Towards Personalized Offers by Means of Life Event Detection on Social Media and Entity Matching

Paulo Cavalin
IBM Research - Brazil
pcavalin@br.ibm.com

Maíra Gatti
IBM Research - Brazil
mairacg@br.ibm.com

Claudio Pinhanez
IBM Research - Brazil
csantosp@br.ibm.com

ABSTRACT

In this paper we present a system for personalized offers based on two main components: a) a hybrid method, combining rules and machine learning, to find users that post life events on social media networks; and b) an entity matching algorithm to find out possible relation between the detected social media users and current clients. The main assumption is that, if one can detect the life events of these users, a personalized offer can be made to them even before they look for a product or service. This proposed solution was implemented on the IBM InfoSphere BigInsights platform to take advantage of the MapReduce programming framework for large scale capability, and was tested on a dataset containing 9 million posts from Twitter. In this set, 42K life event posts sent by 19K different users were detected, with an overall accuracy of 89% e precision of about 65% to detect life events. The entity matching of these 19K social media users against an internal database of 1.6M users returned 983 users, with accuracy of about 90%.

Keywords

Social Media Networks, Life Event Detection, Natural Language Processing, Machine Learning, Entity Matching

1. INTRODUCTION

Social Media Networks (SMN), such as Twitter and Facebook, engage thousands of people that post, on a daily basis, a huge amount of content represented by texts, images, videos, etc [5, 10]. Often the content can be intimately related to the person the publishes it, in such a way that is can expose behavioral traits or events that are happening in the individual's life. As a consequence, the proper exploration of this type of content not only can be a way to better understand the users on SMNs, but also can leverage many applications that require adequate user profiling, for instance credit risk analysis, marketing campaigns, and personalized product and/or service offers.

One way to find potential customers for services or products is by detecting life events from public user activities on SMNs, in special microbloggings. Generally, a life event can be defined as something important that happened, is happening, or will be happening, in a particular individual's life, such as getting married, get graduated, having a baby, buying a house, and thus forth. That is, if a life event is properly detected, a product or service can be offered to someone even before she looks for it, anticipating her needs. For instance, if a person posts on the SMN that her marriage will be happening in a few days (or weeks or months), a loan or an insurance (for the honey moon trip for example) can be offered to her in advance. Furthermore, as state in [6], marketers know that people mostly shop based on habits, but that among the most likely times to break those habits is when a major life event happens.

For this reason, this work focuses on presenting a system that can detect life events from textual posts on SMNs, and can match the corresponding users with an existing database, i.e. entity matching with current clients, using basic information such as the name and the location available on the SMN. Entity matching is important to understand whether a given user of a SMN is already a customer or not, and adapt the way the person can be approached.

Both life event detection and entity matching are complex tasks which are subject of various research in fields such as artificial intelligence, machine learning [6], natural language processing and large scale analysis of unstructured data (popularly known as *Big Data*) [12]. Performing natural language processing on microbloggings' posts presents several challenges, such as dealing with the short and asynchronous nature of the messages, making it difficult to extract contextual information, and dealing with a very unnormalized vocabulary due to the frequent use of slangs, acronyms, abbreviations, and informal language often with misspelling errors [1, 7, 13]. Nonetheless, one study that supports the possibility of detecting life events from textual posts has been presented in [4]. In that work, the author conducted a study on the behavior of mothers during pregnancy, and they observed that these mothers can be distinguished by linguistic changes captured by shifts in a relatively small number of words in their social media posts.

In the light of this, in this work we describe and evaluate our proposed solution to tackle the life event detection problem and the entity matching. For the first task, we propose a

hybrid system combining rules and machine learning (ML). In contrast to the system specifically focused on life event detection presented in [6] (the only one for this problem to the best of our knowledge), which uses only ML, our system allows for dealing with the life event classes independently. The rule-based phase acts as a mechanism to filter most posts that do not contain life events, since all those posts not matching the desirable rules are eliminated. Then, binary classifiers (one for each type of life event) are applied to validate the possible life events. Greater detail is provided in Section 3.1. For entity matching, a combination of string distance functions is used to compare the names and locations of the users. This method is better described in Section 3.2.

