GRIAS: an Entity-Relation Graph based Framework for Discovering Entity Aliases

Lili Jiang[†], Ping Luo[II], Jianyong Wang[§], Yuhong Xiong[II], Bingduan Lin[‡], Min Wang[≀], Ning An[‡]
[†]Max Planck Institute for Informatics, Saarbruecken, Germany
ljiang@mpi-inf.mpg.de
[II] HP Labs China, Beijing, China
{ping.luo, yuhong.xiong}@hp.com
[§]Department of Computer Science and Technology, Tsinghua University, Beijing, China
jianyong@tsinghua.edu.cn
[‡]School of Computer and Information, Hefei University of Technology, China
brandonlin.bl@gmail.com, ning.an@hfut.edu.cn
[≀] Google Research, 1600 Amphitheatre Pkwy Mountain View, CA, USA
minwang@google.com

***Abstract*—Recognizing the various aliases of an entity is a critical task for many applications, including Web search, recommendation system, and e-discovery. The goal of this paper is to accurately identify entity aliases, especially the long tail ones in the unstructured data. Our solution GRIAS (abbr. for a Graph-based framework for discoveRing entIty AliaseS) is motivated by the entity relationships collected from both the structured and unstructured data. These relationships help to build an entity-relation graph, and the graph-based similarity is calculated between an entity and its alias candidates which are first chosen by our proposed candidate selection method. Extensive experimental results on two real-world datasets demonstrate both the effectiveness and efficiency of the proposed framework.**

***Keywords*-entity alias; alias similarity; entity-relation graph**

I. Introduction

Entities (e.g., persons, organizations, and products) commonly exist in various types of data and identifying entity aliases is necessary in many applications [1] [2] [3] [4]. For example, given an entity query "IBM" to a search engine, "International Business Machines" or "International Business Machines Corp." may occur in the search results. Accurately identifying these aliases could help get more relevant pages for the given search query. Another important commercial application is e-discovery [5]. When a company is involved in a lawsuit about a product accident (say micro-wave oven explosion), this company is required to disclose all relevant information, where a solution based on e-discovery could include collecting, preparing, reviewing and producing evidences in the electronic form. Often, entities such as persons, organizations, and products are significant in the legal matters. One challenge is that these entities are often referenced in reports, emails, and other types of unstructured text. Their references are often casual, include abbreviations, initials, or partial references. Especially, it is difficult to discover some long tail aliases, which rarely co-occur with the concerned entity. It is sure that the investigation committee does not want to miss any relevant reference that are important to the case. These reference may be the engineers, inspectors, departments, or any accessories of the striken microwave-oven in millions of electronic documents. Traditional keyword search techniques are ineffective to handle these variations. Additionally, since the evidence is always required as much as possible for a lawsuit, the coverage of entity extraction must be highly considered in these applications and the entity aliases are needed.

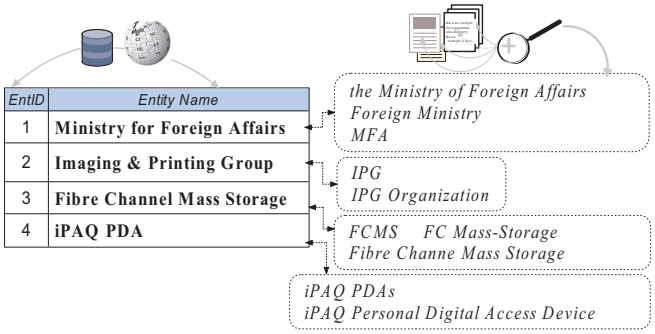


Fig. 1. Examples of Entity-Alias Pairs

Figure 1 clearly illustrates that the real-world entities often appear in various forms in the same/different documents: for each entity on the left side of Figure 1, there is a dashed box containing some variants extracted from related document corpus. The failure of effectively identifying the important aliases leads to evidence missing in the applications mentioned above. Given a concerned entity and millions of documents, the goal of this paper is to discover all the aliases for the concerned entity.

There have been some research efforts that address the task of entity alias discovery [6] [1] [7] [8] [9]. Most of these

1550-4786/13 $31.00 © 2013 IEEE
DOI 10.1109/ICDM.2013.50

studies assume that the relatively full name of the concerned entity is given, which may not hold in real world. Although the concerned entity are often obtained from the authoritative source from structured data (e.g., existing knowledge bases, enterprise databases, or official websites), these entity names are not always cleansed to be the full names. It is quite often that acronyms, word abbreviation, and even any entity variant appear in the given data source. Thus, it is not enough to only consider the shorter forms than the given entity name as the alias candidates.

In this paper we propose a framework, called GRIAS (abbr. for a Graph-based framework for discoveRing entIty AliaseS),to address these challenges by combining our Entity Relation Graph Model (*ERGM*) with our Alias Candidate Selection (*ACS*) method, which explores the entity relationships from both the structured and unstructured data. First, a growing number of structured data can be obtained from various sources (e.g., Wikipedia, Freebase, and Yago), which contain plenty of hierarchy entity information and entity relationships. This is the first type of relationship repository. Second, the unstructured documents provide quantities of entity co-occurrence information, which can be used as another relationship repository. Co-occurrence statistics indicate associative relationship between entities and are helpful for estimating the strength of entities association. Identifying these two types of entity relationships leads to our observation that the aliases of a concerned entity e may frequently co-occur with the entities which are closely related to e in the document corpus. For instance, the product's commonly used variants may often co-occur with the person, who is the designer (manager or salesman) of this product. Therefore, a graph of entity-to-entity relationships can be built to measure the relevance between any entity and its alias candidates determining whether they represent the same real world entity.

The graph contains three types of nodes representing three entity roles: *concerned entity*, *alias candidate* and *attribute entity*. We observed that some concerned entities are connected with their alias candidates through some intermediate nodes. When an entity is to be concerned for finding its aliases, it is concerned entity; excluding concerned entities and their alias candidates, other entities are regarded as attribute entities. In this way, we convert the problem of alias discovery into the task of the graph-based entity similarity matching. More details about this graph will be described in Section V.

