Linking Named Entities in Tweets with Knowledge Base via User Interest Modeling

Wei Shen†, Jianyong Wang†, Ping Luo‡, Min Wang♯
†Tsinghua National Laboratory for Information Science and Technology (TNList),
Department of Computer Science and Technology, Tsinghua University, Beijing, China
‡HP Labs China, Beijing, China; ♯Google Research, Mountain View, USA
†chen-wei09@mails.tsinghua.edu.cn, jianyong@tsinghua.edu.cn
‡ping.luo@hp.com; ♯minwang@google.com

ABSTRACT

Twitter has become an increasingly important source of information, with more than 400 million tweets posted per day. The task to link the named entity mentions detected from tweets with the corresponding real world entities in the knowledge base is called tweet entity linking. This task is of practical importance and can facilitate many different tasks, such as personalized recommendation and user interest discovery. The tweet entity linking task is challenging due to the noisy, short, and informal nature of tweets. Previous methods focus on linking entities in Web documents, and largely rely on the context around the entity mention and the topical coherence between entities in the document. However, these methods cannot be effectively applied to the tweet entity linking task due to the insufficient context information contained in a tweet. In this paper, we propose KAURI, a graph-based framework to collectively linK all the nAmed entity mentions in all tweets posted by a user via modeling the UseR's topics of Interest. Our assumption is that each user has an underlying topic interest distribution over various named entities. KAURI integrates the intra-tweet local information with the inter-tweet user interest information into a unified graph-based framework. We extensively evaluated the performance of KAURI over manually annotated tweet corpus, and the experimental results show that KAURI significantly outperforms the baseline methods in terms of accuracy, and KAURI is efficient and scales well to tweet stream.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Storage and Retrieval—*Information Search and Retrieval*

Keywords

Tweet entity linking; Knowledge base; User interest modeling

KDD'13, August 11–14, 2013, Chicago, Illinois, USA.

1. INTRODUCTION

Twitter, a popular micro-blogging platform, has been growing nearly exponentially recently. With more than 400 million tweets posted daily, Twitter offers an abundant information source of personal data. On Twitter, users can publish and share information in short posts or tweets (with a limitation of 140 characters) about topics ranging from daily life to news events and other interests. This huge collection of tweets embody invaluable information and knowledge about named entities, which appear in tweets frequently. However, as being free text form, named entity mentions in tweets are potentially ambiguous: the same textual name may refer to several different real world entities. As the example shown in Figure 1, the entity mention "Sun" in tweet t_2 can refer to the star at the center of the Solar System, a multinational computer company, a fictional character named "Sun-Hwa Kwon" or many other entities named "Sun".

Recently, the success of Wikipedia and the proposed vision of Linked Open Data (LOD) [2] have facilitated the automated construction of large scale general-purpose machine-understanding knowledge base about the world's entities, their semantic categories and the relationships between them. Such kind of notable endeavors include DBpedia [1], YAGO [21], Freebase [3], and Probase [23]. Bridging these knowledge bases with the collection of tweets is beneficial for exploiting and understanding this huge corpus of valuable personal data on the Web, and also helps to populate and enrich the existing knowledge bases [18]. We define tweet entity linking as the task to link the textual named entity mentions detected from tweets with their mapping entities existing in the knowledge base. If the matching entity of certain entity mention does not exist in the knowledge base, NIL (denoting an unlinkable mention) should be returned for this entity mention.

The tweet entity linking task is of practical importance and can be beneficial for various applications of Twitter mining, such as news and trend detection, commercial brand management, and personalized recommendation [22, 5]. For instance, detecting and linking named entities Twitter users mention in their tweets are significantly helpful for the task of user interest discovery [16, 24], which further allows for recommending and searching Twitter users based on their topics of interest [22]. In addition, there are also applications that utilize user interest to recommend and rank tweets for Twitter users [5, 6]. As another example, the needs to collect opinions or information about some products, celebrities

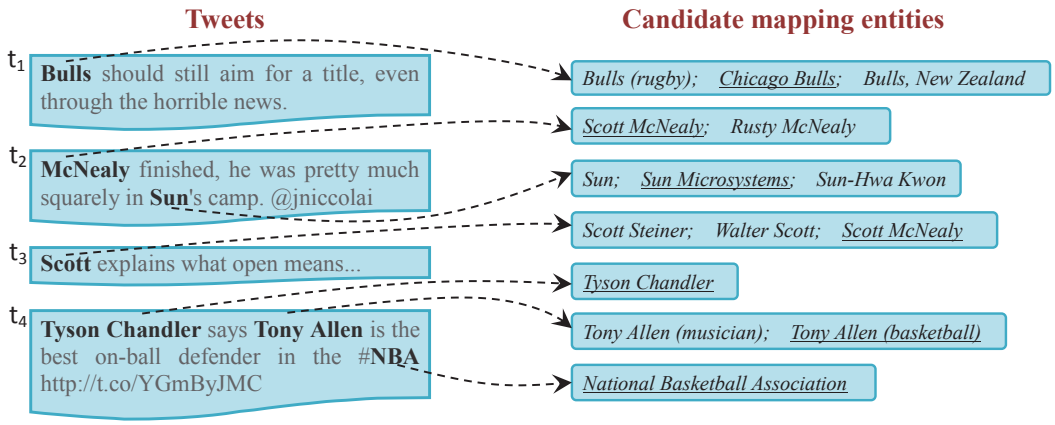

Figure 1: An illustration of the tweet entity linking task. Named entity mentions detected in tweets are in bold; candidate mapping entities for each entity mention are ranked by their prior probabilities in decreasing order; true mapping entities are underlined.

or some other named entities from Twitter also require the process of linking entities in tweets with knowledge base.

In recent years, considerable progresses have been made in linking named entities in Web documents with a knowledge base [4, 7, 12, 11, 9, 20]. These previous methods mainly focus on entity mentions detected from Web documents, and the main idea is to define a context similarity measure between the text around the entity mention and the document associated with the entity in the knowledge base. However, for the tweet entity linking task, these context similarity based methods may result in unsatisfactory performance owing to the informal language usage and the limited length of each individual tweet, which is confirmed by our experiments. For the example in Figure 1, the contexts in tweets t_1 and t_3 are particularly short and ambiguous, which lack sufficient context information in order to link mentions "Bulls" in t_1 and "Scott" in t_3 correctly.

Besides leveraging context similarity, many previous approaches assume that entities mentioned from a single document are likely to be topically coherent, and they consider the interdependence between different entity assignments within a document to jointly link these mentions [7, 12, 11, 9, 20]. Despite that each individual tweet satisfies this assumption, these previous methods still may not work well for the tweet entity linking task due to the limited number of entities appearing within each single tweet, which is also confirmed by our experiments. For the example in Figure 1, there is only one entity detected in tweet t_1 and tweet t_3, respectively. Therefore, we cannot leverage the topical coherence between different entities within the single tweet to help link these mentions.