The entire system has been implemented on the IBM InfoSphere BigInsights platform [9], to take advantage of the MapReduce programming paradigm for large scale data processing. A dataset containing 9 million posts in portuguese, extracted from Twitter, has been used to evaluated the system. To evaluate the entity matching, a database with 1.6 million users has been constructed. More details about the experiments are present in Section 4.

2. BACKGROUND AND RELATED WORK

Since the work proposed in this paper is a hybrid solution on which we integrate a ML-based classifier with an Entity Matching solution, the background and related work is presented separated for both as follows:

Life Event Detection: as already mentioned, a life event can be defined as something important regarding the users' lives in SMNs. It is important to differentiate it from some related work which uses the *event detection* expression to refer to the problem of detecting unexpected event exposed by several users in SMNs like a rumor, a trend, or emergent topic. In the case of the work proposed in this paper, detection means to classify a short post, like Twitter's or Facebook's status messages in one of the life event categories, which could be considered, for instance, topics. Therefore, as related work, any approach of topic classification of short messages could be considered like [6], which is the most related to our work. Regarding ML-based solutions, other supervised or unsupervised methods for topic classification are also related, although not yet used for short messages but long documents. And regarding semantic-rule-based solutions, AQL rules combined with dictionaries are known approaches for topic classification with the usage of templates. Ontologies have also been applied for long documents.

Entity Matching: in SMNs there are two problems one can find Entity Matching solutions for. One is, given a set containing user features on SMNs, like user information and activities, and another set containing real people information, the goal is to try to match the users within both sets. The second problem is, given two sets containing user features on two different SMNs, the goal is to try finding corresponding users, i.e., the biggest possible number of social profiles that refer to the same person between both social networks. The latter can also be called Entity Resolution (ER) problem, and in the past few years some work has been proposed to solve this problem. For instance, [14] proposed supervised learning techniques and extracted features to build different classifiers, which were then trained and used to rank the probability that two user profiles from two different OSNs belong to the same individual.

The former problem can be considered a subset of the latter if we ignore the fact that the second set contains real people information rather than SMN's profiles. And generally, as summarized by [15], there are two approaches for handling this: (i) syntactic-based similarity approaches: providing exact or approximate lexicographical matching of two values; and (ii) semantic-based similarity approaches: used to measure how two values, lexicographically different, are semantically similar. For instance, Foaf-o-matic[1] and OKKAM[2] projects aim at social profiles integration by means of formal FOAF (Friend-of-a-friend) semantics.

Regarding, syntactic-based similarity approach, we summarize here the ones most used for URI, numeric-based attributes and, in the context of SNMs, two users' full names. *Levenshtein* or *Edit Distance* [11] is defined to be the smallest number of edit operations, inserts, deletes, and substitutions required to change one string into another. In addition, Jaro is an algorithm commonly used for name matching in data linkage systems. A similarity measure is calculated using the number of common characters (i.e., same characters that are within half the length of the longer string) and the number of transpositions. Winkler (or Jaro-Winkler) improves upon Jaro's algorithm by applying ideas based on empirical studies which found that fewer errors typically occur at the beginning of names [3][2].

Another approach is the N-Gram name similarity, on which N-grams are sub-strings of length n and an n-gram similarity between two strings is calculated by counting the number of n-grams in common (i.e., n-grams contained in both strings) and dividing by either the number of n-grams in the shorter string (called Overlap coefficient), or the number of n-grams in the longer string (called Jaccard similarity), or the average number of n-grams in both strings. 2-grams and 3-grams have been used to calculate the similarity between the two users' full names. Finally, the VMN name similarity approach proposed by [18] was designed for full and partial matches of names consisting of one or more words. VMN supports the case of swapped names and the cases of partial matches.

In this paper, we use two versions of ED preceded by Jaro's similarity as described in the next section.

3. METHODOLOGY

In this section we describe in detail both systems for life event detection system and entity matching.

