Entity Linking on Graph Data

Minghe Yu
Supervised by Prof. Jianhua Feng
Department of Computer Science and Technology, Tsinghua University, Beijing, China
yumh12@mails.tsinghua.edu.cn

ABSTRACT

With the emergence of massive information networks, graph data have become ubiquitous for various applications. Although many graph processing problems have been studied recently, entity linking on graph data has not received enough attention by the academia and industry, which finds vertex pairs that refer to the same entity from two graphs. There are two main research challenges arising in this problem. The first one is how to determine whether two vertices refer to the same entity which is rather hard for graph data, especially uncertain data, e.g., social networks. The second challenge is to efficiently link the vertices. As existing graph data are rather large, it is very important to devise efficient algorithms to achieve high performance. To address these challenges, in this paper we propose a similarity-based method which takes the vertex pairs with similarity larger than a given threshold as linked entities. We extend existing textual similarity and structural similarity to evaluate similarity between vertices from different graphs. To achieve high quality, we also combine them and propose a hybrid similarity. We also discuss new algorithms to efficiently link entities. We conduct experimental studies on real datasets and the results proves show that our hybrid method achieves high performance and outperform the baseline approaches.

Keywords

Entity Linking; Graph Data; Multiple Graph

1. INTRODUCTION

With the emergence of massive information networks, graph data have become ubiquitous for various applications. For example, social networks, e.g., Twitter, Facebook, Linkedin, have widely accepted. Obviously social networks can be represented by graphs where vertices are users and edges are follower/followee relationships. In addition, knowledge bases, e.g., Freebase [1], Yago, DBPedia, also play an important role in information retrieval. Knowledge bases can also be represented by graphs where vertices are entities and edges are relationships between entities.

Although many graph processing problems, e.g., graph matching, graph search, have been studied recently, entity linking on graph data has not received enough attention by the academia and industry, which finds vertex pairs that refer to the same entity from two graphs. Entity linking is rather important not only on textual data [11] but also on graph data. First we can enrich the graph data. For example, Given two social networks, e.g., Facebook and Twitter, if we can correlate the users from the two networks, we can obtain more complete profile for users, e.g., user preferences, friends, hobbies. Take knowledge bases as another example. If we can link entities across different knowledge bases, we can germane a more accurate, large, mature knowledge base. Second, we can enable link prediction and recommendation across different graphs. For example, based on user relationships in a social network, we can predict and recommend friends for another social network. Third, we can improve the advertising quality. As we know much information about an entity, we can use the enriched data to improve the search and recommendation quality in advertising.

There are two main research challenges arising in the problem of entity linking on graph data. The first one is how to determine whether two vertices refer to the same entity which is rather hard for graph data, especially uncertain data, e.g., social networks. On social networks, some users will not publish their privacy data, e.g., age and education, or post some random or error data. It is rather hard to link the uncertain data. The second challenge is to efficiently link the vertices. As existing graph data are rather large, for example there are more than half a billion users in Twitter, it is very important to devise efficient algorithms to achieve high performance.

In recent years, there are some studies on entity linking across social networks [15, 4]. However these methods only use some heuristics to link entities. For example, Zafarani et. al. [15] utilized users' naming rules in their accounts to correlate users and Goga et. al. [4] uses timestamp and location to identify the similar users. They only use a small portion of data and do not fully utilize the textual information of users and structures between users to link the entities. In addition, they do not develop principle methods to address this problem throughly.

To address these limications, in this paper we propose a similarity-based method which takes the vertex pairs with similarity larger than a given threshold as linked entities. We extend existing textual similarity and structural simi-

WWW'14 Companion, April 7–11, 2014, Seoul, Korea.
ACM 978-1-4503-2745-9/14/04. .

larity to evaluate similarity between vertices from different graphs. To achieve high quality, we also combine them and propose a hybrid similarity. We also discuss new algorithms to efficiently link entities. We conduct experimental studies on real datasets and the results proves show that our hybrid method achieves high performance and outperform the baseline approaches.

The reminder of paper is organized as follows. We first formalize our problem in Section 2, and then survey related works in Section 3. Our similarity-based method is proposed in Section 4. The experimental results are shown in Section 5. Finally, we conclude the paper in Section 6.

