Entity Linking with a Knowledge Base: Issues, Techniques, and Solutions

Wei Shen,
Jianyong Wang, *Senior Member, IEEE,*
and Jiawei Han, *Fellow, IEEE*

Abstract—The large number of potential applications from bridging Web data with knowledge bases have led to an increase in the entity linking research. Entity linking is the task to link entity mentions in text with their corresponding entities in a knowledge base. Potential applications include information extraction, information retrieval, and knowledge base population. However, this task is challenging due to name variations and entity ambiguity. In this survey, we present a thorough overview and analysis of the main approaches to entity linking, and discuss various applications, the evaluation of entity linking systems, and future directions.

Index Terms—Entity linking, entity disambiguation, knowledge base

1 Introduction

1.1 Motivation

The amount of Web data has increased exponentially and the Web has become one of the largest data repositories in the world in recent years. Plenty of data on the Web is in the form of natural language. However, natural language is highly ambiguous, especially with respect to the frequent occurrences of named entities. A named entity may have multiple names and a name could denote several different named entities.

On the other hand, the advent of knowledge sharing communities such as Wikipedia and the development of information extraction techniques have facilitated the automated construction of large scale machine-readable knowledge bases. Knowledge bases contain rich information about the world's entities, their semantic classes, and their mutual relationships. Such kind of notable examples include DBpedia [1], YAGO [2], Freebase [3], KnowItAll [4], ReadTheWeb [5], and Probase [6].

Bridging Web data with knowledge bases is beneficial for annotating the huge amount of raw and often noisy data on the Web and contributes to the vision of Semantic Web [7]. A critical step to achieve this goal is to link named entity mentions appearing in Web text with their corresponding entities in a knowledge base, which is called entity linking.

- *W. Shen and J. Wang are with the Department of Computer Science and Technology, Tsinghua University, Beijing, 100084, China. E-mail: weishen09@gmail.com;jianyong@tsinghua.edu.cn.*
- *J. Han is with the Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801. E-mail: hanj@cs.uiuc.edu.*

Entity linking can facilitate many different tasks such as knowledge base population, question answering, and information integration. As the world evolves, new facts are generated and digitally expressed on the Web. Therefore, enriching existing knowledge bases using new facts becomes increasingly important. However, inserting newly extracted knowledge derived from the information extraction system into an existing knowledge base inevitably needs a system to map an entity mention associated with the extracted knowledge to the corresponding entity in the knowledge base. For example, relation extraction is the process of discovering useful relationships between entities mentioned in text [8,9,10,11], and the extracted relation requires the process of mapping entities associated with the relation to the knowledge base before it could be populated into the knowledge base. Furthermore, a large number of question answering systems rely on their supported knowledge bases to give the answer to the user's question. To answer the question "What is the birthdate of the famous basketball player Michael Jordan?", the system should first leverage the entity linking technique to map the queried "Michael Jordan" to the NBA player, instead of for example, the Berkeley professor; and then it retrieves the birthdate of the NBA player named "Michael Jordan" from the knowledge base directly. Additionally, entity linking helps powerful join and union operations that can integrate information about entities across different pages, documents, and sites.

The entity linking task is challenging due to name variations and entity ambiguity. A named entity may have multiple surface forms, such as its full name, partial names, aliases, abbreviations, and alternate spellings. For example, the named entity of "Cornell

University" has its abbreviation "Cornell" and the named entity of "New York City" has its nickname "Big Apple". An entity linking system has to identify the correct mapping entities for entity mentions of various surface forms. On the other hand, an entity mention could possibly denote different named entities. For instance, the entity mention "Sun" can refer to the star at the center of the Solar System, a multinational computer company, a fictional character named "Sun-Hwa Kwon" on the ABC television series "Lost" or many other entities which can be referred to as "Sun". An entity linking system has to disambiguate the entity mention in the textual context and identify the mapping entity for each entity mention.

1.2 Task Description

Given a knowledge base containing a set of entities E and a text collection in which a set of named entity mentions M are identified in advance, the goal of entity linking is to map each textual entity mention $m \in M$ to its corresponding entity $e \in E$ in the knowledge base. Here, a named entity mention m is a token sequence in text which potentially refers to some named entity and is identified in advance. It is possible that some entity mention in text does not have its corresponding entity record in the given knowledge base. We define this kind of mentions as unlinkable mentions and give NIL as a special label denoting "unlinkable". Therefore, if the matching entity e for entity mention m does not exist in the knowledge base (i.e., $e \notin E$), an entity linking system should label m as NIL. For unlinkable mentions, there are some studies that identify their fine-grained types from the knowledge base [12,13,14,15], which is out of scope for entity linking systems. Entity linking is also called Named Entity Disambiguation (NED) in the NLP community. In this paper, we just focus on entity linking for English language, rather than cross-lingual entity linking [16].

Typically, the task of entity linking is preceded by a named entity recognition stage, during which boundaries of named entities in text are identified. While named entity recognition is not the focus of this survey, for the technical details of approaches used in the named entity recognition task, you could refer to the survey paper [17] and some specific methods [18,19,20]. In addition, there are many publicly available named entity recognition tools, such as Stanford NER[1], OpenNLP[2], and LingPipe[3]. Finkel et al. [18] introduced the approach used in Stanford NER. They leveraged Gibbs sampling [21] to augment an existing Conditional Random Field based system with long-distance dependency models, enforcing label consistency and extraction template consistency

1. http://nlp.stanford.edu/ner/
2. http://opennlp.apache.org/
3. http://alias-i.com/lingpipe/

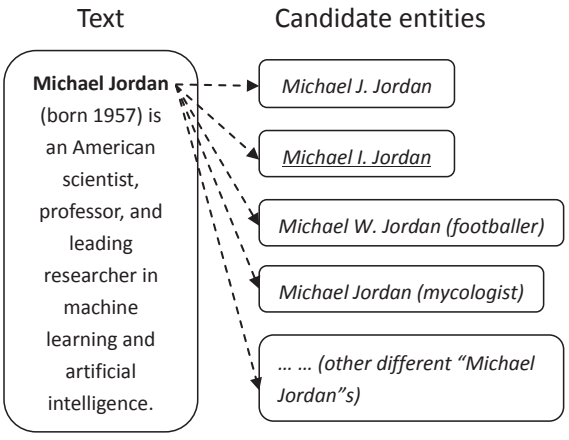

Fig. 1. An illustration for the entity linking task. The named entity mention detected from the text is in bold face; the correct mapping entity is underlined.

constraints. Recently, some researchers [22,23,24] proposed to perform named entity recognition and entity linking jointly to make these two tasks reinforce each other, which is a promising direction especially for text where named entity recognition tools perform poorly (e.g., tweets).

Now, we present an example for the entity linking task shown in Figure 1. For the text on the left of the figure, an entity linking system should leverage the available information, such as the context of the named entity mention and the entity information from the knowledge base, to link the named entity mention "Michael Jordan" with the Berkeley professor *Michael I. Jordan*, rather than other entities whose names are also "Michael Jordan", such as the NBA player *Michael J. Jordan* and the English football goalkeeper *Michael W. Jordan*.

When performed without a knowledge base, entity linking reduces to the traditional entity coreference resolution problem. In the entity coreference resolution problem [25,26,27,28,29,30], entity mentions within one document or across multiple documents are clustered into several different clusters each of which represents one specific entity, based on the entity mention itself, context, and document-level statistics. Compared with entity coreference resolution, entity linking requires linking each entity mention detected in the text with its mapping entity in a knowledge base, and the entity information from the knowledge base may play a vital role in linking decision.

In addition, entity linking is also similar to the problem of word sense disambiguation (WSD) [31]. WSD is the task to identify the sense of a word (rather than a named entity) in the context from a sense inventory (e.g., WordNet [32]) instead of a knowledge base. WSD regards that the sense inventory is complete, however, the knowledge base is not. For example, many named entities do not have the corresponding

entries in Wikipedia. Furthermore, named entity mentions in entity linking vary much more than sense mentions in WSD [33].

Another related problem is record linkage [34,35,36, 37,38,39,40,41] (also called duplicate detection, entity matching, and reference reconciliation) in the database community. Record linkage is the task of matching records from several databases or sources that refer to the same entities, such as matching two publication records referring to the same paper, which is a crucial task for data integration and data cleaning. Each record describing an entity contains a set of attribute values. For instance, a record describing a person entity may have attributes, such as person name, birthday and address. Most record linkage approaches are based on the assumption that duplicate records should have equal or similar attribute values. They typically compare different attribute values of the records using a set of similarity measures and the resulting similarity scores may be combined using different aggregation functions. The pair of records whose aggregated similarity score exceeds some threshold is considered as describing the same entity. Specifically, Dong et al. [38] proposed a novel record linkage algorithm based on a general framework for propagating information from one linkage decision to another by leveraging context information, similarities computed on related entities, and enriched references. Isele and Bizer [39] proposed GenLink, a supervised learning algorithm which employs genetic programming to learn linkage rules from a set of existing record links. Their algorithm is capable of generating linkage rules which select discriminative attributes for comparison, apply chains of data transformations to normalize attribute values, choose appropriate similarity measures and thresholds and combine the results of multiple comparisons using non-linear aggregation functions.

While in the entity linking problem, the entity mention which needs to be linked resides in the unstructured text and does not have attribute values with it. The entity in the knowledge base has many associated relations which indicate its attributes. For each entity mention, if we could leverage some information extraction technique to accurately extract its corresponding attribute values from the unstructured text, the existing record linkage approaches could be adopted to address the entity linking problem. However, the corresponding attribute values for the entity mention may not exist in the text and such information extraction task is difficult. Additionally, the string comparison methods [34,42,43] proposed in record linkage could be exploited to generate candidate entities in the Candidate Entity Generation module introduced in Section 2. In summary, entity linking is different from entity coreference resolution, word sense disambiguation, and record linkage.

Generally speaking, a typical entity linking system consists of the following three modules:

- **Candidate Entity Generation**
  In this module, for each entity mention $m \in M$, the entity linking system aims to filter out irrelevant entities in the knowledge base and retrieve a candidate entity set E_m which contains possible entities that entity mention m may refer to. To achieve this goal, a variety of techniques have been utilized by some state-of-the-art entity linking systems, such as name dictionary based techniques, surface form expansion from the local document, and methods based on search engine. A detailed survey for techniques used in this module is given in Section 2.
- **Candidate Entity Ranking**
  In most cases, the size of the candidate entity set E_m is larger than one. Researchers leverage different kinds of evidence to rank the candidate entities in E_m and try to find the entity $e \in E_m$ which is the most likely link for mention m. In Section 3, we will review the main techniques used in this ranking process, including supervised ranking methods and unsupervised ranking methods.
- **Unlinkable Mention Prediction**
  To deal with the problem of predicting unlinkable mentions, some work leverages this module to validate whether the top-ranked entity identified in the Candidate Entity Ranking module is the target entity for mention m. Otherwise, they return NIL for mention m. In Section 4, we will give an overview of the main approaches for predicting unlinkable mentions.

1.3 Applications

As briefly introduced in Section 1.1, entity linking is essential to many different tasks. Here we present several typical applications.

1.3.1 Information Extraction

Named entities and relations extracted by information extraction systems are usually ambiguous. Linking them with a knowledge base is a good way to disambiguate and fine-grained typing them, which is essential for their further exploitation. Lin et al. [44] proposed an efficient entity linking technique to link entity mentions in 15 million textual extractions from the Web with Wikipedia. They stated that entity linking for these extracted relations would offer benefits, such as semantically typing textual relations, integration with linked data resources, and inference rule learning. PATTY [45] is another good example for this case. Its goal is to construct a taxonomy of relational patterns with semantic types. PATTY first extracts binary relations between entities from the Web. In order to leverage these extracted relations to construct the relational pattern taxonomy, it first

employs entity linking techniques to link entities in the extracted relations with YAGO2 knowledge base [46] to disambiguate them.

1.3.2 Information Retrieval

The trend to advance the traditional keyword-based search to the semantic entity-based search has attracted a lot of attention in recent years. Semantic entity-based search [47,48,49,50] certainly benefits from entity linking, as it inherently needs disambiguated entity mentions appearing in Web text to deal with the semantics of entities and Web documents more precisely. In addition, query ambiguity is among the problems that undermine the quality of search results. Named entities usually appear in search queries and they are undoubtedly ambiguous [51]. For example, the entity mention "New York" in the search query could mean many different entities, such as the state of New York, the city of New York, a historical novel by Edward Rutherfurd whose name is "New York", and many songs whose names are "New York". Linking these ambiguous entity mentions in search queries with a knowledge base using the query context and the user's search history could potentially improve both the quality of search results as well as the user click-through experience.

