Query Reformulation using String Transformation

Sheetal More[1]
[1]Department of Computer Engineering, SVIT COE Chincholi, Nashik, Savitribai Phule Pune University,India

ABSTRACT: *Generally there is a need to change the input string in data mining, natural language processing, information retrieval and bioinformatics. Many times the user is not from the technical background so he may enter the incorrect input string. Depending on the input string the system will generate some topmost k output strings. Most of the time for better results of the search, there is need to transform the input string. Sometimes the user also enters the shortforms that is abbreviations for the search in that case there is need of transforming these abbreviations into their original forms or meanings. Many times while entering the input string user does the spelling mistakes depending upon the pronunciations. So in this case there is need to correct these spelling mistakes and then search for the results. Thus there is need of converting abbreviations into their original form, correction of spelling errors and also replacing the word with its synonym if needed for better search results. Thus these conversions of strings can be stated as string transformation. String transformation can be conducted in two different ways depending upon the use of dictionary that is whether the dictionary is used or not. In this paper log linear model is expressed in terms of an input and output strings. The paper shows an approach for string transformation which is both accurate and efficient. The proposed method is applicable for string transformation and reformation of query.*

Keywords- *Log Linear Model, Spelling Error Correction, String Transformation, Query Reformulation.*

1. INTRODUCTION

For obtaining the perfect results in the search there is need to reconstruct the query using string transformation and then execute the query. Different users enter different input strings which sometimes include words or characters. Thus string transformation is a necessary problem in many applications. There are some users which are from non-technical background so they may enter the input string with spelling mistakes. Thus this forces us to transform the input string in such a form which will give correct and better results. The fields where such transformation of string is required is natural language processing, data mining, information retrieval etc. String transformation can be used in query reformation and query suggestion in search. In data mining string transformation can be considered as mining synonyms and database record matching. The transformation should be conducted not only accurately but also efficiently.

String transformation exactly means applying set of operators for converting input strings into *k* most likely output strings. The strings can be string of words, characters or any type of tokens. And the operators are the rules that define the replacement of a substring with another substring. The main goal of this paper is enhancing both accuracy and efficiency. In natural language processing the operators can be pronunciation generation, spelling error corrections, word transliteration and word stemming.

String transformation can be conducted with two different settings, whether or not a dictionary is used. If the output string exists in the given dictionary, then the dictionary should be very large. In the case where the string comprises of characters there is need of dictionary. Correcting spelling errors in queries usually consist of two steps: candidate generation and candidate selection. To find most likely corrections of a misspelled word from dictionary is considered as candidate generation. In this case the input can be string of characters and the operators can be insertion, deletion or substitution of characters.

As discussed above, there is need to replace the abbreviations with their original meaning for better and correct search result. Thus if the system in not case sensitive and user enters “HCL” or “hcl” then depending upon the next string the HCL should be replaced by either “hydrogen chloride” or “Hindustan Computers Limited” for better matching of query and document. In the previous work developed, efficiency is not considered as an important factor. This paper aims to learn a model for string transformation which can achieve both high accuracy and efficiency. There are three fundamental problems with string transformation: 1) how to define a model which can achieve both high accuracy and efficiency. 2) how to accurately and efficiently train the model. 3) how to generate the top *k* output strings from given input string.

This paper proposes a probabilistic approach to the task. This method used in this approach is novel and unique in the following way. It uses a log-linear model for string transformation. Also uses an effective and accurate algorithm for model learning and an effective algorithm for string generation.

2. RELATED WORK

String transformation has many applications in data mining, natural language processing, information retrieval, and bioinformatics. String transformation has been studied in different specific tasks such as database record matching, spelling error correction, query reformulation and synonym mining. The major difference between our work and the existing work is that we focus on enhancement of both accuracy and efficiency of string transformation.

String transformation can be defined as generating one string from another string such as from "office" we can generate office or officer both are correct. In [2] some transformation rules are applied on the input string. In this work the main aim was applying different and maximum rules for string transformation. In [3] the system proposed used the method of applying transformation rules such as stemming, prefix, suffix and acronym which are predefined. In [4] a log-linear model for string transformation is designed mainly for getting accurate results. In this system finite state transducers are applied to generate the candidates.

