CoRE: A Context-Aware Relation Extraction Method for Relation Completion

Zhixu Li, Mohamed A. Sharaf, Laurianne Sitbon, Xiaoyong Du and
Xiaofang Zhou *Senior Member, IEEE*

Abstract—We identify Relation Completion (RC) as one recurring problem that is central to the success of novel big data applications such as *Entity Reconstruction* and *Data Enrichment*. Given a semantic relation $\mathcal{R}$, RC attempts at linking entity pairs between two entity lists under the relation $\mathcal{R}$. To accomplish the RC goals, we propose to formulate search queries for each query entity α based on some auxiliary information, so that to detect its target entity β from the set of retrieved documents. For instance, a Pattern-based method (PaRE) uses extracted patterns as the auxiliary information in formulating search queries. However, high-quality patterns may decrease the probability of finding suitable target entities. As an alternative, we propose CoRE method that uses context terms learned surrounding the expression of a relation as the auxiliary information in formulating queries. The experimental results based on several real-world web data collections demonstrate that CoRE reaches a much higher accuracy than PaRE for the purpose of RC.

Index Terms—Context-Aware Relation Extraction, Relation Completion, Relation Query Expansion

1 Introduction

THE abundance of Big Data is giving rise to a new generation of applications that attempt at linking related data from disparate sources. This data is typically unstructured and naturally lacks any binding information (i.e., foreign keys). Linking this data clearly goes beyond the capabilities of current data integration systems (e.g., [7], [4]). This motivated novel frameworks that incorporate *Information Extraction (IE)* tasks such as *Named Entity Recognition (NER)* [20], [8] and *Relation Extraction (RE)* [31], [23]. Those frameworks have been used to enable some of the emerging data linking applications such as *Entity Reconstruction* [13], [9] and *Data Enrichment* [5].

In this work, we identify *Relation Completion (RC)* as one recurring problem that is central to the success of the novel application mentioned above. In particular, an underlying task that is common across those applications can be simply modeled as follows: for each *query entity* α from a *Query List* L_α, find its *target entity* β from a *Target List* L_β where (α, β) is an instance of some semantic relation $\mathcal{R}$. This is precisely the Relation Completion task, which is the focus of the work presented in this paper. To further illustrate that task, consider the following scenarios:
Scenario 1: A research institution needs to evaluate the quality of publications of its researchers *w.r.t.* a given list of conference and journal ranking. Many researchers, however, may not provide the exact venue names within their publications record as per the ranking list . In this case, an RC task is performed between the list of publication titles and the list of venues. This is clearly an example of an entity reconstruction problem, in which each paper entity is reconstructed from different data sources.
Scenario 2: Two on-line book stores in different languages, such as English and Japanese, want to merge their databases to provide bilingual information for each book. Literal translation is not acceptable, especially when some books already have popular and quite different names in different languages. This problem is naturally defined as an RC task between the two book lists in English and Japanese, which is an example of a data integration problem in the absence of foreign key information.

To accomplish the RC task, a straightforward approach can be described as follows: 1) formulate a web search query for each query entity α, 2) process the retrieved documents to detect if it contains one of the entities in the target list L_β, and 3) if more than one candidate target entities is found, a ranking method is used to break the ties (e.g., frequency-based [14]). Clearly, however, this approach suffers from the following drawbacks: First, the number of retrieved documents is expected to be prohibitively large and in turn, processing them incurs a large overhead. Second, those documents would include significant amount of noise, which might eventually lead to a wrong β.

In contrary to the basic approach above, our goal is to formulate effective and efficient search queries based on RE

- *Z. Li, and M. A. Sharaf are with the School of Information Technology and Electrical Engineering, The University of Queensland, Brisbane QLD 4072 Australia. Email: zhixuli@itee.uq.edu.au, m.sharaf@uq.edu.au,*
- *L. Sitbon is with the school of Electrical Engineering and Computer Science, Queensland University of Technology, Brisbane, Australia. Email: laurianne.sitbon@qut.edu.au*
- *X. Du is with the Key Laboratory of Data Engineering and Knowledge Engineering, MOE China, and School of Information, Renmin University of China, Beijing 100872 China. Email: duyong@ruc.edu.cn*
- *X. Zhou is with the School of Information Technology and Electrical Engineering, The University of Queensland, Brisbane QLD 4072, Australia. He is also an Adjunct Professor with the School of Computer Science and Technology, Soochow University, China. Email: zxf@itee.uq.edu.au*
- *Part of this work has appeared as a short paper in CIKM'11[18]*

methods. In general, given some semantic relation $\mathcal{R}$ (e.g., (`Lecturer`, `University`)), general RE tasks target at obtaining relation instances of the relation $\mathcal{R}$ from free text. Clearly, our approach is motivated by the observation that RC can be perceived as a more specialized and constrained version of the more general RE task. Specifically, while RE attempts to find arbitrary entity pairs that satisfy a semantic relation $\mathcal{R}$, RC attempts to match sets of given entities α and β under a semantic relation $\mathcal{R}$. In that respect, existing general RE methods can potentially solve the more specialized RC problem.

For instance, consider employing the state-of-the-art Pattern-based semi-supervised Relation Extraction method (PaRE) [1], [4] for the purpose of RC. In general, given a small number of seed instance pairs, PaRE is able to extract patterns of the relation $\mathcal{R}$ from the web documents that contain those instances. Hence, a web search query can be formulated as a conjunction of a PaRE extracted pattern together with an entity query α and the target entity β is extracted from the returned documents. For example in Figure 1(a), given seed instances of the relation (`Lecturer`, `University`) such as *(Jack Davis, Cambridge), (Tom Smith, Oxford) and (Bill Wilson, U. of Sydney)*, patterns shared by these instances in text, such as "`[Lecturer]` *joined* `[University]` *in* ...", can be found. Based on the pattern, we could formulate a query for an incomplete instance, such as query ("Ama Jones joined" + "in") for (Ama Jones, ?). From the returned documents, we could then easily extract "UCLA" as the linked entity.

The PaRE method, however, relies on high-quality patterns which may decrease the probability of finding suitable target entities. That probability is further reduced when an entity query α is used in conjunction with a high-quality pattern. In other words, while an entity query α provides more context for finding a target entity β, the PaRE method falls short in leveraging that context and instead it formulates a very strict search query, which could possibly return very few and irrelevant documents. For example, Figure 1(a) shows that no documents have been retrieved for the query ("Bob Brown joined" + "in") and hence, an incomplete instance (Bob Brown, ?). In fact, our experimental evaluation on real datasets shows that no more than 60% of query entities can be successfully linked to their target entities under the PaRE method. The remaining 40% query entities were mainly entities appeared in very few web pages (i.e., long tail). Though some of those pages contained the correct target entities, PaRE fell short in finding those pages since they failed to satisfy the strict patterns used in formulating the PaRE-based search queries.

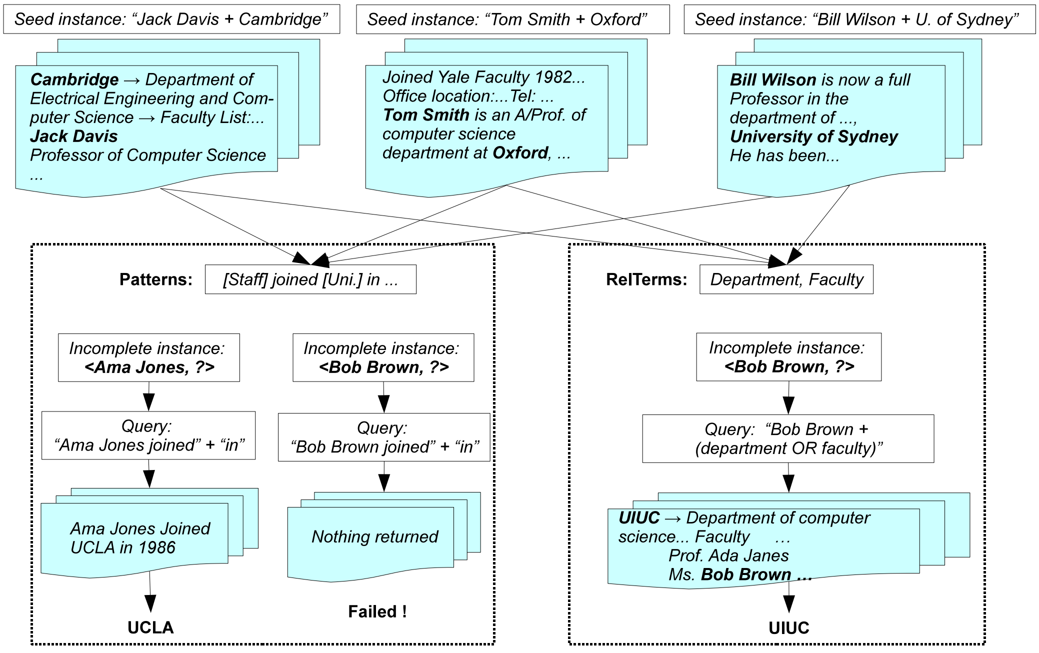
Given such limitations of directly adopting PaRE, we propose a novel *Context-Aware Relation Extraction method (CoRE)*, which is particularly designed for the RC task. CoRE recognizes and exploits the particular context of an RC task. Towards this, instead of representing a relation in the form of strict high-quality patterns, CoRE uses context terms, which we call *Relation-Context Terms (RelTerms)*. For example in Figure 1(b), CoRE searches the web for documents that contain each of the seed instance pairs and from those documents it learns some RelTerms such as "department" and "faculty". Based on those RelTerms, CoRE can formulate a query such as "Bob Brown + (department OR faculty)" for the incomplete instance (Bob Brown, ?). From the returned documents, we can then obtain "UIUC" as the target entity.