To solve the problem of insufficient information in a single tweet, many Twitter mining approaches simply gather the tweets published by an individual user into a big document for processing [22, 5]. However, this simple method cannot be applied to the tweet entity linking task because this aggregated document violates the assumption that entities mentioned from a single document are topically coherent. As the example shown in Figure 1, tweets t_1 and t_4 talk about the NBA teams and basketball players, while tweets t_2 and t_3 talk about IT companies and businessmen which are totally unrelated to the topics in tweets t_1 and t_4. To sum up, it can be seen that the tweet entity linking task is challenging and different from the task of linking entities in Web documents.

Given a knowledge base about the world's entities, for the named entity mentions detected from all tweets of a given Twitter user, our goal is to collectively link them to their mapping entities existing in the knowledge base by leveraging the following two categories of information:

Intra-tweet local information. Intuitively, a candidate mapping entity is "good" with respect to some entity mention within a single tweet if it has three key properties: (1) the prior probability (its definition is given in Formula 2) of the entity being mentioned is high; (2) the similarity between the context around the entity mention in the tweet and the context associated with the candidate mapping entity is high; (3) the candidate mapping entity is topically coherent with the mapping entities of the other entity mentions (if the tweet has) within the same tweet. Therefore, we could leverage the three intra-tweet local features to link the entity mentions in tweets. For example, in Figure 1, for the entity mentions in tweet t_2, we can easily figure out the mention "McNealy" refers to *Scott McNealy*, the CEO of Sun Microsystems, and the mention "Sun" refers to the company entity *Sun Microsystems*, as the prior probability of the entity *Scott McNealy* for the mention "McNealy" is very high, and the mapping entities *Scott McNealy* and *Sun Microsystems* are highly topically coherent. Similarly, for the entity mentions in tweet t_4, knowing the mention "Tyson Chandler" refers to the NBA player *Tyson Chandler* and the mention "NBA" refers to *National Basketball Association*, it could help us link the mention "Tony Allen" to the NBA player *Tony Allen (basketball)* rather than the musician that has higher prior probability compared with the NBA player entity.

Inter-tweet user interest information. As stated above, a single tweet may be too short and noisy to provide sufficient context information for linking the entity mentions with their corresponding mapping entities correctly (e.g., the entity mentions "Bulls" in t_1 and "Scott" in t_3). To deal with this problem, we exploit the user interest information across tweets published by one individual user by modeling the user's topics of interest. We assume each user has an underlying topic interest distribution over various topics of named entities. If a candidate mapping entity is highly

topically related to entities the user is interested in, we assume this user is likely to be interested in this candidate entity as well. For the example in Figure 1, since the NBA players *Tyson Chandler*, *Tony Allen (basketball)* and the entity *National Basketball Association* are deemed likely to be the entities the user mentioned in tweet t_4, we consider this user is interested in these entities. Henceforth, with regard to the entity mention "Bulls" in t_1, we predict that this user is more likely to be interested in the NBA team *Chicago Bulls* rather than the South African rugby union team *Bulls (rugby)* and the small town *Bulls, New Zealand*, as the entity *Chicago Bulls* is highly topically related to the entities mentioned in t_4 that the user is interested in. Similarly, since the CEO *Scott McNealy* is likely to be the entity user mentioned in tweet t_2, among the candidate entities for the mention "Scott" in t_3, the user is more likely to be interested in the same entity *Scott McNealy*, rather than the wrestler *Scott Steiner* and the novelist *Walter Scott*.

Contributions. The main contributions of this paper are summarized as follows.

- We investigate the tweet entity linking task, a new and increasingly important issue due to the proliferation of tweet data and its broad applications.
- To the best of our knowledge, KAURI is the first framework to collectively link all the named entity mentions in all tweets published by one user with a knowledge base via modeling this user's topics of interest.
- KAURI is a novel graph-based framework that unifies two categories of information (i.e., intra-tweet local information and inter-tweet user interest information).
- To verify the effectiveness and efficiency of KAURI, we evaluated KAURI over manually annotated tweet corpus, and the experimental results show that KAURI significantly outperforms the baseline methods in terms of accuracy, and our framework KAURI is efficient and scales well to tweet stream.

2. PRELIMINARIES AND NOTATIONS

In this section, we begin by describing the tweet and the task of tweet entity linking. Next, we introduce the generation of candidate mapping entities for each entity mention.

Tweet. A tweet is a short textual post comprising a maximum of 140 characters published by a Twitter user. In Figure 1, there are four tweets published by a Twitter user. When a user wants to refer to another user in his tweet, he needs to add the '@' symbol before the referred user name to *mention* that user according to Twitter standards. Here, the meaning of *mention* is different from the meaning of entity mention. From now on, we call it *Twitter mention*. In tweet t_2 in Figure 1, "@jniccolai" is a *Twitter mention* of the Twitter user whose name is "jniccolai". Word prefixed with the '#' symbol, like "#NBA" in tweet t_4 in Figure 1, is hashtag, usually referring to a topic. In addition, there are some shortened URLs in tweets, such as "http://t.co/YGmByJMC" in tweet t_4 in Figure 1. Twitter users often mention named entities in their tweets. As shown in the example in Figure 1, there are totally seven named entities in these four tweets. According to the statistics of the manually annotated tweet corpus used in our experiments, about 45.08% of tweets have at least one named entity. Recently, several methods [14, 13] have been proposed to address the problem of named entity recognition for tweets. However, this problem is orthogonal to our task because our task takes the recognized named entity mentions as input.

Tweet entity linking. According to the task setting, we take (1) a collection of tweets posted by some Twitter user (denoted by T), and (2) named entity mentions recognized in the given tweets T (denoted by M) as input. Let $|T|$ be the number of tweets in T. We use $1 \leq i \leq |T|$ to index the tweet in T, and the tweet with index i is denoted by t_i. The set of named entity mentions recognized in each tweet $t_i \in T$ is denoted by M^i, and $M^i \subset M$. Let $|M^i|$ be the number of entity mentions in the set M^i. We use $1 \leq j \leq |M^i|$ to index the entity mention in M^i, and the entity mention with index j in mention set M^i is denoted by m^i_j. Each entity mention m^i_j is a token sequence of a named entity that is potentially linked with an entity in the knowledge base. Here, we formally state the tweet entity linking task as follows:

DEFINITION 1 (**Tweet entity linking**). *Given the tweet collection T posted by some Twitter user and named entity mention set M, the goal is to identify the mapping entity e^i_j in the knowledge base for each entity mention $m^i_j \in M$. If the mapping entity of entity mention m^i_j does not exist in the knowledge base, we should return NIL.*

In this paper, the knowledge base we adopt is YAGO [21], an open-domain ontology combining Wikipedia and WordNet with high coverage and quality. YAGO uses unique canonical strings from Wikipedia as the entity names. Currently, YAGO contains over one million entities. For illustration, we show a running example of the tweet entity linking task.