To identify the *alias candidates*, we propose an "alias candidate selection (*ACS*) method", which considers many entity alias patterns and identifies the most potential alias candidates. The *ACS* method takes the token weight into account, which may vary in different entities. Note that this study focuses on the situations that the concerned entity and its aliases are similar in terms of string match to a certain degree. There are some entity alias pairs whose string match similarities are quite low, such as a few Dow Jones companies: Exxon Mobil and XON, Coca-Cola and KO. The detection of these entity aliases is not the focus of this study. The main contributions of this paper are as follows:

- We analyze the challenges in entity alias discovery for the low-redundant unstructured document, and propose a graph-based solution called GRIAS by leveraging the entity relationships in both the structured and unstructured data.
- We develop a selector for the initial entity alias candidate selection, covering a wide range of alias patterns, then use the identified candidates in the proposed graph model.
- We evaluate the proposed framework on two real-world datasets: an enterprise dataset including all its internal Web pages and the structured databases for the alias discovery of organization and product. The graph we build based on this dataset consists of more than 170,000 nodes and 13,250,000 edges; a government dataset from WikiLeak[1], which consists of various entities (i.e., organization, person, location), and entity relations. A graph of 155,339 nodes and 10,357,642 edges was built to discover all the aliases of organizations.

Outline. The rest of this paper is organized as follows. Section II discusses the related work. Section III formulates the problem of entity alias discovery and briefly describes the overall process of our proposed framework. Section IV introduces a candidate selection method that selects high quality alias candidates to form nodes in the entity-relation graph model. And in Section V, we model a graph to exploit the entity relationships, which effectively detects the aliases for the entity names. Section VI presents the experimental studies, including data preprocessing and experimental evaluation on a WikiLeak dataset and an enterprise dataset from enterprise. Finally, we conclude our paper in Section VII.

II. Related Work

The studies on entity resolution and record linkage are similar with our work [2] [10] [11] [3] [12] [13]. Also entity linking [14] [15] matches the strings in unstructured data with entities from Wikipedia. Some of their solutions can be referred as Web alias discovery, while alias discovery could be one of the core task in these work. As for the solutions, most of them extract these aliases assuming that the entity and its aliases co-occurrence frequently. Entity resolution is the process of identifying the records that represent the same entity and reconciling them to obtain one record. While record linkage is the process of finding related entries in multiple relations in a database and linking them. Some researchers have used the alias candidate selection method [16]. Danushka et al. [6] [17] proposed a lexical pattern based approach to extract alias candidates for a given person, and measure the extracted candidates using the hyperlink structure and page-counts retrieved from a Web search engine. Additionally, they presented a word co-occurrence graph to represent the mutual relations between anchor text from Web search engines, and models the problem of alias extraction as ranking nodes in the proposed graph with respect to a given name. But their methods are based on the assumption that the full name and its aliases always co-occur in different contexts, which is not true in our case. [7] is special to detect email. [1] works

[1] http://wikileaks.ozazar.org/cablegate.html/

on identifying the aliases of the given product name, which consists of a subset of the tokens in the given product name. It focuses on the efficiency of the proposed solution for this task. Moreover, the research on synonyms/acronym extraction [18] [19] [20] is relevant with alias discovery.

With the ever-increasing popularity of information networks such as the Web, citation networks, or social network, many studies on graph-based similarity address the requirement for measuring similarity between objects [21] [22], among which random-walk based and simrank are most popularly used [23] [24] [25]. [22] identifies the unique entity in the structured database. It views the database as the corresponding entity-relationship graph and then analyzes paths that exist between nodes. In their work, all the entities are provided from databases. The method used in this paper is a combination of string match and graph-based match.

III. Overview of GRIAS

A. Notations and Preliminaries

Entity. Any object existing in real world can be called entity. In this paper, the extracted entity types mainly include organization, person, location, product, and e-mail.

Entity Type. There are different entity type classification, including coarse-granularity (e.g., people, organization, location, and MISC) and fine-granularity (e.g., politician, musician, university and etc.). In this paper, we consider coarse-granularity entity types and extract the entities of people, organization, location, and product.

Token. Given an entity ent, its tokens are defined as a list of words, which are obtained by splitting ent with non-letter characters. For example, the entity of "Human-Resources Office" generates $T = \{\text{Human, Resources, Office}\}$, and then each element in T, denoted by t, is defined as a token of ent.

Token Importance(TI). Inspired by the notion of IDF (Inverse Document Frequency) in information retrieval, *TI* is defined to measure the importance of each unique token occurring in certain type of entities to be resolved. All the non-repetitive tokens in these entity names are indexed, and *TI* is then defined as $TI(t) = \ln \frac{m}{|E_t|}$, where m is the total number of the concerned entity names of a unique entity type, $|E_t|$ is the number of the concerned entities containing token t. When token t frequently occurs in different entities in a document corpus, its TI value is very small, such as "corporation" in organization name, "street" in location name, and "camera" in product name. That is, these types of entity tokens are of little importance for alias discovery.

Alias Candidate. Given an entity ent from E and a document corpus D, all the named entities are obtained from D using the multi-word units [26] and entity recognition model[2], and some filtering rules are used to remove the noisy entities. The proposed alias candidate selection method, *ACS* (described in Section IV), is performed to compare ent and all the cleaned named entities to obtain a set of *alias candidates* in a descending order, $C = \{ent.c_1, ent.c_2, ...ent.c_n\}$, each of which has a string similarity with ent above a certain threshold.

B. Problem Formulation

$F(v, w)$ is defined to estimate whether *v* and *w* are aliases with each other and $f_i(v, w)$ denotes a similarity function based on feature i $(i = 1, 2, \ldots n)$. In this paper, we define **Entity Alias Discovery** as such a task: it takes a concerned entity set E and a document corpus D as inputs, extracts named entities from D to form an alias candidates set C, and computes $F(ent, c)$ $(ent \in E, c \in C)$ through aggregating $f_i(ent, c)$ to generate a set of aliases for ent.

C. The Proposed Solution

Figure 2 depicts the GRIAS architecture. Given a set of concerned entities and a document corpus D, the *preprocessing* step first extracts the named entities from D, cleans them, and records all the statistical occurrences in documents of the given concerned entities and the extracted entities. Next, we design an alias candidate selection method, *ACS*, to initially measure the similarities between all the named entities and each given entity, and select some of them as alias candidates for each entity. After that, we model different entities into an entity relation graph, where the entity relationships from structured data and unstructured documents are mapped to the connecting edges. By exploiting the entity relationship, the proposed solution combines the string similarity f_s from *ACS* with the graph similarity f_g to determine the final aliases for each concerned entity.

In the following section, we discuss the main components of GRIAS in Alias Candidate Selection Method and Entity Relation Graph Model in section IV and V respectively.

