not test having this value set higher than their highest suggested value. The reasoning being that by increasing it, you increase the number of clusters and reduce the number of tweets landing in the same clusters. Meanwhile is the ME parameter set higher than the suggested value of the paper. This seems

ISSN 2395-8618

TABLE I
BASELINE PARAMETERS FOR FILTERING TWEETS AND THE LSH CLUSTERING

Baseline	
Parameter	**Value**
MIN_TOKENS (MT)	2
IGNORE	[“i”, “im”, “me”, “mine”, “you”, “yours”]
MIN_TWEETS_IN_SLICE (MTIS)	1
MIN_COSSIM_FOR_EXISTING (MCFE)	0.5
MIN_ENTROPY (ME)	3.5
MIN_UNIQUES (MU)	2
WINDOW_SIZE_IN_SECONDS (WSIS)	900s

to indicate that news tweets have higher entropy than ordinary tweets, which seem reasonable. Even though the value is set higher, there was not a huge difference in the paper's results between their suggested value and the value we found to give the highest precision for detecting news tweets.

Another parameter for which previous research seemingly has found an optimal value is the one for excluding tweets containing less than a certain number of tokens (*MIN_TOKENS* - MT). Both increasing and decreasing this value led to a slight decrease in precision.

One parameter we thought ahead of time would be important in detecting news is the *MIN_TWEETS_IN_SLICE* (MTIS) parameter. This is the number of tweets assigned to a cluster during the current time window. Although it seemingly has little impact on the detection of breaking news, this could change if tested on a data set that is recorded when some catastrophe or similar crisis happens. This is because the parameters work as sort of a dial; the higher it is set, the more tweets must be generated in a cluster during a window for it to be detected, which indicates the importance of the event the cluster describes.

The parameter for excluding clusters with less than a certain amount of unique users posting to it (*MIN_UNIQUES* - MU) sees little change in precision by increasing it by one from the baseline. Although precision increases slightly, by looking at the raw number one can see that the number of relevant clusters and the number of not relevant clusters decrease almost uniformly. Keeping it low, but still above 1 to filter out an ocean of clusters containing only singular messages, seem like the best course of action.

Finally, we have the parameter governing the size of the time window we examine (*WINDOW_SIZE_IN_SECONDS* - *WSIS*). This parameter, like MTIS, did not greatly affect the results. Still it demonstrated that unlike what we had assumed in advance, having a shorter time window did not in general impact the precision negatively. As we want to detect breaking news as soon as possible we want this value to be as low as possible.

The test labeled *FINAL* in Table II is thus a test where we have combined the optimal values for ME, MCFE, WSIS in addition to filtering the clusters using NER.

VI. DISCUSSION AND CONCLUSION

In this paper, we have looked at how one can use an algorithm intended for clustering documents (locality-sensitive hashing) to filter tweets. We have shown that, by tweaking the parameters of LSH and including a named-entity recognition engine in the pipeline, it is possible to achieve a high precision value for news events in tweets.

The LSH implementation itself performs well, and its memory footprint only grows linearly with the size of the vocabulary of the input. With the rate of the free Twitter streaming API, the implementation processes the tweets quicker than they arrive, meaning the implementation is usable in a real-time system.

Many of the relevant tweet clusters found by our algorithm were tweets and retweets from the U.S. Geological Survey about small earthquakes occurring in the San Francisco area (where our tweets were gathered from). These, to us, are indeed interesting events, however they would only likely be picked up in areas with seismic activity. Thus, our findings should be tested on data sets gathered from other geographic areas.

News detection from Twitter is important in the SmartMedia project[18], in which semantic technologies and various recommendation strategies are combined into a mobile news aggregator[19]. When a new event is detected from Twitter, the relevant tweets are fed into the news index and subjected to the same personalization approach as traditional news stories[20]. Whether these tweets provide new insight or faster insight into breaking news will be tested in future experiments.

ACKNOWLEDGEMENTS

This work is a part of the SmartMedia[7] Program in NTNU, Norway. The author Özlem Özgöbek is supported by the ERCIM Alain Bensoussan Fellowship Programme.

REFERENCES

[1] S. Vieweg, A. L. Hughes, K. Starbird, and L. Palen, “Microblogging during two natural hazards events: What twitter may contribute to situational awareness,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI ’10. New York, NY, USA: ACM, 2010, pp. 1079–1088. [Online]. Available: http://doi.acm.org/10.1145/1753326.1753486

[7]http://research.idi.ntnu.no/SmartMedia

ISSN 2395-8618

TABLE II
TEST WITH PRECISION MEASURE FOR THE DIFFERENT TESTS WHERE FINAL IS THE BEST LSH TUNING COMBINED WITH NER.