Example 1. In this example, we just consider the tweet entity linking task for the tweets t_1 and t_4 shown in Figure 1 for the purpose of simplicity. There are four entity mentions (i.e., "Bulls" in t_1, "Tyson Chandler", "Tony Allen", and "NBA" in t_4), which need to be linked in this example. The candidate mapping entities for each entity mention are shown using arrows in Figure 1. For each entity mention, we should output its true mapping entity, which is underlined in Figure 1.

Candidate mapping entity. For each entity mention $m^i_j \in M$, its mapping entity e^i_j should be the entity that may be referred by the token sequence of m^i_j. Therefore, we firstly identify all the entities that may be referred by m^i_j, and denote this set of entities as the candidate mapping entity set R^i_j for the entity mention m^i_j. To identify R^i_j for each m^i_j, we need to build a dictionary D that contains vast amount of information about various surface forms of the named entities, like name variations, abbreviations, spelling variations, nicknames, etc. The dictionary D is a $\langle$key, value$\rangle$ mapping, where the column of the key K is a list of surface forms and the column of the mapping value $K.value$ is the set of named entities which can be referred by the key K. We construct the dictionary D by leveraging the four structures of Wikipedia: **Entity page**, **Redirect page**, **Disambiguation page** and **Hyperlink in Wikipedia article**. The detailed construction method is introduced in [19, 20]. A part of the dictionary D is shown in Table 1.

For each entity mention $m^i_j \in M$, we look up the dictionary D and search for m^i_j in the column of key K. If a hit is found, i.e., $m^i_j \in K$, we add the set of entities $m^i_j.value$ to the candidate mapping entity set R^i_j. We denote the size of R^i_j as $|R^i_j|$, and use $1 \leq q \leq |R^i_j|$ to index the candidate entity in R^i_j. The candidate mapping entity with index q in R^i_j is denoted by $r^i_{j,q}$. Let $R = \{R^i_j | m^i_j \in M\}$ be the set of

Table 1: A part of the dictionary D

K (Surface form)	$K.value$ (Mapping entity)
IBM	*IBM*
NBA	*National Basketball Association*
Sun	*Sun* *Sun Microsystems* *Sun-Hwa Kwon* ...
Bill Hewlett	*William Reddington Hewlett*

candidate mapping entity sets for all the entity mentions in M. For the example in Figure 1, for each entity mention, we show its candidate mapping entity set generated from the dictionary D using arrow, and the true mapping entity is underlined. It is noted that in Figure 1 each candidate entity has the name which is the unique canonical string for that entity in Wikipedia, and the size of the candidate mapping entity set for most of the entity mentions (5 out of 7) is larger than 1. Henceforth, for each entity mention, we have to figure out which entity in its candidate mapping entity set is the mostly proper link for it, which will be addressed in the next section.

3. TWEET ENTITY LINKING

To address the tweet entity linking task, we propose a novel graph-based framework KAURI to collectively link all the entity mentions in all tweets of one user by modeling this user's topics of interest. We have the following three assumptions:

Assumption 1. Each Twitter user has an underlying topic interest distribution over various topics of named entities. That is to say, for each Twitter user, each named entity is associated with an interest score, indicating the strength of the user's interest in it.

Assumption 2. If some named entity is mentioned by a user in his tweet, that user is likely to be interested in this named entity.

Assumption 3. If one named entity is highly topically related to the entities that a user is interested in, that user is likely to be interested in this named entity as well.

Based on these three assumptions, we model the tweet entity linking problem into a graph-based interest propagation problem. Firstly, for each Twitter user, we construct a graph of which the structure encodes the interdependence information between different named entities, which will be introduced in Section 3.1. Next, we estimate the initial interest score for each named entity in the graph based on the intra-tweet local information according to Assumption 2, which will be depicted in Section 3.2. Lastly, we propose a graph-based algorithm, called *user interest propagation algorithm*, to propagate the user interest score among different named entities across tweets using the interdependence structure of the constructed graph according to Assumption 3, which will be illustrated in Section 3.3. Thus, our proposed framework KAURI integrates the intra-tweet local information with the inter-tweet user interest information into a unified graph-based model via modeling the user's interest.

3.1 Graph construction

We propose to turn the tweet entity linking problem into a graph-based interest propagation problem. Thus, we firstly

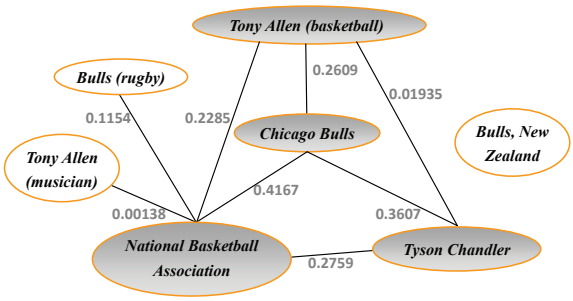

Figure 2: The graph constructed for Example 1

introduce how to construct the graph that captures the interdependence information between different named entities.

For one Twitter user, given the candidate mapping entity set R, we construct a graph $G = (V, A, W)$ to represent all the interdependence information between these candidate mapping entities in R, where V denotes the set of nodes, A is the set of edges, and $W : A \rightarrow \mathbf{R}^+$ is the weight function which assigns a positive weight to each edge in A. The node set V consists of the nodes which come from all the candidate mapping entities in R. One node in the graph represents a candidate mapping entity $r^i_{j,q}$ and it is associated with an interest score $s^i_{j,q}$ indicating the strength of the user's interest in it, as well as an initial interest score $p^i_{j,q}$ estimated from the intra-tweet local information. For each pair of nodes $\langle r^i_{j,q}, r^{i'}_{j',q'} \rangle$ in the graph, if the weight between them is larger than 0, we add an undirected edge $(r^i_{j,q}, r^{i'}_{j',q'})$ to A, with $W(r^i_{j,q}, r^{i'}_{j',q'})$ indicating the strength of interdependence between them. If the two candidate entities of the node pair $\langle r^i_{j,q}, r^{i'}_{j',q'} \rangle$ are candidates of the same entity mention (i.e., $i = i'$ and $j = j'$), we disallow the propagation between them and do not add an edge between them, since only one of them may be the true mapping entity.

