A Collective Approach to Ranking Entities for Mentions

Shunlin Rong
Graduate School of Information, Production and Systems
Waseda University
Fukuoka, Japan
rongshunlin@ruri.waseda.jp

Mizuho Iwaihara
Graduate School of Information, Production and Systems
Waseda University
Fukuoka, Japan
iwaihara@waseda.jp

***Abstract*—Entity linking (EL) is the task of mapping name mentions in web text to their entities in a knowledge base. Most of earlier EL work in the knowledge based approach is usually formulated as a ranking problem, either by (i) non-collective approaches with supervised models, or (ii) collective approaches by leveraging global topical coherence which means semantic relations between entities through graph-based approaches. For the mapping process, we can regard it as selecting an entity to its mention by combining these two methods. In this paper, we propose a probabilistic model that ranks related entities to name mentions where ranking is customized by using three types of data: popularity knowledge of the entity, context similarity between mentions and the entity, and semantic relations between mapping entities. Specifically, we first propose an EL model utilizing global topical coherence that means semantic relatedness between entities, as well as using local mention-to-entity compatibility, to improve recall and precision. The key benefit of our model comes from 1) combination of two methods to provide customized ranking for mentions, 2) the model can save a large amount of calculation by efficiently finding candidate combinations of entities through global semantic coherence.**

Keywords—Wikipedia, Entity Linking, Ranking

I. INTRODUCTION

With the rapid increase of the Internet, the web has become one of the largest data repositories in the world in recent years. A large volume of data is stored on the Internet in the form of natural language texts which are often ambiguous. A named entity may have multiple names and a name could denote several different named entities.

The development of knowledge bases such as Wikipedia provide a possible solution to solve these problems. These knowledge bases contain rich knowledge about the world's entities, their semantic properties, and the semantic relations between each other. These knowledge base contain structured and unambiguous entities and entities relations. By mapping the unstructured web data to the structured and unambiguous. entities, we can solve the ambiguous problem of web data.

A key problem in mapping web data to a knowledge base is linking name mentions in a document with their referent entities in a knowledge base which we often call the Entity Linking (EL) problem. The main difficulty of the task is as follows. The first problem is the name variation problem which means an entity can be mentioned in different ways such as full names, aliases name and so on. The second problem is the name ambiguity problem, such as Michael Jordan could be referred as a famous NBA basketball player or a Berkeley professor. Let us formulate the problem firstly. Let $M = \{m_1, m_2, \ldots, m_k\}$ denote a collection of name mentions. Each name mention m in M is characterized by its name m.s, its local surrounding context *m.c* and the document containing it m.d. Given a knowledge base KB containing a set of entities $E=\{e_1, e_2, \ldots, e_m\}$, the objective of our work is to recommend referent entities in KB of the name mentions in *M*. Let *m.e* denote the referent entity of a mention *m* in *M*. Our aim is to recommend a suitable entity e to a mention m by using a ranking method.

There are also several basic assumptions in this paper. The first assumption is that the more popular an entity is, the more likely it could appear in a document or paragraph. The second assumption is that the referent entity of a name mention should be topically coherent with its neighboring mentions. To be detailed, referent entities in one document should also be co-occurring in other documents. The third assumption is that the mentions and the mentions' referent entities should be similar in some extent. For example, semantic similarities between the mentions and the mention's referent entities should be high.

For the first assumption, we consider that the popularity information of an entity tells us the likelihood of an entity appearing in a document. Also, the popularity of a mention to an entity may suggest a possible candidate entity for an ambiguous mention. For the second assumption, we consider that the referent entities $\{e_1, e_2, \ldots, e_n\}$ of a set of mentions M in a document should be topically coherent. For topical coherence, we utilize web corpus for evaluating global semantic relatedness of entity combinations in a document. For the third assumption, we evaluate local contextual similarities between name mentions and entities.

The main contribution of our work is as follows:

1) Instead of estimating candidate entities to each mention in document independently, we evaluate most likely combinations of entities by evaluating their likelihoods in global contexts through combined search engine lookups, which greatly reduce the search space.

2) Our model works better as the trained wiki dumps becomes larger. Our experiments support this point.

This paper is organized as follows. The related work is described in Section 2. The model is described in Section 3. Experiment results are presented and discussed in Section 4. Finally we conclude this paper in Section 5.