3.1 Hybrid Life Event Detection System

Given a social media network, the life event detection system has as main goal to return a list of users that posted life events within a given time window. This task involves a crawler to gather data, and a system to search for life events on the data. Note that not only accuracy is important in this case, to find the largest list of users with a high precision, but also performance is important since the system is likely

[1]http://www.foaf-o-matic.org/
[2]http://www.okkam.org/

to face a large amount of data. In addition, on a production environment, the system must allow for easy fine-tuning, addition and removal of life events classes.

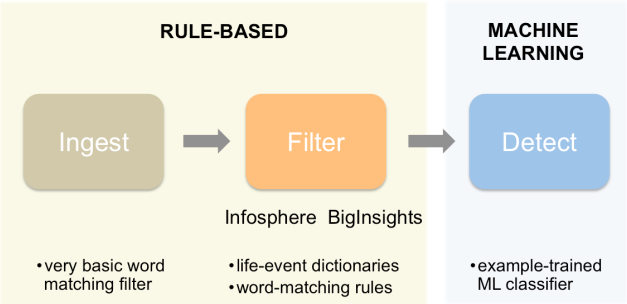

Figure 1: Hybrid Life Event Detection System.

To cope with the aforementioned issues, we propose a hybrid life event detection approach, combining both rules and machine learning (ML). Such a system, depicted in Figure 1, is basically composed of three subsequent phases or modules, namely *Ingest*, *Filter*, and *Detect*. The first phase, i.e. *Ingest*, captures a database of posts to be used for the search for life events. This is done by considering a set of words that can possibly relate to all life events of the system. We assume that the larger this dataset, the larger the set of users that will be returned. Once the set of posts has been totally crawled, the *Filter* module selects the set of posts that is more likely to contain life events. That is, by considering a set of simple rules such as words and combinations of words (but more elaborated rules than those of *Ingest*), but in this case a set of rules for each type of life event, the posts that match these rules are marked with the corresponding possible life events.

Despite these rules can indicate a possible life event, a large portion of these message can be false candidates. For this reason, the *Detected* phase is then carried out to validate the possible life events with their corresponding probability. For each post found in the *Filter* phase, we apply the ML classifier of the corresponding possible life events and compute the probability of that the post contains the given life events. With this information, all posts with life event probability above the threshold θ are selected and users of the corresponding posts are generate as the output of the system.

It is worth noting that currently ML is well-known to produce the best solutions to deal with ambiguous and noisy texts such as microbloggings' posts. However, the proposed hybrid solution takes advantage of the rule-based filtering to reduces the search space for the ML classifier, which can reduce both the number of errors and processing time. Moreover, by treating types of life events independently it makes it easy for fine-tuning, addition and removal of life event classes. For instance, to add a new type of life events, one need to append the corresponding keywords for the *Ingestion* phase, the rules for *Filter*, and a binary classifier in the *Detect* phase. This can be done with no impact on the accuracy of existing life events.

3.2 Entity Matching System

Given the output of the life event detection system, i.e. users (aka entities) that posted life events on social media, the main goal of the entity matching system is to find corresponding people in a database of real names. For achieving this task accurately, the system must use as much information as possible to decrease the level of uncertainty.

Dealing with users found on SMNs, though, is very challenging. First of all, on most SMNs the basic information about the user (e.g. name, location, age) is very limited (on Twitter only the name and location of the user are available). In addition, such personal information may be lacking or not relevant since filling them may be not mandatory, and the content filed is not verified. Besides that, when the information is seriously provided by the user, other difficulty factors can appear, such as the use of simplified names (*Claudio Pinhanez* instead of *Claudio Santos Pinhanez*), the use of social media pen-names (*@cinhanez* instead of *Claudio Santos Pinhanez*), or the use of nickname (*Darth Vader* instead of *Claudio Santos Pinhanez*.