According to Assumption 3, if two candidate entities are more interdependent or topically related to each other, the interest score should be propagated in a larger extent to each other. Thus, the edge weight is defined as the topical relatedness between the two candidate entities. In our framework, since all the candidate entities are Wikipedia articles, we adopt the Wikipedia Link-based Measure (WLM) described in [17] to calculate the topical relatedness between Wikipedia articles. The WLM is based on the Wikipedia's hyperlink structure. The basic idea of this measure is that two Wikipedia articles are considered to be topically related if there are many Wikipedia articles that link to both. Given two Wikipedia articles u_1 and u_2, we define the topical relatedness between them as follows:

$$TR(u_1, u_2) = 1 - \frac{log(max(|U_1|, |U_2|)) - log(|U_1 \bigcap U_2|)}{log(|WP|) - log(min(|U_1|, |U_2|))} \quad (1)$$

where U_1 and U_2 are the sets of Wikipedia articles that link to u_1 and u_2 respectively, and WP is the set of all articles in Wikipedia. This definition gives higher value to more topically related article pair and the value of $TR(u_1, u_2)$ is varied from 0.0 to 1.0.

In Figure 2, we show the constructed graph for Example 1. Each node in Figure 2 represents a candidate mapping entity for some entity mention in Example 1. Each edge in Figure 2 indicates the topical relatedness relationship between

candidate entities, and the value shown beside the edge is the edge weight calculated using Formula 1. For illustration, each true mapping entity is shaded in the figure. From Figure 2, we can see that there is a strong topical relatedness relationship between any two of the true mapping entities, which demonstrates that the measure of topical relatedness in Formula 1 effectively captures the interdependence relationship between them. On the contrary, between the true mapping entity and false mapping entity, the topic relatedness is either none (e.g., between *Tyson Chandler* and *Bulls, New Zealand*), or a little weak (e.g., between *National Basketball Association* and *Tony Allen (musician)*).

It is noted that in our framework, the edge weight can be measured by other features or combination of them, such as the temporal and spatial information embedded in tweets, or the content of the Web page corresponding to the shortened URLs in tweets. This gives flexibility to our framework in an efficient and simple way.

3.2 Initial interest score estimation

Each node $r^i_{j,q}$ in the graph is associated with an initial interest score $p^i_{j,q}$. Based on Assumption 2, we can see that the more likely the candidate entity is the true mapping entity, the more interested the user is in this candidate entity. Thus, given tweet t_i where entity mention m^i_j appears, we estimate the initial interest score $p^i_{j,q}$ for the candidate entity $r^i_{j,q}$ based on the intra-tweet local information in t_i. Specifically, we leverage the following three intra-tweet local features in t_i: (1) the prior probability of the candidate entity $r^i_{j,q}$ being mentioned; (2) the similarity between the context associated with the candidate entity $r^i_{j,q}$ and the context around the entity mention m^i_j in tweet t_i; (3) the topical coherence between candidate entity $r^i_{j,q}$ and the mapping entities for the other entity mentions within tweet t_i.

Prior probability: We have the observation that each candidate mapping entity $r^i_{j,q} \in R^i_j$ having the same surface form m^i_j has different popularity, and some entities are very obscure and rare for the given surface form m^i_j. For the example in Figure 1, for the entity mention "Sun", the candidate entity *Sun-Hwa Kwon*, a fictional character on the ABC television series Lost, is much rarer than the candidate entity *Sun*, the star at the center of the Solar System, and in most cases when people mention "Sun", they mean the star rather than the fictional character simply known as "Sun". We formalize this observation via taking advantage of the count information from Wikipedia, and define the *prior probability* $Pp(r^i_{j,q})$ for each candidate mapping entity $r^i_{j,q} \in R^i_j$ with respect to the entity mention m^i_j as the proportion of the links with the surface form m^i_j as the anchor text which point to the candidate mapping entity $r^i_{j,q}$:

$$Pp(r^i_{j,q}) = \frac{count(r^i_{j,q})}{\sum_{c=1}^{|R^i_j|} count(r^i_{j,c})} \quad (2)$$

where $count(r^i_{j,q})$ is the number of links which point to entity $r^i_{j,q}$ and have the surface form m^i_j as the anchor text. For example, in Figure 1, the candidate mapping entities for each entity mention are ranked by their *prior probabilities* shown on the right of the figure in decreasing order. It can be seen that the notion of *prior probability* suitably expresses the popularity of the candidate mapping entity being mentioned given a surface form.

Context similarity: To calculate the *context similarity* feature $Sim(r^i_{j,q})$ for each candidate mapping entity $r^i_{j,q}$, we compare the context associated with the candidate entity $r^i_{j,q}$ with the context around the entity mention m^i_j in tweet t_i. For each candidate entity $r^i_{j,q}$, we collect each occurrence of entity $r^i_{j,q}$ in the Wikipedia page corpus, and extract the context of $r^i_{j,q}$ in a short window to compose the bag of words vector for $r^i_{j,q}$. To get the bag of words vector for the entity mention m^i_j in tweet t_i, we firstly preprocess the tweet (i.e., remove the *Twitter mentions*, '#' symbols, URLs, and all punctuation symbols). Then we extract the short window of words around each occurrence of m^i_j in tweet t_i to compose the bag of words vector for entity mention m^i_j. Finally, we measure the cosine similarity of these two vectors weighted by TF-IDF as the *context similarity* feature $Sim(r^i_{j,q})$ for each candidate entity $r^i_{j,q} \in R^i_j$.

Topical coherence: Recall that a single tweet satisfies the assumption that entities mentioned in it are likely to be topically coherent. Here, we exploit this topical coherence between entities in the single tweet to define the third feature *topical coherence* for each candidate mapping entity. The *topical coherence* feature $Coh(r^i_{j,q})$ for each candidate mapping entity $r^i_{j,q}$ is measured as the topical coherence between the candidate mapping entity $r^i_{j,q}$ and the mapping entities for the other entity mentions within tweet t_i. In this paper, we measure the topical coherence between entities as the topical relatedness between them defined in Formula 1. Thus, given tweet t_i where the entity mention m^i_j appears, we formally define the notion of *topical coherence* $Coh(r^i_{j,q})$ for each candidate mapping entity $r^i_{j,q} \in R^i_j$ as:

$$Coh(r^i_{j,q}) = \frac{1}{|M^i| - 1} \sum_{c=1, c\neq j}^{|M^i|} TR(r^i_{j,q}, e^i_c) \quad (3)$$

where $|M^i|$ is the number of entity mentions in tweet t_i, e^i_c is the mapping entity for the entity mention $m^i_c \in M^i$, and $TR(r^i_{j,q}, e^i_c)$ is the topical relatedness between the candidate entity $r^i_{j,q}$ and the mapping entity e^i_c for other entity mention $m^i_c \in M^i$ (where $c \neq j$) which can be calculated by Formula 1. However, the mapping entity e^i_c for entity mention m^i_c is unknown to us and needs to be assigned in this task. To solve this problem, we utilize a practical and effective greedy algorithm, called *iterative substitution algorithm*, to jointly optimize the identification of the mapping entities for the entity mentions in the single tweet by leveraging the intra-tweet local information in this tweet. We proposed this *iterative substitution algorithm* to deal with the list linking task [19], and this greedy algorithm starts with a good guess of the initial mapping entity set, and then iteratively improves the quality of the mapping entity set until the algorithm converges. For the details about this algorithm, you could refer to [19].