Test	Parameters	Relevant	Not Relevant	Precision
BASELINE	All at standard (see table I)	199	133	0.472
BASELINE + NER	All at standard	106	49	0.683
MIN_TOKENS #1	MT = 1	117	166	0.413
MIN_TOKENS #2	MT = 3	118	143	0.452
MIN_TWEETS_IN_SLICE #1	MTIS = 2	79	55	0.589
MIN_TWEETS_IN_SLICE #2	MTIS = 3	32	25	0.561
MIN_TWEETS_IN_SLICE #3	MTIS = 4	13	17	0.433
MIN_TWEETS_IN_SLICE #4	MTIS = 5	4	9	0.307
MIN_COSSIM_FOR_EXISTING #1	MCFE = 0.4	137	591	0.188
MIN_COSSIM_FOR_EXISTING #1 + NER	MCFE = 0.4	118	123	0.489
MIN_COSSIM_FOR_EXISTING #2	MCFE = 0.6	86	45	0.656
MIN_COSSIM_FOR_EXISTING #2 + NER	MCFE = 0.6	80	29	0.733
MIN_ENTROPY #1	ME = 3	121	345	0.259
MIN_ENTROPY #1 + NER	ME = 3	28	33	0.259
MIN_ENTROPY #2	ME = 4	53	23	0.697
MIN_ENTROPY #2 + NER	ME = 4	51	11	0.697
MIN_UNIQUES #1	MU = 1	55	56	0.495
MIN_UNIQUES #2	MU = 3	18	31	0.367
WINDOW_SIZE #1	WSIS = 300	107	78	0.578
WINDOW_SIZE #1 + NER	WSIS = 300	98	34	0.742
WINDOW_SIZE #2	WSIS = 600	116	114	0.504
WINDOW_SIZE #3	WSIS = 1200	114	144	0.441
WINDOW_SIZE #4	WSIS = 1500	112	150	0.427
FINAL	MT = 2, MTIS = 1, MCFE = 0.6, ME = 4, MU = 2, WSIS = 300	22	2	0.916

[2] T. Sakaki, M. Okazaki, and Y. Matsuo, "Earthquake shakes twitter users: real-time event detection by social sensors," in *Proceedings of the 19th international conference on World wide web*. ACM, 2010, pp. 851–860.

[3] S. Phuvipadawat and T. Murata, "Breaking news detection and tracking in twitter," in *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on*, vol. 3. IEEE, 2010, pp. 120–123.

[4] M. Hu, S. Liu, F. Wei, Y. Wu, J. Stasko, and K.-L. Ma, "Breaking news on twitter," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2012, pp. 2751–2754.

[5] S. Petrović, M. Osborne, and V. Lavrenko, "Streaming first story detection with application to twitter," in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 2010, pp. 181–189.

[6] J. Kleinberg, "Bursty and hierarchical structure in streams," *Data Mining and Knowledge Discovery*, vol. 7, no. 4, pp. 373–397, 2003.

[7] B. W. Locke, "Named entity recognition: Adapting to microblogging," 2009.

[8] X. Liu, S. Zhang, F. Wei, and M. Zhou, "Recognizing named entities in tweets," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, ser. HLT '11. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011, pp. 359–367. [Online]. Available: http://dl.acm.org/citation.cfm?id=2002472.2002519

[9] L. Chiticariu, R. Krishnamurthy, Y. Li, F. Reiss, and S. Vaithyanathan, "Domain adaptation of rule-based annotators for named-entity recognition tasks," ser. EMNLP '10, 2010, pp. 1002–1012.

[10] C. Li, J. Weng, Q. He, Y. Yao, A. Datta, A. Sun, and B.-S. Lee, "Twiner: Named entity recognition in targeted twitter stream," in *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '12. New York, NY, USA: ACM, 2012, pp. 721–730. [Online]. Available: http://doi.acm.org/10.1145/2348283.2348380

[11] A. Ritter, S. Clark, Mausam, and O. Etzioni, "Named entity recognition in tweets: An experimental study," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, ser. EMNLP '11. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011, pp. 1524–1534. [Online]. Available: http://dl.acm.org/citation.cfm?id=2145432.2145595

[12] P. Agarwal, "Prediction of trends in online social netwok," Ph.D. dissertation, Indian Institute of Technology New Delhi, 2013.

[13] P. Indyk and R. Motwani, "Approximate nearest neighbors: towards removing the curse of dimensionality," in *Proceedings of the thirtieth annual ACM symposium on Theory of computing*. ACM, 1998, pp. 604–613.

[14] C. E. Shannon, "A note on the concept of entropy," *Bell System Tech. J*, vol. 27, pp. 379–423, 1948.

[15] G. G. Chowdhury, "Natural language processing," *Annual review of information science and technology*, vol. 37, no. 1, pp. 51–89, 2003.

[16] G. Zhou and J. Su, "Named entity recognition using an hmm-based chunk tagger," in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, 2002, pp. 473–480.

[17] M. Vogiatzis, "Using storm for real-time first story detection," Master's thesis, University of Edinburgh, School of Informatics, 11 Crichton Street, Edinburgh, Midlothian EH8 9LE, Great Britain, 2012, retrieved online on May 14 2015 at url=https://micvog.files.wordpress.com/2013/06/vogiatzis_storm.pdf.

[18] M. Tavakolifard, K. C. Almeroth, and J. A. Gulla, "Does social contact matter?: modelling the hidden web of trust underlying twitter," in *Proceedings of the 22nd international conference on World Wide Web companion*. International World Wide Web Conferences Steering Committee, 2013, pp. 981–988.

[19] J. E. Ingvaldsen, J. A. Gulla, and Ö. Özgöbek, "User controlled news recommendations," in *Proceedings of the Joint Workshop on Interfaces and Human Decision Making for Recommender Systems co-located with ACM Conference on Recommender Systems (RecSys 2015)*, 2015.

[20] J. A. Gulla, A. D. Fidjestøl, X. Su, and H. Castejon, "Implicit user profiling in news recommender systems," 2014.