Compared to PaRE, CoRE provides two main advantages: 1) it allows more flexibility in formulating the search queries based on context terms instead of patterns, and 2) it seamlessly allows including any query entity as one of the context terms, which further improves the chances of finding a matching target rather than lowering it. This is particularly important for RC tasks in which the objective is to maximize the number of correctly matched entities under a relation $\mathcal{R}$, rather than finding a large number of arbitrary entities that might satisfy $\mathcal{R}$ but are not part of the input lists, which would be the case when employing a general-purpose RE method such as PaRE.

In comparison to PaRE, given the large number of possible RelTerms, and in turn the large number of possible query formulations, realizing an effective and efficient CoRE involves further challenges: 1) learning high-quality RelTerms: as for PaRE, it is quite straightforward to learn patterns which are exactly the same sequences of words surrounding some pairs of linked entities across different web pages. RelTerms, however, can be any terms that are mentioned frequently with some entity pairs, and 2) query formulation: as for PaRE, each pattern can be used to formulate one search query for each query entity. RelTerms, however, can be used in different combinations, and each combination corresponds to a potential search query. Meanwhile, not every combination can be used to formulate an effective search query for a given query entity. Given those challenges, we proposed different techniques that are employed by CoRE so as to maximize both efficiency and effectiveness. Our main contributions in this work are summarized as following:

- We propose CoRE, a novel *Context-Aware Relation Extraction method (CoRE)*, which is particularly designed for the RC task.
- We propose an integrated model to learn high-quality *Relation-Context Terms (RelTerms)* for CoRE. This model incorporates and expands methods that are based on terms' frequency, positional proximity and discrimination information.
- We propose a tree-based query formulation method, which selects a small subset of search queries to be issued as well as schedules the order of issuing queries.
- We propose a confidence-aware method that estimates the confidence that a candidate target entity is the correct one. This enables CoRE to reduce the number of issued search queries by terminating the search whenever it extracts a high-confidence target entity.

As demonstrated by our experimental evaluation, CoRE provides more flexibility in extracting relation instances while maintaining high accuracy, which are desirable fea-


Seed instance: "Jack Davis + Cambridge"
Seed instance: "Tom Smith + Oxford"
Seed instance: "Bill Wilson + U. of Sydney"
Cambridge → Department of Electrical Engineering and Computer Science → Faculty List:... Jack Davis Professor of Computer Science ...
Joined Yale Faculty 1982... Office location:...Tel: ... Tom Smith is an A/Prof. of computer science department at Oxford, ...
Bill Wilson is now a full Professor in the department of ..., University of Sydney He has been...
Patterns:
[Staff] joined [Uni.] in ...
Incomplete instance: <Ama Jones, ?>
Incomplete instance: <Bob Brown, ?>
Query: "Ama Jones joined" + "in"
Query: "Bob Brown joined" + "in"
Ama Jones Joined UCLA in 1986
Nothing returned
UCLA
Failed !
RelTerms:
Department, Faculty
Incomplete instance: <Bob Brown, ?>
Query: "Bob Brown + (department OR faculty)"
UIUC → Department of computer science... Faculty ... Prof. Ada Janes Ms. Bob Brown ...
UIUC

(a) PaRE (b) CoRE

Fig. 1. Comparing PaRE and CoRE in the Context of RC

tures for fulfilling the RC task. We also demonstrate the effectiveness and efficiency of our proposed techniques in learning relation terms and formulating search queries.

Roadmap: We give an overview of CoRE in Sec. 2. The RelTerms learning algorithm is introduced in Sec. 3 while the Query Formulation algorithm is presented in Sec. 4. The experimental setup is described in Sec. 5, and the experimental results are in Sec. 6. We cover related work in Sec. 7, and then conclude in Sec. 8.

2 Background and CoRE Overview

Relation Completion (RC) is rapidly becoming one of the fundamental tasks underlying many of the emerging applications that capitalize on the opportunities provided by the abundance of big data (e.g., Entity Reconstruction [13], [9], Data Enrichment [5], [16], etc). We formally define the Relation Completion (RC) task as follows.

Definition *1:* (**Relation Completion (RC)** Given two entity lists L_α and L_β and a semantic binary relation $\mathcal{R}$, the goal of Relation Completion (RC) is to identify for each entity $\alpha \in L_\alpha$ an entity $\beta \in L_\beta$ which satisfies $(\alpha, \beta) \in \mathcal{R}$. Accordingly, L_α is a *query list*, L_β is a *target list*, α is a *query entity* and β is α's *target entity*.

Similar to classical semi-supervised Relation Extraction (RE) [1], [7], the semantic binary relation $\mathcal{R}$ is expressed in terms of a few seed linked entity pairs between L_α and L_β. Differently, however, the goal of *Relation Extraction (RE)* is to detect semantic relationship mentions in natural language. Formally, given a binary relationship $\mathcal{R}$ between two types of entities, then an entity pair (α, β) is linked under $\mathcal{R}$, i.e. $(\alpha, \beta) \in \mathcal{R}$, if α and β satisfy the semantic relation $\mathcal{R}$. For instance, given $\mathcal{R} =$ *(Company, Headquarter)*, we have *(Microsoft, Redmond)* $\in \mathcal{R}$.

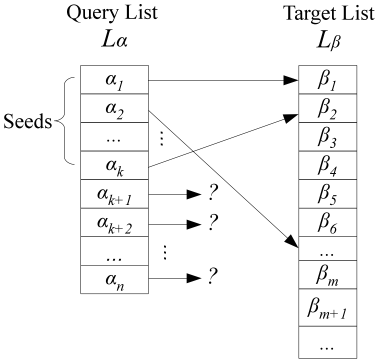


Fig. 2. Demonstration of Relation Completion

Hence, relation completion is a more specialized and constrained version of the more general RE task. In particular, RC is a *targeted* task, which is driven by a set of predefined entities (i.e., query list L_α as shown in Figure 2). RC attempts to match sets of given entities α and β under a relation $\mathcal{R}$. For instance, consider employing the state-of-the-art Pattern-based semi-supervised Relation Extraction method (PaRE) [1], [7], [4] for the purpose of RC. Hence, a web search query can be formulated as a conjunction of a PaRE extracted pattern together with an entity query α, and the target entity β is extracted from the returned documents. The PaRE method, however, relies on high-quality patterns which may decrease the probability of finding suitable target entities. That probability is further reduced when α is used in conjunction with a high-quality pattern.

To overcome the limitations of PaRE, we propose a novel Context-Aware Relation Extraction method (CoRE), which can recognize and exploit the particular context of an RC task. CoRE represents a semantic relation R in the form of context terms, which we call *Relation-Context*

Terms (RelTerms). In our work, RelTerms provide the basis for formulating web search queries that are especially composed for the purpose of RC.

Specifically, CoRE employs what we call a *Relation Query (RelQuery)*, which is basically a web search query that is specially formulated for the purpose of relation completion. Such RelQuery is formally defined as follows:

Definition *2:* (**Relation Query (RelQuery)** A *Relation Query (RelQuery)* is a web search query formulated to retrieve documents containing the target entity β for the query entity α using some auxiliary information Aux.

Further, we denote a retrieved document that contains the correct target entity as *RelDoc*, which is defined as follows:

Definition *3:* (**Relation-Cotext Document (RelDoc)** A retrieved document is denoted *Relation-Context Document (RelDoc)* if and only if it contains the target entity β for the query entity α.

Clearly, a RelQuery is a keyword-based search query. However, the supplied auxiliary information Aux determines the specific nature of such RelQuery. Similarly, the choice of Aux determines the number of retrieved RelDocs as well as the probability of finding the correct matching target in those documents. In particular, consider formulating a RelQuery for the incomplete instance (Bob Brown, ?) illustrated in Figure 1. For which, we consider the following choices of Aux:

1) `Query-based`: A single RelQuery is posed, which is based only on the query entity α (i.e., Bob Brown) and is relation-oblivious (similar to the straightforward approach presented in Sec. 1). It is expected that this approach will return an overwhelming number of web documents, out of which very few are RelDocs that contain the correct target entity.
2) `Pattern-based`: Multiple RelQueries are posed, each of which is based on the query entity α in conjunction with one of the patterns extracted by the PaRE method (e.g., ("Bob Brown joined" + "in"), ("Bob Brown works at"'), etc). Using patterns as auxiliary information will generate very strict RelQueries, which will return the least number of web documents, but most of which are RelDocs. Hence, if a query entity α happened to appear in a web page under one of the used patterns, it will be quickly matched with its correct target entity. However, such assumption is unrealistic for many query entities that appear in very few web pages (i.e., long tail). For those entities, no web pages will be returned and will remain unmatched.
3) `Target-based`: This formulation is orthogonal to the Pattern-based one above, where Multiple RelQueries are posed, each of which is based on the query entity α and an entity β_c from the target list. Hence, each of the retrieved documents is processed to detect any of the patterns extracted by the PaRE method to justify whether $(\alpha, \beta_c) \in \mathcal{R}$. Obviously, this formulation incurs a large overhead as it requires posing a large number of RelQueries for each query entity as well as processing the documents retrieved by those queries.
4) `Context-based`: This formulation is based on our proposed CoRE, in which multiple RelQueries are posed, each of which is based on the query entity α in conjunction with several RelTerms extracted by the CoRE method (e.g., ("Bob Brown" + "Department"), ("Bob Brown" + "Faculty"), etc). By using RelTerms, a limited number of documents are retrieved, among which some are RelDocs that contain the correct target entity.