II. RELATED WORK

Most of earlier entity linking problems can be formed as a ranking problem, either by (i) non-collective approaches which could be divided into two different parts.

978-1-5090-0806-3/16/$31.00 copyright 2016 IEEE
ICIS 2016, June 26-29, 2016, Okayama, Japan

The first is ***Local Compatibility Based Approach*** which is also the initial method by extracting the discriminative features of an entity from its textual description, then linking a name mention to the entity which has the highest contextual similarity with it. Mihalcea et al. [1] proposed a bag of words (BoW)-based methods, where the compatibility between a name mention and an entity was measured as the cosine similarity between them. Cucerzan et al. [2] and Bunescu et al.[3] extended the BoW model by incorporating more entity knowledge such as entities' categories. One of its largest problems is that the dimension of vectors of the words sometimes becomes too big to calculate. Also they do not take into account the interdependence between EL decisions.

The second is ***Simple Relational Approaches:***

Considering the entity linking decisions in one document have no influence with each other, we can utilize the semantic relations between different entities in one document for linking decision. The core assumption is that the referent entity of a name mention should have a strong semantic relationship with its unambiguous contextual entities (Medelyan et al.,[4]). The main problem of this method is that they can only exploit pairwise interdependence between a name mention and its unambiguous contextual entities.

These methods deal with one mention at each time relying on prior popularity, context similarity, and other local features with supervised models without taking account of the global semantic relations between entities.

The second part is the ***Collective Methods*** which deal with the related mentions in parallel by leveraging the global semantic relationship between entities through graph-based approaches (X. Han et al[5]). This approach needs to model the global semantic relations by an iterative method in one document which is not as efficient as the first method. On the other hand, it could achieve higher accuracy in entity linking decisions.

III. The Details of The Model

In this section, we propose a collective model for linking an entity to a mention. As far as we know, the closest work to us is X.Han et al.[5] and X. Han et al.[6]. In the following sections we will introduce how to capture the popularity of entities in a knowledge base, how to calculate the local mention-to-entity compatibility, and how to measure the global semantic relation in a document.

A. Popularity of Entities

The reason why we want to capture popularity of entities in a knowledge base is to utilize the popularities for selecting candidate entities. The more popular an entity is, the more likely it could be a referent entity of a mention and appear in a document. As tried in the literature, we use the frequencies of an entity in the whole knowledge base to estimate the popularity of this entity. In this paper, we account the Wikipedia's redirect links which contain entities as the frequencies. Sometimes one redirect link may contain several entities which share the same meanings. In this case, we choose the first one appears in this redirect link and account it appearing one time. This formula can be defined as follows:

$$P(e) = \frac{Count(e)+1}{|M|+N} \quad (1)$$

Where *Count(e)* is the count of the name mentions whose referent entity is e, and the $|M|$ is the total name mention count. The estimation is further smoothed by using the add-one method. Parameter N is the number of entities appearing in the whole Wiki dump.

As we can see this function needs a large number of documents to capture similarities. As the size of the knowledge base becomes larger, the model could measure the popularity better.

B. Collective semantic relatiedness between entities

As we have mentioned in Section 1, multiple mentions in the same document are contextually related, so assignments of entities to these mentions need to be solved consistently. In other words, we need to collectively treat combinations of candidate entities, instead of resolving each mention independently.

However, the most difficult problem in this approach is that we cannot obtain candidate entities for all the mentions when we are processing one name mention at a one-time pass.

Let us consider an example sentence: 'During his standout career at Bulls, Jordan also acts in the movie Space Jam.' Figure 1 shows this situation. Here, Jordan, Bulls, Space Jam are three name mentions that need to be linked to correct entities.

Now suppose every name mention has only three candidate entities. For example, Jordan's candidate entities are Michael Jordan (a very famous basketball player), Jordan Grand Prix and Jordan River. We use the Fig 1 to show this situation. Then there should be 3^3=27 combinations or tuples of possible combinations of entities in total. One example of the tuples or pairs is (Michael Jordan (Basketball Player), Chicago Bulls (NBA Basketball Team), Space Jam (Movie); Michael Jordan, Chicago Bulls, Space Jam (band)). The most likely tuple of the 27 tuples under certain criteria should be chosen as the solution. However, if each mention has 20 candidate entities and there are n mentions in the document, then the number of tuples, or the size of the search space, becomes 20^n, which is prohibitively high to find an optimum tuple.