Initial interest score: Based on these three intra-tweet local features illustrated above, we calculate the initial interest score $p^i_{j,q}$ for each candidate mapping entity $r^i_{j,q} \in R^i_j$ as the weighted sum of *prior probability*, *context similarity* and *topical coherence* as follows:

$$p^i_{j,q} = \alpha * Pp(r^i_{j,q}) + \beta * Sim(r^i_{j,q}) + \gamma * Coh(r^i_{j,q}) \quad (4)$$

where $\alpha + \beta + \gamma = 1$, and α, β and γ are weight factors that give different weights for different feature values in the calculation of initial interest score $p^i_{j,q}$. Here, we denote $\overrightarrow{w}$ as the weight vector, and $\overrightarrow{w} = \langle \alpha, \beta, \gamma \rangle$. In order to automatically learn the weight vector $\overrightarrow{w}$, we use a max-

Table 2: The initial interest scores for candidate mapping entities

$r^i_{j,q}$	*Bulls (rugby)*	*Chicago Bulls*
$p^i_{j,q}$	0.13	0.0492
$r^i_{j,q}$	*Bulls, New Zealand*	*Tyson Chandler*
$p^i_{j,q}$	0.0208	0.318
$r^i_{j,q}$	*Tony Allen (musician)*	*Tony Allen (basketball)*
$p^i_{j,q}$	0.145	0.155
$r^i_{j,q}$	*National Basketball Association*	
$p^i_{j,q}$	0.402	

Table 3: The final interest scores for candidate mapping entities

$r^i_{j,q}$	*Bulls (rugby)*	*Chicago Bulls*
$s^i_{j,q}$	0.0624	0.189
$r^i_{j,q}$	*Bulls, New Zealand*	*Tyson Chandler*
$s^i_{j,q}$	0.00682	0.194
$r^i_{j,q}$	*Tony Allen (musician)*	*Tony Allen (basketball)*
$s^i_{j,q}$	0.0476	0.122
$r^i_{j,q}$	*National Basketball Association*	
$s^i_{j,q}$	0.297	

margin technique based on the training data set, which is similar to the max-margin technique utilized in [19].

Using the above method, we can compute the initial interest score $p^i_{j,q}$ for each candidate mapping entity $r^i_{j,q}$ in Example 1 based on its intra-tweet local information according to Formula 4. In Table 2, we show the result of the initial interest score for each candidate mapping entity in Example 1. From Table 2, we can see that the notion of initial interest score effectively captures the intra-tweet local information. Specifically, despite that the *prior probability* of the candidate entity *Tony Allen (basketball)* (i.e., 0.278) is much lower than the *prior probability* of the candidate entity *Tony Allen (musician)* (i.e., 0.722) with respect to the entity mention "Tony Allen" in tweet t_4, the initial interest score of the true mapping entity *Tony Allen (basketball)* (i.e., 0.155) is higher than the initial interest score of the candidate entity *Tony Allen (musician)* (i.e., 0.145). On the other hand, for tweet t_1 which lacks sufficient intra-tweet context information to link entity mention "Bulls", the definition of initial interest score is unable to give the higher score to the true mapping entity (i.e., *Chicago Bulls*) compared with the candidate entity (i.e., *Bulls (rugby)*), since the initial interest score only encodes the intra-tweet local information. Thus, it is very necessary to consider the inter-tweet user interest information for linking this mention.

3.3 User interest propagation algorithm

In this section, we introduce the *user interest propagation algorithm*, a graph-based algorithm to propagate the interest score among different candidate mapping entities across tweets using the interdependence structure of the constructed graph G. For each Twitter user, we construct the graph $G = (V, A, W)$, and let $|V|$ be the number of nodes in V. We use $1 \leq k \leq |V|$ to index the node in V, and the node with index k is denoted by v_k. For each node v_k in graph G, we can compute the initial interest score p_k according to Formula 4. Then we can easily normalize it by the sum of the initial interest scores of all nodes in graph G:

$$np_k = \frac{p_k}{\sum_{c=1}^{|V|} p_c} \tag{5}$$

The edge between each pair of nodes $\langle v_k, v_{k'} \rangle$ in the graph G provides a path to propagate interest between these two nodes, and the edge weight $W(v_k, v_{k'})$ indicates the strength of the interest propagation between each other. Therefore, a node will propagate more interest score to the node which has a stronger edge with it. However, we cannot directly use the edge weight $W(v_k, v_{k'})$ computed using Formula 1 as the interest propagation strength because it is unnormalized raw weight. Formally, we define the interest propagation strength $NW(v_k, v_{k'})$ from node v_k to node $v_{k'}$ as:

$$NW(v_k, v_{k'}) = \frac{W(v_k, v_{k'})}{\sum_{v_c \in V_{v_k}} W(v_k, v_c)} \tag{6}$$

where V_{v_k} is the set of nodes each of which has an edge with node v_k. For the graph shown in Figure 2, we can compute the interest propagation strength between nodes using Formula 6. For example, the interest propagation strength from node *National Basketball Association* to node *Bulls (rugby)* is 0.111, while the interest propagation strength from node *Bulls (rugby)* to node *National Basketball Association* is 1.0.

