String Transformation-A Query based Approach

[1]Shijina J Salim,[2]Diliya M Khan
[1]Student, [2]Assistant Professor
[1]Department of Computer Science and Engineering,
[1]LBSITW, Poojappura, Trivandrum, Kerala, India

***Abstract*— Data mining is the process of analyzing data from different perspectives and it is a powerful area, which computerizes the process of searching valuable information from a large database. Its wide range of applications promises a future, where the data grow rapidly. Many problems in natural language processing, data mining, information retrieval, and bioinformatics can be formalized as string transformation. Proposed system implements string transformation in data mining field with the help of selected algorithms. String transformation includes a set of operators to transform a given string into most likely output strings. Insertion, deletion, transposition, and substitution are the operators for transformation. Transformation rules and predefined rule indexes are used here to avoid unwanted searches and time delay. Here the users can view the formation of possible outcomes from the given string. Another important feature is query reformulation. By using efficient methods, proposed system can introduce query reformulation with useful description about the given query. Query reformulation is also a transformation technique and it deals with the term mismatch problem. Here similar query pairs can mine from training data. Proposed system tries to transform a given query to original query and therefore make a better match between the query and the document and also act as a library, which give a brief description about books of authors.**

***Index Terms*—String transformation, query reformulation.**

I. INTRODUCTION

Data mining is a step in the process of knowledge discovery from data (KDD). KDD concerns the acquisition of new, important, valid and useful knowledge from data. KDD is not a single step solution of applying a machine learning method to a dataset (see figure 1), but continuous process with many loops and feedbacks. The main steps in the process include business or problem understanding, data understanding, data preparation (data cleaning and pre-processing), modelling (applying machine learning and data mining algorithms), evaluation (checking the performance of algorithms) and deployment. Data-mining tools promise to discover knowledge. It is a proactive process that automatically searches data for new relationships and anomalies on which to base business decisions in order to gain competitive advantage. Although data mining might always require some interaction between the investigator and the data-mining tool, it may be considered as an automatic process because 'data mining tools automatically search the data for anomalies and possible relationships, thereby identifying problems that have not yet been identified by the end user', while mere data analysis' relies on the end users to define the problem, select the data, and initiate the appropriate data analyses to generate the information that helps model and solve problems they uncovered'. Traditional query and report tools have been used to describe and extract what is in a database. The user forms a hypothesis about a relationship and verifies it or discounts it with a series of queries against the data. For example, an analyst might hypothesize that people with low income and high debt are bad credit risks and query the database to verify or disprove this assumption. Data mining can be used to generate a hypothesis. Data mining techniques are the result of a long process of research and product development. This evolution began when business data was first stored on computers, continued with improvements in data access, and more recently, generated technologies that allow users to navigate through their data in real time. Data mining takes this evolutionary process beyond retrospective data access and navigation to prospective and proactive information delivery. Data mining is ready for application in the business community because it is supported by three technologies that are now sufficiently mature [1]:

- Massive data collection
- Powerful multiprocessor computers
- Data mining algorithms