1.3.3 Content Analysis

The analysis of the general content of a text in terms of its topics, ideas, categorizations, etc., could definitely benefit from the application of entity linking. Content-based news recommendation systems [52,53] require the topical analysis of news articles to recommend interesting news for users. Linking entities in news articles with a knowledge base makes better topical content analysis. In addition, Twitter[4] has become an increasingly important source of information recently. Discovering the topics of interest for a particular Twitter user allows for recommending and searching Twitter users based on their topics of interest [54]. Researchers [55] discovered Twitter users' topics of interest by first detecting and linking named entities mentioned in their tweets with a knowledge base. Then they utilized the categories of linked entities obtained from the knowledge base to characterize the users' topics of interest. As another example, the needs to collect opinions or information about some products, events, celebrities, or some other named entities across documents also require the process of linking named entity mentions with a knowledge base [56].

1.3.4 Question Answering

As stated above, most question answering systems leverage their supported knowledge bases to give the answer to the user's question. To answer the question such as "Which university is the professor Michael Jordan affiliated with?", the system has to first disambiguate the entity mention "Michael Jordan". They could leverage the entity linking technique to map the queried "Michael Jordan" to the Berkeley professor, and then it retrieves his affiliated university from the knowledge base directly to answer the user's question. Gattani et al. [56] interpreted a user query on kosmix.com via linking entities in the query with a knowledge base. Additionally, some question answering systems like Watson [57] exploit the entity linking technique to predict the types of questions and candidate answers, and obtain promising results.

1.3.5 Knowledge Base Population

As the world evolves, new facts are generated and digitally expressed on the Web. Automatically populating and enriching existing knowledge bases with newly extracted facts have become a key issue for Semantic Web and knowledge management techniques. Entity linking is inherently considered as an important subtask for knowledge base population. Given a relation or fact which needs to be populated into a knowledge base, if the entity mention associated with the relation has its corresponding entity record in the knowledge base, the entity linking task should be conducted and this entity mention should be linked with its corresponding entity in the knowledge base. Therefore, the knowledge base population task could potentially benefit from the entity linking problem.

1.4 Preliminaries

A knowledge base is a fundamental component for the entity linking task. Knowledge bases provide the information about the world's entities (e.g., the entities of *Albert Einstein* and *Ulm*), their semantic categories (e.g., *Albert Einstein* has a type of *Scientist* and *Ulm* has a type of *City*), and the mutual relationships between entities (e.g., *Albert Einstein* has a relation named *bornIn* with *Ulm*). In the following, we provide a brief introduction to four knowledge bases which have been widely exploited in the field of entity linking.

- **Wikipedia**[5] is a free online multilingual encyclopedia created through decentralized, collective efforts of thousands of volunteers around the world. At present, Wikipedia has become the largest and most popular Internet encyclopedia in the world and is also a very dynamic and quickly growing resource. The basic entry in Wikipedia is an article, which defines and describes an entity or a topic, and each article in Wikipedia is uniquely referenced by an identifier. Currently, English Wikipedia contains over 4.4 million articles. Wikipedia has a high coverage of named

4. https://twitter.com/

5. http://www.wikipedia.org/

entities and contains massive knowledge about notable named entities. Besides, the structure of Wikipedia provides a set of useful features for entity linking, such as entity pages, article categories, redirect pages, disambiguation pages, and hyperlinks in Wikipedia articles.

- **YAGO** [2] is an open-domain knowledge base combining Wikipedia and WordNet [32] with high coverage and quality. On one hand, YAGO has a large number of entities in the same order of magnitude as Wikipedia. On the other hand, it adopts the clean taxonomy of concepts from WordNet. Currently, the latest version of YAGO contains more than 10 million entities (such as people, organizations, locations, etc.), and has 120 million facts about these entities[6], including the Is-A hierarchy (such as *type* relation and *subclassOf* relation) as well as non-taxonomic relations between entities (such as *livesIn* relation and *graduatedFrom* relation). In addition, the *means* relation in YAGO denotes the reference relationship between strings and entities (for example, "Einstein" *means Albert Einstein*). Hoffart et al. [58] harnessed this *means* relation in YAGO to generate candidate entities.
- **DBpedia** [1] is a multilingual knowledge base constructed by extracting structured information from Wikipedia such as infobox templates, categorization information, geo-coordinates, and links to external Web pages. The English version of the DBpedia knowledge base currently describes 4 million entities, out of which 3.22 million are classified in a consistent ontology[7]. Moreover, it automatically evolves as Wikipedia changes.
- **Freebase** [3] is a large online knowledge base collaboratively created mainly by its community members. Freebase provides an interface that allows non-programmers to edit the structured data in it. Freebase contains data harvested from many sources including Wikipedia. Currently, it contains over 43 million entities and 2.4 billion facts about them[8].

1.5 Outline

In this survey, we carefully review and analyze the main techniques utilized in the three modules of entity linking systems as well as other critical aspects such as features and evaluation. To the best of our knowledge, this is the first paper which systematically surveys entity linking systems.

The rest of this article is organized as follows. We present and analyze the algorithms and features used in the three modules of entity linking systems (i.e., Candidate Entity Generation, Candidate Entity Ranking, and Unlinkable Mention Prediction) in Section 2, 3 and 4, respectively. Then, we introduce the evaluation of entity linking systems in Section 5. Finally, we conclude this paper and discuss future directions in Section 6.

6. http://yago-knowledge.org
7. http://dbpedia.org/About
8. http://www.freebase.com/

2 Candidate Entity Generation

As briefly introduced in Section 1.2, in the Candidate Entity Generation module, for each entity mention $m \in M$, entity linking systems try to include possible entities that entity mention m may refer to in the set of candidate entities E_m. Approaches to candidate entity generation are mainly based on string comparison between the surface form of the entity mention and the name of the entity existing in a knowledge base. This module is as important as the Candidate Entity Ranking module and critical for a successful entity linking system according to the experiments conducted by Hachey et al. [33]. In the remainder of this section, we review the main approaches that have been applied for generating the candidate entity set E_m for entity mention m.

Specifically, in Section 2.1, we describe the name dictionary based techniques. In Section 2.2, we present the surface form expansion approaches to expanding the surface form of an entity mention into a richer form from the local document where the entity mention appears. In Section 2.3, we list the approaches that are based on search engines.

2.1 Name Dictionary Based Techniques

Name dictionary based techniques are the main approaches to candidate entity generation and are leveraged by many entity linking systems [22,56,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74, 75,76,77,78,79]. The structure of Wikipedia provides a set of useful features for generating candidate entities, such as entity pages, redirect pages, disambiguation pages, bold phrases from the first paragraphs, and hyperlinks in Wikipedia articles. These entity linking systems leverage different combinations of these features to build an offline name dictionary D between various names and their possible mapping entities, and exploit this constructed name dictionary D to generate candidate entities. This name dictionary D contains vast amount of information about various names of named entities, like name variations, abbreviations, confusable names, spelling variations, nicknames, etc.

Specifically, the name dictionary D is a $\langle key, value \rangle$ mapping, where the *key* column is a list of names. Suppose k is a name in the *key* column, and its mapping value $k.value$ in the *value* column is a set of named entities which could be referred to as the name k. The dictionary D is constructed by leveraging features from Wikipedia as follows:

- **Entity pages.** Each entity page in Wikipedia describes a single entity and contains the information focusing on this entity. Generally, the title of each page is the most common name for the entity described in this page, e.g., the page title "Microsoft" for that giant software company headquartered in Redmond. Thus, the title of the entity page is added to the *key* column in D as a name k, and the entity described in this page is added as $k.value$.
- **Redirect pages.** A redirect page exists for each alternative name which could be used to refer to an existing entity in Wikipedia. For example, the article titled "Microsoft Corporation" which is the full name of *Microsoft* contains a pointer to the article of the entity *Microsoft*. Redirect pages often indicate synonym terms, abbreviations, or other variations of the pointed entities. Therefore, the title of the redirect page is added to the *key* column in D as a name k, and the pointed entity is added as $k.value$.
- **Disambiguation pages.** When multiple entities in Wikipedia could be given the same name, a disambiguation page is created to separate them and contains a list of references to those entities. For example, the disambiguation page for the name "Michael Jordan" lists thirteen associated entities having the same name of "Michael Jordan" including the famous NBA player and the Berkeley professor. These disambiguation pages are very useful in extracting abbreviations or other aliases of entities. For each disambiguation page, the title of this page is added to the *key* column in D as a name k, and the entities listed in this page are added as $k.value$.
- **Bold phrases from the first paragraphs.** In general, the first paragraph of a Wikipedia article is a summary of the whole article. It sometimes contains a few phrases written in bold. Varma et al. [63,64] observed that these bold phrases invariably are nick names, alias names or full names of the entity described in this article. For instance, in the first paragraph of the entity page of *Hewlett-Packard*, there are two phrases written in bold (i.e., "Hewlett-Packard Company" and "HP") which are respectively the full name and the abbreviation for the entity *Hewlett-Packard*. Thus, for each of the bold phrases in the first paragraph of each Wikipedia page, it is added to the *key* column in D as a name k, and the entity described in this page is added as $k.value$.
- **Hyperlinks in Wikipedia articles.** An article in Wikipedia often contains hyperlinks which link to the pages of the entities mentioned in this article. The anchor text of a link pointing to an entity page provides a very useful source of synonyms and other name variations of the pointed entity, and could be regarded as a name of that linked entity. For example, in the entity page of *Hewlett-Packard*, there is a hyperlink pointing to the entity *William Reddington Hewlett* whose anchor text is "Bill Hewlett", which is an alias name of the entity *William Reddington Hewlett*. Hence, the anchor text of the hyperlink is added to the *key* column in D as a name k, and the pointed entity is added as $k.value$.

TABLE 1
A part of the name dictionary D

k (Name)	$k.value$ (Mapping entity)
Microsoft	*Microsoft*
Microsoft Corporation	*Microsoft*
Michael Jordan	*Michael Jordan* *Michael I. Jordan* *Michael Jordan (footballer)* *Michael Jordan (mycologist)* ...
Hewlett-Packard Company	*Hewlett-Packard*
HP	*Hewlett-Packard*
Bill Hewlett	*William Reddington Hewlett*

Using these features from Wikipedia described above, entity linking systems could construct a dictionary D. A part of the dictionary D is shown in Table 1. Besides leveraging the features from Wikipedia, there are some studies [80,81,82] that exploit query click logs and Web documents to find entity synonyms, which are also helpful for the name dictionary construction.

Based on the dictionary constructed in this way, the simplest approach to generating the candidate entity set E_m for entity mention $m \in M$ is exact matching between the name k in the *key* column and the entity mention m. If some k equals m, the set of entities $k.value$ are added to the candidate entity set E_m.

Besides exact matching, some methods [67,69,70,83, 84] utilize partial matching between the entity name k in the dictionary D and the entity mention m. The common rules used by these approaches include:

- The entity name is wholly contained in or contains the entity mention.
- The entity name exactly matches the first letters of all words in the entity mention.
- The entity name shares several common words with the entity mention.
- The entity name has a strong string similarity with the entity mention. Many string similarity measures have been used, such as character Dice score, skip bigram Dice score, Hamming distance, etc. Since string comparison techniques are not the focus of this survey, some comprehensive surveys of these techniques could be found in the papers [34,42,43].

For each entity mention $m \in M$, if some entity name k in the *key* column satisfies one of the above rules, the set of entities $k.value$ are added to the candidate entity set E_m. Compared with exact matching, partial matching leads to higher recall, but more noise in the

candidate entity set.

Before matching with the dictionary, some approaches address the misspelling problem existing in the entity mention, which is very serious and needs to be addressed particularly. Varma et al. [63] used the metaphone algorithm [85] to identify spelling variations for a given entity mention. Chen et al. [67] obtained the suggested correct string using the spellchecker in Lucene. Nemeskey et al. [86] tackled this misspelling problem by the built-in spell corrector of SZTAKI information retrieval engine [87]. Zhang et al. [65] proposed to use one feature from Wikipedia search engine (i.e., *Did you mean*) to identify spelling variations for a given entity mention. Additionally, several systems [64,68,78] try to correct the misspelling existing in the entity mention using the query spelling correction service supplied by Google search engine.