Suppose that we want to compute the final interest score vector $\overrightarrow{\mathbf{s}} = (s_1, ..., s_k, ..., s_{|V|})$ where s_k is the final interest score for node v_k to indicate the strength of the user's interest in it. The final interest score s_k of each node is decided by the combination of its initial interest score and the propagated score of other related nodes, and the final interest score vector $\overrightarrow{\mathbf{s}}$ can be computed efficiently by matrix multiplication. Let $\overrightarrow{\mathbf{p}}$ be the initial interest score vector and $\overrightarrow{\mathbf{p}} = (np_1, ..., np_k, ..., np_{|V|})$, where np_k is the normalized initial interest score for node v_k which can be calculated using Formula 5. Let B be a $|V| \times |V|$ interest propagation strength matrix, where $b_{k',k}$ is the interest propagation strength from node v_k to node $v_{k'}$ and $b_{k',k} = NW(v_k, v_{k'})$, which can be calculated using Formula 6. Thus, the matrix B is a column-normalized matrix. Formally, we define the computation of the final interest score vector $\overrightarrow{\mathbf{s}}$ as:

$$\overrightarrow{\mathbf{s}} = \lambda \overrightarrow{\mathbf{p}} + (1 - \lambda) B \overrightarrow{\mathbf{s}} \tag{7}$$

where $\lambda \in [0, 1]$ is a parameter that balances the relative importance of the two parts (i.e., the initial interest score $\overrightarrow{\mathbf{p}}$ and the propagated score of other related nodes $B\overrightarrow{\mathbf{s}}$). In the experiments, we show that the performance of our framework is insensitive to the parameter λ. From this definition, we can see that the final interest score vector $\overrightarrow{\mathbf{s}}$ combines evidences from the initial interest score vector $\overrightarrow{\mathbf{p}}$, the interdependence information between nodes B and the interest propagation process encoded as the product of B and $\overrightarrow{\mathbf{s}}$. As Formula 7 is a recursive form, in order to calculate the final interest score vector $\overrightarrow{\mathbf{s}}$, our *user interest propagation algorithm* firstly initializes the final interest score vector $\overrightarrow{\mathbf{s}}$ by setting $\overrightarrow{\mathbf{s}} = \overrightarrow{\mathbf{p}}$ and then applies Formula 7 iteratively until $\overrightarrow{\mathbf{s}}$ stabilizes within some threshold. From Formula 7, we can see that our *user interest propagation algorithm* is a little similar to the topic-sensitive PageRank algorithm [10] and the PageRank-like collective inference algorithm used in [9]. Since the interest propagation strength matrix B is square, stochastic, irreducible and aperiodic, our *user interest propagation algorithm* is guaranteed to **converge** according to [10].

Using the *user interest propagation algorithm* introduced above, we can compute the final interest score $s^i_{j,q}$ for each candidate mapping entity $r^i_{j,q}$ in Example 1 based on the

graph shown in Figure 2 and the initial interest scores shown in Table 2. We show the results in Table 3. From Table 3, we can see that the *user interest propagation algorithm* effectively gives the highest final interest score to the true mapping entity among all the candidate mapping entities for each entity mention in Example 1. In addition, for the true mapping entity *Chicago Bulls*, of which the initial interest score is not the highest among all the candidate mapping entities with regard to the entity mention "Bulls" shown in Table 2, the *user interest propagation algorithm* can effectively leverage the inter-tweet user interest information to give it the highest final interest score among all the candidate entities.

Output of the tweet entity linking task: For each entity mention $m_j^i \in M$ and its corresponding candidate mapping entity set R_j^i, we compute the final interest score $s_{j,q}^i$ for each candidate mapping entity $r_{j,q}^i \in R_j^i$ using our framework introduced above. Then we identify the mapping entity e_j^i for entity mention m_j^i as:

$$e_j^i = \arg\max_{r_{j,q}^i \in R_j^i} s_{j,q}^i \tag{8}$$

Since the mapping entity of some entity mention does not exist in the knowledge base, we have to deal with the problem of predicting unlinkable mention. Firstly, if the size of candidate mapping entity set R_j^i generated for entity mention m_j^i is equal to zero, we predict entity mention m_j^i as an unlinkable mention and return NIL for mention m_j^i undoubtedly. Otherwise, we can obtain e_j^i for entity mention m_j^i according to Formula 8. Subsequently, we have to validate whether the entity e_j^i is the true mapping entity for mention m_j^i. We adopt a simple method and learn a nil threshold τ to validate the entity e_j^i. If the final interest score $s_{j,q}^i$ of entity e_j^i is greater than the learned nil threshold τ, we return e_j^i as the mapping entity for entity mention m_j^i, otherwise we return NIL for m_j^i. The nil threshold τ is learned by linear search based on the training data set.

4. EXPERIMENTS

To evaluate the effectiveness and efficiency of our framework KAURI, we present a thorough experimental study in this section. All the programs were implemented in JAVA and all the experiments were conducted on an IBM server (eight 2.00GHz CPU cores, 10GB memory) with 64-bit Windows. In this paper, we do not utilize the parallel computing technique and just consider single machine implementation.

4.1 Experimental setting

To the best of our knowledge, there is no publicly available benchmark data set for the tweet entity linking task. Thus, we created a gold standard data set for the tweet entity linking task. We used the Twitter API to collect 3,200 most recent tweets for each of randomly sampled 71,937 Twitter users. Since the annotation task for tweet entity linking consists of detecting all the named entity mentions in all the tweets and identifying their corresponding mapping entities existing in YAGO(1)[1] of version 2008-w40-2, the annotation task is very time consuming. Therefore, labeling all tweets for all users is impractical. To create the gold standard data set, we randomly sampled 20 non-verified Twitter users and for each sampled user, we annotated the 200 most recent tweets. For the Twitter user who has fewer than 200 tweets, we annotated all his tweets. Finally, we obtained 3,818 tweets from 20 users forming the gold standard data set. A summary of this data set is shown in Table 4. From Table 4, we can see that 1,721 out of 3,818 tweets (about 45.08%) have at least one named entity mention. For 241 named entity mentions among the total 2,918 detected named entity mentions, we are uncertain to point out their mapping entities as the information contained in their tweets is so limited and noisy. Thus we dropped them and used the remaining 2677 mentions for evaluation.

[1]http://www.mpi-inf.mpg.de/yago-naga/yago/

Table 4: Summary of the gold standard data set

# Twitter user	20
# tweets	3818
# tweets having at least one named entity mention	1721
# named entity mentions	2918
# uncertain named entity mentions	241
# test named entity mentions	2677
# linkable named entity mentions	2240
# unlinkable named entity mentions	437

We downloaded the May 2011 version of Wikipedia to construct the dictionary D introduced in Section 2. Meanwhile, we employed these downloaded 3.5 million Wikipedia pages to obtain the context for each candidate entity for the calculation of *context similarity* feature introduced in Section 3.2. We set $\lambda = 0.4$ in the following experiments. The weight vector $\overrightarrow{w}$ and the nil threshold τ are learned using 2-fold cross validation. To evaluate the effectiveness of KAURI, we calculated the *accuracy* as the number of correctly linked entity mentions divided by the total number of all entity mentions.

4.2 Experimental results

Evaluation of effectiveness. To the best of our knowledge, KAURI is the first framework to deal with the tweet entity linking task. However, if we regard each tweet as a single document, we can apply all the previous methods for linking entities in Web documents [4, 7, 12, 11, 9, 20] to our data set. As it is impractical to implement all these previous methods, we chose one of the state-of-the-art methods LINDEN [20] as our baseline method, which leverages both the semantic context similarity and the topical coherence between entities within a single Web document to link entity mentions with knowledge base. Besides our proposed framework KAURI, we also evaluated the performance of our framework which regards the initial interest score as the final score, and we call this truncated framework as LOCAL.

