ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

Volume 3, Special Issue 3, March 2014

2014 International Conference on Innovations in Engineering and Technology (ICIET'14)

On 21st&22ndMarch, Organized by

K.L.N. College of Engineering, Madurai, Tamil Nadu, India

Efficient Ranked Keyword Search Using AME

K.Padmapriya, A.Ambika , A.Gayathiri

M.E, Department Of CSE(Final Year), Bharath Niketan Engineering College, Aundipatti, Tamil Nadu, India.

Abstract— Entity Recognition is process of identifying predefined entities such as person names, products, or locations in a given document. This is done by finding all possible substrings from a document that match any reference in the given entity dictionary. Approximate Membership Extraction (AME) method was used for finding all substrings in a given document that can approximately match any clean references but it generates many redundant matched substrings because of approximation (rough calculation), thus rendering AME is not suitable for real-world tasks based on entity extraction. We propose a web-based join framework which combines a web search along with the approximate membership localization. Our process first provides a top n number of documents fetched from the web using a general search using the given query and then approximate membership localization(AML) is applied on these documents using the clear reference table and extracts the entities form the document to form the intermediate reference table using Edit distance Vector, Score Correlation.

Keywords— Data Mining, Approximate Membership Localization Algorithm, P-Prune, Entity Recognition.

I. INTRODUCTION

The task of Entity Recognition is to identify predefined entities such as person names, products, or locations in this document. With a potentially large dictionary, this entity recognition problem transforms into a Dictionary-based Membership Checking problem, which aims at finding all possible substrings from a document that match any reference in the given dictionary. With the growing amount of documents and the deterioration of documents' quality on the web, the membership checking problem is not trivial given the large size of the dictionary and the noisy nature of documents, where the mention of the references can be approximate and there may be mentions of non-relevant references.

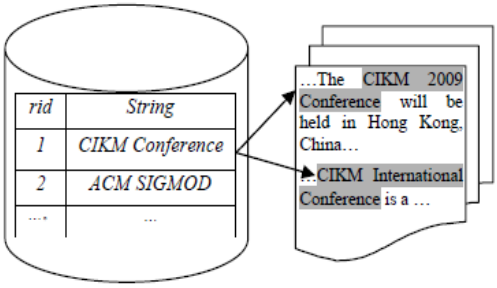

Figure: 1 using AME

The approximation is usually constrained by a similarity function (such as edit distance, jaccard, cosine similarity, etc.) and a threshold within [0, 1], such that slight mismatches are allowed between the substring and its corresponding dictionary reference. For instance, given a list of conference names like "CIKM Conference," "ACM SIGMOD Conference," Conference" as shown on the left part of Fig. 1, the task is to find matches from the text on the right, such as "CIKM 2009 Conference" and "CIKM International Conference," although they do not match the string "CIKM Conference" in the dictionary exactly. The dictionary-based approximate membership checking process is now expressed by the Approximate Membership Extraction (AME), finding all substrings in a given document that can approximately match any clean references.

In this project, we propose a new type of membership checking problem: Approximate Membership Localization (AML). AML only aims at locating true mentions of clean references. AML targets at locating non-overlapped substrings in a given document that can approximately match any clean reference. We also increase the performance of

the AML process making it as a Web-Based Join Framework.

We propose the project as a web-based join framework which uses a list of elements T with an attribute T:X and a clean reference list R with an attribute R:A, the problem is to create from the web an intermediary table RT containing valued correlations between two attributes T.X and R.A in order to perform a join between T and R. Given that the information available on the web can be dirty and noisy, RT shall contain the likelihood associated with its entries. Based on the hypothesis that there exist web documents containing elements of T.X that also contain the elements of R.A, we use the elements of T.X as a query for a search engine to retrieve the ranked list of documents.

First, given that the mention attribute in web pages might not be exact, we need to locate clean references approximately mentioned in given documents, for this AML method is presented second, besides given attribute, some other attributes might also occur in the web pages. To link given attribute with them, we use score correlation. We use Edit Distance Vector which is the simple function for finding the similarity when compare with cosine similarity. Edit Distance Vector reduce the extra complexity in computation because it is secondary step of extraction there is no need of high evaluation.