2.2 Surface Form Expansion From The Local Document

Since some entity mentions are acronyms or part of their full names, one category of entity linking systems use the surface form expansion techniques to identify other possible expanded variations (such as the full name) from the associated document where the entity mention appears. Then they could leverage these expanded forms to generate the candidate entity set using other methods such as the name dictionary based techniques introduced above. We categorize the surface form expansion techniques into the heuristic based methods and the supervised learning methods.

2.2.1 Heuristic Based Methods

For the entity mention in the form of the acronym, some approaches [61,67,69] expand it by searching the textual context around the entity mention through the heuristic pattern matching. The most common patterns they leverage are an acronym that is in parenthesis adjacent to the expansion (e.g., Hewlett-Packard (HP)) and an expansion that is in parenthesis adjacent to the acronym (e.g., UIUC (University of Illinois at Urbana-Champaign)). In addition, some researchers [63,66,68,86] identified the expanded form from the whole document where the entity mention is located via N-Gram based approach. They checked whether there exist 'N' continuous words in the whole document after removing stop words having the same initials as the characters of the acronym. If exist, they considered these 'N' continuous words as the expanded form of the acronym. Furthermore, Varma et al. [64] and Gottipati and Jiang [77] used an off-the-shelf named entity recognizer (NER) to identify named entities from the document and if some identified named entity contains the entity mention as a substring, they considered this named entity as an expanded form for the entity mention. For example, if an NER identifies "Michael I. Jordan" as a person name from the document where the entity mention "Jordan" appears, "Michael I. Jordan" is regarded as an expanded form for the entity mention "Jordan". Cucerzan [71] employed an acronym detector [88] that utilizes the Web data to identify expansions for acronyms.

2.2.2 Supervised Learning Methods

Previous heuristic-based methods for surface form expansion could not identify the expanded form for some complicated acronym such as swapped or missed acronym letters (e.g., "CCP" for "Communist Party of China" and "DOD" for "United States Department of Defense"). Zhang et al. [72] proposed a supervised learning algorithm to find the expanded forms for complicated acronyms, which leads to 15.1% accuracy improvement (evaluation measures for entity linking will be introduced in Section 5.1) over the state-of-the-art acronym expansion methods. Specifically, they identified possible candidate expansions from the document through some predefined strategies including text markers (such as "Hewlett-Packard (HP)" and "HP (Hewlett-Packard)") and first letter matching (i.e., all the word sequences in the document which begin with the same first letter as the acronym and do not contain punctuation or more than two stop words are extracted as candidate expansions). For example, from the sentence "Communist Party of China leaders have granted the ...", with respect to the acronym "CCP", they extracted "Communist Party of China leaders have" containing two stop words and all its subtrings beginning with the first matching word as candidate expansions. Then each pair of an acronym and one of its candidate expansions is represented as a feature vector, including part of speech features and the alignment information between the acronym and the expansion. An SVM (Support Vector Machines) classifier is applied to each candidate acronym-expansion pair to output a confidence score. For each acronym, the candidate expansion with the highest score is selected. The training data for this classifier consists of 170 acronyms and their expansions from documents where the acronyms locate.

2.3 Methods Based on Search Engines

Some entity linking systems [61,69,73,83] try to leverage the whole Web information to identify candidate entities via Web search engines (such as Google). Specifically, Han and Zhao [61] submitted the entity mention together with its short context to the Google API and obtained only Web pages within Wikipedia to regard them as candidate entities. Dredze et al. [83] queried the Google search engine using the entity mention and identified candidate entities whose Wikipedia pages appear in the top 20 Google search

results for the query. Lehmann et al. [69] and Monahan et al. [73] stated that the Google search engine is very effective at identifying some of the very difficult mappings between surface forms and entities. They performed the query using the Google API limited to the English Wikipedia site and filtered results whose Wikipedia titles are not significantly Dice or acronym-based similar to the query. Lastly, they utilized the top three results as candidate entities.

In addition, Wikipedia search engine is also exploited to retrieve candidate entities which can return a list of relevant Wikipedia entity pages when you query it based on keyword matching. Zhang et al. [65] utilized this feature to generate infrequently mentioned candidate entities by querying this search engine using the string of the entity mention.

3 Candidate Entity Ranking

In the previous section, we described methods that could generate the candidate entity set E_m for each entity mention m. We denote the size of E_m as $|E_m|$, and use $1 \leq i \leq |E_m|$ to index the candidate entity in E_m. The candidate entity with index i in E_m is denoted by e_i. In most cases, the size of the candidate entity set E_m is larger than one. For instance, Ji et al. [89] showed that the average number of candidate entities per entity mention on the TAC-KBP2010 data set (TAC-KBP tracks and data sets will be introduced in Section 5.2) is 12.9, and this average number on the TAC-KBP2011 data set is 13.1. In addition, this average number is 73 on the *CoNLL* data set utilized in [58]. Therefore, the remaining problem is how to incorporate different kinds of evidence to rank the candidate entities in E_m and pick the proper entity from E_m as the mapping entity for the entity mention m. The Candidate Entity Ranking module is a key component for the entity linking system. We can broadly divide these candidate entity ranking methods into two categories:

- **Supervised ranking methods.** These approaches rely on annotated training data to "learn" how to rank the candidate entities in E_m. These approaches include binary classification methods, learning to rank methods, probabilistic methods, and graph based approaches.
- **Unsupervised ranking methods.** These approaches are based on unlabeled corpus and do not require any manually annotated corpus to train the model. These approaches include Vector Space Model (VSM) based methods and information retrieval based methods.

In this section, all candidate entity ranking methods are illustrated according to the above categorization. In addition, we could also categorize the candidate entity ranking methods into another three categories:

- **Independent ranking methods.** These approaches consider that entity mentions which need to be linked in a document are independent, and do not leverage the relations between the entity mentions in one document to help candidate entity ranking. In order to rank the candidate entities, they mainly leverage the context similarity between the text around the entity mention and the document associated with the candidate entity [44,59,66,67,68,70,83,84].
- **Collective ranking methods.** These methods assume that a document largely refers to coherent entities from one or a few related topics, and entity assignments for entity mentions in one document are interdependent with each other. Thus, in these methods, entity mentions in one document are collectively linked by exploiting this "topical coherence" [22,58,60,62,75,76,78,79,90,91,92,93].
- **Collaborative ranking methods.** For an entity mention that needs to be linked, these approaches identify other entity mentions having similar surface forms and similar textual contexts in the other documents. They leverage this cross-document extended context information obtained from the other similar entity mentions and the context information of the entity mention itself to rank candidate entities for the entity mention [94,95,96,97].

In the remainder of this section, we will review the main techniques used in the candidate entity ranking process. Firstly in Section 3.1, we review the various types of features found to be useful in candidate entity ranking. Then in Section 3.2, we introduce the supervised ranking methods. Specifically, in Section 3.2.1 we depict the binary classification methods, and in Section 3.2.2 we introduce the learning to rank methods. In Section 3.2.3, we describe the probabilistic methods and Section 3.2.4 covers the graph based approaches. In Section 3.2.5, we introduce the model combination strategy for entity ranking and in Section 3.2.6, we illustrate how to automatically generate the annotated training data. In Section 3.3 we introduce the unsupervised ranking methods. Specifically, in Section 3.3.1 we list the VSM based approaches, and in Section 3.3.2 we present the information retrieval based methods.

3.1 Features

In this subsection, we review the various types of features found to be useful in candidate entity ranking. We divide these features into context-independent features and context-dependent features. Context-independent features just rely on the surface form of the entity mention and the knowledge about the candidate entity, and are not related to the context where the entity mention appears. Context-dependent features are based on the context where the entity mention appears. Here, the context means not only the textual context around the entity mention, but also

other entity mentions which need to be linked in the same document.

3.1.1 Context-Independent Features

3.1.1.1 Name String Comparison: The name string comparison between the entity mention and the candidate entity is the most direct feature that one may use. Many string similarity measures have been used in the name comparison, including edit distance [68,96], Dice coefficient score [69,73], character Dice, skip bigram Dice, and left and right Hamming distance scores [83]. The common name comparison features include:

- Whether the entity mention exactly matches the candidate entity name.
- Whether the candidate entity name starts with or ends with the entity mention.
- Whether the candidate entity name is the prefix of or postfix of the entity mention.
- Whether the entity mention is wholly contained in the candidate entity name, or vice-versa.
- Whether all of the letters of the entity mention are found in the same order in the candidate entity name.
- The number of same words between the entity mention and the candidate entity name.
- The ratio of the recursively longest common subsequence [98] to the shorter among the entity mention and the candidate entity name.

In addition, Dredze et al. [83] computed the name similarity by training finite-state transducers similar to those described in [99]. These transducers assign a score to any string pair by summing over all alignments and scoring all contained character n-grams. Finally, the scores are combined using a global log-linear model.

3.1.1.2 Entity Popularity: Another context-independent feature found to be very useful in entity linking is the popularity of the candidate entity with regard to the entity mention, which tells us the prior probability of the appearance of a candidate entity given the entity mention. The observation they have is that, each candidate entity $e_i \in E_m$ having the same mention form m has different popularity, and some entities are very obscure and rare for the given mention form m. For example, with respect to the entity mention "New York", the candidate entity *New York (film)* is much rarer than the candidate entity *New York City*, and in most cases when people mention "New York", they mean the city of New York rather than the film whose name is also "New York". Many state-of-the-art entity linking systems [22,58,62,73,76,78,79,92,96] formalize this observation via taking advantage of the count information from Wikipedia, and define the *popularity* feature $Pop(e_i)$ for each candidate entity $e_i \in E_m$ with respect to the entity mention m as the proportion of links with the mention form m as the anchor text which point to the candidate entity e_i :

$$Pop(e_i) = \frac{count_m(e_i)}{\sum_{e_j \in E_m} count_m(e_j)}$$

where $count_m(e_i)$ is the number of links which point to the entity e_i and have the mention form m as the anchor text.

Some studies [22,56] utilize the Wikipedia page view statistics associated with each candidate entity to estimate the entity popularity. To express the popularity of the candidate entity, Dredze et al. [83] added features obtained from the Wikipedia graph structure for the candidate entity, like the indegree of the node, the outdegree of the node, and the Wikipedia page length in bytes. Moreover, they used Google's PageRank to add a feature indicating the rank of the candidate entity's corresponding Wikipedia page in the Google result page for the query of the entity mention.

Since most entity mentions appearing in text represent salient entities, only using the entity popularity feature could yield correct answers in most cases. The experiments conducted by Ji and Grishman [100] show that a naïve candidate ranking method only based on the Web popularity can achieve 71% accuracy, which is better than 24 system runs in the TAC-KBP2010 track. Therefore, we can say that the entity popularity feature is significantly important and effective for the entity linking task.

3.1.1.3 Entity Type: This feature is to indicate whether the type of the entity mention (i.e., people, location, and organization) in text is consistent with the type of the candidate entity in a knowledge base. Nemeskey et al. [86] used their in-house Named Entity Recognizer [101] to identify the entity type for the entity mention in text and some candidate entity whose type is unavailable in the knowledge base. Dredze et al. [83] inferred entity types for candidate entities from their infobox class information in Wikipedia. Lehmann et al. [69] and Monahan et al. [73] used LCC's CiceroLite NER system [102] to determine the entity type for the entity mention in text, while for the entity type of the candidate entity, they employed a cascade of resources beginning with the knowledge base where the candidate entity exists. If the type is unknown in the knowledge base, DBpedia is consulted. As a last resort, LCC's WRATS ontology resource is consulted. Entity types from CiceroLite, DBpedia, and WRATS are all reduced to the three common entity types (i.e., people, location, and organization).

Although context-independent features are useful, they provide information only from the entity mention and the candidate entity. It is very necessary to use features related to the context where the entity mention appears. We discuss this issue in the following.