To deal with some of the aforementioned difficulty factors, for this work we have developed a system to match names and locations of users using three different string distance functions:

1. *Exact matching (EM)*: a match is found if all the names of an SMN user are identical to those of a client
2. *Entity Distance 1 (ED1)*: designed to consider misspellings and transpositions between adjacent characters as a match. For instance, the user "Jooa Paulo" matches the client "Joao Paulo", and the user "Carolina" matches "Carolina". In this case, the threshold σ_1 is used to define a match only if the similarity value is above this threshold.
3. *Entity Distance 2 (ED2)*: designed to match abbreviations and some nicknames. For example, the user "Joseph S." matches the client "Joseph Salem"; the user "Fabinho" matches the client "Fábio", and "Mari" matches "Mariana". Similarly to ED1, the threshold σ_2 is used to define a match.

The execution of three aforementioned matching algorithms results in three distinct sets of users, denoted Ω_{EM}, Ω_{ED1} and Ω_{ED2}. The resulting set of users Ω_{All} corresponds to the union of those individual sets. That is, $\Omega_{All} = \Omega_{EM} \cup \Omega_{ED1} \cup \Omega_{ED2}$, where $\Omega_{EM} \cap \Omega_{ED1} \cap \Omega_{ED2} \neq \emptyset$ or $\Omega_{EM} \cap \Omega_{ED1} \cap \Omega_{ED2} = \emptyset$, depending on the data.

It is worth mentioning that the Jaro Winkler similarity filtering [20] is used prior to calling ED1 and ED2, to eliminate weak matches such as 'Maria' and 'Maria das Graças Silva'. Furthermore, ED1 and ED2 may return more than one match for the same user, whenever the result is above the given threshold. In this work, only the matching with the highest value is considered.

4. EXPERIMENTS

In this section we present the results of applying the proposed system on a dataset containing 9 millions of posts

from Twitter, which have been produced by about 1.4 million users. This data has been gathered by means of the GNIP social media data provider [8].

Mainly, these experiments have two different purposes. First we aim at evaluating the numbers related to applying the system on this 9 million dataset, i.e. how many posts and users are returned by using the system. And second, we focus on a quality analysis to validate those numbers by means of a manual inspection of samplings of this dataset.

The life event detection system has been implemented for six types of life events: Marriage, Graduation, Travel, Birthday, Birth,and Death. For each one, a training dataset of about 2 thousand samples has been manually labeled as either life event or non life event, and a distinct classifier has been trained. The training data has been obtained with the Twitter Search API [17]. For this work we make use of Naive Bayes classifiers using bag-of-words features [19]. The main parameters, i.e. θ, σ_1 and σ_2, have been set to 0.5, 0.95 and 0.95, respectively.

4.1 Quantitative Results

As we mentioned, the first experiment has as main purpose to evaluate how many posts and users are returned after carrying out each phase of the proposed system. The results of applying the implemented life event detection system on 9-million-tweet dataset is summarized in Figure 2. In this case, the Filter phase has returned 347 thousand posts from about 220 thousand users. Then, after going through the Detect module, 42 thousand posts, from about 19 thousand users, have been detected as life events. It is worth noting the large difference in terms of proportion from one phase to another. The Ingest phase captures a very large dataset, i.e. 9 million posts. Then, Filter finds out that only 3.7% of these posts can be of interest. However, the Detect phase shows that from these 347K% of posts, only 42 thousand (0.45% of 9M or 12% of 347K) are really those that the application is looking for. Considering that many of the current search system are rule-based, these results indicate that our proposed system can avoid a useless search on about 88% of the posts returned, 307 thousand posts in this case.

In Table 1 we present the results of the experiment above for each type of life event. We can observe that about 12% of the posts filtered have been generally confirmed as life events, but this proportion can vary according to the type of life event. For instance, for the Marriage class, from the 182,096 posts that the filter considered as possible life event, the machine learning algorithm detected 19,475 (10.6%) as being actually life events, which is close to the average. The Graduation type, on the other hand, presented a much larger proportion (43.21%), while Death and Travel smaller ones (5.47% and 8.26% respectively). We believe that this difference can happen either due to the period of the year in which the data is gathered (Graduation supposedly has more posts in certain periods of the year), or even due to the type of life event that may contain more non life events (Travel for example, which may present many posts from marketing agencies) or even less life event posts (for instance Death, whereas people might to be more introspective).