Clearly, each of the choices mentioned above affects both the efficiency and effectiveness of the RC task. Our CoRE context-based formulation tries to strike a fine balance between a very strict RelQuery formulation (i.e., pattern-based) and a very relaxed one (i.e., query-based). Towards this, CoRE exploits RelTerm towards a flexible query formulation in which a RelQuery is formulated based on the query entity α in conjunction with one or more RelTerms.

However, given the large number of possible RelTerms, and in turn the large number of possible RelQuery formulations and their corresponding retrieved documents, realizing an effective and efficient CoRE requires addressing the following challenges:

- Learning RelTerms: CoRE utilizes the existing set of linked pairs towards learning *Relation Expansion Terms* (i.e., *RelTerms*) for any relation R. This task involves two main challenges: (i) learning a set of high-quality candidate RelTerms from each existing linked pair (**Sec. 3.1**), and (ii) Consolidating and pruning those individual candidate sets into a minimal global set of RelTerms that are used in the formulation of RelQueries (**Sec. 3.2**).
- Formulating RelQueries: CoRE formulates and issues a set of Relation Queries (i.e., RelQueries) for each query entity α based on the set of learned RelTerms. However, there are many possible formulations, each of which is based on α and a conjunction of RelTerms. Clearly, formulating and issuing all those queries will incur a large overhead, which is impractical. Hence, one major challenge is to minimize the number of issued RelQueries while at the same time maintaining high-accuracy for the RC task. Towards achieving that goal, we propose two orthogonal techniques: 1) a confidence-aware termination condition, which estimates the confidence that a candidate target entity is the correct one (**Sec. 4.1**), and 2) a tree-based query formulation method, which selects a small subset of RelQueries to be issued as well as schedules the order of issuing those RelQueries (**Sec. 4.2**).

3 Learning Relation Expansion Terms

CoRE utilizes the existing set of linked pairs towards learning the Relation Expansion Terms (i.e., RelTerms) for any given relation $\mathcal{R}$. This task involves two main steps: 1) learning a set of candidate RelTerms for each existing linked pair, and 2) selecting a global set of RelTerms from those individual candidate sets.

3.1 Learning Candidate RelTerms

Several factors such as frequency, position, and discrimination, are typically considered in selecting good expansion terms in the conventional *Query Expansion (QE)* models [14], [24], [30]. In learning the candidate RelTerms for a given linked pair, we also take those factors into account and they are summarized as follows:

1) **Frequency:** The RelTerm is mentioned frequently across a number of different RelDocs that are relevant to the given linked pair.
2) **Position:** The RelTerm is mentioned closely to the two entities in the given linked pair, such that it could help bridging the query entity to its target entity.
3) **discrimination:** The RelTerm is mentioned much less in irrelevant documents (or non-RelDocs) than in RelDocs.

These factors naturally lead to three formal selection models as described below. Meanwhile, for the remainder of this section, we use Q_+ to denote a web search query, which takes as an argument a linked pair α+β and returns only the set of relevant documents F_+ containing both α and β. Similarly, Q_- denotes a web search query, which takes as an argument α-β and returns only the set of non-relevant documents F_- containing α but not β.

3.1.1 Frequency-based Model

The frequency-based model we propose is an adaptation of the classical relevance model [14]. Specifically, the work in [14] assumes different levels of document relevance based on some criteria (e.g., search engine ranking), whereas in our work all retrieved documents are considered equally relevant as long as they contain α+β. This adaptation enables CoRE to enrich the set of RelTerms with useful terms that might as well appear beyond the top-ranked documents. Accordingly, in our model F_+ is simply the set of all retrieved documents and the probability that a term e is a RelTerm for a given linked pair is estimated as follows:

$$P(e|Q_+) = \frac{1}{|F_+|} \sum_{D \in F_+} P_{freq}(e|D) \tag{1}$$

where $P_{freq}(e|D)$ is the probability of the term e being mentioned in document D, which can in turn be estimated by the Bayesian smoothing using Dirichlet priors proposed in [29].

$$P_{freq}(e|D) = \frac{tf(e,D) + \mu P_{ML}(e|\mathcal{C})}{|D| + \mu} \tag{2}$$

where $tf(e,D)$ is the frequency of term e in document D, $|D|$ is the length of document D. Additionally, $P_{ML}(e|\mathcal{C})$ is the maximum likelihood estimation of the probability of e in the collection of web documents $\mathcal{C}$ indexed by the employed web search engine, which can be approximately estimated with the term frequencies from the Web1T corpus [1]. Finally, μ is the Dirichlet prior parameter of Dirichlet smoothing. In our experiments, we set $\mu = 1500$, which is the average length of the documents in collection $\mathcal{C}$.

3.1.2 Position-based Model

The frequency-based model described above selects RelTerms that might appear in any position within the document. Such approach is most likely to introduce multiple irrelevant terms as RelTerms (i.e., noise) since there are typically multiple topics and irrelevant information within a relevant document. Hence, in this work we also consider a position-based model, which exploits the position and proximity information of terms as cues for assessing if a term is "close" enough to be used as a RelTerm in a RelQuery. Our position-based model is adapted from the one proposed by Lv *et. al.* [19] by defining the location of effective RelTerms in terms of α and β. In particular, under the position-based model we compute $P(e|Q_+)$ as:

$$P(e|Q_+) = \frac{1}{|F_+|} \sum_{D \in F_+} P_{pos}(e|D) \tag{3}$$

where $P_{pos}(e|D)$ can be estimated by:

$$P_{pos}(e|D) = \frac{\sum_{i \in pos(\alpha,\beta,D)} P(e|D,i) + \mu P_{ML}(e|\mathcal{C})}{|D| + \mu} \tag{4}$$

where $pos(\alpha, \beta, D)$ is the set of instances where α and β appear close to one another in D, with the distance between α and β being no larger than a given threshold δ_1. $P(e|D,i)$ is the probability of term e being in the proximity of the i-th instance in document D, which can be simplified according to Eq. 5,

$$P(e|D,i) = \begin{cases} 1.0 & \text{if } e \text{ is within } \delta_2 \text{ to the i-th } (\alpha,\beta) \text{ within } \delta_1 \\ 0.0 & \text{otherwise} \end{cases} \tag{5}$$

As shown in Figure 3, for the i-th pair of α and β in D, if the distance between α and β within a threshold δ_1, then a RelTerms might be found within another threshold δ_2 distance to either side of α or β, or in between α and β, i.e., the range specified in the figure between the left boundary and the right boundary.

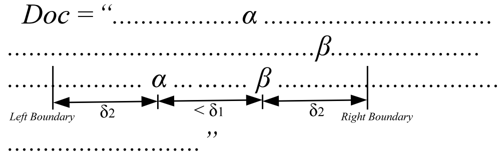

Doc = "..................α ....
....β.....................
.....|............ α .... β ....|.................
Left Boundary
δ2
< δ1
δ2
Right Boundary
..........................."

Fig. 3. The area to find RelTerms in Position-based Model

3.1.3 Discrimination-based Model

Given the two models described above, it is expected to learn the most distinctive set of RelTerms that are able to differentiate between relevant and irrelevant documents on the Web [14], [24]. However, minimizing the number

[1] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2006T13

of documents that contain only α without any candidate β is an important objective in the process of RelQuery Formulation (described in the next section). Therefore, it is necessary to ensure that the selected RelTerms are effective in distinguishing RelDocs from those irrelevant documents. Accordingly, we estimate the probability that a term e is a distinctive RelTerm for a linked pair using Eq. 6,

$$P(e|Q_+) = \frac{1}{|F_+|} \sum_{D \in F_+} P_{dis}(e|D) \tag{6}$$

where $P_{dis}(e|D)$ is estimated as:

$$P_{dis}(e|D) = \frac{tf(e,D) + \mu_n P_{ML}(e|Q_-)}{|D| + \mu_n} \tag{7}$$

where the Dirichlet prior μ_n is still set to 1500 in our experiments, and $P_{ML}(e|Q_-)$ is the maximum likelihood estimation of term e in Q_-, which can be estimated as:

$$P_{ML}(e|Q_-) = \frac{1}{|F_-|} \sum_{D \in F_-} P(e|D) \tag{8}$$

where F_- is the set of non-RelDocs retrieved using Q_-.

3.1.4 Hybrid Model

Putting it all together, we propose a hybrid model that integrates the three individual factors listed above according to:

$$P(e|Q_+) = \frac{1}{|F_+|} \sum_{D \in F_+} [\lambda P_{pos}(e|D) + (1-\lambda) P_{dis}(e|D)] \tag{9}$$

where λ is an interpolation weight, which is a system parameter.