3.1.2 Context-Dependent Features

3.1.2.1 Textual Context: The most straightforward feature about the textual context is to measure the textual similarity between the context around the entity mention and the document associated with the candidate entity. Various forms have been utilized to represent the context:

- **Bag of words.** For each entity mention, the context is represented as a bag of words collected from the entire input document where the entity mention appears [22,66,67,70,90,96] or a suitable window around the entity mention in the document [44,59,62,75,76,79]. For each candidate entity, the context is usually represented as a bag of words from the whole Wikipedia entity page [44,59,62,66,70,75,96], the first description paragraph of its Wikipedia page [62], a suitable window around each occurence of that entity in the Wikipedia page corpus [79], or the top-*k* token TF-IDF summary of the Wikipedia page [22,76].
- **Concept vector.** For the general document where the entity mention appears or the Wikipedia article for the candidate entity, systems extract some keyphrases [58], anchor texts [62], named entities [65,83,94], categories [60,83], descriptive tags [56], and Wikipedia concepts [61,71,73,78] from it to compose a concept vector to represent the semantic content of the document. Moreover, the context for the candidate entity could be represented by its related linked entities in Wikipedia, its attributes, as well as its relevant facts known through Wikipedia infobox [67,69,83].

Based on these different formulations of representations, each text around the entity mention or associated with the candidate entity could be converted to a vector. To calculate the similarity between vectors, different methods have been utilized, including dot-product [22,62,75], cosine similarity [44,59,62,65,66,68, 70,76,79,83,90,94], Dice coefficient [83], word overlap [58,96], KL divergence [58], n-gram based measure [58], and Jaccard similarity [62].

In addition, Han and Sun [74] leveraged unigram language model to encode the context knowledge of each candidate entity, which could tell us the likelihood of an entity appearing in a specific context. Topic modeling [103] was also used to model the underlying semantic topics of documents to calculate the context similarity [70,72,96,104]. He et al. [105] proposed a deep learning technique [106] to automatically learn the context-entity similarity measure for entity linking based on the assumption that the correct mapping entity should be more similar to the context of the entity mention than any other candidate entity. Recently, Li et al. [107] proposed a generative model to augment the context information for entities in a knowledge base in the form of entity-word distribution mined from internal and external corpus.

3.1.2.2 Coherence Between Mapping Entities: The textual context around the entity mention undoubtedly plays a vital role in entity linking. Additionally, for an entity mention, other entity mentions that need to be linked in the same document are also important for its linking. Many state-of-the-art entity linking systems assume that a document largely refers to coherent entities from one or a few related topics, and this topical coherence could be exploited for collectively linking entity mentions in the same document. Therefore, they leverage the feature of topical coherence between mapping entities in one document to aid in linking entities [22,58,60,62,75,76, 78,79,90,91,92,93].

To measure the coherence between mapping entities, Cucerzan [60] first leveraged the agreement between categories of two candidate entities. Additionally, some approaches [58,62,75,76,78,79,91,96] adopt the Wikipedia Link-based Measure (WLM) described in [108,109] to calculate the topical coherence between Wikipedia entities under the assumption that two Wikipedia entities are considered to be semantically related if there are many Wikipedia articles that link to both. The WLM is modeled from the Normalized Google Distance [110]. Given two Wikipedia entities u_1 and u_2, the topical coherence between them is defined as follows:

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

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. In addition to Normalized Google Distance model, Ratinov et al. [76] proposed to use the PMI-like (Point-wise Mutual Information) measure to calculate the topical coherence between Wikipedia entities:

$$Coh_P(u_1, u_2) = \frac{|U_1 \cap U_2|/|WP|}{|U_1|/|WP| \cdot |U_2|/|WP|}$$

Furthermore, Guo et al. [22] calculated the Jaccard distance to measure the topical coherence between Wikipedia entities:

$$Coh_J(u_1, u_2) = \frac{|U_1 \cap U_2|}{|U_1 \cup U_2|}$$

The above three measures [22,76,108] are based on the link structure of Wikipedia. However, for long tail and newly emerging entities that have few or no links associated with them, these three measures cannot work well. To address this problem, Hoffart et al. [111] proposed an efficient measure called KORE that calculates the topical coherence between two entities represented as sets of weighted (multi-word) keyphrases, with consideration of partially overlapping phrases. For efficiency, they used a two-level approximation technique based on min-hash sketches and locality-sensitive hashing.

Recently, Ceccarelli et al. [112] proposed to utilize the learning to rank model to learn the topical coherence between entities for entity linking under the assumption that a good measure should promote the topical coherence between correct mapping entities. The learned measure is the weighted combination of 27 different measures between entities including WLM [108], point-wise mutual information [76] and Jaccard similarity [22] between their in-link article sets. The experimental results show their learned measure performs better than other previously proposed measures. However, it is more time consuming than other measures. In addition, Han and Sun [93] modeled the topical coherence via topic modeling techniques.

To measure the coherence between entities in a Web list, Shen et al. [92] leveraged two categories of information: (1) type hierarchy based similarity that is based on the assumption that two entities are semantically similar if they are in close places in the type hierarchy; (2) distributional context similarity that is based on the assumption that entities that occur in similar contexts are semantically similar, which is an extension of the distributional hypothesis.

Although the feature of coherence between mapping entities is found to be very effective in the entity linking task [22,58,60,62,75,76,78,79,90,91,92,93], the calculation of this feature is not easy and straightforward. To calculate this feature for one entity mention, systems have to be aware of mapping entities for other entity mentions in the same document. Unfortunately, these mapping entities are unknown to us and need to be assigned in this task. Therefore, entity assignments for entity mentions in one document are interdependent with each other. According to the work [58,62,92,96], the optimization of this problem is shown to be NP-hard, which makes this feature computationally expensive and time consuming for real world applications.

3.1.3 Discussion: Features

The large number of features introduced here reflect the large number of aspects an entity linking system could consider when dealing with the entity linking task. Unfortunately, there are very few studies that compare the effectiveness of the various features presented here. However, we emphasize that no features are superior than others over all kinds of data sets. Even some features that demonstrate robust and high performance on some data sets could perform poorly on others. Hence, when designing features for entity linking systems, the decision needs to be made regarding many aspects, such as the tradeoff between accuracy and efficiency, and the characteristics of the applied data set.

3.2 Supervised Ranking Methods

Supervised ranking methods use the annotated data set to "learn" how to assign the proper mapping entity to each entity mention. The training data set typically contains a set of examples in which each entity mention is manually annotated with its mapping entity. In the remainder of this subsection, we review the supervised ranking methods used for ranking candidate entities in detail.

3.2.1 Binary Classification Methods

Some systems [63,65,66,69,73,94,104] formulate the candidate entity ranking problem as a binary classification problem. Given a pair of an entity mention and a candidate entity, they use a binary classifier to decide whether the entity mention refers to the candidate entity. The training or test instance is formed by a pair of an entity mention and a candidate entity $\langle m, e_i \rangle$. The label of this instance is positive if the entity mention m refers to the entity e_i, otherwise it is negative. During the training phase, many labeled $\langle m, e_i \rangle$ pairs are used to learn the classifier. During the test phase, each test $\langle m, e_i \rangle$ pair is presented to the classifier which then outputs a class label indicating positive or negative for it. Each $\langle m, e_i \rangle$ pair is represented as a feature vector consisting of the features described in Section 3.1. For one entity mention, if there are two or more candidate entities that are labeled positive, some techniques are employed to select the most likely one, such as confidence-based methods [63,69,104], VSM based methods [65], and SVM ranking models [66]. For the binary classifier, most systems employ Support Vector Machines (SVM) [65,66,94,104]. Support Vector Machines [113] are based on the idea of learning a hyperplane from the training data set that separates positive examples from negative examples. The hyperplane is located in that point of the hyperspace which maximizes the distance to the closest positive and negative examples. Besides the SVM classifier, Lehmann et al. [69] and Monahan et al. [73] utilized the binary logistic classifier, and Varma et al. [63] used the Naïve Bayes classifier and the K-Nearest Neighbors classifier.

3.2.2 Learning to Rank Methods

Although the binary classification approach is a natural and simple way to deal with the candidate entity ranking task, it has several drawbacks. Firstly, the training data is very unbalanced since the vast majority of candidate entities are negative examples. Moreover, when multiple candidate entities for an entity mention are classified as positive by the binary classifier, they have to utilize other techniques to select the most likely one.

Instead, many entity linking systems [59,62,66,68, 70,72,76,78,83,84,92,94] exploit the learning to rank framework [114] to give a rank to the candidate

entity set and take into account relations between candidate entities for the same entity mention rather than considering them independently like the binary classifier. Learning to rank is a type of supervised techniques whose goal is to automatically construct a ranking model from training data. Training data for the learning to rank model consists of lists of items with some partial order specified between items in each list. While for the problem of entity linking, the approaches just focus on the single correct mapping entity in the candidate entity set and therefore impose a loose requirement, that is, the correct mapping entity should be ranked highest. This formulation addresses the problems with the binary classification. Firstly, training data is balanced since we have a single ranking example for each entity mention. Secondly, methods just need to select the candidate entity which achieves the highest score in the test phase as the mapping entity for each entity mention, rather than resorting to other techniques to select the most likely one. In this learning framework, each instance is also formed by a feature vector consisting of the features described in Section 3.1.

Most entity linking systems that leverage the learning to rank framework [59,62,66,70,72,76,78,83,84,92, 94] utilize the ranking SVM framework [115,116] to learn the ranking model. They use a max-margin technique based on the training data set. They assume that given the ground truth mapping entity $e^m \in E_m$ for each entity mention m, the score of the correct mapping entity $Score(e^m)$ should be higher than the score of any other candidate entity $Score(e_i)$ with a margin, where $e_i \in E_m$ and $e_i \neq e^m$. This gives them the usual SVM linear constraints for all entity mentions:

$$\forall m, \forall e_i \neq e^m \in E_m : Score(e^m) - Score(e_i) \geq 1 - \xi_{m,i}$$

and they minimize over $\xi_{m,i} \geq 0$ and the objective $||w||_2^2 + C\Sigma_{m,i}\xi_{m,i}$ where C is a parameter that allows tradeoff between the margin size and the training error.

LINDEN [78] gives a rank to candidate entities for each entity mention with a linear combination of four features: entity popularity, *semantic associativity* (i.e., semantic context similarity based on Wikipedia hyperlink structure), *semantic similarity* (i.e., semantic context similarity derived from the taxonomy of YAGO), and global topical coherence between mapping entities. LINDEN uses the max-margin technique introduced above to learn feature weights and achieves 84.3% accuracy over the TAC-KBP2009 data set.

Zheng et al. [68] investigated another two different learning to rank frameworks for ranking candidate entities: the pairwise framework Ranking Perceptron [117] and the listwise framework ListNet [118]. In their experiments, the learning to rank methods have been shown to achieve much better results in the candidate entity ranking task compared with the binary classification methods, and ListNet shows slight improvement over Ranking Perceptron. They achieved 84.9% overall accuracy on the TAC-KBP2009 data set. In addition, Chen and Ji [94] also leveraged the listwise ranker ListNet to rank candidates.

3.2.3 Probabilistic Methods

Kulkarni et al. [62] proposed an entity linking system which explicitly links all the entity mentions in one document collectively. Their guiding premise is that a document largely refers to topically coherent entities, and they exploited this "topical coherence" to deal with the candidate entity ranking problem. Their method starts with an SVM-based supervised learner for local context similarity, and models it in combination with pairwise document-level topical coherence of candidate entities using a probabilistic graphical model. The optimization of this model is shown to be NP-hard. To solve this problem, they resorted to approximations and heuristics like hill-climbing techniques and linear program relaxations. The experimental results show that it achieves 69% F_1-*measure* over their created IITB data set. However, even the approximate solution of this optimization model has high computational costs and is time consuming.

In order to deal with the table annotation task, Limaye et al. [119] proposed to simultaneously annotate table cells with entities, table columns with types and pairs of table columns with relations in a knowledge base. They modeled the table annotation problem using a number of interrelated random variables following a suitable joint distribution, and represented them using a probabilistic graphical model. The inference of this task is to search for an assignment of values to variables that maximizes the joint probability, which is NP-hard. They resorted to an approximate algorithm called message-passing [120] to solve this problem. The three subtasks in table annotation are collectively solved, which achieves better results compared with making decisions for each subtask individually.