In approximate string search, the model used is fixed and the objective is to efficiently find all the strings in the dictionary. Other methods are used to find all the candidates within a fixed range and employ n-gram based algorithm or trie-based algorithm. The n-grams algorithm is also applied for finding the top *k* candidates. Spelling error correction is another way of string transformation. Candidate generation is generally concerned with a single word. The method of edit distance is used which performs operations on characters such as deletion, insertion and substitution. Some method generates candidates within fixed range of edit distance or different ranges for string with different lengths. By applying the different transformation rules we can reformulate the whole query which is considered as query reformulation. One of the existing technique first identifies phrase- based transformation rules from query pairs and then segments the input query into phrases and generates a number of candidates using the substitution rules. Another way of query reformulation tried to replace the words in the input query by using patterns.

3. MODEL FOR STRING TRANSFORMATION

The aim of the proposed system is to achieve both high accuracy and efficiency which will be powerful for the large scale databases. Figure 1 shows the overview of the system. In this proposed method the training data consist of large number of input strings and output strings. Further a set of operators are applied for string transformation. Then a model which can assign scores to candidates of output strings is obtained from the training data and the operators. The candidates having highest probabilistic scores are considered as best candidates. The model proposed basically consists of two processes learning and generation. In the learning processes, from the training data rules are found out. Then by using the learning system, the model of string transformation is constructed which consist of rules and weights. In the next generation process, by referring the model which is stored in the rule index the system produces the top-*k* candidates of output string. In this method, the model used is log-linear model which represents the rules and weights, then estimation of the training data is learned and then at last efficiently top-*k* pruning results are generated.

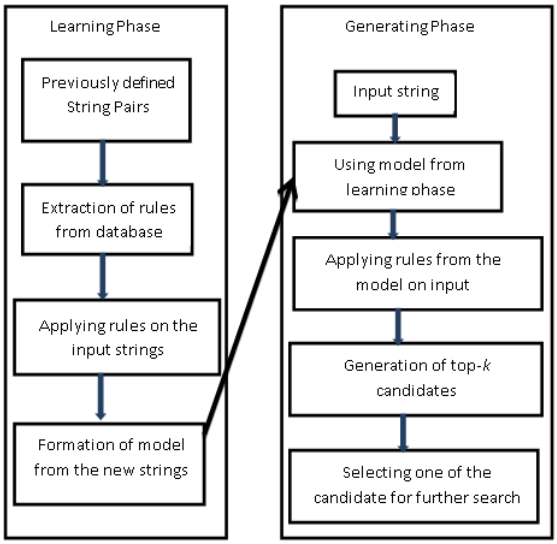

Fig.1 Overview of the System

3.1 Model

The model consists of rules and weights. Here different operations of replacing substrings are performed on the input string and a set of possible strings are obtained. Depending upon the transformation the character level or word level rules are applied for replacement. All the possible rules are derived from the training data based on string alignment.

When a set of rules are utilized to transform the input string to the output target string , then the rule set is said to form a transformation. for that

particular string pair. For a given string pair, there might be many different possible transformations. As shown in the figure 2 the string "types" can be replaced by "ways" and 'f' is inserted in the string "diferent". Replacing 'types' by 'ways' is known as word level rules and other is known as character level rule. It is assumed that without losing the generality, maximum rules are applicable to a string pair is predefined. Thus the number of possible transformation for a string pair is also limited. This is because the difference between an input string and output string should be as small as possible. Also the number of spelling errors in a word should be less.