To overcome the above situation, we take an approach to evaluate global semantic relatedness between entities though co-occurrence on the web corpus. By carefully narrowing candidate tuples over lookups on the web corpus, we expect to reach a semantically coherent tuple quickly.

Existing researches [5] adopt graph-based methods for iteratively obtaining candidate entities for mentions. However just repeating score calculations sometimes fails to obtain an ideal result. Furthermore, the text length is affecting the total cost too.

Actually, the suitable referent entity for the basketball player Michael Jordan should also have a strong semantic relationship with the Bull, however the name Bull itself can have many meanings. To avoid extensive explorations of a knowledge graph, we adopt a web corpus-based semantic relatedness measure, called the Normalized Google Distance (Rudi et al[7]). Only one look-up of a search engine is required here. For

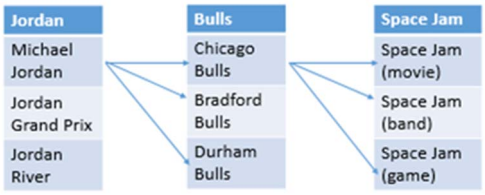

Fig. 1. Semantic Relation between Candidate Entities

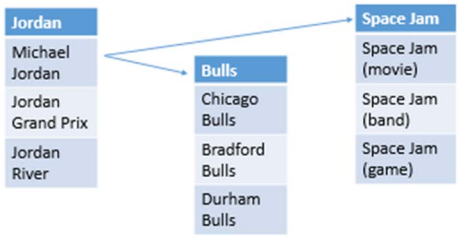

Fig. 2. Semantic Relation between Candidate Entities

example on Michael Jordan, we only need to calculate the semantic relatedness between (Michael Jordan, Bulls, Space Jam). This only needs 1 time calculation for one candidate entity and 3 times calculations for one name mention, when compared with deep explorations of a knowledge base.

If there are 20 candidate entities for every name mention, and n name mentions in the text. We only need 20*1 times calculation. That's to say no matter how many name mentions in the text. The number of tuples is only related to the number of the candidate entities to one name mention. In this paper, we have 20 candidate entities for every name mention at most. In another words, we have at most 20 tuples for every text regardless the length of the text.

We use the Normalized Google Distance (NGD) to measure the semantic relatedness between the referent entities of one name mention and the associated name mentions in the document. The Google Distance is defined as follows:

$$NGD(x,y) = \frac{\max\{logf(x), logf(y)\} - logf(x,y)}{logM - min\{logf(x), logf(y)\}} \quad (2)$$

Where M is the total number of web pages searched by Google, f(x) and f(y) are the number of hits for search terms x and y, respectively, and f(x, y) is the number of web pages on which both x and y co-occur.

Due to the limit on the Google API, we use another search engine Bing to calculate NGD. By using a search engine to calculate NGD, we can deal with new words not in the knowledge base. On the other hand, the hit numbers of a search engine is a mixture on search results of non-disambiguated mentions. A commercial search engine will display all related pages about the name mentions, such as the pages about the Bull, pages mixed about Chicago Bulls and pages about animals named bull, and so on.

However, we need to consider that we use the Normalized Google Distance to calculate the semantic relations. And the Normalized Google Distance could only calculate two terms(like Michael Jordan) at once. We treat each (Michael Jordan(NBA Player), Bulls, Space Jam) as a tuple or pair. We need to calculate the semantic relations between Michael Jordan with Bulls, Michael Jordan with Space Jam. So this needs two times. If there are n name mentions in each tuple or pair, there will be n-1 times NGD calculations for each pair or tuple.

The Normalized Google Distance measures the semantic relation between entities and the remaining name mentions. Actually, what we need is to measure the global semantic relationship in one document. If one candidate entity shares a stronger semantic relation with the other name mention in the same document, the entity has a higher possibility to be linked to the name mention. We address the situation using the following function:

$$\mathrm{Sd}(e_j) = \sum_{m \in d,\ m_j \neq m} \left(1 - NGD(e_j, m)\right) \quad (3)$$

In this formula, d means the document what we will deal with at a time. And e_j is one of the candidate entities of name mention m_j. Also m is the name mentions except m_j in this document d. $Sd(e_j)$ measures the relationship between one entity e_j and the document d.