IV. *ACS*: Alias Candidate Selection Method

A. Entity Alias Patterns

Through a study on various documents, Table I lists seven popular patterns in forming entity aliases. 1) `Abbreviation` is the most popular pattern of entity alias. 2) `Acronym` is usually formed by taking the initials of a phrase or compounded-word. 3) *Abbreviation* and *acronym* are shortened forms, while aliases are sometimes longer by `expanding` the given entity. 4) Some `stop words`[3], such as "and", "of", and "the" are meaningless and often pruned away in forming an alias. 5) Some tokens are slightly different in an entity and its alias. For example, "Graphics" compares with "Graphic" in the fifth row of Table I. 6) Some tokens may appear in different `orders` in an entity and its alias respectively. 7) Some entity names contain an implicit `hierarchy` relationship. For example, "Canon Cameras EOS Digital Rebel XT" and "EOS Rebel XT" denote the same style of camera. As described above, Table I shows the entity alias patterns with the corresponding examples, and these alias patterns are independent of entity types.

Through the observation on the forming rules above, we design a method for alias candidate selection, *ACS*, which

[2]http://alias-i.com/lingpipe/

[3]http://en.wikipedia.org/wiki/Stop_words

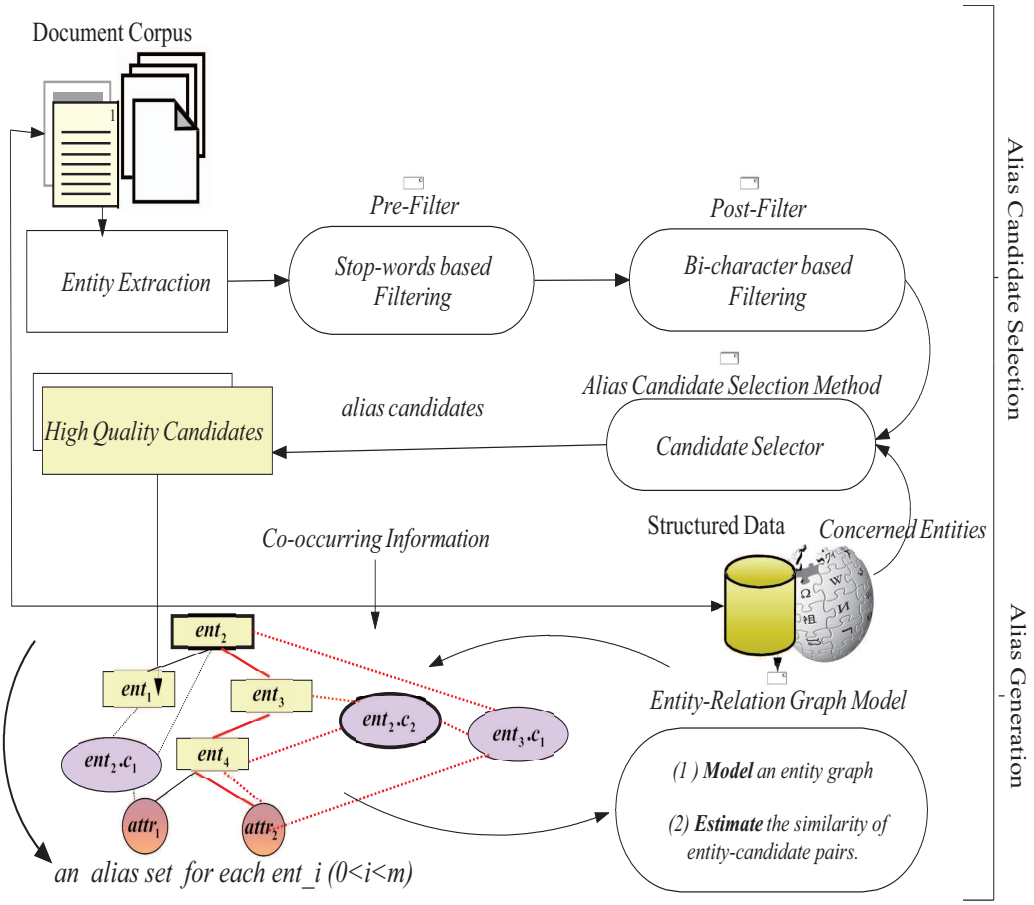


Fig. 2. The Architecture of GRIAS

is expected to achieve two purposes: 1) provides a string similarity score for each entity-candidate pair, and 2) filters the false candidates as many as possible and keeps the potential aliases with high scores, which will be used in the phase of graph similarity computation.

TABLE I
ENTITY ALIAS PATTERNS AND EXAMPLES

	Alias Pattern	Concerned Entity	Alias Example
1	Abbreviation	**Americas** Finance	**AMS** Finance
2	Acronym	The **Technology for Emerging Markets** Research Group	**TEM** Group
3	Expansion	**LAN** Adapter	Network Adapter **Local Area Network**
4	StopWords	PSG AP **&** Japan	PSG AP **and** Japan
5	TokenVariants	**Graphics** Imaging Business	**Graphic** and Imaging Business
6	OrderTransposition	**PDA iPAQ** Pocket	**iPAQ PDA** Pocket
7	Hierarchy	Canon **Cameras EOS Digital Rebel XT**	**EOS Rebel XT**

B. Alias Candidate Selection

The general idea of *ACS* is combining the *rewarding* and *penalization* schemes based on the token importance(*TI*). Given an entity and its alias candidate, *ACS* first splits them into a set of tokens respectively. And, the commonly used stop words (e.g., "and", "for", and "of") are removed from them. Next, we attempt to match the acronym tokens in the given entity and its candidate. After the acronym match, the remaining tokens in the entity are required to find its matching tokens in the candidate. To compute the similarity of each pair of tokens respectively from the entity and its potential alias, we propose the $simi'$ formula which is optimized based on *JaroWinkler* and described in the Appendix. All the *matched* tokens mentioned above will reward the similarity score. If an alias candidate does not belong to any alias type presented in Table I, we will simply use the formula of $simi'$ to get its string similarity. Finally, we impose penalization on the *unmatched* token and normalize the final similarity score. For better understanding, an example will be provided to describe the working process of *ACS* in the following subsection.

C. An Illustrating Running Example

Taking the entity of "HR Information - Technology" and its alias candidate "Information Tech. Office of Human Resources" as example (for convenience, they are named respectively as ent and $candi$.), *ACS* executes the following three

steps to compute the string similarity $score$.