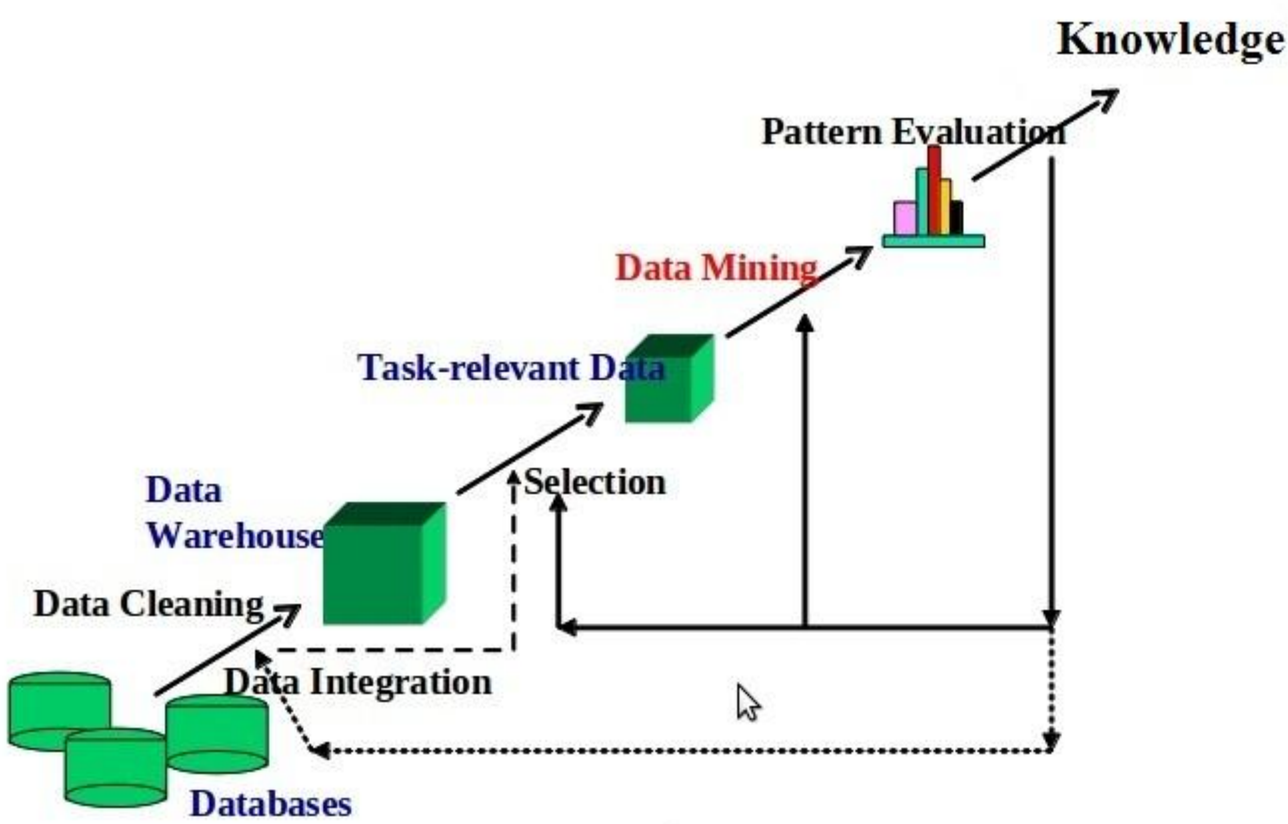

Fig. 1 Knowledge Discovery (KDD)

Text Mining is the discovery by computer of new, previously unknown information, by automatically extracting information from different written resources. A key element is the linking together of the extracted information together to form new facts or new hypotheses to be explored further by more conventional means of experimentation. Text mining is different from what are familiar with in web search. In search, the user is typically looking for something that is already known and has been written by someone else. In text mining, the goal is to discover unknown information, something that no one yet knows and so could not have yet written down. Text mining is a young interdisciplinary field which draws on information retrieval, data mining, machine learning, statistics and computational linguistics. As most information (over 80 %) is stored as text, text mining is believed to have a high commercial potential value. Knowledge may be discovered from many sources of information [2]. Unstructured texts remain the largest readily available source of knowledge. Text mining is also a sibling of data mining.

String transformation is an essential problem in many applications. String transformation can be defined in the following way. Given an input string and a set of operators, we are able to transform the input string to the k most likely output strings by applying a number of operators as shown in figure. Here the strings can be strings of words, characters, or any type of tokens. Each operator is a transformation rule that defines the replacement of a substring with another substring. The likelihood of transformation can represent similarity, relevance, and association between two strings in a specific application. Reformulation of queries in search is aimed at addressing the problem of mismatch. For example, if the query is “IOC” and the document only contains “Indian Oil Corporation”, the query and the document does not fit well and the document does not classified high. Query Reformulation tries to transform “IOC” the “Indian Oil Corporation” and therefore make a better match between the query and the document. Figure 2 shows an overview of string transformation.