C. Local Context Similarities between Mentions and Entities

In this section we discuss capturing similarities between the local context of a mention m and a specific entity e. Traditional methods model this as a bag of words and map to a vector space where the name mention m is represented as a vector of its context words and entity e is represented as a vector of its Wikipedia words. All words are weighted using their TFIDF values. This method sometimes consumes a great amount of time, since the dimensions of entity vectors can be thousands or even larger. To deal with this, we utilize word2vec by Tomas et al.[8] and Ilya et al. [9].

Word2Vec is an open source project released by Google which achieved state-of-the-art performances in many natural language processing tasks. It takes a large text corpus as input and outputs word vectors for each unique word. The resulting word vector file can be used as features in a number of natural language processing and machine learning applications.

Tomas carried out two neural network models for representation learning: Continuous Bag-of-Words Model (CBOW) and Continuous Skip-gram Model. We use the figure 3 to represent these two models. Skip grams are word windows from which one word is excluded, an n-gram with gaps. With skip-grams, given a window size of n words around a word w, word2vec predicts contextual words c; i.e. in the notation of probability $P(c|w)$.

Conversely, CBOW predicts the current word, given the context in the window, $P(w|c)$. To be detailed, suppose that input words to the model are $w_{i-2}, w_{i-1}, w_{i+1}, w_{i+2}$, namely the preceding and following words of the current word w_i we are at. The output of the neural network is to estimate most likely w_i. Hence we can think of the task as "predicting the word given its context". Note that the number of words we use depends on your setting for the window size

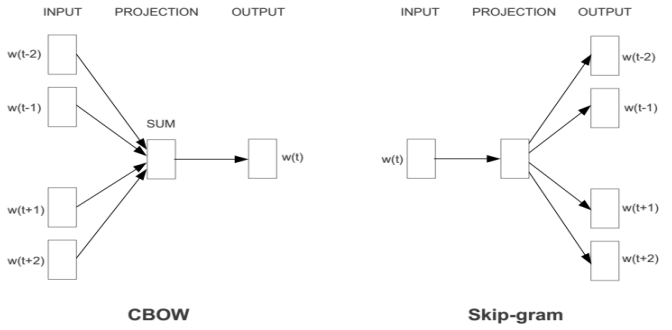

Fig. 3. CBOW and Skip-Gram Model Structure

By using outputs in the vector representation by word2vec, we can calculate the cosine similarity between a mention m and an entity e.

$$CP(m,e) = \frac{m*e}{|m|*|e|} \quad (4)$$

Here, the name mention m is represented as a vector of its context words and the entity e is represented as a vector of its Wikipedia page's words.

For candidate entity selection, we collect all redirect links as the anchor dictionary and count how many times it appears in the whole Wikipedia articles. Once we deal with the mention, we choose the similar one in the anchor dictionary. To be detailed, we use the wildcard to find all similar entities to this mention and filter it according to their popularities. For example, sometimes there are too many candidate entities for one mention. We will rank the candidate entities by their counts in descending order and choose top-20 as the mention's candidate entities.

D. Aim of the Model

In this section, we will describe the aim of our model as follows:

$$\text{rank}(m,e) = P(e) * CP(m,e) * Sd(e) \quad (5)$$

Here, *m.e* is the referent entity of the name mention m. P(e) measures the popularity of the entities e and *CP(m,e)* measures the local similarities between name mentions and entities. As defined earlier, *Sd(e)* measures the relatedness between document *d* and entity *e*. A most likely entity *e* is then determined by these three factors.

IV. EXPERIMENTS

In this section, we evaluate the performance of our method and compare it with the traditional methods. We first explain the experiment settings in Section 4.1, then we will discuss and evaluate the results in Section 4.2.

A. Experimental Setting

- **Knowledge Base**

  We use the Dec.20, 2015 English version of Wikipedia as the knowledge base. In total, the knowledge base contains more than 9,000,000 distinct entities. A name-to-entity dictionary contains over 17,000,000 distinct entity names and the candidate referent entities of each name. There are over 400,000,000 semantic relations between entities.