Step 1: Tokenization and Acronym Match. After splitting ent and $candi$ into tokens, it scans ent to find a sequence of *capitalized tokens*, and then attempts to match a sequence of tokens in $candi$, which may be the expanded form of this *capitalized token*. In this example, "HR" is *matched* with "Human Resources". For each match, the *TI* value of the *matched* token in ent is added to $score$. It is important that we remove all the *matched* tokens from both ent and $candi$ to prevent them from getting involved in the further matching. If there are more than one matches, the left encountered match is prefered. In a reverse manner, we scan $candi$ for acronym tokens and look for their expanded forms in ent.

Step 2: Stop Word Removal and Token Match. After the acronym match, all the stop words are removed and then we scan the remaining tokens again in ent and $candi$. For each token s in ent, it finds any token t in $candi$, which has the same initial letter as s, and then computes the similarity between s and t using the function of $simi'$ in Appendix. If there are multiple matches, the *matched* token with the highest $simi'$ is preferred. In the example, "Information" is firstly matched and then "Technology" in ent finds its matching token "Tech" in $candi$. After that, the *matched* tokens in both strings are removed. If there are still some tokens left in both ent and $candi$, $simi'$ is used to compute a similarity score sl between their remaining tokens.

Step 3: Penalization and Normalization. Steps 1 and 2 reward similarity score by adding the matched tokens' *TI* values. While this step computes the *penalty* score based on the *unmatched* tokens, *penalty*=$(1-sl)\times \sum_{x=1}^{n} TI(t_x)$, where n is the number of *unmatched* tokens and sl is the similarity score of the remaining tokens from ent and $candi$. That is, the higher the similarity of the *unmatched* tokens is, the weaker the *penalty* is. Finally, we use the following formula to compute the string similarity

$$f_s = \frac{reward - penalty}{\sum_{x=1}^{|ent|} TI(t_x)} \quad (1)$$

where, *reward* is the rewarding score, *penalty* is the penalizing score, and the denominator represents the TI sum of tokens in ent. In this example, there is only one *unmatched* token "Office". Finally, the string similarity between ent and $candi$ is *0.84*.

The stop words removal is performed after acronym match since acronym sometimes gets stop words involved. All the *matched* tokens removed add rewarding scores. As for the remaining *unmatched* tokens, we use the optimized *JaroWinkler* function in Appendix to compute their similarity scores and combine the *TI* of these *unmatched* tokens for penalization.

V. *ERGM*: AN ENTITY-RELATION GRAPH MODEL

The entity relationships from the co-occurring information in the structured and unstructured data are naturally modeled as a graph. The proposed graph-based approach attempts to compute the graph similarity between each concerned entity and all its alias candidate.

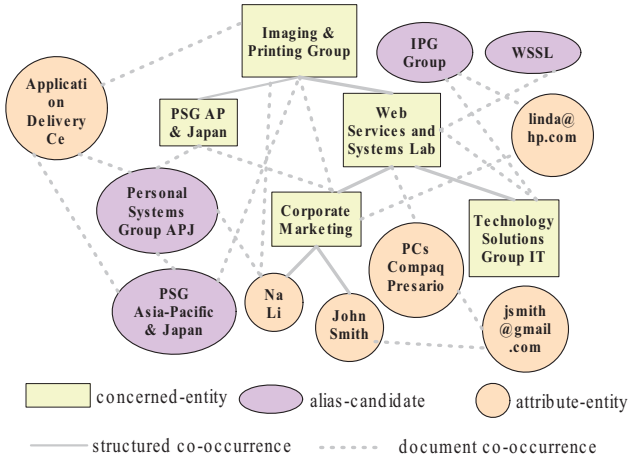


Fig. 3. A Sample Entity-Relation Graph

A. Entity-Relation Graph

As shown in Figure 3, the graph is modeled as $G = (V, E)$, where V denotes a set of nodes and E denotes a set of edges. V contains three types of nodes: *concerned entity*, *alias candidate*, and *attribute entity*. They are represented as the rectangle, ellipse and circle respectively. A given entity concerned for finding its aliases is concerned entity; excluding concerned entities and their alias candidates, other entities are defined as attribute entities. For example, to discover the aliases of a product, the products are concerned entities and attribute entities might include organization, organization alias candidate, employee, and etc. Likewise, when resolving other types of entities, products and their alias candidates can act as attribute entities. The concerned entities are given, alias candidates are extracted from document corpus, and the attribute entities are extracted from both the structured data and unstructured documents. For edges in E, there are two types: 1) *structured co-occurrence* that denotes the entity co-occurrence relationship in the structured data and is represented as the solid line; 2) *document co-occurrence* that denotes the entity co-occurrence relationship in the document corpus and is represented as the dashed line. Therefore, the parallel edges are allowed when both the co-occurrence relationships exist.

B. Edge Weighting

In the entity-relation graph G in Figure 3, the *structured co-occurrence* edge denotes a binary relation from the structured data, while the *document co-occurrence* edge denotes the possibility of two entities co-occurring in the same document. Since the entity relationships from the structured data are significantly reliable and accurate, we assign the highest relevance weight $ew(v, w) = 1.0$ to *structured co-occurrence* edge. Meanwhile we use the notion of $Jaccard$ to measure the weight of the *document co-occurrence* edge as follows.

$$ew(v,w) = \frac{|co\_occur(v,w)|}{|occur(v)| + |occur(w)| - |co\_occur(v,w)|} \quad (2)$$

Herein, $|occur(v)|$ represents the occurrence counts of node v in the whole document corpus, while $|co\_occur(v,w)|$ is the accumulated count of occurrences that nodes w and v simultaneously appear in a single document over the entire document corpus.

$|co\_occur(v,w)|$ requires that these two nodes appear anywhere in the document and represents the accumulative frequency of occurring together in each document over the whole document corpus. For example, node v appears in document d_1 three times and in d_2 once, and w appears twice in d_1 and once in d_2, then the co-occurring value is 2+1=3 using the following formula:

$$|co\_occur(v,w)| = \sum_{i=1,2} minimum(occur(d_i,v), occur(d_i,w)) \quad (3)$$

where $occur(d_i,v)$ denotes the number of times of node v appearing in document d_i.