To evaluate the entity matching, we have a built a dataset

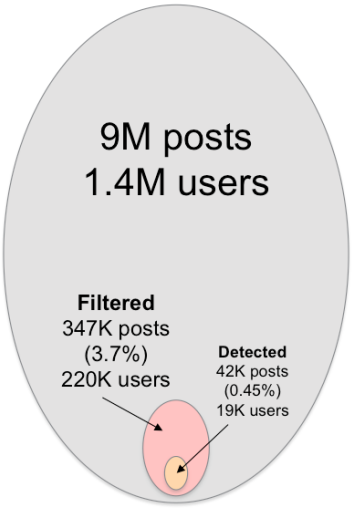

Figure 2: Results on the 9M dataset.

Table 1: Detailed results on the 9 million dataset.

Life event	Filter (% of 350k)	Detect (% of 42k)
Marriage	182,096 (52.4)	19,457 (46.5)
Graduation	25,676 (7.4)	11,097 (26.5)
Travel	22,596 (6.5)	1,868 (4.5)
Birthday	33,305 (9.6)	3,604 (8.6)
Birth	48,687 (14.0)	3,881 (9.3)
Death	35,242 (10.1)	1,929 (4.6)
Total (% of 9M)	347,602 (3.7)	41,836 (0.45)

containing 1.6 million users using publicly-available data. The users on this dataset have been matched against the 19 thousand users that have been detected as the ones that posted life events in the 9M dataset. The results and this process are illustrated in Figure 3. Note that we have conducted two different experiments. The first one matches these users by taking into account only their names, since we consider this as the minimum information we will be able to obtain from the SMN. In this case, 983 users have been found as probable matches. In the second experiment, where both names and locations are considered, only 5 users have been found. This shows that the precision of entity matching can be increased considering more for this process. On the other hand, this will also reduce the size of the resulting matching set.

In order to validate the above results, we performed a random sampling of 23 thousand posts (from the 9 million set) focusing on quality analysis. The number of posts filtered and detected are shown in Figure 4. The total of posts filtered is 1,008, from which 105 have been detected as life events. Similar to the results on the 9 million set, only about 10% of the filtered posts have been detected as life events. Detailed numbers, for each type of life event, are presented in the columns Filtered and Detected in Table 2.

Those 1,008 posts resulting from the Filter module have

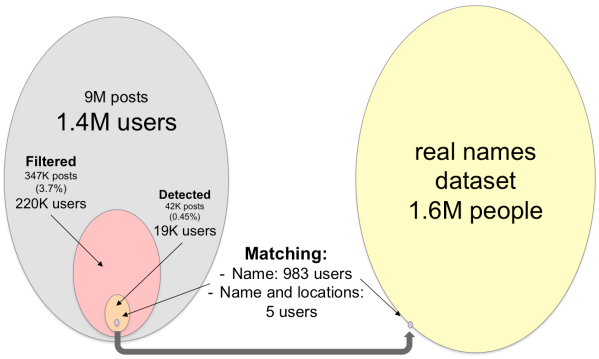

Figure 3: Entity matching results.

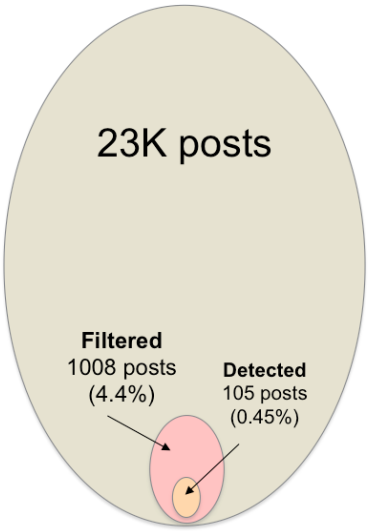

Figure 4: Results on the 23K sampling.

Table 2: Number of posts returned per life event type on the 23K sampling.