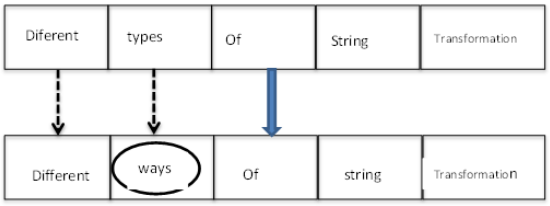

Fig. 2 An example of String Transformation

3.2 Training of Model

The training data consist of many transformation rules, so it is not given in the training data that which rule is to be applied on the input string to get the correct output string. Thus it is not known which transformation is true and it is difficult to derive it automatically from data. In this paper , on the basis of conditional probability of output string for given input string is defined. The conditional probability is marginalized over all possible transformations. The Quasi Newton method is used in the optimization. Also the bounded L-BFGS algorithm is applied as the optimization technique which works well when the number of parameters is large.

3.3 String Generation

In string generation, the main aim is to generate the most nearest *k* output strings from the given input string which can be transformed and have the largest probabilities assigned by the learned model. In this approach the maximum conditional probabilities are taken instead of taking max of the conditional probabilities which can make the generation process very efficient.

4. STRING GENERATION ALGORITHM

In this paper mainly three algorithms are used such as Aho-Corasic tree (AC tree), top *k* pruning and dictionary matching algorithm for rule indexing , obtaining top *k* results and finding words from dictionary respectively.

4.1 AC-TREE

This algorithm is used to store all the rules and their weights, which makes the references of rule very efficient. The Aho-Corasick string matching algorithm can be executed on AC tree which is a trie –with "failure links". The Aho-Corasick algorithm is well known dictionary matching algorithm which can quickly locate the elements of a finite set of strings within an input string.

4.2 Top *k* Pruning

This algorithm is used for obtaining top-*k* candidates that is output strings from the input string. In this algorithm the path is denoted by its position, string and score. A priority queue is used to store all the possible paths. A set is used for storing the best *k* candidates and their scores. The algorithm picks up one path from queue and processes it and then takes the next path. It uses top-*k* pruning strategy for improving efficiency. The path with minimum score will be discarded and not use further. If two paths have same position and string then only path with highest score is kept for further processing.

4.3 Dictionary Matching Algorithm

Sometimes there is need to utilize the dictionary for string transformation such as spelling error correction, database record matching and synonym mining in which the output strings must exist. Here the dictionary is indexed as trie, such that each string in the dictionary corresponds to the path from root node to leaf node.

5. CONCLUSION

In this paper a new statistical approach is proposed for string transformation. This method is novel and unique in its model, learning model and string generation algorithm. Using the proposed system spelling error checking and query reformulation becomes easy. The system improves not only accuracy but also efficiency when the database is large.

REFERENCES

[1]. Ziqi Wang, Gu Xu, Hang Li, and Ming Zhang" A Probabilistic Approach to String Transformation" IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING VOL:PP NO:99 YEAR 2013.

[2]. A.Arasu, S. Chaudhuri, and R. Kaushik, "Learning string transformations from examples," *Proc. VLDB Endow.*, vol. 2, pp. 514– 525, August 2009.

[3]. M. Dreyer, J. R. Smith, and J. Eisner, "Latent-variable modeling of string transductions with finite-state methods," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, ser. EMNLP '08. Stroudsburg, PA, USA: Association for Computational Linguistics, 2008, pp. 1080–1089.

[4]. S. Tejada, C. A. Knoblock, and S. Minton, "Learning domainindependent string transformation weights for high accuracy object identification," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '02. New York, NY, USA: ACM, 2002, pp. 350–359.

[5]. N. Okazaki, Y. Tsuruoka, S. Ananiadou, and J. Tsujii, "A discriminative candidate generator for string transformations," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, ser. EMNLP '08. Morristown, NJ, USA: Association for Computational Linguistics, 2008, pp. 447–456.

[6]. M. Li, Y. Zhang, M. Zhu, and M. Zhou, "Exploring distributional similarity based models for query spelling correction," in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ser. ACL '06. Morristown, NJ, USA: Association for Computational Linguistics, 2006, pp. 1025–1032.

[7]. A.Behm, S. Ji, C. Li, and J. Lu, "Space-constrained gram-based indexing for efficient approximate string search," in *Proceedings of the 2009 IEEE International Conference on Data Engineering*, ser. ICDE '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 604–615.