  As to the measure methods, we use the sax which is an event-based parser for xml documents to deal with the whole wiki dump and we treat one article as one entity. Once we deal with the whole wiki dump, we can get the number of the articles in the dump which means we can get the number of entities in the whole dump. We also treat the redirect words in the wiki dump as names of the entities and regard links between redirect words and the document which contains these redirect words as kind of semantic relations.

- **Dataset**

  Our experiments are composed of three parts. The first part is to calculate the semantic relatedness between entities in one document. We need to use the hit numbers of a commercial search engine, and we use Bing. The second part is to incorporate Wikipedia articles as the ground truth of the entity linking results. To be detailed, we use the contents and redirect links of the Wikipedia as the ground truth. The third part is the corpus we use to evaluate the effects of our model. In this paper, we use the KORE dataset which is used in [10] containing a large number of very ambiguous mentions from five domains (Celebration, Musician, Business, Sports, Politics).

  To evaluate the effects of our model, we first test it on a relative small wiki dump which contains 19105 articles and 4015004 entity relationships and then test it on a big wiki dump which contains 9435689 articles and 419250479 entities relationships to test our assumption that the model performs better on a large data set. Then we will compare our model with the baseline.

- **Evaluation Criteria**

  The measure methods are given as follows:

$$\text{Precision} = \frac{|M \cap M^*|}{|M|} \quad (6)$$

$$\text{Recall} = \frac{|M \cap M^*|}{|M^*|} \quad (7)$$

$$\text{F1} = \frac{2*\text{Precision}*\text{Recall}}{\text{Precision}+\text{Recall}} \quad (8)$$

  Here, *M* is the set of referent entities produced by our entity linking method, *M** is the golden standard set of the referent entities judged by human. In this experiment, I manually checked ranked results and count entities.

- **The Baseline**

  TagMe (Ferragina and Scaiella, 2010) is an end-to-end Wikification system specialized in short texts. TagMe performs best among publicly available Wikification systems. We adopt the TagMe as the baseline to judge our model's performance on short texts. The human judge is done by us according to our common senses.

TABLE I. THE FACTS ABOUT CORPUS

Domain	The Number of Each Parts		
	Mentions	*Candidate Entities*	*Word Count*
Celebration	50	809	119
Musician	68	1026	191
Business	70	1045	183
Sports	30	515	76
Politics	41	713	123

TABLE II. RECALL VALUES OF THREE MODELS

Domain	The Recall Value of models on Five Domains		
	TagMe	*CARE(W)*	*CARE(S)*
Celebration	0.300000	**0.360000**	0.240000
Musician	0.500000	0.470588	0.352941
Business	0.300000	**0.300000**	0.214286
Sports	0.433333	0.300000	0.166667
Politics	0.463415	**0.512195**	0.463415
Total	0.393822	0.389961	0.289573

TABLE III. PRECISION VALUES OF THREE MODELS

Domain	The Precision Value of models on Five Domains		
	TagMe	*CARE(W)*	*CARE(S)*
Celebration	0.357143	**0.450000**	0.300000
Musician	0.548387	**0.627400**	0.444444
Business	0.411765	0.344200	0.245902
Sports	0.448276	0.333300	0.185185
Politics	0.500000	**0.567568**	0.513514
Total	0.459459	**0.467593**	0.342466

TABLE IV. THE F1 VALUES OF THREE MODELS

Domain	The F1 Value of models on Five Domains s		
	TagMe	*CARE(W)*	*CARE(S)*
Celebration	0.326087	**0.400000**	0.266667
Musician	0.523077	**0.537815**	0.393443
Business	0.347107	0.320611	0.229008
Sports	0.440678	0.315789	0.175439
Politics	0.481010	**0.538462**	0.487179
Total	0.424116	**0.425263**	0.313808

B. Experimental Results

We conduct experiments on our model CARE(S)(collective approach ranking entities trained with small wiki dump) first to check the result of our model on short wiki dump. Then we will do experiments on our model CARE(W)(collective approach ranking entities trained with whole wiki dump). The experiments' corpus' details are shown in Table I.