The experimental results are shown in Table 5. Besides the accuracy, we also show the number of correctly linked entity mentions for all methods with different types (i.e., linkable, unlinkable, and all). For the framework LOCAL and the framework KAURI, we not only show their performance by leveraging all the intra-tweet local features introduced in Section 3.2, which we refer as LOCAL_{full} and KAURI_{full} respectively, but also present their performance by leveraging a subset of the intra-tweet local features. For instance, $\text{LOCAL}_{\beta=0,\gamma=0}$ and $\text{KAURI}_{\beta=0,\gamma=0}$ mean that when we calculate the initial interest score using Formula 4, we set $\beta = 0$ and $\gamma = 0$. Thus, in these two methods, we only leveraged the feature of *prior probability* to calculate the initial interest score.

The experimental results in Table 5 demonstrate that our proposed graph-based framework KAURI_{full} significantly

Table 5: Experimental results over the data set

Method	Linkable		Unlinkable		All	
	#	Accu.	#	Accu.	#	Accu.
LINDEN	1852	0.827	353	0.808	2205	0.824
$\text{LOCAL}_{\beta=0,\gamma=0}$	1784	0.796	355	0.812	2139	0.799
$\text{LOCAL}_{\gamma=0}$	1795	0.801	355	0.812	2150	0.803
$\text{LOCAL}_{\beta=0}$	1862	0.831	355	0.812	2217	0.828
LOCAL_{full}	1863	0.832	355	0.812	2218	0.829
$\text{KAURI}_{\beta=0,\gamma=0}$	1882	0.840	356	0.815	2238	0.836
$\text{KAURI}_{\gamma=0}$	1894	0.846	357	0.817	2251	0.841
$\text{KAURI}_{\beta=0}$	1913	0.854	371	0.849	2284	0.853
KAURI_{full}	**1923**	**0.858**	**373**	**0.854**	**2296**	**0.858**

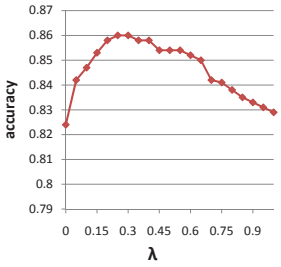

Figure 3: Sensitivity analysis to the parameter

outperforms the baseline method LINDEN and all the different configurations of the framework LOCAL. Also, we can see that each configuration of the framework KAURI obtains much higher accuracy compared with the same configuration of the truncated framework LOCAL, since we utilize the constructed graph described in Section 3.1 and the *user interest propagation algorithm* introduced in Section 3.3 in the framework KAURI. This means, the inter-tweet user interest information is very important and our *user interest propagation algorithm* is very effective for the tweet entity linking task, and integrating intra-tweet local information with the inter-tweet user interest information considerably boosts the performance. Moreover, it can be also seen from Table 5 that every intra-tweet local feature has a positive impact on the performance of the framework KAURI and LOCAL, and with the combination of all intra-tweet local features, KAURI_{full} and LOCAL_{full} can obtain the best results among the different configurations of the framework KAURI and LOCAL, respectively.

Sensitivity analysis. To better understand the performance characteristics of our proposed framework, we conducted sensitivity analysis to understand the influence of parameter λ to the performance of our framework. In Figure 3, we show the performance of KAURI_{full} with varied parameter λ. From the trend plotted in Figure 3, it can be seen that when $0.15 \leq \lambda \leq 0.65$, the accuracy achieved by KAURI_{full} is all greater than 0.85. Thus, we can say that when λ is varied from 0.15 to 0.65, the performance of our framework is not very sensitive to parameter λ and we set $\lambda = 0.4$ in all the other experiments.

Evaluation of efficiency. For the purpose of efficiency, when we want to link the entity mentions in the recent tweet stream published by some Twitter user, it is unnecessary to consider the interdependence relationship between the recent tweet and the very old tweets published by the same user a long time ago. Thus, we define the *tweet sliding window* as the set of sequential tweets published by one Twitter user, and the size of the tweet sliding window means

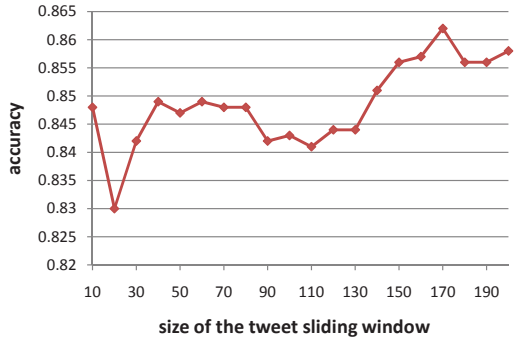

Figure 4: The performance of KAURI_{full} with varied size of the tweet sliding window

the number of tweets contained in this window. Then we want to know the proper size of the tweet sliding window, which means just considering the interdependence information across tweets within such size of window, our framework can achieve high and stable accuracy for this task. Therefore, we conducted an experiment to find the proper size for the tweet sliding window. We show the accuracy achieved by KAURI_{full} with varied size of the tweet sliding window in Figure 4. For illustration, for each user, we assign a tweet index to each tweet sequentially from 1 to 200. For the experimental results shown in Figure 4, all the tweet sliding windows with different size are set to start from the tweet with index 1 for each user. From Figure 4, we can see that when the size of the tweet sliding window is larger than or equal to 150, the accuracy achieved by KAURI_{full} is greater than 0.855 and remains relatively stable.

Accordingly, when we receive some new tweet stream posted by some user, we just need to consider the interdependence relationship across tweets within the latest tweet sliding window of size 150 in order to link the entity mention in the new tweet stream. Moreover, since our graph-based framework KAURI naturally supports the incremental update operation and we can store the constructed graph for the former tweet sliding window, in order to construct the new graph for the latest tweet sliding window, we just need to incrementally modify the existing graph (i.e., remove some nodes out of the latest window and add some new nodes appearing in the latest window), which will save lots of time and cost. To evaluate the incremental update performance of KAURI_{full}, we firstly preformed the tweet entity linking task for the initial tweet sliding window of size 150 for each user (starting from the tweet with index 1 to the tweet with index 150). The overall accuracy is 0.856, and the total annotation time is 251.35s for 1,952 entity mentions contained in this window. Roughly 70% of the annotation time is spent in graph construction. Then, we evaluated the incremental update performance of KAURI_{full} and assume that at each time we received 10 new tweets for each user. Table 6 shows the performance of KAURI_{full} at each time. From Table 6, we can see that at each time when new tweets are received, KAURI_{full} achieves stable high accuracy and spends about linear incremental annotation time to the number of added edges upon incrementally modifying the existing graph, which shows the scalability of KAURI. On average, the incremental annotation time per entity mention is between 14.03ms and 22.53ms, which demonstrates