C. Similarity in ERGM

The aim of *ERGM* as shown in Figure 3 is to compute the graph-based similarity between an entity and its alias candidates. Some previous work has proposed approaches to computing the node similarity in a graph, among which random walk [23] and Simrank [24] are commonly used. Compared with random walk, this method collects a full view of some particular region in the graph and is believed to be representative of the entire graph. The iterations in Simrank is essential for performance but difficult to determine. Moreover, the method of Simrank does not take the edge weight into account. In this section, we propose a new method to compute the graph-based similarity between two nodes. It aims to view all the paths between two entities and then choose the most important ones for consideration. The experimental results in Section VI indicate that our proposed approach is more effective to measure the node similarity between an entity and its candidates than these methods.

It is observed that many entities do not often appear with their alias candidates in the same document. Hence, sometimes we need to utilize other nodes as the structure context to link an entity node and its alias candidate nodes. In other words, the entities co-occurring with an entity in the same context can describe it in a useful way. Therefore, we decide to find out all the non-cycle paths between them using a bi-directed path search strategy, and then compute the graph-based similarity between any two nodes based on the following intuitions: 1) the more the number of edges in each path, the weaker the graph-based association is, 2) the higher the importance of each edge (i.e., edge weight), the stronger the graph-based association between these two nodes is. Herein, the graph-based association is estimated through the graph-based similarity score. According to the intuitions discussed above, we compute the graph-based similarity (i.e.,gs_p) between any two nodes on path p using the following formula.

$$gs_p(v,w) = \sum_{i=0}^{t} \left(\frac{1}{deg(v_i)} \frac{ew(v_i,v_{i+1})}{t^{i+1}} + \frac{1}{deg(w_i)} \frac{ew(w_i,w_{i+1})}{t^{i+1}}\right) \quad (4)$$

Overall, the core idea of this formula is summing all the edge weights on the path, where t is the number of edges on this path between nodes v and w. $v_0 = v$, $w_0 = w$, v_i and v_{i+1} are the connected nodes by the $(i+1)th$ edge originating from v_0, w_i and w_{i+1} are the connected nodes by the $(i+1)th$ edge originating from w_0. Meanwhile, three additional factors are taken into account 1) degree: this formula takes the degree of the start node in each edge (i.e., $deg(v_i)$ and $deg(w_i)$) into account. This consideration originates from the observations as follows: in any graph structure, there are always many nodes on the path between node v and w, and if a node is active with high degree, the relevance of nodes, v and w, passing through it will be decreased, which leads to a low contribution on the task of alias discovery. That is, the graph similarity any two nodes depends on the probability of reaching one node from another; 2) decay factor: as the relevance between nodes v and w is becoming weaker and weaker as transferring along each path between them, we use t^{i+1} as a decay factor, which denotes that the importance of a path will be less if it is constituted by more edges; 3) path search: the purpose of bi-directed path search is to simultaneously search paths originated from v and w as shown in Formula 4.

Following the heuristic that the longer a path is, the smaller the sum value on this path, and the edge closer to entity node should be assigned a higher value comparatively, we compute the graph-based similarity following the direction from the entity node to the candidate node. Then, we compute the graph-based similarity value based on all the paths between nodes v and w as follows.

$$f_g(v,w) = \sum_{p=1}^{s} \frac{1}{2s} \times gs_p(v,w) \quad (5)$$

Where, s is the total number of paths between nodes v and w. $gs_p(v,w)$ is the graph similarity value on the pth path between v and w. As the similarity score are computed twice for each path respectively from the two given end nodes in Formula 4, the value is cut in half.

D. Path Selection Strategies

It is clear that searching all non-cycle paths between two nodes on a large graph must be costly. Thus, we select high-quality paths using some constraints to make a balance between effectiveness and efficiency.

Path length We optimize this graph-based approach by setting a suitable value for the path length β. The experimental studies in Section VI-D2 introduce the setting of parameter β to avoid high cost searching.

Edge weight The edges with weights approaching zero are filtered out since they are treated as a simple chance of co-occurrence instead of relevance with each other.

Standard variance As some nodes may be connected due to a simple chance instead of real relevance, the equal-length paths are ought to be treated in different ways. Therefore, we use standard variance of the edge weights to evaluate the strength of a path. When the standard variance of edge weights for a path is larger than the average edge weight on this path, the path is ignored.

E. Entity Alias Discovery

Ultimately, we compute $F(x, y)$ through aggregating $f_i(x, y)$ (*i*=1,2), and $f_1(x, b)$, $f_2(x, y)$ denote the alias similarity and graph-based similarity respectively. The alias candidate selection method can keep most of the real aliases with high string similarity scores, while the graph-based similarity demonstrates the relevance in entity-relation graph between an entity and its alias candidates. Whereupon we take these two measures into account to compute the final similarity of any entity-candidate pair. More feature functions can be added for extending this solution and this paper uses a linear model as shown in Formula 6.

$$F(v, w) = \sum_{i=1}^{i \leq n} \theta_i \times f_i(v, w) \tag{6}$$

Where, i = 1,2, $f_1(v, w)$=$f_s(v, w)$, denotes the string similarity of the entity v and its candidate w using *ACS*, $f_2(v, w) = f_g(v, w)$, denotes the graph-based similarity between these two nodes. θ_i is used as a scaling factor and $\sum_{i=1}^{i \leq n} \theta_i = 1$ for n similarity functions. Generally, if $F(v, w)$ is above a predefined threshold λ, the candidate w can be determined as the final alias of v. All the parameters are trained through the 10-cross validation [27]. The overall process for alias discovery is presented in Algorithm 1.

Algorithm 1: Graph-based Alias Discovery

Input: $G(V, E)$, entity ent, $extractedEntity\ Set$, thresholds α, β, λ
Output: $aliasSet$ for ent
for *String candi:*$extractedEntity\ Set$ **do**
  if $(f_s(ent, candi) > \alpha)$ **then**
    candiSet.add(*candi)*;
  Identify ent as $entNode$ in G;
  for *(Node candiNode:candiSet)* **do**
    Search G for all the paths with length no longer than β between *entNode* and *candiNode* using the depth first search;
    Compute $F(entNode, candiNode)$ based on Formulae 5 and 6;
  end
end
$aliasSet \leftarrow candiNodes$ with similarity above λ.
Note: all the three parameters are determined by the 10-fold cross-validation.

VI. Experimental Study

In the following subsections, we first describe the used datasets and the data preprocessing procedure. Secondly, we compare our proposed *GRIAS* with other three methods. Finally, we perform scalability evaluation.