2. PROBLEM FORMULATION

Data Model. In many applications, data can be modeled as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a set of vertices and $\mathcal{E}$ is a set of edges. We also call each vertex as an entity. Each entity v_i is associated with a set of attributes $\mathcal{A}_i = \{a_i^1, a_i^2, ..., a_i^{|\mathcal{A}_i|}\}$. For example, in social networks, the entities are users and edges are follower/followee relationships. The attributes of each entity include user profile and its posed tweets. For publication databases, the entities are authors and edges are coauthor relationships. The attributes of each entity include publication of the corresponding author. Next we formalize our problem.

Definition 1 (Entity Linking on Graph Data). *Given two graphs $\mathcal{G}_1, \mathcal{G}_2$, we want to find all vertex pairs from the two graphs that correspond to real-world entities.*

Example 1. *Consider the two datasets in Table 2.1 We can model them as co-authorship graphs shown in Figure 2.1 (Graph A and B). The vertices are authors and edges are co-author relationships, and the attributes of publication as shown in Table 2.2. For instance, In Dataset D_1, author A and B both publish an article P_1, therefore we can add an edge to link them in Graph A. A and A' should denote the same entity as they publish similar papers and have many common co-authors.*

Table 2.1: Publication Databases

(a) Dataset D_1

	Author	Publication
A	*Adele*	P_1, P_2
B	*Jenny*	P_1, P_3, P_4, P_6
C	*Jane*	P_1, P_2, P_4
D	*Leo*	P_2, P_5
E	*Harold*	P_3, P_6
F	*Tomas*	P_3, P_4, P_5
G	*Red*	P_5

(b) Dataset D_2

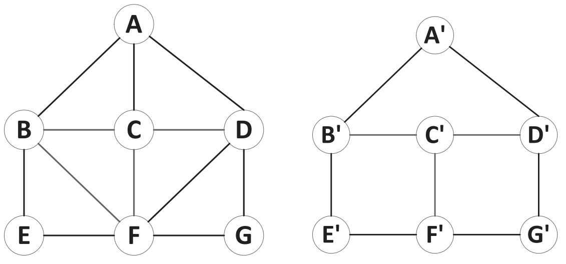
	Author	Publication
A'	*Adele*	P_1,P_2,
B'	*Will*	P_1,P_4,P_6
C'	*Jane*	P_4,P_8,P_9
D'	*Leo*	P_2,P_5,P_9
E'	*Alex*	P_3,P_6
F'	*Tomas*	P_3,P_7,P_8
G'	*Red*	P_5,P_7

3. RELATED WORK

Correlating Users Across Social Networks. There are some recent works on correlating users across multiple social networks integration. Zafarani et. al. [15] proposes a method to find matching entities based on users' unique behavioral patterns. This method analyzes and summarizes the users' naming rules and employs supervised learning to connect users. Although this method can identify users with similar names, it does not consider other information of the user. Thus if a user does not name his account based on the normal naming rules, the method cannot match these account well. Goga et. al. [4] uses timestamp and location to identify the similar users. The basic idea is that the same user usually posts relevant information in different social networks in a time window. This method also utilizes users' active location region to measure the similarity.

Table 2.2: publication details

Publication	Keywords	Year
P_1	integration, entity linking	2012
P_2	textual, entity linking	2011
P_3	alignment,knowledge base	2013
P_4	graph,entity linking	2011
P_5	entity,knowledge base	2012
P_6	graph,textual,similarity	2010
P_7	textual,link	2013
P_8	knowledge base	2013
P_9	alignment,entity, similarity	2012



(a) Graph A (b) Graph B

Figure 2.1: Graph modeling authors' data

Entity Alignment Across Knowledge Graphs. There are some recent studies on aligning large knowledge bases. Comparing with social networks, knowledge bases usually have a specific branch structure and the properties of entities are clean and have unique description with high quality. Julien et. al. [6] propose Simple Greedy Matching (SiGMa), an iterative propagation algorithm to match entities with structure similarity. It also considers the properties' similarity between entities calculated by the Jaccard similarity. However, this method cannot apply to social networks because the data are rather uncertain in social networks. Qi et. al. [10], focus on link prediction cross networks. They use a mature and reliable network as the source to predict links in the target network. To correct the cross-network bias, they resample the source network for ensuring robust link structure. And this method also uses the attributes of vertex to calculate relevance when the target network is lack of links.