Query reformulation try to make a better matching between the query and document. In the task, system needs to generate all similar queries from the original query (strings of words). Previous work on string transformation can be categorized into two groups. Some task mainly considered efficient generation of strings, assuming that the model is given. Other work tried to learn the model with different approaches, such as a generative model, a logistic regression model, and a discriminative model. There are three fundamental problems with string transformation: how to define a model which can achieve both high accuracy and efficiency, how to train the model accurately and efficiently from training instances, how to efficiently generate the top k output strings given the input string, with or without using a dictionary.

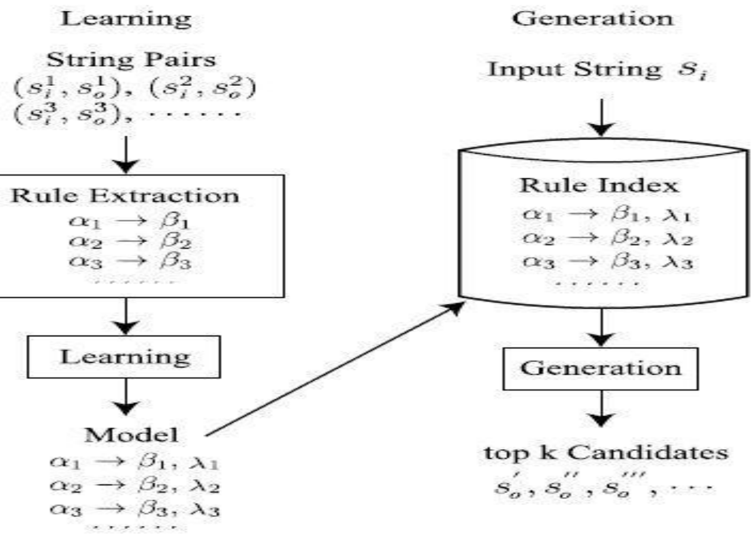

Fig. 2 Overview of String Transformation

Proposed method is novel and unique. It uses a log-linear model for string transformation, an efficient algorithm for string generation, and for query reformulation. The log linear model defined as a conditional probability distribution of an output string and a rule set for the transformation given an input string. Proposed model is trained toward the objective of generating strings with the largest likelihood given input strings. The generation algorithm efficiently performs the top k candidates' generation using top k pruning. To find the best k candidates pruning is guaranteed without enumerating all the possibilities. Aho-corasick algorithm is also used here to generate string as fast as possible. It is used for finding occurrences of words in text and it is faster than other common algorithms. We can divide the algorithm into two steps. First is to constructing a finite state automaton. That is constructing a Trie with input words and finding the right fall function for each node of trie. Second is to moving through automaton and reading each character from input and storing the results.

II. Literature survey

String transformation concept is introduced in 90s. Various studies took place in early years about string transformation in many fields. Many existing algorithms are used to answer approximate string queries. First study how to solve spell error problems. Later consider time delay and other quality factors. Query reformulation is introduced after few years from spell check. Studies are takes place in natural language processing as well as data mining. In data mining, string transformation can be employed in the mining of synonyms and database record matching.

Online Spelling Correction for Query Completion is a transformation model that is capable of capturing users spelling behaviour. Also estimate the transformation model using clicks on search engine recourse links, which represent user confirmed query misspellings. Search algorithm used here is configured to deal with partial queries, so that online search is possible. In this paper, they search queries with a generative model, where the intended query is transformed through a noisy channel into a potentially misspelled query. The distribution from which the target query is selected is estimated from the search engine query log based on frequency. Thus, they are more likely to suggest more popular queries. For the noisy channel, this describes the distribution of spelling errors. Main problem with this function is that it does not deal the untransformed part of the input query. Therefore, this can design a better heuristic by taking into consideration the upper bound of the transformation probability [3].