Han and Sun [74] proposed *entity-mention model*, a generative probabilistic model, to link entity mentions in Web free text with a knowledge base. This model incorporates three types of heterogeneous knowledge (i.e., popularity knowledge, name knowledge, and context knowledge) into a unified probabilistic model for the entity linking task. Specifically, the popularity knowledge tells us the likelihood of an entity appearing in a document. The name knowledge tells us the possible names of an entity and the likelihood of a name referring to a specific entity. The context knowledge tells us the likelihood of an entity appearing in a specific context. In this model, each entity mention to be linked is modeled as a sample generated through a three-step generative story. The experimental results

show that this method could achieve as high as 86% accuracy over the TAC-KBP2009 data set.

Demartini et al. [121] proposed a system called *ZenCrowd* and attempted to take advantage of human intelligence to improve the quality of the entity linking result. They developed a probabilistic reasoning framework to dynamically make sensible decisions about entity linking with consideration of results from both human workers on the crowdsourcing platform and automatic machine-based techniques. If some entity linking results generated by machine-based techniques are deemed promising but uncertain, they are then used to dynamically generate micro-tasks, which are then published on a crowdsourcing platform. When the human workers on the crowdsourcing platform have performed these micro-tasks, their results are fed back to the probabilistic reasoning framework, which could generate the final result after combining the inconsistent output from arbitrary human workers.

3.2.4 Graph Based Approaches

Compared with the previous research work [62] which models document-level topical coherence of candidate entities in a pairwise fashion, Han et al. [75] proposed a graph based collective entity linking method to model the global topical interdependence (rather than the pairwise interdependence) between different entity linking decisions in one document. Firstly, they proposed a graph based representation, called *Referent Graph*, which could model both the textual context similarity and the global topical interdependence between entity linking decisions (i.e., the feature of coherence between mapping entities introduced in Section 3.1.2.2) as its graph structure. Then they utilized a purely collective inference algorithm over the *Referent Graph* to jointly infer mapping entities of all entity mentions in the same document, which is similar to the topic-sensitive PageRank algorithm [122]. The experimental results show that by modeling and leveraging the global interdependence, Han et al. [75] could further improve the entity linking performance than the pairwise interdependence model [62] with an F_1-*measure* of 73% over the IITB data set.

In the meantime, Hoffart et al. [58] also proposed a graph based approach for collective entity linking. This model combines three features into a graph model: entity popularity, textual context similarity as well as coherence between mapping entities. They built a mention-entity graph, a weighted and undirected graph with entity mentions and candidate entities as nodes. In this mention-entity graph, a mention-entity edge is weighted with a combination of the entity popularity feature and the textual context similarity feature, and an entity-entity edge is weighted by the Wikipedia hyperlink structure based coherence (see Section 3.1.2.2). Given this constructed graph, their goal is to compute a dense subgraph that contains exactly one mention-entity edge for each entity mention. However, this problem is NP-hard as it generalizes the well studied Steiner-tree problem. To solve this problem, Hoffart et al. [58] developed a greedy algorithm with the extension of the algorithm proposed in [123]. The experimental results show that it outperforms the collective entity linking system [62] and the approach of Cucerzan [60], and achieves 81.8% accuracy over their own *CoNLL* data set.

Shen et al. [79] proposed a graph-based framework called KAURI 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. Their 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 model. For the intra-tweet local information, KAURI leverages three features: entity popularity, textual context similarity, and coherence between entities within a tweet. As a single tweet may be too short and noisy to provide sufficient context information for entity linking, KAURI exploits the user interest information across tweets by modeling the user's topics of interest. The experimental results show that it significantly outperforms LINDEN [78] and many baselines in terms of accuracy, and scales well to tweet stream.

3.2.5 Model Combination

Model combination, also called ensemble method, typically aggregates together learning algorithms with significantly different nature and characteristics [124, 125], and seeks to obtain better predictive performance than any of the models they combine [126]. Model combination becomes increasingly popular as it allows one to overcome the weakness of a single model. Recently, the increasing number of diverse entity linking systems based on the various resources provide new opportunities to benefit from model combination for the entity linking task.

Zhang et al. [66] is the first to use the model combination strategy for the entity linking task. They developed three single systems (i.e., an information retrieval based system (see Section 3.3.2), a learning to rank based system, and a binary classification system), and combined them into a final system using a supervised method. An SVM three-class classifier was chosen to judge which of the three systems should be trusted. The experimental results show that the combined system performs better than each individual component and achieves 79.4% accuracy over the TAC-KBP2010 data set. Additionally, Ji and Grishman [100] also applied a voting approach on the top nine entity linking systems in the TAC-KBP2010 track and found that all combination orders achieve significant gains, with the highest absolute improvement of 4.7%

in accuracy over the top entity linking system in the TAC-KBP2010 track. Chen and Ji [94] used simple composite functions (e.g., *majority voting* and *weighted average*) to integrate eight baseline methods including four supervised approaches and four unsupervised approaches. The empirical results show that the combined model obtains absolute accuracy gain of 1.3% (*majority voting* function) and 0.5% (*weighted average* function) over the best baseline method. Furthermore, CUNY-UIUC-SRI system [95] combines the collaborative ranking framework described in [94] and the entity linking system described in [76] based on majority voting. This combined system achieves 77.1% F_1*-measure* on the TAC-KBP2011 data set.

3.2.6 Training Data Generation

One problem with the supervised ranking methods is the requirement of many annotated training examples to train the classifier. Moreover, entity linking annotation is expensive and very time consuming because of the large size of the referenced knowledge base. Some supervised ranking approaches train their models on a small manually created data set consisting of thousands of labeled entity mentions [78,83,84,127,128]. Some systems [59,76,129] use hyperlinks in Wikipedia articles to construct training data sets. However, these training data sets are created from Wikipedia which could not work well in a targeted new domain [65]. Based on this observation, Zhang et al. [65] proposed a novel method to automatically generate large scale annotation data. Specifically, they leveraged the unambiguous entity mention (i.e., the entity mention associated with only one entity in the knowledge base) in the document collection, and replaced it with its ambiguous name variations to create more training data. Furthermore, they also leveraged Wikipedia documents to provide additional information through a domain adaption approach [130]. At last, from 1.7 million documents, they generated 45,000 labeled instances. By leveraging the generated annotation data, they performed at 83.8% accuracy over the TAC-KBP2009 data set.

However, the distribution of the automatically generated annotation data is not consistent with the real entity linking data set. To solve this problem, Zhang et al. [72] used an instance selection strategy (similar to active learning [131,132]) to select a more balanced and informative subset from the generated instances. Finally, they reported 86.1% accuracy on the TAC-KBP2010 data set.

3.3 Unsupervised Ranking Methods

3.3.1 VSM Based Methods

In order to avoid manually annotating training data which is labor-intensive and costly, one simple way is to rank candidate entities using the unsupervised Vector Space Model (VSM) [133] based methods [60, 61,67]. They first calculate the similarity between the vectorial representation of the entity mention and the vectorial representation of the candidate entity. Then the candidate entity which achieves the highest similarity score is selected as the mapping entity for the entity mention. Those various approaches differ in methods of vectorial representation and vector similarity calculation.

Specifically, Cucerzan [60] extracted all the entity references mentioned in the candidate entity article and all the category tags associated with the candidate entity article to constitute the vector for the candidate entity. For the entity mention, Cucerzan built its vector through identifying the set of entity references appearing in its context. Lastly, this system identifies entity assignments to entity mentions through maximizing the vector similarity between the candidate entity and the entity mention, as well as the agreement among categories associated with candidate entities. Finally, this system achieves 91.4% accuracy over a news article data set.

Han and Zhao [61] first detected all the Wikipedia concepts from the context of the entity mention and the candidate entity article. The vector similarity is computed as the weighted average of all semantic relatedness [108] between Wikipedia concepts in vectors of the entity mention and the candidate entity. They reported 76.7% accuracy on the TAC-KBP2009 data set.

In addition, Chen et al. [67] generated vectors for the entity mention and the candidate entity using bags of words from their context and their related attributes. To calculate the similarity between vectors, they leveraged TF-IDF similarity. They obtained 71.2% accuracy on the TAC-KBP2010 data set.

3.3.2 Information Retrieval Based Methods

Some entity linking systems treat the candidate entity ranking problem as an information retrieval based ranking problem [63,64,66,77,86]. In their models, each candidate entity is indexed as a separate document, and for each entity mention, they generate a search query from the entity mention and its contextual document. Finally, the search query is given to the candidate entity index and the candidate entity which has the highest relevant score is retrieved as the mapping entity for the entity mention.

Gottipati and Jiang [77] leveraged a statistical language model based information retrieval approach to rank candidate entities. Specifically, they adopted the widely used KL-divergence retrieval model [134]. Given a candidate entity e and an entity mention m, they scored e based on the KL-divergence defined below:

$$s(e,m) = -Div(\theta_m||\theta_e) = -\sum_{w \in V} p(w|\theta_m) log \frac{p(w|\theta_m)}{p(w|\theta_e)}$$

where θ_m and θ_e are the entity mention language model and the candidate entity language model, respectively. V is the vocabulary and w is a single word. To estimate θ_e, they used the standard maximum likelihood estimation with Dirichlet smoothing [135] from the candidate entity name string and its disambiguation text. To estimate θ_m, they used the empirical word distribution from the entity mention string. In addition, they also used both the local contexts and the global world knowledge to expand the entity mention language model θ_m. Finally, they picked the candidate entity which has the highest score as the mapping entity for the entity mention m. This system shows competitive performance (i.e., 85.2% accuracy) on the TAC-KBP2010 data set.

4 Unlinkable Mention Prediction

In the previous section, we have reviewed the main techniques used to rank the candidate entities in E_m. The entity linking methods can pick the top-ranked entity e_{top} from E_m as the mapping entity for entity mention m. However, in practice, some entity mention does not have its corresponding record in a knowledge base. Therefore, they have to deal with the problem of predicting unlinkable mentions. In this section, we give a brief overview of the main approaches to predicting unlinkable mentions.

For the purpose of simplicity, many studies [60, 62,75,92,93,119,121] suppose that the knowledge base contains all the mapping entities for all the entity mentions, and thus ignore the unlinkable problem of entity mentions. Some approaches [63,67,86] utilize a simple heuristic method to predict unlinkable entity mentions. If the candidate entity set E_m for the mention m generated by the Candidate Entity Generation module is empty, they predict the mention m as unlinkable and return NIL for m.

Besides those methods, many entity linking systems [59,61,69,77,78,79,91,104,107] employ a NIL threshold method to predict the unlinkable entity mention. In these systems, the top-ranked entity e_{top} is associated with a score s_{top}. If the score s_{top} is smaller than a NIL threshold τ, they return NIL for the entity mention m and predict the mention m as unlinkable. Otherwise, they return e_{top} as the correct mapping entity for the mention m. The NIL threshold τ is usually automatically learned from the training data.

A large number of entity linking systems [66,68, 69,70,72,73,74,76,83,84] leverage supervised machine learning techniques to predict the unlinkable entity mention. Specifically, methods [66,68,69,70,72,73,76] utilize the binary classification technique. Given a pair of an entity mention and its top-ranked candidate entity $\langle m, e_{top}\rangle$, a binary classifier is used to decide whether the top-ranked candidate entity e_{top} is the correct mapping entity for this entity mention m, and outputs a label. If the label of the pair $\langle m, e_{top}\rangle$ is positive, they return the entity e_{top} as the correct mapping entity for m, otherwise they return NIL for the mention m. Each $\langle m, e_{top}\rangle$ pair is represented as a feature vector, and most features used in this module are the same as those used in the Candidate Entity Ranking module described in Section 3.1. Furthermore, Zheng et al. [68] and Ratinov et al. [76] designed some additional features for unlinkable mention prediction, such as the score of the top-ranked candidate and whether the entity mention is detected by some NER as a named entity. For the binary classifier, most systems [66,68,72,76] employ the SVM classifier.

In addition, Dredze et al. [83], McNamee [84], and Han and Sun [74] incorporated the unlinkable mention prediction process into the entity ranking process. Among them, Dredze et al. [83] and McNamee [84] used the learning to rank framework to rank candidate entities which has been introduced in Section 3.2.2. To predict the unlinkable mention, they added a NIL entity into the candidate entity set, and considered NIL as a distinct candidate. If the ranker outputs NIL as the top-ranked entity, this entity mention is considered as unlinkable. Otherwise, the top-ranked entity is returned as the correct mapping entity. The probabilistic model proposed in [74] also seamlessly takes into account the unlinkable entity prediction problem, rather than adding an additional step. The model assumes that for the entity mention which refers to some specific entity, the probability of this entity mention generated by this specific entity's model should be significantly higher than the probability of this mention generated by a general language model. It adds a NIL entity into the knowledge base and assumes that the NIL entity generates a mention according to the general language model. If the probability of some mention generated by the NIL entity is greater than the probability of this mention generated by any other entity in the knowledge base, this mention is predicted as unlinkable.

