Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence

Exacting Social Events for Tweets Using a Factor Graph

Xiaohua Liu [‡ †], Xiangyang Zhou [♯], Zhongyang Fu [§], Furu Wei [†], Ming Zhou [†]
[‡]School of Computer Science and Technology
Harbin Institute of Technology, Harbin, 150001, China
[♯]School of Computer Science and Technology
Shandong University, Jinan, 250100, China
[§]Department of Computer Science and Engineering
Shanghai Jiao Tong University, Shanghai, 200240, China
[†]Microsoft Research Asia
Beijing, 100190, China
[†]{*xiaoliu, fuwei, mingzhou*}*@microsoft.com*
[§] *zhongyang.fu@gmail.com* [♯] *v-xzho@microsoft.com*

Abstract

Social events are events that occur between people where at least one person is aware of the other and of the event taking place. Extracting social events can play an important role in a wide range of applications, such as the construction of social network. In this paper, we introduce the task of social event extraction for tweets, an important source of fresh events. One main challenge is the lack of information in a single tweet, which is rooted in the short and noise-prone nature of tweets. We propose to collectively extract social events from multiple similar tweets using a novel factor graph, to harvest the redundance in tweets, i.e., the repeated occurrences of a social event in several tweets. We evaluate our method on a human annotated data set, and show that it outperforms all baselines, with an absolute gain of 21% in F1.

Introduction

Social events are events that occur between people, in which at least one person is aware of the other and of the event taking place (Agarwal and Rambow 2010). For example, there is a social event in the sentence "John talks to Mary", since John and Mary are aware of each other and both are aware of the talking event. The task of social event extraction is believed to be helpful for a wide range of applications including social network construction, question-answering, summarization and so on.

Apoorv and Rambow (2010) introduce this task. They annotate part of Automatic Content Extraction (ACE) data, and perform social event detection and classification experiments using Support Vector Machines (SVMs) with Kernel methods. Structures derived from phrase structure trees and dependency trees are adopted as features. In their work, social events are classified into two types: 1) Interaction, in which both parties are aware of the social event (e.g., a conversation); and 2) observation, in which only one party is aware of the interaction (e.g., thinking about or spying on someone).

Inspired by this work, we propose the task of social event extraction for tweets. We are interested in tweets, because it has become an important repository of fresh events. Nearly all significant events are first reported in tweets, e.g., the Los Angeles fire in 2007, the Chile earthquake in 2008, and the death of Michael Jackson in 2009. Particularly, our investigation reveals that 15.2% tweets contain a social event [1]. However, this task is hard because: 1) A tweet is often too short to provide sufficient information to the classifier; and 2) a tweet is often so noisy that current natural language processing (NLP) tools, such as chunker and syntax parser which have proved critical for social event extraction, are unable to offer reliable features.

We propose to extract social events from multiple similar tweets using a factor graph to tackle this issue. Our solution is based on the following observation: a social event is likely to appear in multiple tweets, for some of which extraction is easy while for the others it is hard. As an illustrated example, consider the tweets in Table 1. It is straightforward to extract an interaction type social event involving "Aquino" and " Obama" from the first tweet, owing to the strong indicator of "meet"; while it is hard to extract it from the second, because of the limited context. Intuitively, knowing that there is an interaction type social event involving "Aquino" and "Obama" in the first tweet will encourage us to guess the same event for the second tweet.

Our graphical model is built as follows. We first introduce a random variable for each social event candidate, whose

[1]We sample about 10,000 tweets, and for each tweet manually check whether it has any social event. It turns out 1,526 contains social events.

value indicates the social event type of the corresponded candidate. Following Apoorv and Rambow (2010), we discriminate two types of social events, Interaction and Observation, and use a special type None to indicate the candidate is not a social event. Hereafter, x_m^i and y_m^i are used to denote the i^{th} candidate and its associated variable from tweet t_m, respectively. Next we add a factor for every two variables whose corresponded candidates involve the same two entities and come from similar tweets. We use f_{mn}^{ij} to denote the factor connecting y_m^i and y_n^j (t_m and t_n are similar tweets). We say two tweets are similar if and only if they fall into a period of time and their content similarity is above a threshold (0.3 in our work). Figure 1 illustrates an example of our factor graph.