II. ALGORITHMS USED

Data mining can be used for extracting previously unknown patterns like groups of data records as well as to extract word and calculate between two data items. The algorithm used to find these dependencies between the data items is AML algorithm, while the algorithm used to extract groups of records is K-means clustering.

Approximate Membership Localization (AML) Algorithm :

Algorithms for the AML problem based on the two assumptions below:

Assumption 1: any approximate mention m that
matched with a reference consists of consecutive words in a document, i.e., each m is a substring.

Assumption 2: only substrings whose length is up to a length threshold L are of interest, so we may as well require that $|m| \leq L$.

For the purpose of similarity functions, we regard strings as sets of words. For any word w, we use wt(w) to denote its Inverse Document Frequency (IDF) weight. Given a string S= w1, w2. . . wk, the weight of s is the sum of weights of all its constituents and can be denoted as $wt(s)=\sum 1 \leq i \leq k(wt(wi))$. The weighted Jaccard similarity of two strings s1 and s2 can be calculated as: $WJS(s1,s2)=wt(s1 \cap s2)/wt(s1 Us2)$. It is worth mentioning that the algorithms presented in this paper can be applied to any similarity functions such as (weighted) jaccard similarity, edit distance [9], [22], as long as it satisfies: $sim(s1,s2) \leq wt(s1 \cap s2)/wt(s1)$ (and also $sim(s1,s2) \leq wt(s1 \cap s2)/wt(s2)$.

In this section, we introduce two algorithms to solve the AML problem. The first algorithm is based on AME. Since the AML results are a subset of the AME results, we can do AME first, then remove the redundant pairs from the AME results second. However, this algorithm uses a lot of extra time for generating these redundant pairs and then removing them. As an alternative, a more efficient approach is to prune the potential redundant substrings before generating them, so that less time is required for generating and verifying the remaining candidate pairs.

The project as a web-based join framework which uses a list of elements T with an attribute T:X and a clean reference list R with an attribute R:A, the problem is to create from the web an intermediary table RT containing valued correlations between two attributes T:X and R:A in order to perform a join between T and R. Given that the information available on the web can be dirty and noisy, RT shall contain the likelihood associated with its entries. Based on the hypothesis that there exist web documents containing elements of T:X that also contain the elements of R:A, we use the elements of T:X as a query for a search engine to retrieve the ranked list of documents Docs

First, given that the mention attribute in web pages might not be exact, we need to locate clean references approximately mentioned in given documents, for this AML method is presented second, besides given attribute, some other attributes might also occur in the web pages. To link given attribute with them, we use score correlation. We use Edit Distance Vector which is the simple function for finding the similarity when compare with cosine similarity. Edit Distance Vector reduce the extra complexity in computation because it is secondary step of extraction there is no need of high evaluation.

Scoring Correlations:

Scoring correlations for each document depends upon three relevant parameters frequency, distance and document importance.

Frequency freq: the number of times each reference
is mentioned in each document of Docs.

Distance dist: the distance between the mention of each clean reference and the position of T:x.

Document importance imp: documents retrieved on the web are of different importance w.r.t. their relevance to the query, i.e., their ranks in a web search engine results.

The following equation is used to find the score correlation

$$P(r,T.x) = \frac{\sum_{d \in Docs} imp(d) \cdot score(r,d)}{\sum_{d \in Docs} imp(d)|},$$

where imp(d) is the importance of document d and score(r; d) is the local score of r in d.

To calculate imp(d), the N documents of Doc should be partitioned into B (1 < B < N) ranges according to their rank returned by the web search engine, so that the importance imp(d) follows the normalized discount
cumulative gain (NDCG) function, which is popularly
used for assigning degrees of importance to web documents in a ranked list.

$$imp(d) = \frac{log(2)}{log\left(1 + \frac{\lceil rank(d) \rceil}{B}\right)}.$$

The document importance main depends upon the page ranking assigned to document during the web search which is not focused in the existing system.
We use an inverse document based page ranking to assign rank to the documents which enhance the process of scoring correlation.

$$score(r,d) = w_a \cdot \frac{freq}{N} + (1 - w_a) \cdot \sum_{1 \le i \le freq} \frac{|d| - dist_i}{freq \cdot |d|}$$

P- Pruning:

Guarantee that all best match substrings can be found within. For very large reference lists, the number of these window domains, especially overlapped ones, can become an issue. In this section, we introduce how to prune domains that impossible to contain a best match substring, and how to minimize the size of the remaining domains.