Notice that in our integrated model, while $P_{pos}(e|D)$ represents the position-based model, it also covers the frequency-based model since it considers the number of documents. Also notice that given the integrated model above, CoRE learns a number of RelTerms, each with a different probability for different linked pairs. In the next subsection, we propose techniques for the selection of a set of general RelTerms for relation completion based on those probabilities (i.e., $P(e|Q_+)$).

3.2 Selecting General RelTerms for the Relation

After learning all the possible candidate RelTerms from each of the existing individual linked pair, CoRE selects a set of general RelTerms from those candidates. The goal is to select a set of high-quality RelTerms for effective query formulation, and in turn accurate relation completion (i.e., finding target entities). In CoRE, this task takes place in two steps: in the first step, CoRE uses a local pruning strategy to eliminate the least effective RelTerms, and in the second step, CoRE uses a global selection strategy to choose the most effective RelTerms.

During the local pruning step, CoRE verifies the effectiveness of each RelTerm in extracting the target entity for the linked pair from which it was learned. In particular, in the verification of a linked pair such as (α_i, β_i), α_i is considered as a seed RelQuery without auxiliary information and each learned RelTerm $e_{i,j}$ is used as a candidate to such seed query with auxiliary information. That is, to formulate a keyword-based query $\alpha_i + e_{i,j}$. Accordingly, we measure the accuracy achieved for the top-ranked documents that returned and ranked by the employed web search engine, P@N, i.e., the ratio of documents containing the actual target β (top-100 is enough to indicate the performance). To set up a baseline for comparison, we also measure the accuracy of the top-ranked documents which are retrieved with the unexpanded seed query. If the improvement of P@N is evident i.e., the improvement of P@100 is significantly (for example, more than 30%), then the verified RelTerms survive the elimination step and is promoted to the second step (i.e., global selection), which is described next.

During the global selection step, CoRE creates a set of a general RelTerms that are best fit for completing the relation under consideration. Intuitively, the RelTerms belonging to more linked pairs with higher probability should have a better coverage rate. Hence, one possibility is to employ a selection model based on the number of covered linked pairs by each of the RelTerm candidate (which we call as *query-based* model) as given in Eq. 10,

$$P(e|\mathcal{R}) = \sum_{(\alpha,\beta) \in T} P(e|Q_+^{(\alpha,\beta)}) \tag{10}$$

where each (α, β) is a linked pair in the training set T, and $P(e|Q_+^{(\alpha,\beta)})$ can be calculated by Eq. 9.

However, there are many cases in which this query-based model defines a higher coverage rate for anecdotal RelTerm than for actual general RelTerms, especially when we have a relatively biased training set. For example in learning the RelTerms for the relation (Academic Staff, University), if there is a relatively large number of existing linked pairs for academics working at universities located in “London”, then “London” might be learned as RelTerm candidate for all those pairs. Hence, according to Eq. 10, “London” might appear as a general RelTerm for the academics relation.

As an alternative, we propose a cluster-based selection model, in which we cluster the linked pairs in the training set and then estimate the coverage of RelTerms in terms of clusters instead of linked pairs. The purpose of query clustering is to reduce the influence of a possibly skewed distribution of examples.

3.2.1 Clustering Linked Pairs

Similar to any clustering task, linked pairs clustering can be performed according to many possible techniques [10]. In this work, we opt to use the density-based clustering algorithm DBSCAN [6] because of its ability to automatically detect the number of clusters in a data set as well as its efficiency.

Central to the clustering techniques, however, is defining an effective measure of similarity. Given two linked pairs (α_r, β_r) and (α_s, β_s) under relation $\mathcal{R}$, we argue that the similarity between two entities is in terms of their contexts rather than their lexical similarity. To define the context of each linked pair, we exploit the fact that the top-ranked relevant documents F_+ returned by a search engine are the most relevant to a linked pair and in turn define its context. We get all the context terms for the linked pair (α_r, β_r) within the same area defined in Fig. 3 in $F_{+}^{(\alpha_r,\beta_r)}$. Then the similarity between the two pairs is measured by:

$$Sim((\alpha_r, \beta_r), (\alpha_s, \beta_s)) = Cosine(CT_r, CT_s) \quad (11)$$

where $Cosine(.,.)$ measures the cosine similarity between two vectors, and CT_r and CT_s are the term frequency vectors for the context of (α_r, β_r) and (α_s, β_s) over the same dimension of context terms, respectively.

3.2.2 Cluster-based RelTerms Selection

The cluster-based RelTerm selection model is formalized as follows:

$$P(e|\mathcal{R}) = \sum_{C \in Clusters} P(e|C) \quad (12)$$

where $P(e|C)$ measures the utility of RelTerm e in determining the target entities within cluster C, which is defined as:

$$P(e|C) = \frac{1}{|C|} \sum_{(\alpha,\beta) \in C} P(e|Q_{+}^{(\alpha,\beta)}) \quad (13)$$

where $|C|$ is the number of linked pairs in C.

Given the cluster-based selection model, CoRE ranks all candidate RelTerms according to their probability score calculated by Eq. 12. In the next section, we describe how CoRE utilizes those ranked RelTerms towards effective and efficient formulation of RelQueries.

4 RelQuery Formulation

In Section 3, we have addressed the challenge of learning high-quality RelTerms for some semantic relation $\mathcal{R}$. In this section, we address the second major challenge towards realizing CoRE. That is, the formulation of efficient and effective RelQueries. In order to put that challenge in perspective, recall that for each query entity α, there are many possible formulations of a RelQuery, each of which is based on α and a conjunction of RelTerms. In particular, assume that n RelTerms are learned, then there are (n^2-2) different combinations of RelTerms, leading to (n^2-2) different formulation of RelQueries for each α. Obviously, formulating and issuing all those queries will incur a large overhead, which is impractical. Hence, our goal is to minimize the number of issued RelQueries while at the same time maintaining high-accuracy for the RC task.

Towards achieving that goal, we propose the following two orthogonal techniques: 1) a confidence-aware termination condition, which estimates the *confidence* that a candidate target entity β_c is the correct target entity (Section 4.1), and 2) a tree-based query formulation method, which selects a small subset of RelQueries to be issued as well as schedules the order of issuing those RelQueries (Section 4.2). Our termination condition can be used independently or in synergy with our tree-based query formulation method.

When the termination condition is used independently, all the possible RelQueries for a query entity α are ordered arbitrarily and the termination condition is checked after each of those queries is issued. That is, calculate the confidence that one of the candidate target entities β_c extracted from the retrieved documents is the right target entity β. If the confidence is higher than a threshold,

that is the case, CoRE stops issuing more queries and the search for a target entity is terminated successfully.

While the termination condition is expected to eliminate the need for issuing many of the possible RelQueries, further improvements are attainable by tuning the issuing order of such queries. Ideally, the most effective RelQuery for each α in the query list should be issued first. In reality, however, it is impossible to determine which is the most effective RelQuery for each α. But since the different combinations of RelTerms form a hierarchical structure in which some combinations subsume others, it is often possible to predict the effectiveness of one RelQuery based on the perceived estimated effectiveness of another RelQuery that has already been issued. As such, CoRE builds a tree that captures the relationship between the different combinations of RelTerms. Further, it employs a tree-based query formulation method which ranks the promising combinations of RelTerms while pruning those combinations that are predicted to be ineffective.

4.1 Confidence-Aware Termination

Each time we fired a RelQuery for an entity α, we will identify all candidate target entities from the retrieved documents using Named Entity Recognition (NER) method [20], [8]. For a given the entity type such as "Organization", the NER method is expected to identify all phrases that refer to organizations in the documents. However, the state-of-the-art NER methods can only identify limited types of entities such as "Organization", "Time" or "Location" etc. Hence, we use the target list as a *dictionary* to aid the NER process, as they did in the Dictionary-based Entity Extraction method [17]. In particular, we find all approximate mentions of those dictionary entries in each document, such that those mentions form a list of candidate target entities.

When more than one target entities are found, a ranking method is required to get the most possible target entity β for each query entity α. Here we propose a confidence-based ranking method, which calculates a confidence for each candidate target entity β_c. We believe that the following three parameters help define the confidence of β_c w.r.t. being the target entity of α.

- Frequency $freq(\beta_c, d)$: the number of times entity β_c is mentioned in document d. Apparently, the more times β_c is mentioned, that higher confidence that it is the target entity of α;
- Distance $dist(\beta_c, \alpha)$: the distance between each mention of β_c and α is also very important to reflect the relationship between β_c and α;
- Document confidence $conf(d)$: This is the confidence of document d w.r.t. whether it is a RelDoc. Usually, the higher confidence that d is a RelDoc, the higher confidence that β_c found in d is the target entity of α.

For each entity β_c, we measure the probability that β_c is the target entity of α (i.e., β) with a heuristic formula below:

$$P(\beta_c|\alpha) = \frac{\sum_{d \in Docs} conf(d) \cdot s(r,d)}{\sum_{d \in Docs} conf(d)} \tag{14}$$

where $s(r,d)$ is the local score of β_c in d, which can be defined as follows:

$$s(\beta_c, d) = w \cdot \frac{freq(\beta_c, d)}{N} + (1-w) \cdot \sum_{1 \le i \le freq} \frac{|d| - dist_i(\beta_c, \alpha)}{freq(r,d) \cdot |d|} \tag{15}$$

where $|d|$ is the length of document d, $freq$ is the frequency of β_c in d, $dist_i$ is the distance between the i-th mention of β_c and the query entity α in d, N is a normalization factor, w is a scaling factor. The confidence of each retrieved document can be estimated by any of the measures proposed below. The best choice of the confidence measure will be estimated empirically in Section 6.1.