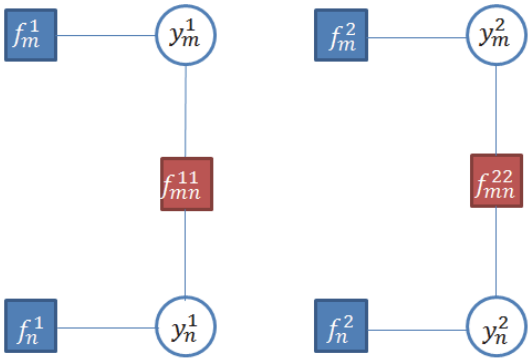

Figure 1: Variables and factors of our factor graph model. Circles represent random variables, each of which denotes the type of a social event candidate; Blue and red rectangles represent factors related to a single variable and two variables, respectively.

Note that the idea of utilizing redundance in tweets has been successfully practiced in some of recent studies on tweets, most of which are based on the two-stage labeling strategy. For example, Liu et al. (2011) cluster similar tweets and conduct two-stage Semantic Role Labeling (SRL) on each cluster of tweets, in which the second round labeler exploits some statistical features derived from the outputs of the first round labeler. In contrast to these studies, our method is based on probabilistic graphical model, which is mathematically well formalized.

We evaluate our method on a manually annotated data set. Experimental results show that our method achieves 77% F1, as compared to 54% F1 of the state-of-the-art baseline that performs the extraction tweet by tweet using SVM based classifiers. Our method also beats a variety of our method that ignores cross-tweet information. Based on the mined social events, we further build an application that visualizes people's social distance closeness , and receive positive feedback.

We summarize our contributions as follows.

1. We introduce the task of social event mining for tweets. One main challenge is that a tweet often cannot provide insufficient information.

Aquino to meet with Obama in May or June - US envoy — Inquirer Global Nation j.mp/xCwmCS
Aquino-Obama in May or June: US envoy - BusinessWorld Online Edition goo.gl/fb/ZXYsp

Table 1: A motivating example of collective social event extraction.

2. We propose to extract social events from multiple similar tweets using a graphical model, which exploits redundance in tweets to make up for the lack of information in a single tweet.
3. We evaluate our method on a manually annotated data set, and show that our method achieves an F1 of 77%, outperforming all baselines.

The rest of this paper is structured as follows. In the next section, we introduce related work. In Section 3, we define the task. In Section 4, we present our method. In Section 5, we evaluate our method. Finally, we conclude in Section 6 with a brief discussion of future work.

Related Work

Event Extraction on Formal Text. Harabagiu et al. (2005) propose a sentence level ACE event extraction system, which combines pattern matching and statistical models. Liao and Grishman (2010) develop existing research on ACE event extraction by using cross-event information. There is also a large body of studies targeting TimeML events, e.g., STEP (Bethard and Martin 2006) which uses a rich set of textual, morphological dependency and WordNet hypergamy features to build a SVMs model. Most recently, Llorens al. (2010) analyze the contribution of semantic roles to TimeML event identification. Studies on event identification besides ACE and TimeML also exist. For example, Yu et al. (2009) utilize association language patterns plus single words as features to classify sentences with negative life events into predefined categories (e.g., *Family*, *Love*, *Work*).

Apoorv and Rambow (2010) first study the task of social event extraction on a corpus derived from ACE data, and propose to use kernel SVMs to achieve it. Our work is inspired by this work. However, there are two remarkable differences. Firstly, we use a graphical model, rather than kernel SVM, to simultaneously infer the labels of all event candidates. One advantage of our model is that it allows us to aggregate across tweet information to make up for the lack of information in a single tweet. Secondly, our model adopts only shallow linguistic features, rather than chunking and dependency parsing related features. This is because tweets are short and often informally written, and as a result current natural language processing tools perform bad on tweets (Ritter et al. 2011), which means such advanced linguistic features are not reliable for tweets.

Event Extraction on Tweets. Sankaranarayanan et al. (2009) extract breaking news from tweets to build a news processing system, called TwitterStand; and particularly, Sakaki et al. (2010) devise a classifier of tweets based on features derived from one tweet (e.g., the keywords in a

tweet and the number of words) to detect a particular type of event like *Earthquake*. Benson et al. (2011) present a graphical model to extract canonical entertainment events by aggregating information across multiple messages.