We list the number of words, the number of mentions and the number of candidate entities for the fifty documents in Table I. Every domain has 10 documents. To improve recall values, we choose the top 20 candidate entities for every mention. Actually not all name mentions have 20 candidate entities. The word count is the number of words in each domain.

In this experiment, we also take use of the Standford University tools postag[11] to preprocess the text to find the name mentions in texts.

The experiment results of TagMe, our model with whole wiki dump CARE(W) and with small wiki dump CARE(S) are shown in Table II, Table III, and Table IV.

According to the experiments, we find that the model with whole wiki dump CARE(W) performs better than the model with part wiki dump CARE(S) in precision values, recall values, F1 values proving our assumption that the model will perform better with bigger data sets.

Comparing the model CARE(W) with TagMe, we find that the model CARE(W) performs better than the TagMe in Celebration, Musician part, and Politics part on precision values and F1 values while the TagMe performs better in the Business part and Sports part on F1 values and precision values. As to the recall values, CARE(W) performs better in Celebration and Politics domain and performs as well as TagMe in Business domain.

In total we can see that the CARE(W) performs a litter better than the TagMe and far better than CARE(S).It is remarkable that our model is wining in almost many domains except in Business part domain and sports part domain.

The main reason why TagMe performs a little better than CARE(W) in Business Part and Sports Part is that the business and Sports corpus are too short compared to other domains. CARE(W)'s semantic relations feature and local context similarities feature doesn't play a main role in these domains. Actually in these domains, the popularities of candidate entities play a very important role in the final ranking. However it is not enough to determine the referent entities of a name mention simply by the popularities of entities. What's more, sometimes our methods to measure the popularities of entities can't reflect the popularities of entities. For example, the candidate entities for David can be many such as David Bowie (British Musician), David Beckham (Famous British Football Player) and so on. Most of guys would know David Beckham and some may know David Bowie. That's to say the David Beckham should be more popular than David Bowie. However, in Wikipedia, there are more links pointing to David Bowie than David Beckham. Our model currently couldn't solve this kind of problem, we will solve it in our future works.

As TagMe performs best among publicly available Wikification systems on short texts. We can see that our system can compete with the state of the art system in most cases. We expect that an even larger dataset can improve our results

CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a 'Collective Approach to Ranking Entities for Mentions' by utilizing global topical coherence that means semantic relations between entities, as well as using local mention-to-entity compatibility, to improve recall and precision. According to our experiments on short texts, our model could compete with the TagMe. In the future, we will perform this model on larger corpus and longer documents.

According to our experiments, we also find that the popularities of entities playing a very important role in the final ranking results. Our current method measuring the popularities of entities sometimes can't reflect the popularities very well, we will improve this method in future work

REFERENCES

[1] Mihalcea, R. & Csomai, A. 2007. Wikify!: linking documents to encyclopedic knowledge. In: Proceedings of the sixteenth ACM CIKM.

[2] Cucerzan, S. 2007. Large-scale named entity disambiguation based on Wikipedia data. In: Proceedings of EMNLPCoNLL.

[3] Bunescu, R. & Pasca, M. 2006. Using encyclopedic knowledge for named entity disambiguation. In: Proceedings of EACL, vol.6.

[4] Medelyan, O., Witten, I. H. & Milne, D. 2008. Topic indexing with Wikipedia. In: Proceedings of the AAAI WikiAI workshop.

[5] X. Han, L. Sun, and J. Zhao. 2011. Collective entity linking in web text: A graph-based method. In Proc.SIGIR2011.

[6] X.Han, L.Sun.2011. A Generative Entity-Mention Model for Linking Entities with knowledge Base in Proc. ACL2011

[7] Rudi L. Cilibrasi and Paul M.B. Vita´ny. 2007. In IEEE Transactions on knowledge and data engineering.

[8] Strube, M. and Ponzetto, S. P. 2006. WikiRelate! Computing Semantic Relatedness Using Wikipedia. In: Proceedings of AAAI.

[9] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. In Proceedings of Workshop at ICLR, 2013.

[10] Johannes Hoffart., CIKM 2012 KORE: Keyphrase Overlap Relatedness for Entity Disambiguation

[11] Manning, Christopher D., Mihai Surdeanu, John Bauer, Jenny Finkel, et al 2014. The Stanford CoreNLP Natural Language Processing Toolkit In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp. 55-60.