Structure-based Entity Linking. There are some efficient algorithms to measure entity similarity, including PageRank [3],SimRank [5] and P-Rank [16]. For the random walk model, The similarity of two vertices v and u is denoted by their the random-walk-related probability value. In PageRank, it is the chance that a surfer from u can reach the v at the i-th step. And it defines the probability of two surfers from u and v are meeting at i-th step in SimRank. P-Rank is the method like SimRank that also measures the similarity of entities by their neighbors' similarity. The difference of P-Rank from SimRank is that it can be applied in heterogeneous graphs in which their vertices can denote different kinds of entities.

Text-based Entity Linking. There are also some studies on text-based entity linking. Metzler et. al. [9] investigate several methods of similarity measure for short text. They evaluate the effectiveness and efficiency of lexicon-level matching, probabilistic measure and hybrid technique dealing with short text similarity. The objective of their work is to estimate performance of the text similarity measure in the web search results which can be consider as the expanded representations of short text segments.

Different from existing studies, we study the problem of entity linking on graph data by using user profiles and graph structures. We emphasize on both efficiency and quality.

4. SIMILARITY-BASED METHOD

In this section, we propose a similarity-based method to link vertices across different graphs. The basic idea is that if two vertices refer to the same similarity, they should have large similarity. Thus two research challenges arise. The first one is how to effectively quantify the similarity. The second one is how to efficiently compute the similar pairs. We first introduce the textual-based similarity (Section 4.1) and then discuss the structural-based similarity (Section 4.2). Finally we propose a hybrid-based similarity (Section 4.3).

4.1 Textual Similarity

Each vertex usually has multiple attributes and each attribute contains a set of keywords. Different attributes usually have different weights (importance). For example, in social networks, the attributes include both of user's profile (e.g., account, birthday, gender, address, email, middle school, high school, university), and tweets posed by the user. We can use a priori knowledge to deduce the "identification attribute" from the user profiles, which can be used to distinguish vertices. That is if two vertices refer to the same entity, the values of their identification attributes should be the same. For instance, email should be an identification attribute, because if the emails of two vertices are the same, they should refer to the same entity. In addition, "birthday + gender + address" may also be a quasi identification attribute, because if the birthday, gender, address of two vertices are same, they have large probability to be the same entity.

If we can generate the identification attribute, we can easily link different vertices. However, to preserve privacy, some users will not publish identification attributes (in other words, vertices have no values for identification attributes). Thus we cannot use identification attributes to link vertices. To address this issue, we can quantify the similarity of their other attributes. The basic idea is that if some attributes for two vertices are similar, they may refer to the entity. Formally, we utilize the following similarity function to quantify the similarity.

$$\textsc{Sim}_{\mathsf{T}}(v, v') = \sum w_i \cdot \mathsf{T}(v.a_i^k, v'.a_i^k) \tag{1}$$

where w_i is the weight (importance) of an attribute and T is a traditional textual similarity function on two keyword sets ($v.a_i^k$ and $v'.a_i^k$). The common used similarity functions include Jaccard and Cosine.

Example 2. *Consider the vertices of Graph A and B in Fig 2.1. And we use Jaccard to measure textual similarity,including the similarity in the attributes.Surpose authors' name and publications' year are the "identification attribute" and the Publications with the same tag are the same. For vertices D and D', they share the same name "Leo" and they publish articles $\{P_2, P_5\}$ and $\{P_2, P_5, P_9\}$. Therefore, the textual similarity of D and D' is* $\textsc{Sim}_{T}$ *(D,D')= 0.375. For vertices E and E', even though they publish the same articles, their names are different. So their similarity is* $\textsc{Sim}_{T}$ *(E,E')= 0.*

Two vertices are taken to be the same entity if there textual similarity is not smaller than a given threshold τ_{T}. We can utilize existing similarity-join-based method to efficiently find similar vertex pairs, including EDJoin [13], PPJoin [14] and PassJoin [7]

4.2 Structural Similarity

If vertices have no enough attribute information, we cannot utilize the textual-based similarity to link different vertices. To address this problem, we introduce a structural-based similarity, e,g., Simrank [5]. Simrank is usually to quantify the similarity between two vertices in a graph based on the graph structure, defined as below.

$$\textsc{Sim}_{\mathsf{S}}(u, v) = R_{x+1}(u, v) = \frac{C}{|I(u)||I(v)|} \sum_{s=1}^{|I(u)|} \sum_{t=1}^{|I(v)|} R_x(I_s(u), I_t(v)) \tag{2}$$

where

$$R_0(u, v) = \begin{cases} 1 & u = v \\ 0 & u \neq v \end{cases}$$

and C is a constant between 0 and 1, x is the iteration number, $I(u)$ is the in-neighbor set of u and $I_s(u)$ is the s-th in-neighbor of u.