Our method is similar to Benson et al. (2011) in the sense that we also adopt a factor graph as our model. However, two significant differences exist. Firstly, our work concentrates on social event extraction for tweets, rather than entertainment events; secondly, feature weights of our factor graph are automatically tuned on the training data, while Benson et al. (2011) manually decide a good part of their model parameters.

Task Description

A tweet is a short text message with no more than 140 characters. Here is an example of tweets: "mycraftingworld: #Win Microsoft Office 2010 Home and Student #Contest from @office http://bit.ly/ $\cdots$ ", where "mycraftingworld" is the name of the user who published this tweet; words beginning with "#" like ""#Win" are hash tags; words starting with "@" like "@office" represent user names; and "http://bit.ly/" is a shortened link.

Given as input a collection of tweets $T = \{t_m\}$, our task is to output $E_m = \{e_m^i\}_i^{N_m}, m = 1, \cdots, |T|$, where e_m^i denotes the i^{th} social event extracted from the m^{th} tweet; N_m stands for the total number of social events extracted from t_m. Following Apoorv and Rambow (2010), we define a social event as a triple $e_m^i = (p_1, p_2, y)$, where: p_1 and p_2 stand for the two persons involved into the event; y denotes the type of the event, which can be Interaction (I), Observation (O) or None (N). Note that we always assume (p_1, p_2, y) and (p_2, p_1, y) are interchangeable. And to concentrate our focus on social event extraction, we assume that person names in the input tweets have been labeled.

As an illustrative example, consider the following two tweets: "$\cdots$ Love is in the air? This pic of the day looks like [Demi Lovato]$_p$ and [Wilmer Valderrama]$_p$ are getting pretty close$\cdots$" and "$\cdots$ [Demi Lovato]$_p$ & [Wilmer Valderrama]$_p$ are caught dating$\cdots$", where each "[$\cdots$]$_p$" denotes a person name. The expected output is: $E_1 = \{$(Demi Lovato,Wilmer Valderrama, I)$\}$ and $E_2 = \{$(Demi Lovato,Wilmer Valderrama, I)$\}$.

Our Method

Overview. Our method consists of two steps. In the first step, it generates social event candidates as follows. For each tweet $t_m \in T$, it creates a social event candidate (p_1, p_2, y) for each pair of person names in t_m. Note that when p_1 or p_2 occurs multiple times in a tweet, only one candidate is generated; and in this case (p_1, p_2) refers to the closest pair. For each tweet, we index its candidates. We use x_m^i to denote the person name pairs of the i^{th} candidate, i.e., $x_m^i \cdot p_1$ and $x_m^i \cdot p_2$ correspond to p_1 and p_2, respectively, and y_m^i to denote the type label of the i^{th} candidate. For example, for the tweet "[Michael Lohan]$_p^1$ accuses [Lindsay Lohan]$_p$ of smoking crack and [Donald Trump]$_p$ Accuses [Jon Stewart]$_p$ of "Racist", [Michael Lohan]$_p^2$ again.", the generated candidates are listed below: ([Michael Lohan]$_p^1$, [Lindsay Lohan]$_p$), ([Lindsay Lohan]$_p$,[Donald Trump]$_p$), ([Lindsay Lohan]$_p$,[Jon Stewart]$_p$), ([Donald Trump]$_p$,[Jon Stewart]$_p$), ([Donald Trump]$_p$,[Michael Lohan]$_p^2$) and ([Jon Stewart]$_p$,[Michael Lohan]$_p^2$).

In the second step, our method constructs a factor graph $\mathcal{G} = (Y, F, E)$, where: $Y = \{y_m^i\}_{m,i}$ represents all the event type variables; F stands for factor vertices, consisting of $\{f_m^i(y_m^i)\}$ and $\{f_{mn}^{ij}(y_m^i, y_n^j)\}, \forall x_m^i = x_n^j$ [2]; E represents edges, consisting of edges between y_m^i and f_m^i, edges between y_m^i and f_{mn}^{ij}, and edges between y_m^j and f_{mn}^{ij}.