5 Evaluation

In this section, we introduce some issues related to the evaluation of entity linking systems: evaluation measures and entity linking data sets. With respect to the experimental performance of the state-of-the-art entity linking systems, we have discussed them when those systems were introduced in Section 3.

5.1 Evaluation Measures

The assessment of entity linking systems is usually performed in terms of evaluation measures, such as *precision*, *recall*, F_1*-measure*, and *accuracy*. The *precision* of an entity linking system is computed as the fraction of correctly linked entity mentions that are generated by the system:

$$precision = \frac{|\{correctly\ linked\ entity\ mentions\}|}{|\{linked\ mentions\ generated\ by\ system\}|}$$

Precision takes into account all entity mentions that are linked by the system and determines how correct entity mentions linked by the entity linking system are. Precision is usually used with the measure *recall*, the fraction of correctly linked entity mentions that should be linked:

$$recall = \frac{|\{correctly\ linked\ entity\ mentions\}|}{|\{entity\ mentions\ that\ should\ be\ linked\}|}$$

Recall takes into account all entity mentions that should be linked and determines how correct linked entity mentions are with regard to total entity mentions that should be linked. These two measures are sometimes used together in F_1-*measure* to provide a single measurement for a system. F_1-*measure* is defined as the harmonic mean of precision and recall:

$$F_1 = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

For many entity linking systems [59,60,63,66,68,69, 72,74,77,78,83,84,92,119], entity mentions that should be linked are given as the input of these systems, so the number of linked mentions generated by the system equals the number of entity mentions that should be linked. In this situation, researchers usually use *accuracy* to assess the system's performance. Accuracy is calculated as the number of correctly linked entity mentions divided by the total number of all entity mentions. Therefore, here $precision$=$recall$=F_1=$accuracy$. Moreover, accuracy is also regarded as the official evaluation measure in the TAC-KBP track, which will be introduced in the remainder of this section.

5.2 Entity Linking Data Sets

Some researchers [58,60,62,83,136] have manually annotated some data sets and made them publicly available. Thus, these data sets are good benchmark data sets for the entity linking task. Some detailed summaries of these data sets could be found in the papers [33,137]. Additionally, Cornolti et al. [137] recently proposed a publicly available benchmarking framework for comparison of entity-annotation systems that include some entity linking systems.

The Knowledge Base Population (KBP) track conducted as part of NIST Text Analysis Conference (TAC)[9] is an international entity linking competition held every year since 2009. Entity linking is regarded as one of the two subtasks in this track. These public entity linking competitions provided some benchmark data sets [89,100,138,139] to evaluate and compare different entity linking systems. The TAC-KBP track requires the systems who participate in the track to process entity mentions independently from one to another, which means they require that systems cannot leverage the topical coherence among the set of entity mentions according to the task description. Moreover, taking the TAC-KBP2009 data set as an example, the total 3904 entity mentions are scattered in 3688 documents, each of which has at most two mentions in its context according to its statistics. Therefore, almost all systems which mainly leverage the topical coherence feature for linking entities [58,62,75,93] were not evaluated over the TAC-KBP data set.

9. http://www.nist.gov/tac/about/index.html

6 Future Directions and Conclusions

In this article, we have presented a comprehensive survey for entity linking. Specifically, we have surveyed the main approaches utilized in the three modules of entity linking systems (i.e., Candidate Entity Generation, Candidate Entity Ranking, and Unlinkable Mention Prediction), and also introduced other critical aspects of entity linking such as applications, features, and evaluation.

Although there are so many methods proposed to deal with entity linking, it is currently unclear which techniques and systems are the current state-of-the-art, as these systems all differ along multiple dimensions and are evaluated over different data sets. A single entity linking system typically performs very differently for different data sets and domains. Although the supervised ranking methods seem to perform much better than the unsupervised approaches with respect to candidate entity ranking, the overall performance of the entity linking system is also significantly influenced by techniques adopted in the other two modules (i.e., Candidate Entity Generation and Unlinkable Mention Prediction) [33]. Supervised techniques require many annotated training examples and the task of annotating examples is costly. Furthermore, the entity linking task is highly data dependent and it is unlikely a technique dominates all others across all data sets. For a given entity linking task, it is difficult to determine which techniques are best suited. There are many aspects that affect the design of the entity linking system, such as the system requirement and the characteristics of the data sets, which is similar to the problem of feature selection introduced in Section 3.1.3.

Although our survey has presented many efforts in entity linking, we believe that there are still many opportunities for substantial improvement in this field. In the following, we point out some promising research directions in entity linking.

Firstly, most of the current entity linking systems focus on the entity linking task where entity mentions are detected from unstructured documents (such as news articles and blogs). However, entity mentions may also appear in other types of data and these types of data also need to be linked with the knowledge base, such as Web tables [140,141], Web lists [142,143], and tweets [144,145]. As different types of data have

various characteristics (e.g., Web tables are semi-structured text and have almost no textual context, and tweets are very short and noisy), it is very meaningful and necessary to develop specific techniques to deal with linking entities in them. Although some researchers have preliminarily addressed the entity linking task in Web tables [119], Web lists [92], and tweets [22,56,79,96,146], respectively, we believe there is still much room for further improvement. Moreover, a repository of benchmark data sets with these different types should be made available to researchers in order for them to develop and evaluate their methods for linking entities in these diverse types of data.

Secondly, most work on entity linking lacks an analysis of computational complexity, and they usually do not evaluate the efficiency and scalability of their systems. However, for real-time and large-scale applications, efficiency and scalability are significantly important and essential. Furthermore, the increasing amount of Web data is going to make this issue more prevalent in the future. Therefore, a promising direction for future research is to devise techniques that can substantially improve the efficiency and scalability while remaining the high accuracy. Although recently Lin et al. [44] investigated entity linking over millions of textual extractions, the overall linking accuracy was not high (about 70%) and there is potentially much space for substantial improvement. In addition, as a particular focus of record linkage in the database community has been on efficiency, their speed-up techniques could be leveraged for the highly efficient entity linking approaches. Recently, a large scale entity linking data set (i.e., Google's Wikilinks Corpus[10] [147]) has been released publicly, which contains over 40 million disambiguated mentions within over 10 million Web pages. This is a great opportunity for developing and evaluating large-scale entity linking systems.

Thirdly, the increasing demand for constructing and populating domain-specific knowledge bases (e.g., in the domains of biomedicine, entertainment, products, finance, tourism, etc.) makes domain-specific entity linking important as well. Domain-specific entity linking focuses on a specific domain of data, and the domain-specific knowledge bases may have different structures with the general-purpose knowledge bases (e.g., Wikipedia and YAGO). So far, Pantel and Fuxman [148] have addressed the task of associating search engine queries with entities from a large product catalog, and Dalvi et al. [149] have exploited the geographic aspects of tweets to infer the matches between tweets and restaurants. Dai et al. [150] employed a Markov logic network to model interweaved constraints to deal with the task of gene mention linking, which links each gene entity mention with a large scale gene database. In addition, Shen et al. [151] proposed a probabilistic model which unifies the entity popularity model with the entity object model to link the named entities in Web text with the DBLP bibliographic network. We strongly believe that this direction deserves much deeper exploration by researchers.

Finally, it is expected that more research or even a better understanding of the entity linking problem may lead to the emergence of more effective and efficient entity linking systems, as well as improvements in the areas of information extraction and Semantic Web.

10. https://code.google.com/p/wiki-links/

7 Acknowledgments

This work was supported in part by National Key Basic Research Program of China (973 Program) under Grant No. 2014CB340505, National Natural Science Foundation of China under Grant No. 61272088, a Samsung Research Grant, Tsinghua University Initiative Scientific Research Program, the U.S. Army Research Laboratory under Cooperative Agreement No. W911NF-09-2-0053 (NS-CTA), the U.S. Army Research Office under Cooperative Agreement No. W911NF-13-1-0193, U.S. National Science Foundation grants CNS-0931975, IIS-1017362, IIS-1320617, IIS-1354329, DTRA, NASA NRA-NNH10ZDA001N, and MIAS, a DHS-IDS Center for Multimodal Information Access and Synthesis at UIUC.

References

[1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, and Z. Ives, "Dbpedia: A nucleus for a web of open data," in *ISWC*, 2007, pp. 11–15.

[2] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: A core of semantic knowledge unifying wordnet and wikipedia," in *WWW*, 2007, pp. 697–706.

[3] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: a collaboratively created graph database for structuring human knowledge," in *SIGMOD*, 2008, pp. 1247–1250.

[4] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates, "Web-scale information extraction in knowitall: (preliminary results)," in *WWW*, 2004, pp. 100–110.

[5] A. Carlson, J. Betteridge, R. C. Wang, E. R. Hruschka, Jr., and T. M. Mitchell, "Coupled semi-supervised learning for information extraction," in *WSDM*, 2010, pp. 101–110.

[6] W. Wu, H. Li, H. Wang, and K. Q. Zhu, "Probase: a probabilistic taxonomy for text understanding," in *SIGMOD*, 2012, pp. 481–492.

[7] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic web," *Scientific American*, 2001.

[8] E. Agichtein and L. Gravano, "Snowball: Extracting relations from large plain-text collections," in *ICDL*, 2000, pp. 85–94.

[9] D. Zelenko, C. Aone, and A. Richardella, "Kernel methods for relation extraction," *J. Mach. Learn. Res.*, vol. 3, pp. 1083–1106, 2003.

[10] T. Hasegawa, S. Sekine, and R. Grishman, "Discovering relations among named entities from large corpora," in *ACL*, 2004, pp. 415–422.

[11] W. Shen, J. Wang, P. Luo, M. Wang, and C. Yao, "Reactor: a framework for semantic relation extraction and tagging over enterprise data," in *WWW*, 2011, pp. 121–122.

[12] W. Shen, J. Wang, P. Luo, and M. Wang, "A graph-based approach for ontology population with named entities," in *CIKM*, 2012, pp. 345–354.

[13] X. Ling and D. S. Weld, "Fine-grained entity recognition," in *AAAI*, 2012.

[14] N. Nakashole, T. Tylenda, and G. Weikum, "Fine-grained semantic typing of emerging entities," in *ACL*, 2013, pp. 1488–1497.

[15] T. Lin, Mausam, and O. Etzioni, "No noun phrase left behind: Detecting and typing unlinkable entities," in *EMNLP*, 2012, pp. 893–903.

[16] T. Zhang, K. Liu, and J. Zhao, "Cross lingual entity linking with bilingual topic model," in *IJCAI*, 2013, pp. 2218–2224.

[17] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Lingvisticae Investigationes*, vol. 30, no. 1, pp. 3–26, Jan. 2007.

[18] J. R. Finkel, T. Grenager, and C. Manning, "Incorporating non-local information into information extraction systems by gibbs sampling," in *ACL*, 2005, pp. 363–370.

[19] A. Mikheev, M. Moens, and C. Grover, "Named entity recognition without gazetteers," in *EACL*, 1999, pp. 1–8.

[20] D. Klein, J. Smarr, H. Nguyen, and C. D. Manning, "Named entity recognition with character-level models," in *CONLL*, 2003, pp. 180–183.

[21] S. Geman and D. Geman, "Stochastic relaxation, gibbs distributions, and the bayesian restoration of images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 6, no. 6, pp. 721–741, Nov. 1984.

[22] S. Guo, M.-W. Chang, and E. Kiciman, "To link or not to link? a study on end-to-end tweet entity linking," in *NAACL*, 2013.

[23] A. Sil and A. Yates, "Re-ranking for joint named-entity recognition and linking," in *CIKM*, 2013, pp. 2369–2374.

[24] K. Q. Pu, O. Hassanzadeh, R. Drake, and R. J. Miller, "Online annotation of text streams with structured entities," in *CIKM*, 2010, pp. 29–38.

[25] A. Bagga and B. Baldwin, "Entity-based cross-document coreferencing using the vector space model," in *COLING*, 1998, pp. 79–85.