Table 1 Types of misspellings

CAUSE	MISSPELLING	CORRECTION
Typing quickly	exxit	exit
Keyboard adjacency	importamt	important
Inconsistent rules	recieve	receive
Ambiguous word breaking	up date	update

A Unified and Discriminative Model for Query Refinement describes new CRF model for performing query refinement, called CRFQR. The model is unique in that it predicts a sequence of refined query words as well as corresponding operations given a sequence of query words. And show that employing a unified and discriminative model in query refinement is effective. They propose exploiting a unified and discriminative model in query refinement, specifically conducting various query refinement tasks in a unified framework, and employing a special model called CRF-QR to accomplish the tasks. One advantage of employing this model is that the accuracy of query refinement can be enhanced. This is because the tasks of query refinement are often mutually dependent, and need to be addressed at the same time. Lexicon-based feature representing whether a query word or a refined query word is in a lexicon or a stop word list. Position-based feature representing whether a query word is at the beginning, middle, or end of the query. Corpus-based feature representing whether the frequency of a query word or a query word in the corpus exceeds a certain threshold. Query-based feature representing whether the query is a single word query or multi-word query [4].

Table 2 Types of misspellings

ORIGINAL QUERY	REFINED QUERY
sytem requirement	system requirement
You tube	YouTube
Data mine	Data mining
Onlin	Online

Space-Constrained Gram-Based indexing model explain how to reduce the size of index structures, while still maintaining a high query performance. The setting of approximate string search is unique in that a candidate result needs to occur at least a certain number of times among all the inverted lists, and not necessarily on all the inverted lists. The first approach is based on the idea of discarding some of the lists. They study several technical challenges that arise naturally in this approach. One issue is how to compute a new lower bound on the number of common grams (whose lists are not discarded) shared by two similar strings, the formula of which becomes technically interesting. These models partition an inverted list into fixed-size segments and compress each segment with a word-aligned integer coding scheme. Also studied how to adopt existing inverted-list compression techniques to achieve the goal, and proposed two novel methods for achieving the goal. The trie based pruning is not included in this model; hence performance of this model is not so efficient [5].

In 2006 study in string transformation concentrate on the problem of learning improved query spelling correction model by integrating distributional similarity information automatically derived from query logs. The key contribution of work is identifying that they can successfully use the evidence of distributional similarity to achieve better spelling correction accuracy. They present efficient methods that are able to take advantage of distributional similarity information. This system extends a string edit based error Model with probabilities within a generative source channel model and it explores the effectiveness of approach by integrating distributional similarity based features [6].

Another model is discriminative model, which addresses many challenges by exploring the discriminative training of candidate generators. More specifically, they build a binary classifier that, when given a source strings, decides whether a candidate should be included in the candidate set or not. This approach appears straightforward, but it must resolve two practical issues. First, the task of the classifier is not only to make a binary decision for the two strings, but also to enumerate a set of positive strings for the string. Another issue arises when they prepare a training set. A discriminative model requires a training set in which each instance (pair of strings) is annotated with a positive or negative label. They design features that express transformations from a source string to its destination string. And also present an algorithm that utilizes the feature weights to enumerate candidates of destination strings efficiently [7].

Learning string transformations from example is another method, which can learn a set of transformation rules that explain most of the given examples. Increasing the coverage of the rule set was the primary focus. This method analyzes the difference between the strings and used the hypothesis that consistent differences occurring across many examples are indicative of a transformation rule. Based on this intuition, they formulated a rule learning problem where seek a concise set of transformation rules that accounts for a large number of differences. Most approaches to record matching rely on textual similarity of the records, typically computed using a similarity function such as edit distance to determine if two records are matches or not. However, textual similarity can be an imperfect indicator of whether or not two records are matches; in particular, two matching records can be textually dissimilar [8].

III. PROPOSED SYSTEM

Proposed method is used to solve two problems, spelling error correction and reformulation of queries in web search. The difference between the two problems is that string transformation is performed at a character level in the former task and at a word level in the latter task. Aho-Corasick algorithm is used to generate words. The effect of using the Aho-Corasick algorithm is tested here. The time complexity of Aho-Corasick algorithm is determined by the length of query word plus the number of matches. Also examined how the number of matches on query word changes when the size of the rule set increases. The number of matches is not largely affected when the size of the rule set increases in the rule index. It implies that the time for searching applicable rules is close to a constant and does not change much with different numbers of rules. This method has been applied to two applications, spelling error correction of queries and reformulation of queries in web search. Experiments have been conducted between this method and the baselines methods. The result shows that proposed method performs consistently better than the baselines in terms of accuracy. Moreover, the accuracy of this method is constantly better than the baselines in different experiment settings, such as size of the rule set, maximum number of applicable rules, and dictionary size. Experiments on efficiency have been conducted with running time as the measure. The running times of this method are smaller than those of the baselines, which indicate that the pruning strategy in this method is very efficient.