$\mathcal{G} = (Y, F, E)$ defines a probability distribution $P(Y|\mathcal{G}, T)$ as defined in Formula 1.

$$\begin{aligned} \ln P(Y|\mathcal{G}, T) = &- \ln Z(\mathcal{G}, T) + \\ &\sum_{m,i} \ln f_m^i(y_m^i) + \\ &\sum_{m,n,i,j} \delta_{mn}^{ij} \cdot \ln f_{mn}^{ij}(y_m^i, y_n^j) \end{aligned} \tag{1}$$

where $\delta_{mn}^{ij} = 1$ if and only if $x_m^i = x_n^j$, otherwise zero; $Z(\mathcal{G}, T)$ is the partition function as defined in Formula 2.

$$\begin{aligned} Z(\mathcal{G}, T) = &\sum_{Y} \prod_{m,i} f_m^i(y_m^i) \cdot \\ &\prod_{m,n,i,j} f_{mn}^{ij}(y_m^i, y_n^j)^{\delta_{mn}^{ij}} \end{aligned} \tag{2}$$

A factor factorizes according to a set of features, as defined in Formula 3.

$$\begin{aligned} \ln f_m^i(y_m^i) &= \sum_k \lambda_k^{(1)} \phi_k^{(1)}(y_m^i) \\ \ln f_{mn}^{ij}(y_m^i, y_n^j) &= \sum_k \lambda_k^{(2)} \phi_k^{(2)}(y_m^i, y_n^j) \end{aligned} \tag{3}$$

$\{\phi_k^{(1)}\}_{k=1}^{K_1}$ and $\{\phi_k^{(2)}\}_{k=1}^{K_2}$ are two sets of features. Each feature has a real value as its weight. $\Theta = \{\lambda_k^{(1)}\}_{k=1}^{K_1} \bigcup \{\lambda_k^{(2)}\}_{k=1}^{K_2}$ denotes the feature weight set, which is also called parameters of $\mathcal{G}$.

Our method jointly decides the values of Y. With y_m^i resolved, outputting E_m is straightforward:

$$E_m = \{(x_m^i, y_m^i) | \forall i, y_m^i \neq N\} \tag{4}$$

Training. Given a set of training data T, in which every possible social event is annotated, Θ is learnt by maximizing the log data likelihood, as defined in Formula 5.

$$\Theta^* = \arg\max_{\Theta} \ln P(Y|\Theta, T) \tag{5}$$

To solve this optimization problem, we first calculate its gradient:

$$\begin{aligned} \frac{\partial \ln P(Y|T;\Theta)}{\partial \lambda_k^1} = &\sum_{m,i} \phi_k^{(1)}(y_m^i) - \\ &\sum_{m,i} \sum_{y_m^i} p(y_m^i|T;\Theta) \phi_k^{(1)}(y_m^i) \end{aligned} \tag{6}$$

[2] We use case insensitive comparison. For example, "wilmer valderrama" and "Wilmer Valderrama" are regarded to be equal.

$$\frac{\partial \ln P(Y|T;\Theta)}{\partial \lambda_k^2} = \sum_{m,n,i,j} \delta_{mn}^{ij} \cdot \phi_k^{(2)}(y_m^i, y_n^j) - \sum_{m,n,i,j} \delta_{mn}^{ij} \sum_{y_m^i, y_n^j} p(y_m^i, y_n^j|T;\Theta) \cdot \phi_k^{(2)}(y_m^i, y_n^j) \quad (7)$$

where, the two marginal probabilities $p(y_m^i|T;\Theta)$ and $p(y_m^i, y_n^j|T;\Theta)$ are estimated using the loopy belief propagation algorithm (Murphy, Weiss, and Jordan 1999). Once we have computed the gradient, Θ^* can be figured out by standard techniques such as steepest descent, conjugate gradient and the limited-memory BFGS algorithm (L-BFGS). L-BFGS is adopted because it is particularly well suited for optimization problems with a large number of variables.

Inference. Given a set of tweets T for testing, we can construct a factor graph $\mathcal{G}$. Suppose its parameters Θ are fixed to Θ^* , the inference problem is to find the most possible assignment of Y, i.e.,

$$Y^* = \arg\max_Y \ln P(Y|\Theta^*, T) \quad (8)$$

We adopt the max-product algorithm to solve this inference problem. The max-product algorithm is nearly identical to the loopy belief propagation algorithm, except that the sums are replaced by maxima in the definitions. Note that in both the training and testing stage, we construct the factor graph in the same way as described in Section .