Uniform: As a baseline, we can use a uniform value to the confidence of all retrieved documents, that is,

$$conf(d) = 1 \tag{16}$$

Page Rank: The N documents of Doc can be partitioned into B ($1 \le B \le N$) ranges according to their ranks returned by the web search engine, so that the confidence of a document $conf(d)$ follows the Normalized Discount Cumulative Gain (NDCG) function [11], which is popularly used for assigning degrees of importance to web documents in a ranked list. More specifically,

$$conf(d) = \frac{log(2)}{log(1 + \frac{\lceil rank(d) \rceil}{B})} \tag{17}$$

Number of RelTerms: The confidence of a retrieved document can be decided by the number of RelTerms it contains, that is,

$$conf(d) = \frac{|Terms(d)|}{|ET(\mathcal{R})|} \tag{18}$$

where $E(\mathcal{R})$ is the set of learned RelTerms of $\mathcal{R}$, and $Terms(d)$ is the set of terms in document d.

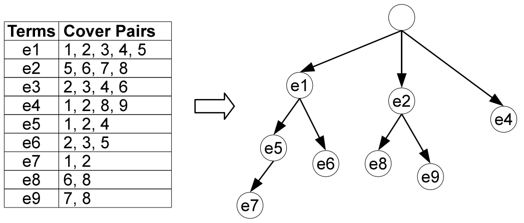
Confidence of RelTerms Combination: We can retain for the confidence of a document the highest confidence of any RelTerm combinations contained in this document, that is,

$$conf(d) = \underset{E \subseteq Terms(d)}{\text{ArgMax}} \; conf(E) \tag{19}$$

The confidence of a given RelTerm combination is estimated from the set of linked entity pairs. More specifically, we get the distribution of each learned RelTerm amongst the retrieved web documents, either RelDocs or non-RelDocs, of each linked pair. Based on this distribution, we can estimate the confidence of a RelTerm combination E as:

$$conf(E) = \frac{\sum_{p \in T} N_D(E,p,+)}{\sum_{p \in T} N_D(E,p)} \tag{20}$$

where T is the linked pairs set, p is a linked pair, $N_D(E,p)$ is the number of retrieved documents for p which contains all RelTerms in E, while $N_D(E,p,+)$ is the number of retrieved RelDocs among them.



Terms	Cover Pairs
e1	1, 2, 3, 4, 5
e2	5, 6, 7, 8
e3	2, 3, 4, 6
e4	1, 2, 8, 9
e5	1, 2, 4
e6	2, 3, 5
e7	1, 2
e8	6, 8
e9	7, 8

(a) Cover Table (b) Sorted RelTerm Tree

Fig. 4. Example Cover-based Sorted RelTerm Tree

In the RelQuery Formulation (QF) process, once we detect that the $P(\beta_c|\alpha)$ of a candidate target entity β_c is higher than a given threshold, we could terminate to use more RelTerms to formulate more RelQueries for the target entity α. The value of the threshold will be discussed in Section 6.4.

4.2 Tree-Based QF Method

In this section, we first introduce how we construct a tree with RelTerms based on the set of linked pairs each RelTerm covers in the training set (Sec. 4.2.1). We call this tree as a *Cover-based Sorted RelTerm Tree (CSRTree)*, which is expected to capture the relationship between different combinations of RelTerms. Based on the CSRTree, we then present our Tree-based QF method, which skips over ineffective RelTerms, and also generates effective combinations of RelTerms as expansion terms in QF (Sec. 4.2.2).

4.2.1 Sorted RelTerm Tree Construction

Basically, the CSRTree is formulated according to the "cover-based relation" between RelTerms. In particular, when a RelTerm e is learned from a linked pair in the training set, we say the RelTerm *covers* the linked pair. We say a RelTerm e_b is a *SubCoverTerm* of a RelTerm e_a (e_a is a *SuperCoverTerm* of e_b), if e_b only covers a subset of the linked pairs covered by e_a in the training set.

Given a set of linked pairs $\mathcal{S}$, if a set of RelTerms could cover the maximum number of distinct linked pairs in $\mathcal{S}$ with the least number of RelTerms, we call this set of RelTerms as the *Minimum Cover Set (MinCoverSet)* of $\mathcal{S}$, which can be established following a traditional greedy algorithm: We list all RelTerms in a descendent order according to the number of their covered linked pairs in $\mathcal{S}$. At each iteration, we select the RelTerm that covers the largest number of uncovered linked pairs into the *MinCoverSet*, until no more pairs can be covered. For example in Fig. 4, e_1 is firstly selected into the MinCoverSet, then e_2 is the next RelTerm that covers the most uncovered pairs (3 pairs: 6, 7, 8). After that, e_4 is the third one selected into the MinCoverSet, which covers all the left uncovered pairs. Finally, we have the MinCoverSet as $\{e_1, e_2, e_4\}$.

Now we introduce how we construct the *Cover-based Sorted RelTerm Tree (CSRTree)*: The root of the tree is a blank node which is supposed to cover all linked pairs. Except the root node, all other nodes in this tree is a RelTerm. Assume a node n_x covers a set of linked pairs

$\mathcal{S}(n_x)$, then the children nodes of n_x is the MinCoverSet of $\mathcal{S}(n_x)$. Specifically, the MinCoverSet of the whole training set are children nodes of the root node. Finally, each node covers no less linked pairs than its brothers lying on its right.

For example in Fig. 4, the RelTerms in the MinCoverSet of the whole training set $\{e_1, e_2, e_4\}$ are taken as children nodes of the root. Since e_1 covers more entity pairs than e_2, and e_4 covers less entity pairs than e_2, we put e_1 on the left-most position, and e_4 on the right-most position. Then, for each node such as e_1, we find the MinCoverSet of linked pairs set $\mathcal{S}(e_1)$ as its children nodes in this tree, until no more nodes can be included in the tree.

4.2.2 Tree-Based QF Method

We now introduce the Tree-based QF method based on the CSRTree. For each query entity α, we begin with the root node, and then traverse the whole tree in a depth-first manner. We will keep a *Current Expansion Term Set (CETs)* to store the expansion terms that are used to expand α together in the current RelQuery. For example, if CETs contains $\{e_1, e_5, e_7\}$, the RelQuery will be $e_1 + e_5 + e_7 + \alpha$.

In the beginning, at the root node, the CETs is empty, so the first RelQuery is an unexpanded query to α. Each time we traverse to a node, we will add the RelTerm in this node into CETs, and then construct a new RelQuery accordingly. We then submit the current RelQuery to the web search engines, and find out all candidate target entities from the returned top-K web pages. Since the web search engines maximumly return 100 web pages at a time, for efficiency issue, we also set $K = 100$ here. Three situations might arise then (we use node e_5 as an example, the current RelQuery should be $e_1 + e_5 + \alpha$):

Situation 1: There are at least one candidate target entities whose confidence is higher than a given threshold. According to the Confidence-Aware Termination condition, we will exit the QF process. When several target entity candidates are found, only the one with the highest confidence will be taken as the target entity.

Situation 2: Otherwise, assume there are N_D documents returned by the current RelQuery, and $N_D \leq K$, then we have already gone through all returned documents without finding good candidates. We skip over all the descendent nodes under the node (such as e_7 under e_5), and the RelTerm in this node (e_5) will be removed from CETs. Next, we move to its first brother node (e_6) on the right. If there is no brother node on the right, we skip to the first un-traversed brother node of its parent node on the right, and the RelTerm in its parent node will also be removed.

Situation 3: Otherwise, we have $N_D > K$ (most of the time $N_D \gg K$), for efficiency issues, we won't go through the documents after top-K. Instead, we move to the first of its un-traversed child node (e_7) without touching CETs. However, if there is no un-traversed child, we skip to the next un-traversed brother of its father node. Meanwhile, we remove the RelTerms in this node and its parent node.

Note that in situation 2, although it can not be guaranteed that the SubCoverTerms of a failed RelTerm will also fail when it is applied to the same query entity, they lose their priority to be selected as the next RelTerm. Also note that in situation 3, when a RelQuery can't return RelDocs in the top-ranked documents, we don't drop the RelTerm in this RelQuery directly, but to refine the results with its SubCoverTerms. Although this combination may remove some RelDocs, but it is supposed to remove a lot more irrelevant documents and bring the left RelDocs to top-ranked ones.

5 Experimental Setup

5.1 Data Sets

We perform RC on four real-world data sets below:

Academic Staff & University (Staff): About 25k academic staff's full names (from 20 different universities) and their universities have been collectedWe also collected 500 university names from the SHJT world university ranking[2].

Book & Author (Book): This data set contains more than 43k book titles collected from Google Books[3]. These books are of more than 20 different categories including education, history *etc.* We have also collected about 20k book writers' names (including the chief authors of the 43k books) from Google Books.

Invention & Inventor (Invention): This data set contains 512 inventions' names with their chief inventors' full names (311 different people) from an inventor list[4] in Wikipedia.

Drug & Disease (Drug): This data set contains 200 drug names and the names of 183 different diseases they can cure. It was extracted from a drugs list[5].