Top k Pruning

Top k Pruning 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. Unwanted paths are removed by the help of this algorithm and thereby we can reduce complexity and time delay. 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 [1]. Here we consider the following heuristics, which are highly reasonable and work well in practice:

- A path is preferred if no rule is applicable at position, pos-1 of it
- A path is preferred if its position, pos is larger
- A path is preferred if its score is higher

Dictionary Matching Algorithm

Aho-Corasick string matching algorithm is a string searching algorithm. It is a kind of dictionary-matching algorithm that locates elements of a finite set of strings (the "dictionary") within an input text. It matches all patterns simultaneously. The complexity of the algorithm is linear in the length of the patterns plus the length of the searched text plus the number of output matches. Informally, the algorithm constructs a finite state machine that resembles a trie with additional links between the various internal nodes. These extra internal links allow fast transitions between failed pattern matches to other branches of the trie that share a common prefix. This allows the automaton to transition between pattern matches without the need for backtracking. 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 [1]. The rule index stores all the rules and their weights, which can make the references of rules very efficient as shown in figure 3. The AC tree is a trie with failure links, on which the Aho-Corasick string matching algorithm can be executed. The Aho-Corasick algorithm is a well-known dictionary matching algorithm which can quickly locate the elements of a finite set of strings within an input string [10].

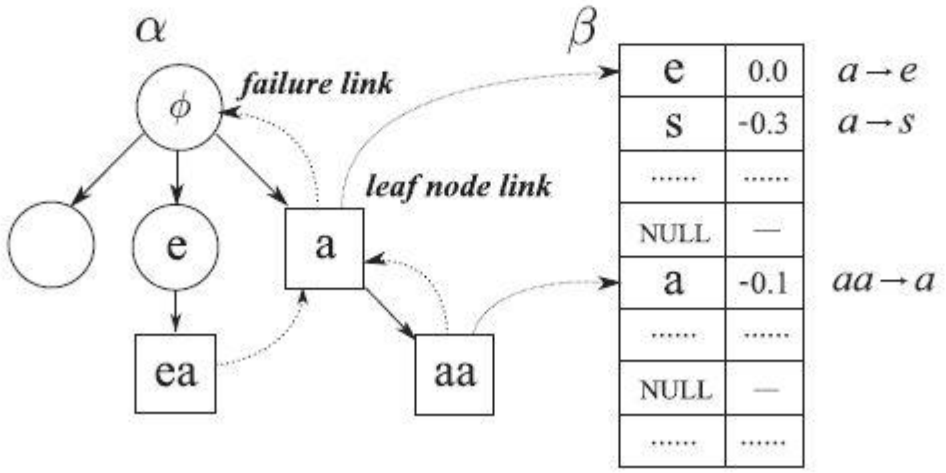

Fig. 3 Rule index based on Aho-Corasick tree

Log linear model

Log linear model is a mathematical model, takes the form of function which has first degree polynomial function of parameter. Log linear actually uses binary features to indicate whether or not rules are applied. Conditional probability distribution is used here, which can transform the input string to the output string. To improve generation efficiency, assume that all the weights are non-positive. Hence higher probability of a word is the result. The chance of making more errors should be lower than that of making fewer errors. So we can say that the log linear model increase the efficiency of proposed system.

$$P(S_o|S_i) = \sum_R (S_{i,} S_o) P(S_o, R(S_i, S_o)| S_i)$$

Here we take the maximum of the conditional probabilities $P(So, R(S_i, S_o)| S_i)$ instead of taking $P(S_o|S_i)$, because in this way we can make the generation process very efficient and for each possible transformation, simply take the summation of weights of the rules used in transformation [1].