The basic idea of Simrank is that if two vertices are similar they must share many neighbors (or neighbors' neighbors). It computes the similarity iteratively until convergence.

However different from traditional Simrank, our method requires to compute the similarity of vertices across two different graphs. Thus we cannot get the initialization value $R_0(u, v)$ as we do not know whether $u = v$ is true or not. To address this issue, we can use the textual similarity as the initialization value of Simrank, that is

$$R_0(u, v) = \textsc{Sim}_{\mathsf{T}}(u, v) \tag{3}$$

Example 3. *Consider the vertices of Graph A and B in Fig 2.1, and the textual similarity is measured with above method. Suppose the iteration number is 1 and constanct C is 0.85. For the vertices A and A', their structure similarity is based on the similarity of their linked vertices, $\{B, C, D\}$ and $\{B', D'\}$. After 1 time iteration, we can get the structure similarity* $\textsc{Sim}_{S}$ *(u,v)=0.10625.*

Thus two vertices are taken to be the same entity if there structural similarity is not smaller than a given threshold τ_{S}. We can utilize existing simrank-based method to efficiently find similar vertex pairs,including Single-Pair SimRank [8] and IDJ [12].

4.3 Hybrid Similarity

Obviously the textual similarity neglects the structure information. Although the structural similarity uses the textual information, the proportion of textual similarity is negligible compared with the structural similarity (as the structural similarity is usually small after several iterations). To address this problem, we propose two hybrid similarity by combing the textual similarity and structural similarity.

4.3.1 Naive Hybrid Similarity

The naive hybrid similarity method combines these two similarities with a tuning coefficient, i.e.,

$$\textsc{Sim}_{\textsf{HN}}(u, v) = \alpha\textsc{Sim}_{\textsf{T}}(u, v) + (1 - \alpha)\textsc{Sim}_{\textsf{S}}(u, v) \qquad (4)$$

where α is a tuning parameter to balance the importance of textual similarity and structural similarity in the function.

4.3.2 Iterative Hybrid Similarity

When we use SimRank to measure similarity, the middle results in each iteration may have bias. To address this issue, we add textual similarity into structural similarity and the new similarity function is as follow:

$$\begin{aligned}\textsc{Sim}_{\textsf{HI}}(u, v) = \mathcal{R}'_{x+1}(u, v) = \alpha\textsc{Sim}_{\textsf{T}}(u, v)\\ + (1 - \alpha) * \frac{C}{|I(u)||I(v)|} \sum_{s=1}^{|I(u)|} \sum_{t=1}^{|I(v)|} R'_x(I_s(u), I_t(v))\end{aligned} \qquad (5)$$

The reasons causing bias in the structural similarity include two aspects: first is over-dense entities, such as a social community in the social networks. And second is the large structural different among graphs. Entities have this problems can be improved with utilizing this hybrid similarity function to distinguish each other.

Two vertices are taken to be the same entity if their hybrid similarity is not smaller than a given threshold $\tau_{\textsf{H}}$. It is worth noting that given a vertex v in a graph, there may be multiple vertices in another graph that have similarity to v larger than the threshold. However there is usually one vertex that is most similar to vertex v. To find the most similar vertex pairs from many results pairs with similarity larger than a threshold, we model the problem as a maximum weighted matching problem as follows.

Given several vertex pairs (v, v') with a weight $w(v, v')$. We can model the pairs as a bipartite graph (bigraph) $G = (U, V, E)$ where U is a set of vertices for v, V is a set of vertices for v', and E is the edge set (Each edge also has a weight). Without loss of generality, suppose $|U| \leq |V|$. A matching of the digraph is a set of $|U|$ pairs that contain all the vertices in U. A maximum weighted matching is the matching with the maximum weight (the sum of the weight of edges in the matching). The maximum weighted matching problem can be resolved in polynomial time [2].

4.4 Research Challenges

Efficiency. For two large graphs, there may be large numbers of vertices. It is rather expensive to compute the structural similarity. To address this problem, there are two possible solutions. The first one is to devise pruning-based algorithms which can prune the pairs with similarity smaller than the threshold. The second one is to find the top-k pairs.
Selecting Appropriate Parameters. There are some parameters in the functions, e.g., α, w_i, to evaluate the importance of the textual similarity, the importance of different attributes. It is challenging to devise an automatic method to compute these parameters.
Temporal-based Similarity. The above similarity functions do not consider the temporal information. In real applications, the attributes (e.g., user profiles) may be dynamically changed. We should consider the temporal information to compute the similarity.