Life event	Filtered (% of 1008)	Detected (% of 105)	Ground-Truth (% of 142)
Marriage	162 (16.2)	8 (7.6)	7 (4.9)
Graduation	70 (7.0)	26 (24.7)	15 (10.6)
Travel	474 (47.4)	55 (52.4)	99 (69.7)
Birthday	102 (10.2)	11 (10.5)	12 (8.5)
Birth	107 (10.7)	4 (3.8)	7 (4.9)
Death	93 (9.3)	1 (9.5)	2 (1.4)
Total (% of 23K)	1008 (4.4)	105 (0.45)	142 (0.6)

been then manually inspected in order to verify whether the Detect phase has assigned the correct probability or not. The total of posts for each type of life event are listed in the Ground-Truth column in Table 2. It can be observed that our system presents numbers that are close to what was found by the manual inspection. By comparing the manual inspection with the results of the system, we have been able to compute the confusion matrix presented in Table 3, which contains the total number of true positives, true negative, false positives and false negatives. This has allowed us to compute the values for accuracy, precision and recall [16], which were at about 89%, 65% and 48%, respectively. In this case, a true positive consists of a posts that contains a life event (according to the manual inspection) and is correctly detected by the system, a true negative is not a life event and is correctly ignored by the system, a false positive is not a life events but is detected by the system, and a false negative is a life event but is not detected by the system. As a consequence, the precision represents the proportion of detected posts that contain life events, and the recall the proportion of life events that have been found by the system. It is worth noting that there is a trade-off between precision and recall that is set according to the value of θ, where lower values can increase recall and large values increase the precision (see Figure 5).

Table 3: Confusion matrix of the 1008 filtered posts found on the 23K sampling.

		Manual labeling	
		Life Events	**Non Life Events**
Life Event Detection System	**Positive**	68 (6.7%)	37 (3.7%)
	Negative	74 (7.3%)	829 (82.4%)

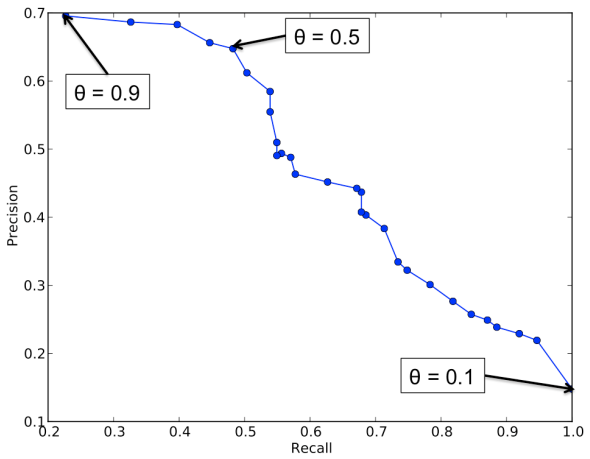

Figure 5: Precision/Recall trade-off by varying θ from 0.1 to 0.9.

Similarly, to validate the quality of the entity matching algorithm we have done a random sampling of 500 users and manually inspected the correctness of the matchings found. In this case, the entity matching algorithm returned 72 users, being 43 found by EM, 13 by ED1 and 16 by ED2. But, as we mentioned, both ED1 and ED2 can return more than one matching per user if the matching algorithm returns a value above the threshold σ_1 and σ_2. For a better analysis of the algorithm, in Table 4 and Table 5 we present the confusion matrices of both ED1 and ED2 considering all matches. The former has found a total of 476 matches, with an accuracy of about 91%, precision of 10.4% and recall of 71.4%, while the latter has returned a total of 452 matches, 94% of accuracy, precision of 50% and recall of 94%.

Table 4: Confusion matrix for ED1 on 500 users.

		Manual labeling	
		Match	**Non Match**
Entity Matching System	**Positive**	5 (1.10%)	43 (9.0%)
	Negative	2 (0.4%)	426 (89.5%)

Table 5: Confusion matrix for ED2 on 500 users.