[26] G. S. Mann and D. Yarowsky, "Unsupervised personal name disambiguation," in *CONLL*, 2003, pp. 33–40.

[27] J. Artiles, J. Gonzalo, and S. Sekine, "The SemEval-2007 WePS Evaluation: Establishing a benchmark for the Web People Search Task," in *the Fourth International Workshop on Semantic Evaluations*, 2007, pp. 64–69.

[28] X. Yin, J. Han, and P. S. Yu, "Object distinction: Distinguishing objects with identical names," in *ICDE*, 2007, pp. 1242–1246.

[29] L. Jiang, J. Wang, N. An, S. Wang, J. Zhan, and L. Li, "Grape: A graph-based framework for disambiguating people appearances in web search," in *ICDM*, 2009, pp. 199–208.

[30] X. Han and J. Zhao, "Named entity disambiguation by leveraging wikipedia semantic knowledge," in *CIKM*, 2009, pp. 215–224.

[31] R. Navigli, "Word sense disambiguation: A survey," *ACM Comput. Surv.*, vol. 41, no. 2, pp. 10:1–10:69, Feb. 2009.

[32] C. Fellbaum, Ed., *WordNet: An Electronic Lexical Database*. The MIT Press, 1998.

[33] B. Hachey, W. Radford, J. Nothman, M. Honnibal, and J. R. Curran, "Evaluating entity linking with wikipedia," *Artif. Intell.*, vol. 194, pp. 130–150, Jan. 2013.

[34] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios, "Duplicate record detection: A survey," *IEEE Trans. on Knowl. and Data Eng.*, vol. 19, no. 1, pp. 1–16, Jan. 2007.

[35] M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar, and S. Fienberg, "Adaptive name matching in information integration," *IEEE Intelligent Systems*, vol. 18, no. 5, pp. 16–23, Sep. 2003.

[36] H. Köpcke and E. Rahm, "Frameworks for entity matching: A comparison," *Data Knowl. Eng.*, vol. 69, no. 2, pp. 197–210, Feb. 2010.

[37] P. Christen, "A survey of indexing techniques for scalable record linkage and deduplication," *IEEE Trans. on Knowl. and Data Eng.*, vol. 24, no. 9, pp. 1537–1555, Sep. 2012.

[38] X. Dong, A. Halevy, and J. Madhavan, "Reference reconciliation in complex information spaces," in *SIGMOD*, 2005, pp. 85–96.

[39] R. Isele and C. Bizer, "Learning expressive linkage rules using genetic programming," *Proc. VLDB Endow.*, vol. 5, no. 11, pp. 1638–1649, Jul. 2012.

[40] J. Volz, C. Bizer, M. Gaedke, and G. Kobilarov, "Discovering and maintaining links on the web of data," in *ISWC*, 2009, pp. 650–665.

[41] J. Wang, T. Kraska, M. J. Franklin, and J. Feng, "Crowder: Crowdsourcing entity resolution," *Proc. VLDB Endow.*, vol. 5, no. 11, pp. 1483–1494, Jul. 2012.

[42] M. Hadjieleftheriou and D. Srivastava, "Approximate string processing," *Found. Trends databases*, vol. 2, no. 4, pp. 267–402, Apr. 2011.

[43] O. Hassanzadeh, M. Sadoghi, and R. J. Miller, "Accuracy of approximate string joins using grams," in *the 5th International Workshop on Quality in Databases (QDB'07) - VLDB Workshop*, 2007.

[44] T. Lin, Mausam, and O. Etzioni, "Entity linking at web scale," in *AKBC-WEKEX*, 2012, pp. 84–88.

[45] N. Nakashole, G. Weikum, and F. Suchanek, "Patty: a taxonomy of relational patterns with semantic types," in *EMNLP-CoNLL*, 2012, pp. 1135–1145.

[46] J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum, "Yago2: A spatially and temporally enhanced knowledge base from wikipedia," *Artif. Intell.*, vol. 194, pp. 28–61, Jan. 2013.

[47] T. Cheng, X. Yan, and K. C.-C. Chang, "Entityrank: Searching entities directly and holistically," in *VLDB*, 2007, pp. 387–398.

[48] G. Demartini, T. Iofciu, and A. P. De Vries, "Overview of the inex 2009 entity ranking track," in *INEX*, 2009, pp. 254–264.

[49] K. Balog, P. Serdyukov, and A. P. de Vries, "Overview of the trec 2010 entity track," in *TREC*, 2010.

[50] I. Bordino, Y. Mejova, and M. Lalmas, "Penguins in sweaters, or serendipitous entity search on user-generated content," in *CIKM*, 2013, pp. 109–118.

[51] J. Guo, G. Xu, X. Cheng, and H. Li, "Named entity recognition in query," in *SIGIR*, 2009, pp. 267–274.

[52] O. Phelan, K. McCarthy, and B. Smyth, "Using twitter to recommend real-time topical news," in *RecSys*, 2009, pp. 385–388.

[53] J. Liu, P. Dolan, and E. R. Pedersen, "Personalized news recommendation based on click behavior," in *IUI*, 2010, pp. 31–40.

[54] J. Weng, E.-P. Lim, J. Jiang, and Q. He, "Twitterrank: finding topic-sensitive influential twitterers," in *WSDM*, 2010.

[55] M. Michelson and S. A. Macskassy, "Discovering users' topics of interest on twitter: a first look," in *the fourth workshop on Analytics for noisy unstructured text data*, ser. AND, 2010, pp. 73–80.

[56] A. Gattani, D. S. Lamba, N. Garera, M. Tiwari, X. Chai, S. Das, S. Subramaniam, A. Rajaraman, V. Harinarayan, and A. Doan, "Entity extraction, linking, classification, and tagging for social media: A wikipedia-based approach," *Proc. VLDB Endow.*, vol. 6, no. 11, pp. 1126–1137, Aug. 2013.

[57] C. Welty, J. W. Murdock, A. Kalyanpur, and J. Fan, "A comparison of hard filters and soft evidence for answer typing in watson," in *ISWC*, 2012, pp. 243–256.

[58] 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*, 2011, pp. 782–792.

[59] R. Bunescu and M. Pasca, "Using Encyclopedic Knowledge for Named Entity Disambiguation," in *EACL*, 2006, pp. 9–16.

[60] S. Cucerzan, "Large-Scale Named Entity Disambiguation Based on Wikipedia Data," in *EMNLP-CoNLL*, 2007, pp. 708–716.

[61] X. Han and J. Zhao, "Nlpr_kbp in tac 2009 kbp track: A two-stage method to entity linking," in *TAC 2009 Workshop*, 2009.

[62] S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti, "Collective annotation of Wikipedia entities in web text," in *SIGKDD*, 2009, pp. 457–466.

[63] V. Varma, P. Bysani, K. Reddy, V. Bharat, S. GSK, K. Kumar, S. Kovelamudi, K. K. N, and N. Maganti, "Iiit hyderabad at tac 2009," in *TAC 2009 Workshop*, 2009.

[64] V. Varma, P. Bysani, K. Reddy, V. B. Reddy, S. Kovelamudi, S. R. Vaddepally, R. Nanduri, K. K. N, S. Gsk, and P. Pingali, "Iiit hyderabad in guided summarization and knowledge base population," in *TAC 2010 Workshop*, 2010.

[65] W. Zhang, J. Su, C. L. Tan, and W. T. Wang, "Entity linking leveraging automatically generated annotation," in *COLING*, 2010.

[66] W. Zhang, Y. C. Sim, J. Su, and C. L. Tan, "Nus-i2r: Learning a combined system for entity linking," in *TAC 2010 Workshop*, 2010.

[67] Z. Chen, S. Tamang, A. Lee, X. Li, W.-P. Lin, M. Snover, J. Artiles, M. Passantino, and H. Ji, "Cuny-blender tac-kbp2010 entity linking and slot filling system description," in *TAC 2010 Workshop*, 2010.

[68] Z. Zheng, F. Li, M. Huang, and X. Zhu, "Learning to link entities with knowledge base," in *NAACL*, 2010, pp. 483–491.

[69] J. Lehmann, S. Monahan, L. Nezda, A. Jung, and Y. Shi, "Lcc approaches to knowledge base population at tac 2010," in *TAC 2010 Workshop*, 2010.

[70] W. Zhang, J. Su, B. Chen, W. Wang, Z. Toh, Y. Sim, Y. Cao, C. Y. Lin, and C. L. Tan, "I2r-nus-msra at tac 2011: Entity linking," in *TAC 2011 Workshop*, 2011.

[71] S. Cucerzan, "Tac entity linking by performing full-document entity extraction and disambiguation," in *TAC 2011 Workshop*, 2011.

[72] W. Zhang, Y. C. Sim, J. Su, and C. L. Tan, "Entity linking with effective acronym expansion, instance selection and topic modeling," in *IJCAI*, 2011, pp. 1909–1914.

[73] S. Monahan, J. Lehmann, T. Nyberg, J. Plymale, and A. Jung, "Cross-lingual cross-document coreference with entity linking," in *TAC 2011 Workshop*, 2011.

[74] X. Han and L. Sun, "A generative entity-mention model for linking entities with knowledge base," in *ACL*, 2011, pp. 945–954.

[75] X. Han, L. Sun, and J. Zhao, "Collective entity linking in web text: a graph-based method," in *SIGIR*, 2011, pp. 765–774.

[76] L. Ratinov, D. Roth, D. Downey, and M. Anderson, "Local and global algorithms for disambiguation to wikipedia," in *ACL*, 2011, pp. 1375–1384.

[77] S. Gottipati and J. Jiang, "Linking entities to a knowledge base with query expansion," in *EMNLP*, 2011, pp. 804–813.

[78] W. Shen, J. Wang, P. Luo, and M. Wang, "Linden: linking named entities with knowledge base via semantic knowledge," in *WWW*, 2012, pp. 449–458.

[79] ——, "Linking named entities in tweets with knowledge base via user interest modeling," in *SIGKDD*, 2013, pp. 68–76.

[80] K. Chakrabarti, S. Chaudhuri, T. Cheng, and D. Xin, "A framework for robust discovery of entity synonyms," in *SIGKDD*, 2012, pp. 1384–1392.

[81] T. Cheng, H. W. Lauw, and S. Paparizos, "Entity synonyms for structured web search," *TKDE*, 2011.

[82] B. Taneva, T. Cheng, K. Chakrabarti, and Y. He, "Mining acronym expansions and their meanings using query click log," in *WWW*, 2013, pp. 1261–1272.

[83] M. Dredze, P. McNamee, D. Rao, A. Gerber, and T. Finin, "Entity disambiguation for knowledge base population," in *COLING*, 2010, pp. 277–285.

[84] P. McNamee, "Hltcoe efforts in entity linking at tac kbp 2010," in *TAC 2010 Workshop*, 2010.

[85] S. Deorowicz and M. G. Ciura, "Correcting spelling errors by modeling their causes," *International Journal of Applied Mathematics and Computer Science*, vol. 15, pp. 275–285, 2005.

[86] D. Nemeskey, G. Recski, A. Zséder, and A. Kornai, "Budapestacad at tac 2010," in *TAC 2010 Workshop*, 2010.

[87] B. Daróczy, Z. Fekete, M. Brendel, S. Rácz, A. Benczúr, D. Siklósi, and A. Pereszlényi, "Cross-modal image retrieval with parameter tuning," in *Evaluating Systems for Multilingual and Multimodal Information Access-9th Workshop of the Cross-Language Evaluation Forum*, September 2009.

[88] A. Jain, S. Cucerzan, and S. Azzam, "Acronym-expansion recognition and ranking on the web," in *IRI*, 2007, pp. 209–214.

[89] H. Ji, R. Grishman, and H. T. Dang, "Overview of the tac2011 knowledge base population track," in *TAC 2011 Workshop*, 2011.

[90] T. Štajner and D. Mladenić, "Entity resolution in texts using statistical learning and ontologies," in *ASWC*, 2009, pp. 91–104.

[91] P. Ferragina and U. Scaiella, "Tagme: On-the-fly annotation of short text fragments (by wikipedia entities)," in *CIKM*, 2010, pp. 1625–1628.

[92] W. Shen, J. Wang, P. Luo, and M. Wang, "Liege: Link entities in web lists with knowledge base," in *SIGKDD*, 2012, pp. 1424–1432.