Features. Features $\{\phi_k^{(1)}(y_m^i)\}_{k=1}^{K_1}$ can be divided into local features and global features. Local features are related to tweet t_m, including: 1) The number of words between $x_m^i \cdot p_1$ and $x_m^i \cdot p_2$; 2) whether $x_m^i \cdot p_1$ and $x_m^i \cdot p_2$ are in the same sentence [3]; 3) the verb nearest to $x_m^i \cdot p_1$ and $x_m^i \cdot p_2$, and its position, which can be L, M or R, meaning being on the left, in the middle and on the right of the two persons, respectively; 4) whether t_m has any hash tag; 5) whether $x_m^i \cdot p_1$ appears before $x_m^i \cdot p_2$; and 6) the word before / after $x_m^i \cdot p_1$ / $x_m^i \cdot p_2$.

Global features are statistic information collected from the whole corpus (training and testing data set), including: 1) Co-occurrence times of $x_m^i \cdot p_1$ and $x_m^i \cdot p_2$ in the same tweet/sentence; and 2) content similarity between $x_m^i \cdot p_1$ and $x_m^i \cdot p_2$, as defined in Formula 9, where P_1 (P_2) refers to the person names each of which appears together with p_1 (p_2) in some tweet.

$$\text{sim}(p_1, p_2) = \frac{|P_1 \bigcap P_2|}{|P_1 \bigcup P_2|} \quad (9)$$

Features $\{\phi_k^{(2)}(y_m^i, y_n^j)\}_{k=1}^{K_2}$ include: 1) The similarity between t_m and t_n, as defined in Formula 10, where $\vec{t_m}$ ($\vec{t_n}$) is the bag-of-words vector representation of t_m (t_n) with stop words removed. The stop words are mainly from http://www.textfixer.com/resources/common-english-words.txt; 2) whether t_m and t_n fall into the same time period e.g., a day or 12 hours; 3) whether t_m and t_n have one common hash tag/verb; 4) whether re-tweet/reply relationship exists between t_m and t_n; 5) whether t_m and t_n contain the same link; and 6) whether $x_m^i \cdot p_1$ / $x_m^i \cdot p_2$ and $x_n^i \cdot p_1$ / $x_n^i \cdot p_2$ have some common neighboring word in a size three text window.

$$sim(t_m, t_n) = \frac{\vec{t_m} \cdot \vec{t_n}}{|\vec{t_m}||\vec{t_n}|} \quad (10)$$

Three things are worth mentioning here. Firstly, we map a feature with a real value r to a binary feature with value 1 if and only if $P_{\text{norm}}(x > r) \leq 0.2$. Here we assume a normal distribution of $P_{\text{norm}}(\cdot|\hat{\mu}, \hat{\sigma}^2)$ for any real value feature, as defined in Formula 11. Secondly, we conduct some pre-processing before feature extraction, e.g., removing stop words, extracting tweet meta data such as hash tags and links. Thirdly, we have converted each word to its lowercase lemma form for string comparison and feature generation. For example, "Dating" and "dated" are consider to be equal since they have the same lemma "date"; and "date" is the value of the following feature, the verb closest to "[Demi Lovato]$_p$" and "[Wilmer Valderrama]$_p$" in the tweet "$\cdots$ [Demi Lovato]$_p$ Is Dating [Wilmer Valderrama]$_p \cdots$".

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} r_i, \hat{\sigma}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(r - \hat{\mu})^2 \quad (11)$$

Experiments

Data Preparation. We sample 1,923 English tweets from November 1^{st}, 2011 to November 5^{th}, 2011. Two annotators [4] are asked to independently annotate person names and social events, if any, for every tweet. For each pair of person names in the same tweet, they are required to annotate at most one social event, and to use I or O to indicate its type (Interaction or Observation). The inter-rater agreements measured by kappa for person name recognition, whether a tweet expresses a social event involving two persons, and the type of social event are 0.78, 0.69 and 0.72, respectively. Each inconsistently annotated case is discussed by the two annotators and a consensus is reached.

In total, 5,128 different person names are annotated, and 926 tweets are obtained which have at least two person names. From the annotated tweets, we get 4,631 social event candidates, among which 212 are interaction type social events, 241 are observation type social events, and the remainder are not social events. 126 randomly selected tweets are used for development, and the remainder are used for 5-fold cross validation.

Evaluation Metrics. We adopt the widely used Precision, Recall and F1 to measure the classification performance for a particular type of social event (including None). Precision measures to what percentage the output classification labels are correct, and recall means to what percentage the classification labels in the gold-standard data set are correctly labeled, while F1 is the harmonic mean of precision and recall as defined in Formula 12. We use the average Precision, Recall and F1 to measure the overall classification performance, where the weight of each event type is proportional

[3] Two words are considered to be in the same sentence if and only if they are not separated by "!","." or "?".