		Manual labeling	
		Match	**Non Match**
Entity Matching System	**Positive**	17 (3.7%)	17 (3.7%)
	Negative	1 (3.9%)	427 (96.1%)

5. CONCLUSIONS

In this work we presented a system for personalized offer based on life event detection. Once the system detects users posting life events on a social media network, these users are matched against an internal database of clients to decide what is the best approach to offer them a service or product. We described a way to implement the entire system, and presented the results of applying the system on a dataset of 9 million posts. From this set, a total of 42 thousands life events have been found, with a projected accuracy of 88.90% and precision of 65%. This indicates that, in a normal day of 20 million posts published by Brazilian users, for instance, the system presents the ability to detect around 91 thousand posts a day, being about 60 thousand of them correct. Besides that, it is worth mentioning that the system is scalable since it has been implement with the MapReduce programming paradigm.

Future work can follow many different and complementary paths. Accuracy is important and could be improved by evaluating other types of classifiers and features, as well as increasing training data. The addition and evaluation of other types of life events could be important to better understand the way people behave on the SMNs. Furthermore, the adaptation to a real-time streaming platform such as the IBM InfoSphere Streams would allow the system react very quickly (near to real-time) once the users post life events.

6. REFERENCES

[1] Atefeh, F., and Khreich, W. A survey of techniques for event detection in twitter. *Computational Intelligence* (2013), n/a–n/a.

[2] Bilenko, M., Mooney, R., Cohen, W., Ravikumar, P., and Fienberg, S. Adaptive name matching in information integration. *IEEE Intelligent Systems 18*, 5 (Sept. 2003), 16–23.

[3] Cohen, W. W., Ravikumar, P., and Fienberg, S. E. A comparison of string distance metrics for name-matching tasks. pp. 73–78.

[4] De Choudhury, M., Counts, S., and Horvitz, E. Major life changes and behavioral markers in social media: Case of childbirth. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work* (New York, NY, USA, 2013), CSCW '13, ACM, pp. 1431–1442.

[5] Ehrlich, K., and Shami, N. S. Microblogging inside and outside the workplace. In *ICWSM* (2010).

[6] Eugenio, B. D., Green, N., and Subba, R. Detecting life events in feeds from twitter. *2012 IEEE Sixth International Conference on Semantic Computing 0* (2013), 274–277.

[7] Felt, A. P., and Wagner, D. Phishing on mobile devices. In *In W2SP* (2011).

[8] GNIP. GNIP, 2014. [Online; accessed 28-May-2014].

[9] IBM. IBM InfoSphere BigInsights, 2014. [Online; accessed 28-May-2014].

[10] Kwak, H., Lee, C., Park, H., and Moon, S. What is twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web* (New York, NY, USA, 2010), WWW '10, ACM, pp. 591–600.

[11] Levenshtein, V. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady 10* (1966), 707.

[12] Lin, J., and Dyer, C. *Data-Intensive Text Processing with MapReduce*. Claypool Publishers, 2010.

[13] Liu, F., Weng, F., and Jiang, X. A broad-coverage normalization system for social media language. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1* (Stroudsburg, PA, USA, 2012), ACL '12, Association for Computational Linguistics, pp. 1035–1044.

[14] Peled, O., Fire, M., Rokach, L., and Elovici, Y. Entity matching in online social networks. In *Social Computing (SocialCom), 2013 International Conference on* (Sept 2013), pp. 339–344.

[15] Raad, E., Chbeir, R., and Dipanda, A. User profile matching in social networks. In *Network-Based Information Systems (NBiS), 2010 13th International Conference on* (Sept 2010), pp. 297–304.

[16] Sokolova, M., and Lapalme, G. A systematic analysis of performance measures for classification tasks. *Information Processing and management*, 45 (2009), 427–437.

[17] Twitter. Using the Twitter Search API, 2014. [Online; accessed 28-May-2014].

[18] Vosecky, J., Hong, D., and Shen, V. User identification across multiple social networks. In *Networked Digital Technologies, 2009. NDT '09. First International Conference on* (July 2009), pp. 360–365.

[19] Weiss, S. M., Indurkhya, N., and Zhang, T. *Fundamentals of Predictive Text Mining*. Springer London, 2010.

[20] Winkler, W. E. String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage. In *Proceedings of the Section on Survey Research Methods (American Statistical Association* (1990), pp. 354–359.