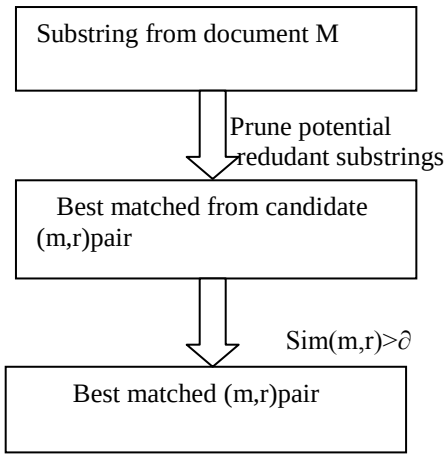

Figure: 2 pruning

Given an input document M and an entity reference list R, the task of approximate membership localization is to find all match substrings m in M for each reference r in R, such that:

1. sim(m,r), where sim() is a given similarity function, and tow is a given threshold between [0, 1].
2. there is no substring m' that overlaps with m in any position of M which satisfies sim(m', r')> sim(m, r),

where r' is also a reference from R.

M- to denote the input document, R to denote the entity reference list, and
Tow-to denote the similarity threshold to the string similarity function sim(a; b), where a and b are two strings

We now present the detailed algorithm of P-Prune
(Potential redundancy prune). Given a reference list R, an

Inverted index is built over the words of all references in R and the strong words of each reference are also selected. For a coming document M, we generate window domains from it. Each domain is represented by (posb; pose; r), where posb and pose refer to the starting and ending position of the domain in M, r is the unique reference it corresponds to. All domains are sorted according to their starting positions and stored in a set Dsort. We also use set Dact to store active domains being processed. Starting from the beginning position of M, we do:

Step 1: We step to the next new word position poscur,
with the word wcur in that position.
Step 2: We remove the first domain from Dsort and put it into Dact, until it can't satisfy Dsort[1].posb= poscur.

Step 3: For each domain D =(posb, pose, r) in Dact, if
wcur presents in r, it becomes a word in D's segment, otherwise it becomes a word in D's interval. The Interval Pruning strategy is applied to all generated intervals. If poscur = D.pose, the Boundary Pruning and Weight Pruning are applied as well.
Step 4: If poscur is not included in any range of domains or it already reaches the ending position of M, then the function Locate Best Match is called to generate the best match substrings from all domains in Dact.
Step 5: We do step 1 to step 4 iteratively, until poscur
reaches the ending position of M. We output all best match substrings we learn.

III. WORKING AND RESULTS

Over all process is the extracting document without overlap.
Data set has been collected from:
1.http://www.cnts.ua.ac.be/conll2003/ner
2.http://tjongkimsang@ua.ac.be
3.http://fien.demeulder@ua.ac.be
4.http://projects.students3k.com/ieee-base-paper-and-document-download-on-networking-domain.html
5.http://ieeeprojectsworld.blogspot.in/2013/09 /to-get-abstracts-pdf-base-paper-review.html

From our data set we will extract the desire document with efficient manner which support of score correlation and AML algorithm. The stepwise process is below theory,

First, To extract the documents from the web using a content based search processed using the given document attributes as query input. Content based document retrieval is process of retrieving a document by search for the given keyword within each document in the document set. We select top K resultant documents from the search results for our next level of processing. Each document contains certain rank value assigned to it based on the content matching the given query attributes.