5. INITIAL RESULTS

5.1 Experimental Setting

We extracted 1000 authors and 25,908 papers published by these authors from DBLP and constructed graphs where vertices are authors with the attributes of these authors' publications and edges are co-author relationships. The attributes of authors included three properties, $\{title, year, booktitle\}$.

In our implementation, we separated the data set into two graphs. Each graph had almost 70% different attributes of the authors and there were 20% different authors in two graphs. In other words, there was about 46% overlap attributes shared by the two graphs. Details of the graphs are shown in the Table 5.1.

Table 5.1: DBLP Dataset

	Vertex	Edge	#AVG papers	#AVG links
Network1	770	12,092	44.18	15.70
Network2	900	11,516	25.72	12.79

5.2 Experimental Results

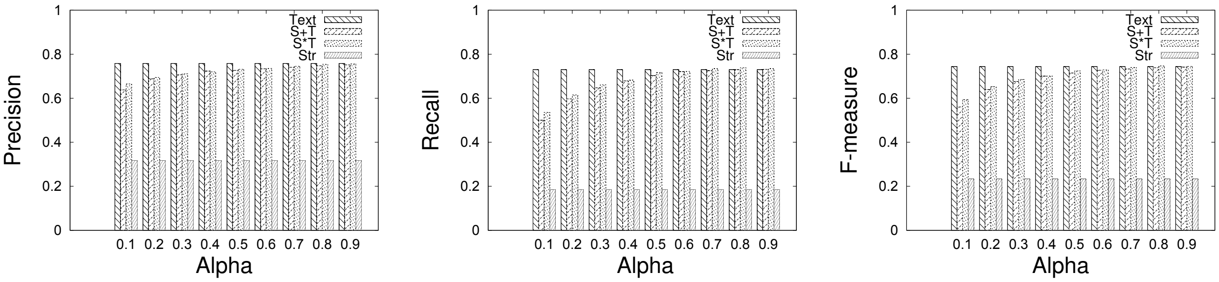
We implemented the four algorithms: **Text** and **Str** which only utilized textural and structural similarity respectively, **S+T** which used naive hybrid similarity, and **S*T** which used iterative hybrid similarity.

To evaluate the quality, we utilized standard metrics of precision, recall, and F-measure. We evaluated these algorithms by varying different parameters: constant C, threshold τ and tuning coefficient α. The results are shown in Figures 5.1, 5.2, 5.3.

From the experimental results, we can make the following observations. First, **S*T** achieved the best performance, especially in recall. This is because it can seamlessly combine textual similarity and structural similarity. Second, **Str** achieved the worst result because it was very hard to link entities from two graphs based on structures (as many authors may share enough common co-authors). Third, `Text` also achieved high performance since in this dataset the determinant of linking entities is textual similarity. Fourth, with the increase of α, recall increased as the textual similarity is more important. Fifth, with the increase of constant C, the precision decreased since the important of structural similarity increased. Sixth, with the increase of threshold τ, the precision increased and recall decreased because for larger threshold, there are less results.

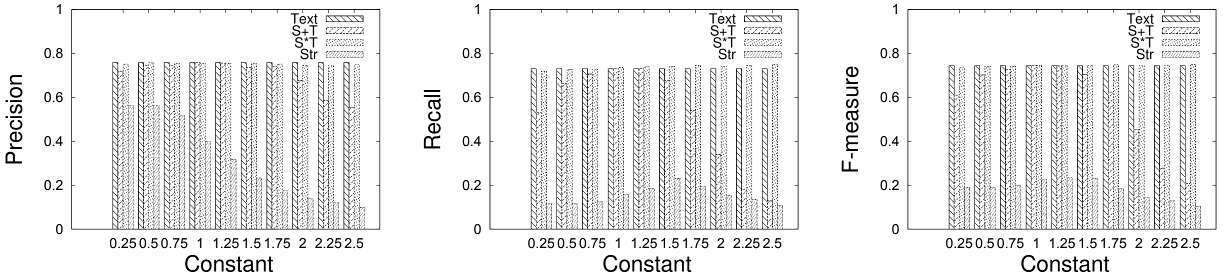
6. CONCLUSION AND FUTURE WORK

In this paper, we studied the problem of entity linking on Graph data. We discussed how to use existing techniques to support our problem. We found that existing method cannot effective link the entities. We also proposed a new