A. Data Description

The enterprise dataset consists of entity databases containing 2,000 organization names, 2,000 product names, and 300,000 employees, and an unstructured document corpus containing 2,800,000 documents (i.e., Web pages, email documents, and server logs). The government dataset from Wikileak consists of 250,000 documents and some collected structured data (i.e., 1,000 international government/non-government organizations, 25,963 officer names, and 220 countries) obtained from Wikipedia[4], and the US State official homepage[5]. The core task of the following experimental study is exploring the document corpus to detect all the aliases for the organization names and product names respectively.

B. Evaluation Measure

There may be numerous aliases for an entity, and people always prefer to obtain the most similar and commonly used ones. Moreover, for each entity, it is extremely difficult, if possible, to obtain a full set of its aliases ground truth in a large scale document corpus. Thus, an ideal alias discovery approach should generate the real aliases for each entity in the top positions of the generated alias set. Therefore, we also use the measure $precision@N$ as evaluation measure, which computes the ratio of the correctly identified aliases on the top N results in the identified alias set of each concerned entity.

C. Data Preprocessing

Given the document corpus, we first parse the HTML markups and convert the various documents to plain texts. After that, the following three tasks should be performed:

Indexing entities. For all the given and extracted entities (i.e., organization, person, location, products, and e-mail), we have to build a document-entity index, which records all their occurrence information in the each document (i.e., occurrence count, position, and the corresponding document number).

Extracting entity alias candidates. Entity candidates are initially extracted using the multi-word units [26] and entity recognition model[6]. Here, stop words are case-insensitive. The ultimate purpose of the alias candidate selection method is keeping a high recall for further entity-relation based graph .

Filtering. Before performing the proposed *ACS* approach for entity alias candidate selection, we employ two filters to remove the candidates distinctively dissimilar with the concerned entity. 1) A pre-filter removes out the extracted candidates, in which all the tokens are stop words. This procedure filters a large amount of non-entity candidates. 2)

[4]http://en.wikipedia.org/wiki/International_organization/
[5]http://www.state.gov/
[6]http://alias-i.com/lingpipe/

A post-filter inversely indexes all the entities based on bi-character and retains the candidates that have at least two common bi-characters with an entity. This restriction does not apply to candidates consisting of only capitalized letters. These filters can remove a large number of irrelevant candidates, which reduces the burden of the nested loop comparison of entity-candidate pairs. After preprocessing, the proposed graph-based approach is used for entity alias discovery.

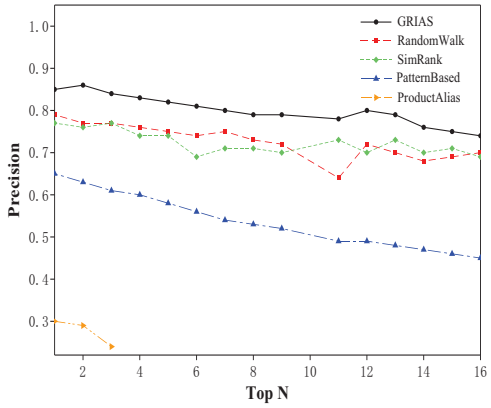
For parameter training, we randomly take 100 documents and 100 organizations from enterprise dataset as training data. For evaluation, we randomly select 100 organization names, 100 product names from the enterprise dataset, and 100 organizations from the government dataset as test cases. In the following three subsections, we present the experimental results respectively in terms of string match similarity based alias candidate selection methods, the graph-based similarity algorithm, and its scalability on both datasets. Note that all parameters in this study are obtained using the 10-fold cross validation.

D. Evaluation of Alias Discovery Approaches

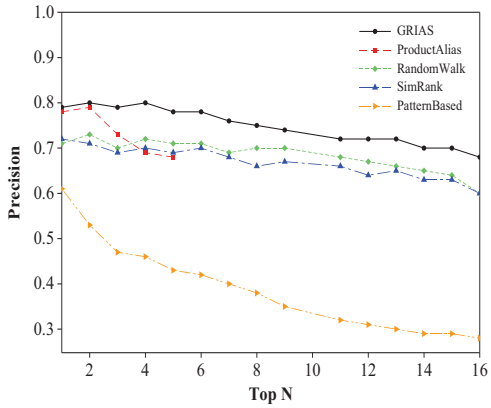
1) Comparison of Alias Discovery Approaches: Figure 4 presents the evaluation results of alias discovery on enterprise organization, product, and government organization respectively using five different methods, *RandomWalk* [23], *SimRank* [24], *ProductAlias* [1], *PatternBased* [28], and our proposed graph-based approach (*GRIAS*). Since *SimRank* [24] does not take the edge weight into account, we use the optimized weighted *SimRank* [29] for comparison. We use *RandomWalk* and *SimRank* to replace Formula 4 and 5 to compute the graph similarity and other settings are the same with *GRIAS*.

Based on the comparative results as shown in Figure 4, we have the following observations:

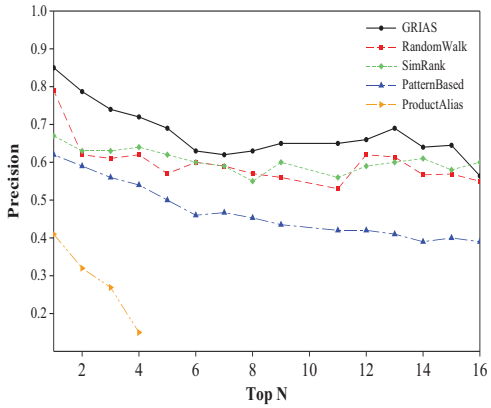
- The *PatternBased* method has to explore the popular patterns from large amount of documents, which does not work well in the low redundancy documents.
- The experimental results indicate that the proposed graph-based similarity calculation is more significantly effective than the weighted *SimRank* and *RandomWalk*. Compared with random walk, the proposed graph method could view all the paths between two entities and then choose the most important ones for consideration.
- As shown in the three sub-figures in Figure 4, their curves stop when N merely reaches four, three and five respectively. That's because *ProductAlias* assumes that the extracted alias is a token subset of the given entity, which leads to only a small number of aliases. Figure 4(a) and 4(c) shows that *ProductAlias* runs on organization alias discovery with a very poor performance. Comparatively, its $precision@N$ evaluation on the product alias extraction in Figure 4(b) is better. It is observed that many product names are often referred merely using the brand name plus a model number, such as "ThinkPad X61 Note-book" and its alias "ThinkPad X61". Our experimental results show that *ProductAlias* works well on this type of products specially. However, compared with *GRIAS*, *ProductAlias* ignores far more entity alias patterns.
- As the average number of tokens in an entity is often no more than ten, the number of real alias variants for each entity is not very large. Moreover, all the aliases commonly used by users are also limited. Therefore, the precision of all the five methods consistently decreases when N becomes bigger than a certain value.