Table 6: Incremental update performance of KAURI$_{full}$

Time	1	2	3	4	5
Tweet index	11-160	21-170	31-180	41-190	51-200
# mentions	1979	1988	2005	2013	2016
# added nodes	827	598	726	775	780
# added edges	172296	139398	122824	217440	201368
Accuracy	0.853	0.859	0.859	0.855	0.858
Incremental annot. time (s)	35.58	30.22	28.13	45.36	42.50
Incremental annot. time per mention (ms)	17.98	15.20	14.03	22.53	21.08

the efficiency of KAURI. In addition, taking time 5 as an example, if we construct the graph for the tweet sliding window from empty rather than incrementally modify the existing graph for the tweet sliding window at time 4, the needed annotation time is 304.51s, about 7.2 times of the incremental annotation time at time 5 (i.e., 42.50s), which also demonstrates the efficiency of KAURI.

5. RELATED WORK AND DISCUSSION

Twitter has received more and more attention from research community recently. The work relevant to our tweet entity linking task include named entity recognition for tweets [14, 13], influential Twitter user identification [22], tweet recommendation [5, 6], and user interest discovery [16, 24]. The task similar to our tweet entity linking task is concept linking with Wikipedia for tweets [15]. The goal is to identify relevant concepts (such as *Education*, *Shopping*, *Meeting*, *Student*, etc.) in tweets and generate links to the corresponding Wikipedia articles. Another recent work related to our work is object matching for tweets proposed in [8]. In [8], the authors exploited the geographic aspects of tweets, and used a probabilistic model to infer the matches between tweets and restaurants from a list of restaurants. Compared with these two work, our tweet entity linking task focuses on linking named entities of general types (e.g., person, location, organization, and event) rather than the concepts or the tweets, and links named entity mentions with the general-purpose knowledge base rather than the restaurant list.

Another thread of related work is linking named entities in Web documents (or Web lists) with knowledge base [4, 7, 12, 11, 9, 19, 20]. The work [12] models local mention-to-entity compatibility in combination with the pairwise intra-document topical coherence of candidate entities using a probabilistic graphical model. In [9], they proposed a graph-based collective entity linking method to model the topical interdependence between different entity linking decisions within one document. LINDEN [20] gives a rank to the candidate entities with the linear combination of four features (i.e., the entity popularity, semantic associativity, semantic similarity and the global topical coherence between mapping entities). The intra-tweet local features adopted by KAURI, *prior probability*, *context similarity*, and *topical coherence*, have been used in various ways by these previous work. However, applying the local features alone leads to unsatisfactory performance over the tweets due to the noisy, short and informal nature of tweets, which has been discussed before and confirmed by our experiments. In KAURI we propose a unified graph-based framework by combining intra-tweet local information with the inter-tweet user interest information, which is very effective in the new problem setting of tweet entity linking.

6. CONCLUSION

In this paper, we have studied a new problem on tweet entity linking. We propose KAURI, a framework which can effectively link named entity mentions detected in tweets with a knowledge base via user interest modeling. KAURI combines both the intra-tweet local information and the inter-tweet user interest information into a unified graph-based framework. The experimental results show that KAURI achieves high performance in terms of both accuracy and efficiency, and scales well to tweet stream.

7. ACKNOWLEDGMENTS

This work was supported in part by National Natural Science Foundation of China under Grant No. 61272088 and National Basic Research Program of China (973 Program) under Grant No. 2011CB302206.

8. REFERENCES

[1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, and Z. Ives. Dbpedia: A nucleus for a web of open data. In *ISWC'07*.
[2] C. Bizer, T. Heath, and T. Berners-Lee. Linked Data - The Story So Far. *IJSWIS*, 5(3), 2009.
[3] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD'08*.
[4] R. Bunescu and M. Pasca. Using Encyclopedic Knowledge for Named Entity Disambiguation. In *EACL'06*.
[5] J. Chen, R. Nairn, L. Nelson, M. Bernstein, and E. Chi. Short and tweet: experiments on recommending content from information streams. In *CHI'10*.
[6] K. Chen, T. Chen, G. Zheng, O. Jin, E. Yao, and Y. Yu. Collaborative personalized tweet recommendation. In *SIGIR'12*.
[7] S. Cucerzan. Large-Scale Named Entity Disambiguation Based on Wikipedia Data. In *EMNLP-CoNLL'07*.
[8] N. Dalvi, R. Kumar, and B. Pang. Object matching in tweets with spatial models. In *WSDM'12*.
[9] X. Han, L. Sun, and J. Zhao. Collective entity linking in web text: a graph-based method. In *SIGIR'11*.
[10] T. H. Haveliwala. Topic-sensitive pagerank. In *WWW'02*.
[11] J. Hoffart, M. A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. Robust disambiguation of named entities in text. In *EMNLP'11*.
[12] S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti. Collective annotation of wikipedia entities in web text. In *SIGKDD'09*.
[13] 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 *SIGIR'12*.
[14] X. Liu, S. Zhang, F. Wei, and M. Zhou. Recognizing named entities in tweets. In *ACL'11*.
[15] E. Meij, W. Weerkamp, and M. de Rijke. Adding semantics to microblog posts. In *WSDM '12*, pages 563–572.
[16] M. Michelson and S. A. Macskassy. Discovering users' topics of interest on twitter: a first look. In *AND'10*.
[17] D. Milne and I. H. Witten. An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. In *WIKIAI'08*.
[18] W. Shen, J. Wang, P. Luo, and M. Wang. A graph-based approach for ontology population with named entities. In *CIKM'12*, pages 345–354.
[19] W. Shen, J. Wang, P. Luo, and M. Wang. Liege: Link entities in web lists with knowledge base. In *SIGKDD'12*.
[20] W. Shen, J. Wang, P. Luo, and M. Wang. Linden: linking named entities with knowledge base via semantic knowledge. In *WWW'12*.
[21] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A core of semantic knowledge unifying wordnet and wikipedia. In *WWW'07*.
[22] J. Weng, E.-P. Lim, J. Jiang, and Q. He. Twitterrank: finding topic-sensitive influential twitterers. In *WSDM'10*.
[23] W. Wu, H. Li, H. Wang, and K. Q. Zhu. Probase: a probabilistic taxonomy for text understanding. In *SIGMOD'12*.
[24] Z. Xu, L. Ru, L. Xiang, and Q. Yang. Discovering user interest on twitter with a modified author-topic model. In *WI-IAT'11*.