[4] Two native English speakers.

to the number of events of that type.

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \quad (12)$$

Baselines. The system most similar to ours is the one proposed by Apoorv and Rambow (2010). However, it is trained on formal text and exploits advanced linguistic features such as dependency parsing. Not surprisingly it performs bad on tweets, a new genre of texts, which are short and noisy. On our data set, its average F1 is 28.5%, as compared to 35.0% F1 of the rule based system which outputs a social event if the number of words between the two person names is less than three and randomly assigns I or O to the event as its type.

Following Apoorv and Rambow (2010), we build a baseline system that first detects whether a candidate is a social event and if yes further decides its type (I or O). Similarly, the baseline adopts SVMs as classification models for both tasks. However it uses $\{\phi_k^{(1)}(y_m^i\}_{k=1}^{K_1}\}$ as its features, rather than dependency tree derived features, considering that current NLP tools cannot extract such advanced linguistic features from noise-prone tweets (Ritter et al. 2011; Liu et al. 2011). Hereafter, B_{ar} is used to denote this baseline system.

We build another system, denoted by B_{nr}, which can be considered as a variety of ours but with across tweet features $\{\phi_k^{(2)}(y_m^i, y_m^j\}_{k=1}^{K_2}\}$ ignored. Different from B_{ar}, B_{nr} conducts social event detection and classification in one step, by assigning I, O or N (meaning not a social event) to each candidate, and it uses a graphical model rather than SVMs.

The part-of-speech tagger fine tuned for tweets (Ritter et al. 2011) is used to extract verbs, and the OpenNLP toolkit is used to get lemmas. Both tools are publicly available.

Results. Table 2 reports the overall performance of our method and the baselines, in terms of Precision (P), Recall (R) and F1, respectively. It can be seen that our method significantly outperforms the baselines (with $p < 0.04$), suggesting the efficiency of conducting social event extraction collectively from multiple tweets. We have observed that largely owing to the insufficient information provided by a single tweet, the baselines fail to output any social event for a number of tweets, for which our method succeeds by leveraging information from similar tweets. As an illustrative example, consider the following two tweets in the same time scope and with similar contents: "$\cdots$ [Demi Lovato]$_p$ Is Dating [Wilmer Valderrama]$_p$ #1Dfacts $\cdots$" and "$\cdots$ Love is in the air? looks like [Demi Lovato]$_p$ and [Wilmer Valderrama]$_p$ pretty close #1Dfacts $\cdots$". For the first tweet, all systems successfully extract a social event ([Demi Lovato]$_p$,[Wilmer Valderrama]$_p$, I), owing to the strong signal provided by the verb "date". The second tweet presents a challenging case since the verb closest to the two person names is "look", which is not informative, and the rare co-occurrence of the two person names in our data set (only twice), which yields no reliable statistic evidence. The baselines output nothing for this tweet; in contrast, our method correctly outputs ([Demi Lovato]$_p$,[Wilmer Valderrama]$_p$, I), encouraged by the fact that a social event involving the same two persons is recognized in some similar tweets.

Table 2: Overall performance.

System	P	R	F1
Ours	0.78	0.76	0.77
B_{ar}	0.69	0.48	0.56
B_{nr}	0.67	0.45	0.53

Table 3: Performance for none social events.

System	P	R	F1
Ours	0.79	0.77	0.78
B_{ar}	0.68	0.49	0.57
B_{nr}	0.65	0.47	0.54

From Table 2, we can also see that B_{ar} performs slightly better than B_{nr} ($p < 0.05$). We guess this can be largely explained by the unbalanced distribution of social event candidates, i.e., 90% candidates in our data set are not social events. Owing to this data skewness problem, B_{nr} is more likely to favor "None" than B_{ar}.

Tables 3- 5 show Precision (P), Recall (R) and F1 of our method and the baselines for three types of social events, respectively. We can see that our method consistently outperforms the baselines for any type of social event ($p < 0.01$); and that B_{ar} achieves better performance than B_{nr} on every type of event ($p < 0.05$), suggesting that B_{ar} better handles data skewness problem than B_{nr}.