(a) Evaluation on Organization from Enterprise



(b) Evaluation on Product from Enterprise



(c) Evaluation on Organization from Wikileak

Fig. 4. Precision@N Evaluation

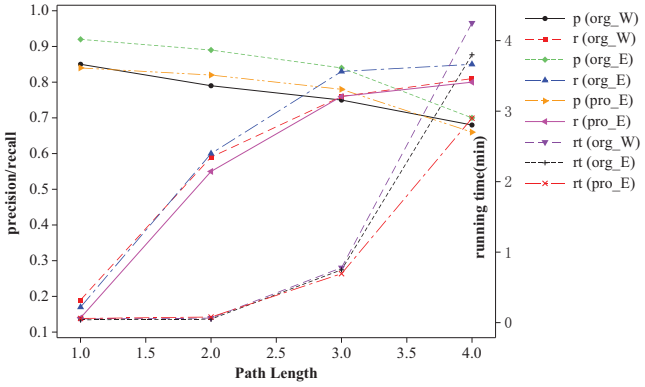


Fig. 5. Path Length Selection

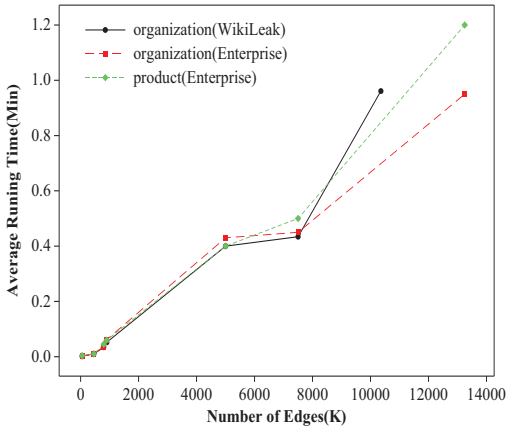


Fig. 6. Scalability Test

2) Path Length Selection in Entity-Relation Graph: The proposed graph-based approach explores the relevance between entities through finding all the paths between a concerned entity and its candidates. According to random walk theory, a long path between two nodes (i.e., v and w) will lower the probability of reaching w from v within limited time. Formula 4 also indicates that a longer path may result in a smaller graph-based similarity value. Moreover, searching a long path is time consuming. Therefore, it is reasonable in the literatures on graph mining to find the limited-length paths between two nodes for the efficiency with no/little loss of accuracy. So the path selection method mentioned in Section V-E is employed, which chooses some fixed length paths through the performance evaluation on training dataset. The experimental results on both organization and product alias discovery in Figure 5 show that a path of length three can achieve a good balance between effectiveness and efficiency. In this figure, "p" and "r" denotes "precision" and "recall", while "org_W", "org_E", and "pro_E" denote organization_WikiLeak, organization_Enterprise, and product_Enterprise respectively. In addition, "rt" denotes "running time".

While some true alias should have the paths with various lengths from their concerned entities, very few of them only have long paths. Hence, taking the short path approach cover most of true aliases for each given entity. The overall performance under path length one or two is significantly lower than that under paths with longer length. However, when the path length grows to four, the efficiency goes beyond the reasonable time tolerance on a large graph. Most importantly, the overall performance of the graph-based approach with path length no longer than three and four make little difference. Therefore, three is chosen as the path length threshold in this study.

E. Scalability Test

Using a fixed path length of three, we perform the proposed alias discovery algorithm on different scale of graphs, and compute the average time of finding all paths for each entity-candidate pair.

With the growing volume of document corpus, the number of nodes grows steadily while the number of edges increases sharply. The reason is that the entity types and the quantity for each type slightly change when an entity-oriented data corpus reaches a certain scale.

F. Discussion

The method of alias candidate selection takes more alias types into account, which keeps a high coverage for the alias candidates selection and avoid the positive alias loss in the final results. While the graph-based model, *GRIAS* effectively combine the alias candidates into the graph model and refines them for each concerned entity. *GRIAS* explores the existing structured data and can be extended through adding more similarity functions based on the additional resources. The method of *ProductAlias* assumes that a document which contains an entity alias usually refers the remaining tokens of the entity within the alias' vicinity. Since some documents in the official publications or the enterprise applications are mostly written technically for expert readers (not for public reading and discussion) they are less descriptive and narrative. Thus, the co-occurrence frequency is much lower in these documents.

The aforementioned path selection strategies show that there are only less than a half of the entities are referred in the large-scale dataset and most of them are rarely co-occurring with its true aliases. Therefore, the lower coverage on the shorter paths proves the fact that there are few entity-candidate pairs co-occur in an enterprise document.

VII. Conclusions and Future Work

Entity alias discovery is a critical task in many real world applications. In this paper, we propose a graph-based approach, called *GRIAS*, to perform entity alias discovery in free documents. To enhance the effectiveness of alias discovery and improve its efficiency, we introduce the Alias Candidate Selection (*ACS*) method to select high-quality candidates. We then utilize entity information from both entity databases and document corpus to form the Entity Relation Graph Model. Assuming that the candidate has a high similarity with the

true counterpart of a given entity, we compute the graph-based similarity for each entity-candidate pair and combine it with the string match similarity to determine the final aliases. The key contribution of *GRIAS* is an efficient graph-based model combined with the method of alias candidate selection. Taking organizations from Wikileak and organization as well as products from enterprise as real dataset, our experiments have demonstrated the effectiveness and efficiency of *GRIAS*.

Entity alias discovery could be formulated as a machine learning problem in future. Namely, given a set of concerned entities and a set of alias candidates, we might train a classifier to learn how the set of aliases for a concerned entity could be ranked. The proposed features used in alias discovery could be used in classifier training.

ACKNOWLEDGMENT

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

REFERENCES

[1] S. Chaudhuri, V. Ganti, and D. Xin, "Exploiting web search to generate synonyms for entities," in *Proceedings of the International Conference on World Wide Web (WWW)*, 2009, pp. 151–160.

[2] O. Benjelloun, H. Garcia-Molina, D. Menestrina, Q. Su, S. E. Whang, and J. Widom, "Swoosh: a generic approach to entity resolution," *VLDB J.*, vol. 18, no. 1, pp. 255–276, 2009.