All four data sets exhibit 1-1 semantic relations. That is, each query entity has only one target entity in target list.

5.2 Metrics

Three Metrics are used to estimate the effectiveness or efficiency of our proposed techniques and models. (1) **P@N**: The precision of top N documents, that is, the percentage of RelDocs in the top N retrieved results. (2) **RC Accuracy**: To estimate the effectiveness of CoRE and PaRE, we apply them in the Relation Completion task. The accuracy of RC is the percentage of initially unlinked pairs that could be correctly linked. (3) **AvgQueryNum**: The average number of processed queries for each RelQuery; The first two metrics are designed for measure effectiveness, the third one measures efficiency.

5.3 Implementation

1. CoRE v.s. PaRE We mainly compare CoRE with PaRE in the context of RC. For CoRE, we use Google API[6] to retrieve documents and associated snippets from the web. The number of feedback documents we use for each query is fixed to 100, which is the maximum that the search

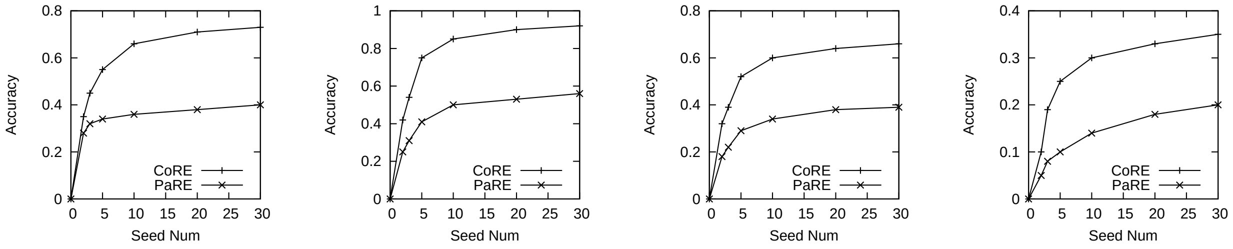
[2] http://www.arwu.org/

[3] http://books.google.com/

[4] http://en.wikipedia.org/wiki/List_of_inventors

[5] http://en.wikipedia.org/wiki/List_of_bestselling_drugs

[6] http://www.google.com



(a) Acd. Staff & Univ. (b) Book & Author (c) Invention & Inventor (d) Drug & Disease

Fig. 5. Comparing the Join Accuracy of RC by Using CoRE and PaRE Respectively

engine returns at a time. We learn RelTerms from single relation query with the Hybrid Positional+Semi-Negative model, and then do the tree-based QF method. For PaRE, we adapt the state-of-the-art method proposed in NELL (Never-Ending Language Learner) [4] for PaRE.

Both CoRE and PaRE are semi-supervised methods, which require a small number of seed linked entity pairs. Therefore, we perform experiments with several numbers of seed linked pairs ($|T|$=2, 3, 5, 10, 20, 30). For each size, cross-validation is achieved by generating 5 different random sets for training, while all the remaining pairs in each data set are used for testing.

2. Models for Learning Candidate RelTerms: Four models were proposed for learning RelTerms from single linked pairs in Sec. 3.1, including: (1) The **Frequency-based** model; (2) The **Position-based** model; (3) The **Discrimination-based** model; (4) The **Hybrid** of all the above models. To demonstrate the effectiveness of the four models, we evaluate them in two dimensions: (1) RC accuracy; and (2) P@100.

3. Models for Selecting General RelTerms: Three possible models are compared in selecting general RelTerms: (1) The **Query-Based** Model; (2) The **Cluster-Based** Model; (3) The **Unexpanded** Model. We evaluate in two dimensions: (1) RC Accuracy; and (2) P@100.

4. Confidence-Aware Termination: Four different options are available to estimate the confidence of each retrieved document, including: (1) The **Uniform** option; (2) The **Page Rank** option; (3) The **Number of RelTerms** option; (4) The **Conf. of RelTerms Combination** option. We will work out a proper threshold for the Confidence-Aware Termination (CA-Term) for each of the options, and then compare the four options in one dimension: RC Accuracy. Finally, to demonstrate the effectiveness of the Confidence-Aware Termination strategy, we compare the RC Accuracy and AvgQueryNum of our Tree-based QF method with or without using the Confidence-Aware Termination strategy.

5. RelQuery Formulation Methods: We will compare proposed efficient QF method with two baselines. Thus we have three methods as following: (1) The **Linear Coverage-based** QF uses top-K RelTerm (combinations) for expansion one after one; (2) The **Linear MinCoverSet-based** QF uses RelTerms in the MinCoverSet to do the expansion one after one; (3) The **Tree-based** QF selects RelTerms and their combinations with the guidance of the Sorted Graph. We will evaluate them in two dimensions: (1) RC Accuracy; and (2) AvgQueryNum.

6 Experimental Results

We present all the experimental results in this section.

6.1 CoRE v.s. PaRE

We now compare the RC accuracy of applying either CoRE or PaRE in the context of RC with different number of seed instances. As can be observed in Figure 5, CoRE always reaches a higher RC accuracy than PaRE on all the four data sets with different number of seeds.

To further illustrate our accuracy results, Table 1 provides a more comprehensive comparison based on the precision, recall and F_1 metrics, in which the seed size is set to 10. Here recall is the percentage of linked pairs, precision is the percentage of pairs that are correctly linked, while $F_1 = 2 \times \frac{precision \times recall}{precision + recall}$.

TABLE 1
Comparing CoRE and PaRE Comprehensively

Acd. Staff & Univ.	Precision	Recall	F1	Accuracy
PaRE	0.965	0.425	0.589	0.41
CoRE	0.730	1.000	**0.843**	**0.73**
Book & Author	Precision	Recall	F1	Accuracy
PaRE	0.955	0.607	0.742	0.58
CoRE	0.920	1.000	**0.958**	**0.92**
Invention & Inventor	Precision	Recall	F1	Accuracy
PaRE	0.930	0.452	0.607	0.42
CoRE	0.660	1.000	**0.795**	**0.66**
Drug & Disease	Precision	Recall	F1	Accuracy
PaRE	0.940	0.170	0.288	0.16
CoRE	0.350	1.000	**0.518**	**0.35**

As shown in Table 1, the precision of PaRE is usually very high($\geq$ 90%), but its recall is typically low. To the contrast, the recall provided by CoRE is always high (=1.0). This is because in the absence of a confidence threshold, CoRE can always find some target entity for each query entity α even if it is a false positive. But as expected, this comes as the expense of a lower precision when compared to PaRE. Overall, however, the F_1 score achieved by CoRE is always greater than PaRE, which emphasizes the the advantage provided by CoRE over PaRE.

Through observations to the linked results, we found that all the pairs that can be successfully linked by PaRE were also linked by CoRE, but CoRE could link about 20%-30% more pairs than PaRE. These 20%-30% entity pairs are deemed as "long tail" entity pairs that appeared in very few web pages. Though some pages may mention the correct target entities, PaRE fell short in finding these pages due to the strictness of PaRE-based search queries.

The experimental results also demonstrate that as the number of the seeds increases from 2 to 10, the performance of both CoRE and PaRE improves dramatically, whereas, only slight improvements are observed after the seed number becomes larger than 10. We conclude that a small number of seeds are enough to launch CoRE or PaRE in the context of RC. While the number of seed instances is small ($\simeq$ 10), the total number of training sentences is sufficiently large ($\simeq$ 1000), which is the reason for the accuracy of both PaRE and CoRE tends to stabilize after the seed number is larger than 10. Such behaviour is expected since our method is a web-based semi-supervised method instead of a supervised one.

From the experiments, we also observe that both PaRE and CoRE reach a relatively higher RC accuracy on the Book data set than that on the other three data sets, probably because book and author are more commonly to be mentioned in some formal formats. As a result, patterns or RelTerms are better shared amongst different instances. The worst performance is on the Drug data set, since different kinds of drugs are described in different words, thus they share less patterns as well as RelTerms.

6.2 Models for Learning Candidate RelTerms

In all the future experiments, we set the default size of seeds to 30. The evaluation results to the four learning models are listed in Table 2. The frequency-based model only takes frequency into account, thus it gets the lowest P@100 and RC accuracy. Both the position-based and discrimination-based models work better than frequency-based model. However, the combination of the three models reaches the best performance. In the following subsection, the hybrid model will be set as the default model for learning the RelTerms for single queries.

TABLE 2
Comparing Models for Learning Candidate RelTerms.