To study the contribution of local and global features in $\{\phi_k^{(1)}(y_m^i)\}_{k=1}^{K_1}$, we modify our method to adopt only local and global features, respectively. Note that modified methods also conduct jointly inference to leverage cross tweet information. Table 6 presents Precision, Recall and F1 of the two modified methods, from which we can see that: 1) Using only local features yields better performance than the baselines, which use both local and global features but not cross-tweet features; 2) the local features seem to be more effective than the global features. We guess one reason is that our data set is relatively small and cannot give reliable statistic information; and 3) combining all features gives the best performance.

Discussion. A great portion of errors made by our method, about 47.5%, are related to the fact that mining social event is essentially a task that requires understanding the meaning of tweets, which goes beyond the capability of the shallow features adopted by our method. As an illustrative example, consider the following tweet: "$\cdots$ Retweet if you love [Justin Bieber]$_p$, [Selena Gomez]$_p \cdots$" Owning to the extracted feature "love", the verb nearest to "[Justin Bieber]$_p$" and "[Selena Gomez]$_p$", our method incorrectly outputs an

Table 4: Performance for interaction type social events.

System	P	R	F1
Ours	0.82	0.75	0.78
B_{ar}	0.67	0.49	0.56
B_{nr}	0.63	0.41	0.50

Table 5: Performance for observation type social events.

System	P	R	F1
Ours	0.77	0.78	0.77
B_{ar}	0.71	0.46	0.56
B_{nr}	0.65	0.43	0.52

Features	P	R	F1
Local	0.67	0.58	0.62
Global	0.42	0.38	0.40

Table 6: Overall performance of our method with different feature sets.

social event ([Justin Bieber]$_p$, [Selena Gomez]$_p$, I). This error could be corrected, if we know the subject of "love" is "you". Correcting this kind of errors, ideally, requires to adapt existing NLP tools to tweets (Ritter et al. 2011), or to normalize tweets to accommodate existing NLP tools, as demonstrated by Han and Baldwin (2011). We plan to study these approaches and exploit semantic features in future.

The remaining errors are largely related to the relatively small size of training data. For example, for the tweet "[Kris Jenner]$_p$ Defend [Kim Kardashian]$_p$ http://t.co/3bEsWd6f via @yahooomg", our method does not recognize the social event ([Kris Jenner]$_p$,[Kim Kardashian]$_p$, O), for the reason that the verb "Defend" never occurs in our training data. To fix these errors, we are going to annotate more data. Particularly, we are interested in compiling a comprehensive list of social event trigger words.

Statistic information on even a small size of annotated data set shows helpful. To get more reliable global features, we can run named entity recognition system for tweets (Ritter et al. 2011) on large scale tweets. One limitation of our method is that normalization of person names is not considered. That means, no collections are established between event candidates involving variations of person names, e.g., ([Demi]$_p$,[Wilmer]$_p$) and ([Demi Lovato]$_p$,[Wilmer Valderrama]$_p$). Intuitively, normalizing person names helps our method since it allows to collect more cross-tweet information. We will leave this to our future work.

Application. Based on the mined social events, we build a preliminary application that visualizes people's social distance to setup an end to end test bed. Given two person names, denoted by p_1 and p_2, their social distance is computed according to Formula 13, where $\mathbb{I}_{\text{condition}} = 1$ if condition is satisfied, otherwise 0. Note that in general $dist(p_1, p_2) \neq dist(p_2, p_1)$ owing to the different denominators.

$$dist(p_1, p_2) = \frac{\sum_m \mathbb{I}_{(p_1,p_2,\cdot)\in E_m \bigvee (p_2,p_1,\cdot)\in E_m}}{\sum_m \mathbb{I}_{(p_1,\cdot,\cdot)\in E_m \bigvee (\cdot,p_1,\cdot)\in E_m}} \quad (13)$$

The application, as illustrated in Figure 2, consists of circles representing persons, and edges connecting person pairs involved in any social event. The size of a circle is proportional to the number of tweets mentioning the corresponded person name, while the length of an edge is proportional to the social distance between the two persons connected by the edge. Each edge is assigned with a label using the verb that co-occurs most frequently with the two person names related to the edge. We have deployed this application internally, and received positive feedback.