[3] D. G. Brizan and A. U. Tansel, "A survey of entity resolution and record linkage methodologies," *Communications of IIMA*, vol. 6, no. 3, 2006.

[4] V. R. Jakkula, A. S. Crandall, and D. J. Cook, "Knowledge discovery in entity based smart environment resident data using temporal relation based data mining," in *ICDM Workshops*, 2007, pp. 625–630.

[5] D. Grosvenor and A. Seaborne, "Using hybrid search and query for e-discovery identification," HP Labs, Technical Reports HPL-2009-155, 2009.

[6] D. Bollegala, T. Honma, Y. Matsuo, and M. Ishizuka, "Automatically extracting personal name aliases from the web," *Proceedings of the International Conference on Natural Language Processing(GoTAL)*, vol. 5221, pp. 77–88, 2008.

[7] R. Hölzer, M. Bradley, and S. Latanya, "Email alias detection using social network analysis," in *Proceedings of the KDD Workshop on Link Analysis and Link Detection (LinkKDD)*, 2005, pp. 52–57.

[8] E. Sapena, L. Padró, and J. Turmo, "Alias assignment in information extraction," *Procesamiento del Lenguaje Natural*, vol. 69, no. 39, pp. 105–112, 2007.

[9] S. Chaudhuri, V. Ganti, and D. Xin, "Mining document collections to facilitate accurate approximate entity matching," *Proceedings Of the International Conference on Very Large Data Bases (VLDB) Endow*, vol. 2, no. 1, pp. 395–406, 2009.

[10] P. Singla and P. Domingos, "Entity resolution with markov logic," in *Proceedings of the International Conference of Data Mining (ICDM)*, 2006, pp. 572–582.

[11] L. Niu, J. Wu, and Y. Shi, "Entity resolution with attribute and connection graph," *2012 IEEE 12th International Conference on Data Mining Workshops*, vol. 0, pp. 267–271, 2011.

[12] C. Wang, K. Chakrabart, T. Cheng, and S. Chaudhuri, "Targeted disambiguation of ad-hoc, homogeneous sets of named entities," in *Proceedings of the International Conference on World Wide Web (WWW)*, 2012, pp. 719–728.

[13] W. E. Winkler, "String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage," in *Proceedings of the Section on Survey Research Methods, American Statistical Association*, 1990, pp. 354–359.

[14] R. Mihalcea and A. Csomai, "Wikify!: linking documents to encyclopedic knowledge," in *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, 2007, pp. 233–242.

[15] D. Milne and I. H. Witten, "Learning to link with wikipedia," in *Proceedings of the 17th ACM conference on Information and knowledge management*, ser. CIKM '08, 2008, pp. 509–518.

[16] W. W. Cohen, P. D. Ravikumar, and S. E. Fienberg, "A comparison of string distance metrics for name-matching tasks," in *Proceedings of the IJCAI Workshop on IIWeb*, 2003, pp. 73–78.

[17] D. Bollegala, Y. Matsuo, and M. Ishizuka, "A co-occurrence graph-based approach for personal name alias extraction from anchor texts," in *Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP)*, 2008, pp. 865–870.

[18] K. Chakrabarti, S. Chaudhuri, T. Cheng, and D. Xin, "A framework for robust discovery of entity synonyms," in *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD)*, 2012, pp. 1384–1392.

[19] S. Feng, Y. Xiong, C. Yao, L. Zhang, and W. Liu, "Acronym extraction and disambiguation in large-scale organizational web pages," in *Proceeding of the Conference on Information and Knowledge Management (CIKM)*, 2009, pp. 1693–1696.

[20] Z. Manuel, "A (acronyms)," Ph.D. thesis, School of Computing Science, Simon Fraser University, 2004.

[21] L. Getoor and C. P. Diehl, "Link mining: a survey," *ACM SIGKDD Explorations Newsletter*, vol. 7, pp. 3–12, 2005.

[22] D. V. Kalashnikov and S. Mehrotra, "A probabilistic model for entity disambiguation using relationships," in *Proceedings of the SIAM International Conference on Data Mining (SDM)*, 2005, pp. 21–23.

[23] F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens, "Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, pp. 355–369, 2007.

[24] J. Glen and W. Jennifer, "Simrank: a measure of structural-context similarity," in *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD)*, 2002, pp. 538–543.

[25] P. Li, H. Liu, J. X. Yu, J. He, and X. Du, "Fast single-pair simrank computation," in *Proceedings of the SIAM International Conference on Data Mining (SDM)*, 2010, pp. 571–582.

[26] J. F. da Silva and G. P. Lope, "A local maxima method and a fair dispersion normalization for extracting multi-word units from corpora," in *Proceedings of Sixth MOL*, 1999, pp. 369–381.

[27] P. A. Devijver and J. Kittler, "Pattern recognition: A statistical approach," *Prentice-Hall, London*, 1982.

[28] D. Bollegala, T. Honma, Y. Matsuo, and M. Ishizuka, "Identification of personal name aliases on the web," in *Proceedings of the International Conference on World Wide Web (WWW)*, 2008, p. 1107C1108.

[29] I. Antonellis, H. Molina, and C. chao Chang, "Simrank++: query rewriting through link analysis of the click graph," in *PVDLB*, 2008, pp. 408–421.

APPENDIX

Given two case-insensitive strings $str1$ and $str2$, let string $com1$ consist of all the characters in $str1$ that also appear in $str2$, string $com2$ consist of all the characters in $str2$ that also appear in $str1$, and $L(com1)$ denotes the length of $com1$ (note that repetitiveness is allowed in $str1$ and $str2$). We first measure whether there is any character order transposition from $com1$ to $com2$, if true, $simi$ is assigned as zero. Otherwise, we need to compute a similarity between $str1$ and $str2$ using the following formula.

$$simi(str1, str2) = \frac{1}{2.0} \times (\frac{L(com1)}{L(str1)} + \frac{L(com2)}{L(str2)})$$

Moreover, [13] claims that the strings with similar prefixes ought to carry similar meanings. Similarly, we give more favorable ratings to the compared strings that match from the beginning for a set prefix length. Finally, as presented in the following formula, $simi'$ is used to compute the ultimate string similarity between the given string pair, where $prefixL$ is the length of common prefix at the start of the two strings, and 0.1 is used as a scaling factor for how much the score is adjusted upwards for having common prefix's.

$$simi'(str1, str2) = simi + prefixL * 0.1 * (1 - simi)$$