[93] X. Han and L. Sun, "An entity-topic model for entity linking," in *EMNLP*, 2012.

[94] Z. Chen and H. Ji, "Collaborative ranking: A case study on entity linking," in *EMNLP*, 2011.

[95] T. Cassidy, Z. Chen, J. Artiles, H. Ji, H. Deng, L.-A. Ratinov, J. Zheng, J. Han, and D. Roth, "Cuny-uiuc-sri tac-kbp2011 entity linking system description," in *TAC 2011 Workshop*, 2011.

[96] X. Liu, Y. Li, H. Wu, M. Zhou, F. Wei, and Y. Lu, "Entity linking for tweets," in *ACL*, 2013.

[97] Y. Guo, B. Qin, T. Liu, and S. Li, "Microblog entity linking by leveraging extra posts," in *EMNLP*, 2013.

[98] P. Christen, "A comparison of personal name matching: Techniques and practical issues," in *ICDMW*, 2006, pp. 290–294.

[99] M. Dreyer, J. R. Smith, and J. Eisner, "Latent-variable modeling of string transductions with finite-state methods," in *EMNLP*, 2008, pp. 1080–1089.

[100] H. Ji and R. Grishman, "Knowledge base population: successful approaches and challenges," in *ACL*, 2011, pp. 1148–1158.

[101] D. Varga and E. Simon, "Hungarian named entity recognition with a maximum entropy approach," *Acta Cybern.*, vol. 18, no. 2, pp. 293–301, Feb. 2007.

[102] J. Lehmann, P. Aarseth, L. Nezda, S. Fayyaz, A. Jung, S. Monahan, and M. Oberoi, "Language computer corporation's ace 2007 system description," in *2007 Automatic Content Extraction Conference*, 2007.

[103] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Mar. 2003.

[104] A. Pilz and G. Paaß, "From names to entities using thematic context distance," in *CIKM*, 2011, pp. 857–866.

[105] Z. He, S. Liu, M. Li, M. Zhou, L. Zhang, and H. Wang, "Learning entity representation for entity disambiguation," in *ACL*, 2013.

[106] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006.

[107] Y. Li, C. Wang, F. Han, J. Han, D. Roth, and X. Yan, "Mining evidences for named entity disambiguation," in *SIGKDD*, 2013, pp. 1070–1078.

[108] D. Milne and I. H. Witten, "Learning to link with wikipedia," in *CIKM*, 2008, pp. 509–518.

[109] ——, "An open-source toolkit for mining wikipedia," *Artif. Intell.*, vol. 194, pp. 222–239, Jan. 2013.

[110] R. L. Cilibrasi and P. M. B. Vitanyi, "The google similarity distance," *IEEE Trans. on Knowl. and Data Eng.*, vol. 19, pp. 370–383, March 2007.

[111] J. Hoffart, S. Seufert, D. B. Nguyen, M. Theobald, and G. Weikum, "Kore: Keyphrase overlap relatedness for entity disambiguation," in *CIKM*, 2012, pp. 545–554.

[112] D. Ceccarelli, C. Lucchese, S. Orlando, R. Perego, and S. Trani, "Learning relatedness measures for entity linking," in *CIKM*, 2013, pp. 139–148.

[113] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *the fifth annual workshop on Computational learning theory*, 1992, pp. 144–152.

[114] T.-Y. Liu, "Learning to rank for information retrieval," *Found. Trends Inf. Retr.*, vol. 3, no. 3, pp. 225–331, Mar. 2009.

[115] R. Herbrich, T. Graepel, and K. Obermayer, "Large margin rank boundaries for ordinal regression," in *Advances in Large Margin Classifiers*, A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, Eds. Cambridge, MA: MIT Press, 2000, pp. 115–132.

[116] T. Joachims, "Optimizing search engines using clickthrough data," in *SIGKDD*, 2002, pp. 133–142.

[117] L. Shen and A. K. Joshi, "Ranking and reranking with perceptron," *Mach. Learn.*, vol. 60, no. 1-3, pp. 73–96, Sep. 2005.

[118] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li, "Learning to rank: from pairwise approach to listwise approach," in *ICML*, 2007, pp. 129–136.

[119] G. Limaye, S. Sarawagi, and S. Chakrabarti, "Annotating and searching web tables using entities, types and relationships," *Proc. VLDB Endow.*, vol. 3, pp. 1338–1347, September 2010.

[120] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[121] G. Demartini, D. E. Difallah, and P. Cudré-Mauroux, "Zencrowd: leveraging probabilistic reasoning and crowdsourcing

techniques for large-scale entity linking," in *WWW*, 2012, pp. 469–478.
[122] T. H. Haveliwala, "Topic-sensitive pagerank," in *WWW*, 2002, pp. 517–526.
[123] M. Sozio and A. Gionis, "The community-search problem and how to plan a successful cocktail party," in *SIGKDD*, 2010, pp. 939–948.
[124] D. Opitz and R. Maclin, "Popular ensemble methods: An empirical study," *Journal of Artificial Intelligence Research*, vol. 11, pp. 169–198, 1999.
[125] L. Rokach, "Ensemble-based classifiers," *Artif. Intell. Rev.*, vol. 33, no. 1-2, pp. 1–39, Feb. 2010.
[126] G. J. J. Adeva, U. B. Cerviño, and R. A. Calvo, "Accuracy and Diversity in Ensembles of Text Categorisers," *CLEI Electronic Journal*, vol. 9, 2005.
[127] F. Li, Z. Zheng, F. Bu, Y. Tang, X. Zhu, and M. Huang, "Thu quanta at tac 2009 kbp and rte track," in *TAC 2009 Workshop*, 2009.
[128] P. McNamee, M. Dredze, A. Gerber, N. Garera, T. Finin, J. Mayfield, C. Piatko, D. Rao, D. Yarowsky, and M. Dreyer, "Hltcoe approaches to knowledge base population at tac 2009," in *TAC 2009 Workshop*, 2009.
[129] E. Agirrey, A. X. Changz, D. S. Jurafskyz, C. D. Manningz, V. I. Spitkovskyz, and E. Yeh, "Stanford-ubc at tac-kbp," in *TAC 2009 Workshop*, 2009.
[130] H. Daumé III, "Frustratingly easy domain adaptation," in *ACL*, 2007, pp. 256–263.
[131] K. Brinker, "Incorporating diversity in active learning with support vector machines," in *ICML*, 2003, pp. 59–66.
[132] D. Shen, J. Zhang, J. Su, G. Zhou, and C.-L. Tan, "Multi-criteria-based active learning for named entity recognition," in *ACL*, 2004.
[133] G. Salton, A. Wong, and C. S. Yang, "A vector space model for automatic indexing," *Commun. ACM*, vol. 18, no. 11, pp. 613–620, 1975.
[134] C. Zhai and J. Lafferty, "Model-based feedback in the language modeling approach to information retrieval," in *CIKM*, 2001, pp. 403–410.
[135] ——, "A study of smoothing methods for language models applied to information retrieval," *ACM Trans. Inf. Syst.*, vol. 22, no. 2, pp. 179–214, Apr. 2004.
[136] L. Bentivogli, P. Forner, C. Giuliano, A. Marchetti, E. Pianta, and K. Tymoshenko, "Extending english ace 2005 corpus annotation with ground-truth links to wikipedia," in *the COLING Workshop*, 2010.
[137] M. Cornolti, P. Ferragina, and M. Ciaramita, "A framework for benchmarking entity-annotation systems," in *WWW*, 2013, pp. 249–260.
[138] P. McNamee, H. Simpson, and H. T. Dang, "Overview of the tac 2009 knowledge base population track," in *TAC 2009 Workshop*, 2009.
[139] H. Ji, R. Grishman, H. T. Dang, K. Griffitt, and J. Ellis, "Overview of the tac 2010 knowledge base population track," in *TAC 2010 Workshop*, 2010.
[140] M. J. Cafarella, A. Halevy, D. Z. Wang, E. Wu, and Y. Zhang, "Webtables: exploring the power of tables on the web," *Proc. VLDB Endow.*, vol. 1, pp. 538–549, August 2008.
[141] H. Elmeleegy, J. Madhavan, and A. Halevy, "Harvesting relational tables from lists on the web," *Proc. VLDB Endow.*, vol. 2, pp. 1078–1089, August 2009.
[142] F. Fumarola, T. Weninger, R. Barber, D. Malerba, and J. Han, "Extracting general lists from web documents: a hybrid approach," in *IEA/AIE*, 2011, pp. 285–294.
[143] Z. Zhang, K. Q. Zhu, and H. Wang, "A system for extracting top-k lists from the web," in *SIGKDD*, 2012, pp. 1560–1563.
[144] X. Liu, S. Zhang, F. Wei, and M. Zhou, "Recognizing named entities in tweets," in *ACL'11*.
[145] A. Ritter, S. Clark, Mausam, and O. Etzioni, "Named entity recognition in tweets: an experimental study," in *EMNLP'11*.
[146] E. Meij, W. Weerkamp, and M. de Rijke, "Adding semantics to microblog posts," in *WSDM*, 2012, pp. 563–572.
[147] S. Singh, A. Subramanya, F. Pereira, and A. McCallum, "Wikilinks: A large-scale cross-document coreference corpus labeled via links to wikipedia," University of Massachusetts, Amherst, Tech. Rep. UM-CS-2012-015, 2012.
[148] P. Pantel and A. Fuxman, "Jigs and lures: associating web queries with structured entities," in *ACL*, 2011, pp. 83–92.
[149] N. Dalvi, R. Kumar, and B. Pang, "Object matching in tweets with spatial models," in *WSDM*, 2012, pp. 43–52.
[150] H.-J. Dai, R. T. Tsai, and W.-L. Hsu, "Entity Disambiguation Using a Markov-Logic Network," in *IJCNLP*, 2011, pp. 846–855.
[151] W. Shen, J. Han, and J. Wang, "A probabilistic model for linking named entities in web text with heterogeneous information networks," in *SIGMOD*, 2014, To appear.

Wei Shen received the bachelor's degree in software engineering from Beihang University, Beijing, China, in 2009. He is currently working toward the PhD degree in the Department of Computer Science and Technology, Tsinghua University, Beijing, China. He is a recipient of the Google PhD fellowship. His research interests include entity linking, knowledge base population, and text mining.

Jianyong Wang is currently a professor in the Department of Computer Science and Technology, Tsinghua University, Beijing, China. He received his PhD degree in Computer Science in 1999 from the Institute of Computing Technology, Chinese Academy of Sciences. He was an assistant professor at Peking University, and visited Simon Fraser University, University of Illinois at Urbana-Champaign, and University of Minnesota at Twin Cities before joining Tsinghua University in December 2004. His research interests mainly include data mining and Web information management. He has co-authored over 60 papers in some leading international conferences and some top international journals. He is serving or ever served as a PC member for some leading international conferences, such as SIGKDD, VLDB, ICDE, WWW, and an associate editor of IEEE TKDE. He is a senior member of the IEEE, a member of the ACM, a recipient of the 2009 and 2010 HP Labs Innovation Research award, the 2009 Okawa Foundation Research Grant (Japan), WWW'08 best posters award, the Year 2007 Program for New Century Excellent Talents in University, The Ministry of Education of China, the Year 2013 second-class Prize for Natural Sciences, China Computer Federation, and the Year 2013 second-class Prize for Natural Sciences, The Ministry of Education of China.

Jiawei Han is Abel Bliss Professor in Engineering, in the Department of Computer Science at the University of Illinois. He has been researching into data mining, information network analysis, and database systems, with over 600 publications. He served as the founding Editor-in-Chief of ACM Transactions on Knowledge Discovery from Data (TKDD) and on the editorial boards of several other journals. Jiawei has received ACM SIGKDD Innovation Award (2004), IEEE Computer Society Technical Achievement Award (2005), IEEE Computer Society W. Wallace McDowell Award (2009), and Daniel C. Drucker Eminent Faculty Award at UIUC (2011). He is a Fellow of ACM and a Fellow of IEEE. He is currently the Director of Information Network Academic Research Center (INARC) supported by the Network Science-Collaborative Technology Alliance (NS-CTA) program of U.S. Army Research Lab. His book "Data Mining: Concepts and Techniques" (Morgan Kaufmann) has been used worldwide as a textbook.