Acd. Staff & Univ.	P@100	Accuracy
Frequency-based	0.074	0.60
Position-based	0.086	0.67
Discrimination-based	0.087	0.68
Hybrid	**0.101**	**0.73**
Book & Author	P@100	Accuracy
Frequency-based	0.180	0.82
Position-based	0.198	0.86
Discrimination-based	0.201	**0.88**
Hybrid	**0.260**	**0.92**

6.3 Models for Selecting General RelTerms

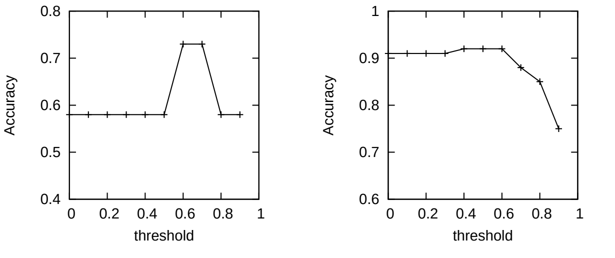
The experimental results of the model comparison are listed in Table 4. The performance of Cluster-based model is always above that of Query-based model, which shows the advantage of estimating the "coverage" of RelTerms amongst clusters instead of queries. Both the two models are better than the baseline. For a better observation, we list the top 10 single RelTerms learned with either Query-based model or Cluster-based model in Table 3. From the listed terms, we also directly observe the advantage of Cluster-based model over Query-based model. Take the (Academic Staff, University) relation for example, RelTerms like "technology" and "australia" are too specific. They are learned by Query-Based since there are some staff members in the linked pairs who work on technology, or their universities are located in Australia. For the same relation Cluster-Based provide other RelTerms like "research", "phd" and "edu". It is reasonable since a large part of staff in universities are also researchers and usually hold a PhD. "edu" should be the suffix of their homepages. A remaining issue is that few RelTerm can work for all queries, since not every academic staff is a "professor" or "lecturer".

TABLE 4
Comparing Models for Selecting General RelTerms

Acd. Staff & Univ.	P@10	P@50	P@100	Accuracy
Unexpanded	0.087	0.054	0.050	0.41
Query-Based	0.094	0.074	0.089	0.70
Cluster-Based	**0.145**	**0.103**	**0.101**	**0.73**
Book & Author	P@10	P@50	P@100	Accuracy
Unexpanded	0.230	0.201	0.109	0.58
Query-Based	0.242	0.241	0.203	0.88
Cluster-Based	**0.285**	**0.281**	**0.260**	**0.92**

6.4 Confidence-Aware Termination Strategy

To get a proper threshold for CA-Term, we get the trend of *RC accuracy* along with the threshold from 0 to 1. As presented in Figure 6, a threshold less than 0.5 makes a very loose termination condition, such that the accuracy of the found target entity is not high; on the other hand, a threshold greater than 0.6 seems too strict to reach a high accuracy. The highest accuracy can be reached when the threshold is between 0.5 and 0.6.



(a) Acd. Staff & Univ.

(b) Book & Author

Fig. 6. The Effect of Confidence Threshold to the Accuracy

We also work out the best threshold for each of the other three document confidence estimation options respectively. As presented in Table 5, the *Conf of RelTerms Combination* option could always reach the best accuracy among all the four options. Although the *Num of RelTerms* option works as good as the *RelTerms Combination Confidence* option on one of the data sets, it can not compete on the other three data sets.

To demonstrate the effectiveness of the Confidence-Aware Termination strategy, we compare the RC accuracy

TABLE 3
Top-10 RelTerms Learned with Query-Based or Cluster-Based (with underlined ones are not good RelTerms)

Acd. Staff & Univ.		Book & Author		Invention & Inventor		Drug & Disease	
Query-Based	**Cluster-Based**	**Query-Based**	**Cluster-Based**	**Query-Based**	**Cluster-Based**	**Query-Based**	**Cluster-Based**
university	university	author	author	inventor	inventor	efficacy	efficacy
professor	professor	book	book	invented	invented	approved	approved
school	school	isbn	isbn	invention	invention	mg	mg
faculty	faculty	hardcover	hardcover	named	developed	safety	safety
technology	dr	paperback	publisher	developed	father	drug	drug
department	research	fiction	published	engineer	scientist	patients	patients
australia	institute	publisher	written	father	did	combination	used
dr	phd	published	wrote	scientist	patented	used	treat
institute	edu	written	edited	famous	created	generic	dose
lecturer	lecturer	wrote	amazon	credited	invent	treat	tablets

TABLE 5
Comparing Options in Estimating the Conf. of Document

Acd. Staff & Univ.	Accuracy
Uniform	0.65
Page Rank	0.68
RelTerms Number	0.70
RelTerms-Combination Conf.	**0.73**
Book & Author	Accuracy
Uniform	0.86
Page Rank	0.86
RelTerms Number	0.86
RelTerms-Combination Conf.	**0.92**

TABLE 6
Comparing Tree-based QF with or without CA-Term

Acd. Staff & Univ.	Accuracy	AvgQueryNum
Tree-based QF without CA-Term	0.73	38.25
Tree-based QF + CA-Term	**0.73**	**6.05**
Book & Author	Accuracy	AvgQueryNum
Tree-based QF without CA-Term	0.92	19.24
Tree-based QF + CA-Term	**0.92**	**3.43**

and AvgQueryNum of our Tree-based QF method with using the Confidence-Aware Termination strategy and without using the strategy. As listed in Table 6, by applying the Confidence-Aware Termination strategy, the RC accuracy won't be decreased, and at the same time, it greatly decreases the AvgQueryNum that used in doing QF.

6.5 Evaluation on QF Methods

We compare the three QF methods combined with CA-Term. As listed in Table 7, the Tree-based QF method always reaches the highest accuracy among the three methods. Besides, it issues much less expanded queries than the Liner Coverage-based method. Although the MinCoverSet-based method issues even less expanded queries than our method, it could not reach as high accuracy as ours in doing RC. To summarize, our Tree-based QF method outperforms the two baselines.

6.6 Extension for Many-to-Many Mapping

TABLE 7
Comparing QF methods working together with CA-Term

Acd. Staff & Univ.	Accuracy	AvgQueryNum
Linear Coverage-based	0.68	26.60
Linear MinCoverSet-based	0.58	2.20
Tree-based QF	**0.73**	**6.05**
Book & Author	Accuracy	AvgQueryNum
Linear Coverage-based	0.84	7.55
Linear MinCoverSet-based	0.90	2.44
Tree-based QF	**0.92**	**3.43**

So far in this paper, we have assumed a many-to-one setting of the RC problem so that to ensure a fair comparison between our proposed context-aware CoRE scheme vs. the pattern-based PaRE scheme. In particular, PaRE has been mainly designed and used for detecting instances in many-to-one mapping problems, such as the set expansion problem [26], [27]. The RC problem, however, naturally lends itself to the more general setting, in which there is a many-to-many mapping between the source and target lists. The main challenge in the many-to-many version of the RC problem is to automatically decide the number of target entities for each query entity. While this challenge is common to both the PaRE and CoRE schemes, the solution is expected to be different for each. In particular, for PaRE, one reasonable starting point in the solution space would be to consider "all" the retrieved entities that satisfy the PaRE patterns as target entities. This solution, however, requires further tuning in case the number of target entities is too large. The simplest form of tuning is to utilize a threshold value k that acts as a knob to control the number of returned target entities. While the same approach could also be incorporated in CoRE, we note that CoRE relies on heuristic probabilistic functions (Eq. 14), instead of patterns, to detect the target entities. Thus, under CoRE, the preliminary solution outlined above can be further tuned using two orthogonal knobs, namely: 1) threshold c on the confidence of each target entity, and 2) threshold k on the number of target entities. The threshold c is simply learned from the linked instances during the learning phase, while the threshold k is similar to that of PaRE.

To evaluate the effectiveness of the solutions outlined above, in this section, we present some preliminary results on a many-to-many version of the Book data set. In that version, each book in the data set has multiple authors (up to 5 authors, and 3.25 on average). To allow for many-to-many relation completion, we extend PaRE to employ a threshold k on the number of target entities (as explained above). In particular, for each query entity, PaRE ranks its candidate target entities in a descending order according to frequency. For a given threshold of k, PaRE returns the top-k entities in that list (compared to only the top-1 in the many-to-one scenario). For CoRE, in addition to the threshold k, it also employs a threshold c on the minimum acceptable confidence. Accordingly, CoRE ranks candidate target entities according to their confidence as calculated by Eq. 14 and returns the first top entities that have confidence greater than c. If more than k entities satisfy the threshold c, then only the top-k are returned. In this experiment, k is set to 5, whereas c is automatically learned from the linked instances.

Table 8 compares PaRE and CoRE in terms of the imputation precision, recall and F1 score. The table shows that under this many-to-many setting, the precision achieved by PaRE decreases to 85.6%, in comparison to the previously achieved 95.5% under the many-to-one setting (as shown in Table 1). The recall also decreases from 60.7% to 55.8% as pattens tend to miss some more target entities when the number of target entities corresponding to each query entity is getting larger. For the same reasons, the precision and recall of CoRE also decreases from 92.0% to 76.1%, and from 100% to 87.4%, respectively, in comparison to the many-to-one setting. Overall, the F1 score achieved by CoRE is still higher than PaRE, which demonstrates the advantage of CoRE compared to PaRE.

TABLE 8
Comparing CoRE and PaRE in Many-to-Many RC

M-M Book&Author	Precision	Recall	F1
PaRE	0.856	0.558	0.477
CoRE	0.761	0.874	**0.665**

7 Related Work

In this work, we identify Relation Completion (RC) as one recurring problem that is central to the success of some emerging applications. The RC problem, although novel, is still related to some well-studied problems in the areas of data management and information extraction. For instance, the conventional Record Linkage (RL) problem whose goal is to find similar entities across two data sets (e.g., [7], [4]) can be considered a special case of the RC problem, in which the semantic relation between those two data sets is always "same as". In RC, however, that semantic relation can take any arbitrary form such as "published in", "study at", "employed by" or "married to" etc.