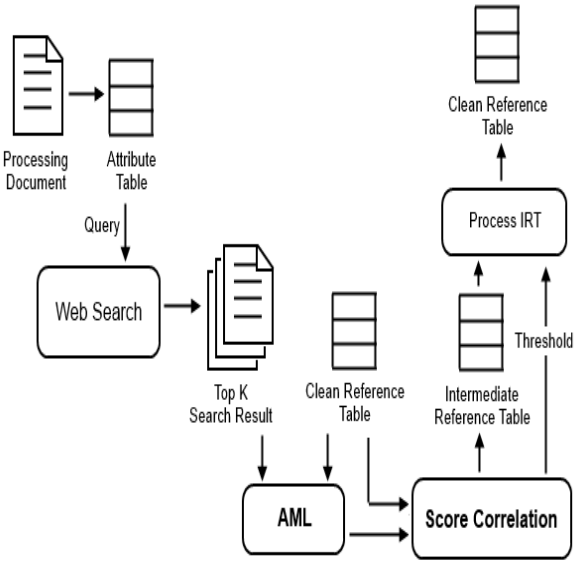

Figure 3: Framework for our method

Secondly, to remove the AML problem from our process this is to extract the clean reference approximately. It uses an optimized P-Prune algorithm, which can prune potential redundant substrings from documents before generating clean reference approximately. This algorithm shows a much higher efficiency than the AME-based algorithm. After the pruning process we use a similarity function for extracting the approximate clean reference from the documents.

Thirdly, scoring correlations for each document depends upon three relevant parameters frequency, distance and document importance. The following equation is used to find the score correlation. We use an inverse document based page ranking to assign rank to the documents which enhance the process of scoring correlation.

Finally, to construct a reference table by using the results produced by the approximate membership location and score correlation modules. In this step we convert the intermediate reference table to a clean reference table considering its score correlation values. This assigned score value is used as a threshold value to filter out the approximate reference from the intermediate reference list. After that CRT will be display to user what they searching.

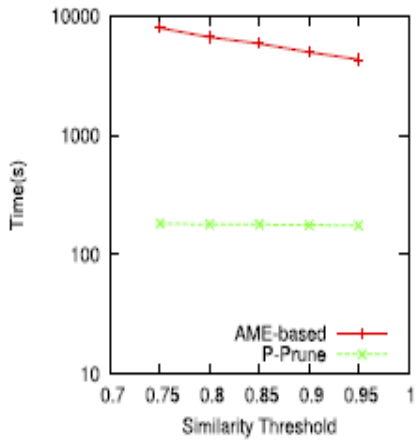

Figure 4 : AML Efficiency

IV. CONCLUSION

I formalize the AML problem and to solve it with an efficient P-Prune algorithm. The improvement of AML over AME, we apply both approaches within our proposed web-based join framework, which is a typical real-world application that greatly relies on the results of membership checking. AML-targeted solutions are more appropriate than the AME-targeted solutions for this type of real-world applications, since the matched pair results of the AML are much closer to the true matched pairs than AME result

REFERENCES

[1] S.Agrawal, K. Chakrabarti, S. Chaudhuri, V. Ganti, A. Konig, andD. Xin, "Exploiting Web Search Engines to Search Structured Databases," *Proc. 18th WWW Int'l Conf. World Wide Web, pp. 501-510, 2009*

[2] A. Arasu, V. Ganti, and R. Kaushik, "EFFICIENT EXACT SET-SIMILARITY JOINS," *Proc. 32nd VLDB Int'l Conf. Very Large Data Bases, pp. 918-929, 2006.*

[3] R. Bayardo, Y. Ma, and R. Srikant, "SCALING UP ALL PAIRS SIMILARITY SEARCH*," Proc. 16th WWW Int'l Conf. World Wide Web, pp. 131-140, 2007.*

[4] B. Bocek, E. Hunt, and B. Stiller, "FAST SIMILARITY SEARCH IN LARGE DICTIONARIES," *Technical Report ifi-2007.02, Dept. of Informatics Univ. of Zurich, 2007.*

[5] H. Chan, T. Lam, W. Sung, S. Tam, and S. Wong, "A LINEAR SIZE INDEX FOR APPROXIMATE PATTERN MATCHING," *Proc. 17th Ann. Symp. Combinatorial Pattern Matching, pp. 49-59, 2006.*

[6] K. Jarvelin and J. Kekalainen, "CUMULATED GAIN-BASED EVALUATION OF IR TECHNIQUES," ACM Trans. Information Systems, vol. 20, no. 4, pp. 422-446, 2002

[7] N. Koudas, S. Sarawagi, and D. Srivastava, "Record linkage: Similarity Measures and Algorithms," *Proc. ACM SIGMOD Int'l Conf. Management of Data, pp. 802-803, 2006.*

[**8**] L. Tanabe and W. Wilbur, "GENERATION OF A LARGE GENE/PROTEIN
LEXICON BY MORPHOLOGICAL PATTERN ANALYSIS," *J. Bioinformatics and
Computational Biology, vol. 1, no. 4, pp. 611-626, 2004.*