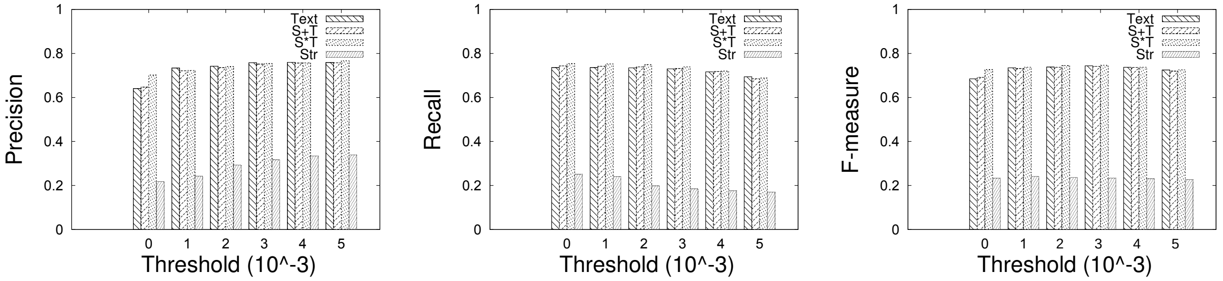
(a) Precision (b) Recall (c) F-Measure

Figure 5.1: Effect on varying α



(a) Precision (b) Recall (c) F-Measure

Figure 5.2: Effect on varying constant C



(a) Precision (b) Recall (c) F-measure

Figure 5.3: Effect on varying similarity threshold τ

hybrid-based method to improve the quality. Experimental results show that our method achieved higher quality than existing solutions. Our study can motivate many new research directions: new effective similarity metrics to evaluate the similarity between graph vertices and new efficient algorithms to link entities.

7. REFERENCES

[1] Freebase. `http://www.freebase.com/`.

[2] Matching (graph theory). `http://en.wikipedia.org/wiki/Maximum_matching_problem`.

[3] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30(1-7):107–117, 1998.

[4] O. Goga, H. Lei, S. H. K. Parthasarathi, G. Friedland, R. Sommer, and R. Teixeira. Exploiting innocuous activity for correlating users across sites. In *WWW*, pages 447–458, 2013.

[5] G. Jeh and J. Widom. Simrank: a measure of structural-context similarity. In *KDD*, pages 538–543, 2002.

[6] S. Lacoste-Julien, K. Palla, A. Davies, G. Kasneci, T. Graepel, and Z. Ghahramani. Sigma: simple greedy matching for aligning large knowledge bases. In *KDD*, pages 572–580, 2013.

[7] G. Li, D. Deng, J. Wang, and J. Feng. Pass-join: A partition-based method for similarity joins. *CoRR*, abs/1111.7171, 2011.

[8] P. Li, H. Liu, J. X. Yu, J. He, and X. Du. Fast single-pair simrank computation. In *SDM*, pages 571–582, 2010.

[9] D. Metzler, S. T. Dumais, and C. Meek. Similarity measures for short segments of text. In *ECIR*, pages 16–27, 2007.

[10] G.-J. Qi, C. C. Aggarwal, and T. S. Huang. Link prediction across networks by biased cross-network sampling. In *ICDE*, pages 793–804, 2013.

[11] V. Stoyanov, J. Mayfield, T. Xu, D. W. Oard, D. Lawrie, T. Oates, and T. Finin. A context-aware approach to entity linking. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, AKBC-WEKEX '12, pages 62–67, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[12] L. Sun, R. Cheng, X. Li, D. W. Cheung, and J. Han. On link-based similarity join. *PVLDB*, 4(11):714–725, 2011.

[13] C. Xiao, W. Wang, and X. Lin. Ed-join: an efficient algorithm for similarity joins with edit distance constraints. *PVLDB*, 1(1):933–944, 2008.

[14] C. Xiao, W. Wang, X. Lin, and J. X. Yu. Efficient similarity joins for near duplicate detection. In *WWW*, pages 131–140, 2008.

[15] R. Zafarani and H. Liu. Connecting users across social media sites: a behavioral-modeling approach. In *KDD*, pages 41–49, 2013.

[16] P. Zhao, J. Han, and Y. Sun. P-rank: a comprehensive structural similarity measure over information networks. In *CIKM*, pages 553–562, 2009.