RC is also very strongly related to the problems arise in Question Answering systems [28], [15]. In those systems, answers are provided to questions such as "*Which country is the city 'Amsterdam' located in?*" , or "*Who is the author of the book 'The world is flat'?*". Currently, question answering systems rely on Relation Extraction (RE) methods to build an offline knowledge base for providing answers to specific questions. RE methods particularly fit the purpose of question answering systems since its goal is to find arbitrary entity pairs that satisfy a semantic relation $\mathcal{R}$. Meanwhile, RC can be perceived as a more specialized and constrained version of the RE task with the objective of matchings two sets of given entities under a relation $\mathcal{R}$.

While general-purpose RE methods, such as PaRE can be adopted to fulfill the RC task, our experimental evaluation shows that our special-purpose CoRE method provides significant improvements in the RC accuracy. This is primarily because PaRE falls short in incorporating the particular context of the RC task, in which query and target entities are given. Like PaRE, other general-purpose RE methods also suffer from the same shortcoming. In the following, we describe those methods and discuss their usage in the context of the RC task.

Generally, most RE methods can be divided into three categories: supervised, unsupervised and semi-supervised. Supervised RE methods [12], [31] formulate RE as a classification task, and decide whether an extracted entity pair belongs to a given semantic relation type by exploiting its linguistic, syntactic and semantic features. Supervised method mostly built the model based on tree and sequence kernels that can also exploit structural information and interdependencies among labels [22]. However, they are expensive to be applied to new relation types for requirement of labeled data [2]. To solve this problem, recent work [21], [22] used large semantic databases such as WordNet or Freebase, to provide "distance supervision". In particular, they can automatically generate a proper training set with sentences containing pairs of entities in the semantic databases. However, the supervised RE methods are still not appropriate to be used in the context of RC, since we need to generate the feature vectors for each entity pair between the query list and the target list. In order to do that, we need to collect a number of web pages for each entity pair by searching for the web documents containing each term in the query list first and then identify sentences containing each entity pairs. However, this way still requires us to do pair-wised search on all retrieved documents, which will lead to a large overhead.

Unsupervised RE methods [25], [3] produce relation-strings for a given relation through clustering the words between linked entities of the relation in large amounts of text. In some sense, the relation-strings are very similar to the RelTerms learned in the CoRE method, but they only limited in learning these strings, instead of using them to formulate effective RelQueries.

Semi-supervised methods [26], [27] only require a small number of seed instances to capture more instances of the same relation type in a bootstrapping manner. The state-of-the-art semi-supervised relation extraction methods are all pattern-based, which rely on syntactic patterns to identify instances of a relation type. Based on similar idea of learning and then using patterns in a bootstrapping manner, we designed the PaRE method for the RC task.

8 Conclusions and Future Work

In this work, we identify Relation Completion (RC) as one recurring problem that is central to the success of novel big data applications. We then propose a Context-Aware Relation extraction (CoRE) method, which is particularly designed for the RC task. The experimental results based on several real-world web data collections demonstrate that CoRE could reach more than 50% higher accuracy than a Pattern-based method (PaRE) in the context of RC. As future work, we will further study the RC problem under the many-to-many mapping, and investigate techniques for maintaining the high precision and recall achieved under the many-to-one case.

Acknowledgements: This research is partially supported by National 863 High-tech Program (Grant No. 2012AA011001) and the Australian Research Council (Grant No. DP120102829).

References

[1] E. Agichtein and L. Gravano. Snowball: Extracting relations from large plain-text collections. In *ACMDL*, pages 85–94, 2000.

[2] N. Bach and S. Badaskar. A survey on relation extraction. *Language Technologies Institute Carnegie Mellon University*, 2007.

[3] M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni. Open information extraction for the web. In *IJCAI*, pages 2670–2676, 2007.

[4] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. Hruschka Jr, and T. Mitchell. Toward an architecture for never-ending language learning. In *AAAI*, pages 1306–1313, 2010.

[5] S. Chaudhuri. What next?: a half-dozen data management research goals for big data and the cloud. In *PODS*, pages 1–4, 2012.

[6] M. Ester, H. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, volume 1996, pages 226–231, 1996.

[7] O. Etzioni, M. Banko, S. Soderland, and D. Weld. Open information extraction from the web. *Communications of the ACM*, 51(12):68–74, 2008.

[8] J. Finkel, T. Grenager, and C. Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *ACL*, pages 363–370, 2005.

[9] R. Gummadi, A. Khulbe, A. Kalavagattu, S. Salvi, and S. Kambhampati. Smartint: using mined attribute dependencies to integrate fragmented web databases. *Journal of Intelligent Information Systems*, pages 1–25, 2012.

[10] J. Hartigan. *Clustering algorithms*. John Wiley & Sons, Inc., 1975.

[11] K. Jarvelin and J. Kekaainen. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002.

[12] N. Kambhatla. Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations. In *ACL*, page 22, 2004.

[13] G. Koutrika. Entity Reconstruction: Putting the pieces of the puzzle back together. *HP Labs, Palo Alto, USA*, 2012.

[14] V. Lavrenko and W. Croft. Relevance based language models. In *SIGIR*, pages 120–127, 2001.

[15] X. Li, W. Meng, and C. Yu. T-verifier: Verifying truthfulness of fact statements. In *ICDE*, pages 63–74, 2011.

[16] Z. Li, M. A. Sharaf, L. Sitbon, S. Sadiq, M. Indulska, and X. Zhou. Webput: Efficient web-based data imputation. In *WISE*, pages 243–256, 2012.

[17] Z. Li, L. Sitbon, L. Wang, X. Zhou, and X. Du. Aml: Efficient approximate membership localization within a web-based join framework. *Knowledge and Data Engineering, IEEE Transactions on*, 25(2):298–310, 2013.

[18] Z. Li, L. Sitbon, and X. Zhou. Learning-based relevance feedback for web-based relation completion. In *CIKM*, pages 1535–1540, 2011.

[19] Y. Lv and C. Zhai. Positional relevance model for pseudo-relevance feedback. In *SIGIR*, pages 579–586, 2010.

[20] A. Mikheev, M. Moens, and C. Grover. Named entity recognition without gazetteers. In *EACL*, pages 1–8, 1999.

[21] M. Mintz, S. Bills, R. Snow, and D. Jurafsky. Distant supervision for relation extraction without labeled data. In *ACL & AFNLP*, pages 1003–1011, 2009.

[22] T. V. T. Nguyen and A. Moschitti. End-to-end relation extraction using distant supervision from external semantic repositories. In *ACL*, pages 277–282, 2011.

[23] P. Pantel, E. Crestan, A. Borkovsky, A. Popescu, and V. Vyas. Web-scale distributional similarity and entity set expansion. In *EMNLP*, pages 938–947, 2009.

[24] S. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-4. In *TREC*, pages 73–97, 1996.

[25] Y. Shinyama and S. Sekine. Preemptive information extraction using unrestricted relation discovery. In *ACL*, pages 304–311, 2006.

[26] R. Wang and W. Cohen. Iterative set expansion of named entities using the web. In *ICDM*, pages 1091–1096, 2008.

[27] R. Wang, N. Schlaefer, W. Cohen, and E. Nyberg. Automatic set expansion for list question answering. In *EMNLP*, pages 947–954, 2008.

[28] X. Yin, J. Han, and P. Yu. Truth discovery with multiple conflicting information providers on the web. *Knowledge and Data Engineering, IEEE Transactions on*, 20(6):796–808, 2008.

[29] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *SIGIR*, pages 334–342, 2001.

[30] C. Zhai and J. Lafferty. Model-based feedback in the KL-divergence retrieval model. In *CIKM*, pages 403–410, 2001.

[31] S. Zhao and R. Grishman. Extracting relations with integrated information using kernel methods. In *ACL*, pages 419–426, 2005.

Zhixu Li is now a postdoctoral fellow at King Abdullah University of Science and Technology. He received his Ph.D. degree from the University of Queensland in 2013, and his B.S. and M.S. degree from Renmin University of China in 2006 and 2009 respectively. His current research interests include Data Integration, Record Linkage, Information Extraction and Information Retrieval.

Mohamed A. Sharaf is a Lecturer of computer science with The University of Queensland, and an adjunct Assistant Professor in Computer Science at the University of Pittsburgh. He received his Ph.D. from the University of Pittsburgh in 2007 and was a Postdoctoral Research Fellow at the University of Toronto until 2009. He is now a member of the Data and Knowledge Engineering (DKE) group at UQ.

Laurianne Sitbon is a lecturer at the Queensland University of Technology in Brisbane, Australia in the discipline of Computer Science. She got her Ph.D. degree from the University of Avignon, France. From 2008 to 2010 she was a postdoctoral fellow of the Queensland Research Laboratory of National ICT Australia, then of the University of Queensland, both located in Brisbane, Australia.

Xiaoyong Du is a Professor of computer science in the School of Information, Renmin University of China. He is the dean of the School of Information, Renmin University of China, and the director of Key Laboratory of Data Engineering and Knowledge Engineering, MOE China. He received his PhD degree of Computer Science from Nagoya Institute of Technology, Japan, in 1997.

Xiaofang Zhou is a Professor of computer science with The University of Queensland. He is the Head of the Data and Knowledge Engineering Research Division, School of Information Technology and Electrical Engineering. He is the Director of ARC Research Network in Enterprise Information Infrastructure (EII), and a Chief Investigator of ARC Centre of Excellence in Bioinformatics.