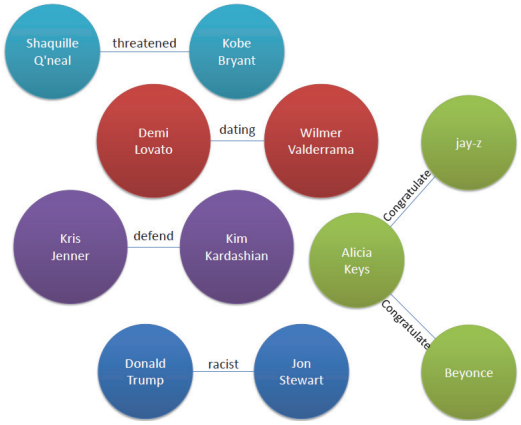

Figure 2: An application to visualize people's social distance based on the mined social events. A circle represents a person, whose size is proportional to the number of tweets containing the corresponded person name; an edge connects two persons involved in any social event, whose length is proportional to the social distance between the two corresponded persons. The label of an edge is the verb that co-occurs most frequently with the two person names connected by the edge.

Conclusions and Future Work

Tweets has become an important source of fresh events. In this work, we introduce the task of social event extraction for tweets, which can help applications such as the construction of people's social network. The main challenge is that one single tweet often provides insufficient information, rooted in the short and informal nature of tweets. We propose a factor graph based method to collectively extract social events from multiple tweets, leveraging the redundancy in tweets to make up for the lack of information in a single tweet. Particularly, our method first generates social event candidates, and then jointly assigns an event type label (I, B or N) to every candidate. Extensive experiments on a manually labeled data set demonstrate the effectiveness of our method, which outperforms all baselines. Based on the mined social events, we build an application that visualizes people's social closeness, as an end to end test bed of our method.

We plan to explore three directions in future: 1) Developing advanced tweet normalization technologies and related tools to extract semantic features; 2) utilizing external knowledge, such as unlabeled tweets and common social event triggers to overcome data spareness; and 3) normalizing person names to aggregate additional cross-tweet information.

References

Agarwal, A., and Rambow, O. 2010. Automatic detection and classification of social events. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, 1024–1034. Cambridge, MA: Association for Computational Linguistics.

Benson, E.; Haghighi, A.; and Barzilay, R. 2011. Event discovery in social media feeds. In *ACL*, 389–398. Portland, Oregon, USA: Association for Computational Linguistics.

Bethard, S., and Martin, J. H. 2006. Identification of event mentions and their semantic class. In *EMNLP*, 146–154. Sydney, Australia: Association for Computational Linguistics.

Han, B., and Baldwin, T. 2011. Lexical normalisation of short text messages: Makn sens a #twitter. In *ACL*, 368–378.

Harabagiu, S.; Moldovan, D.; Clark, C.; Bowden, M.; Hickl, A.; and Wang, P. 2005. Nyu 's english ace 2005 system description. In *ACE 05 Evaluation Workshop*.

Liao, S., and Grishman, R. 2010. Using document level cross-event inference to improve event extraction. ACL '10, 789–797. Morristown, NJ, USA: Association for Computational Linguistics.

Liu, X.; Li, K.; Zhou, M.; and Xiong, Z. 2011. Collective semantic role labeling for tweets with clustering. In *IJCAI*, 1832–1837.

Llorens, H.; Saquete, E.; and Navarro-Colorado, B. 2010. Timeml events recognition and classification: Learning crf models with semantic roles. In *Coling 2010*, 725–733.

Murphy, K. P.; Weiss, Y.; and Jordan, M. I. 1999. Loopy belief propagation for approximate inference: An empirical study. In *In Proceedings of Uncertainty in AI*, 467–475.

Ritter, A.; Clark, S.; Mausam; and Etzioni, O. 2011. Named entity recognition in tweets: An experimental study. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 1524–1534. Edinburgh, Scotland, UK.: Association for Computational Linguistics.

Sakaki, T.; Okazaki, M.; and Matsuo, Y. 2010. Earthquake shakes Twitter users: real-time event detection by social sensors. WWW '10, 851–860.

Sankaranarayanan, J.; Samet, H.; Teitler, B. E.; Lieberman, M. D.; and Sperling, J. 2009. TwitterStand: news in tweets. In *SIGSPATIAL*, GIS '09, 42–51. New York, NY, USA: ACM Press.

Yu, L.-C.; Chan, C.-L.; Wu, C.-H.; and Lin, C.-C. 2009. Mining association language patterns for negative life event classification. In *ACL-IJCNLP 2009*, 201–204.