Spelling error correction

In spelling error correction enters a string and then enters a space. If the input string is not correct then it displayed as red. And combination of characters are generated and displayed as tree structure, and then matching words are listed. The accuracy of the spell check is determined as the ratio of total combinations of characters to the matching candidates. Accuracy = Total combinations of characters/matching candidates. If the word we given is in dictionary and there is no error exist, then the word show as black letters. Another important feature is that proposed system has provision to add a new word to the dictionary.

Query reformulation

Query reformulation is the process of rewriting the original query with its similar queries and enhancing the accuracy of search. It involves evaluating a user's input and expanding the search query to match additional documents. In query reformulation, similar query pairs are used as training. If the coefficient is larger than a threshold, then the two queries are considered similar. This method only works well for head (high frequency) queries. We consider learning a string transformation model trained from head queries and applying the model to tail queries, i.e., conducting top k similar query finding on tail queries. Note that in this case, no dictionary can be used in string transformation; because it is impossible to construct a dictionary of queries in advance in web. When given a query the system generate all similar queries from the original query.

Proposed system can receive a wrong query and can give original queries similar to the given query. Also proposed system gives details about the given query. Here authors and their publishing book details are shown. Anchor text and string transformation are the two modes to choose by the user. If a user choose anchor text, then the accurate query result show else all queries, which are original are shown and hence user can view book details of any author. Proposed system uses efficient methods to get this accurate output. For spell correction Log linear model is used. Log linear model with transformation give accuracy and efficiency to the proposed method.

IV. Experiment and Result

Proposed system counts the total combinations of letters and then counts the matching candidates. Take the ratio of total combinations to matching candidates. That is the accuracy. And the time required to show the matching candidates after enter space is the time delay.

Table 3 Dataset 1 (Combination of two letters)

DATA SET INPUT	TOTAL COMBINATIONS	MATCHING CANDIDATES	TIME DELAY IN SECONDS	ACCURACY
Df	413	8	.25	51.625
Lm	173	3	.50	57.66
Cd	296	4	.75	74
Bv	238	4	1	59.5

The table shows the accuracy of spell check. Here input is two letter words. These words are not correct. So combinations of characters are generated. That is the total combinations. The generation is done by using Aho-Corasick algorithm. The time delay for the generation algorithm is shown in the table. That is the time required to generate combinations of characters after enter the space. Then top k pruning algorithm is performed to select matching candidates. Count number of matching candidates and total combinations of characters. Then calculate the accuracy. From the above examples average accuracy of the system when enter input with two letters. Average accuracy = total accuracy/ 4 = 60.696

Table 4 Dataset 2 (Combination of three letters)

DATA SET INPUT	TOTAL COMBINATIONS	MATCHING CANDIDATES	TIME DELAY IN SECONDS	ACCURACY
asd	938	13	.25	74.46
crn	976	17	.50	57.41
Far	848	14	.75	60.57
ter	1075	20	1	53.75

In the table 4 inputs are three letter strings. So combinations of letters are generated. Average accuracy = 61.547 and average time delay = 0.965 second.

Table 5 Dataset 3 (Combination of four letters)

DATA SET INPUT	TOTAL COMBINATIONS	MATCHING CANDIDATES	TIME DELAY IN SECONDS	ACCURACY
gare	1345	26	.25	51.73
Comn	1237	15	.50	82.42
Fint	876	13	.75	67.38
worg	1149	13	1	88.38

Average accuracy =76.35%, average time delay =1.11 second. From the analysis of these tables as letters in input string increases combinations of characters are also increases. Thus accuracy also increases. But in the case of time delay as number of letters in the input string increases delay also increases due to large number of combinations. That means as efficiency slightly decreases as number of letters in the input string increases. Based on these examples, average accuracy and efficiency of the system can be calculated. Average accuracy = 66.197% and average time delay = 0.763 second.

Query reformulation is the process of rewriting the original query with its similar queries and enhancing the accuracy of search. It involves evaluating users input and expanding the search query to match additional documents. When given a query the system generate all similar queries from the original query. In query reformulation enter a query. The proposed system shows matching queries. From list of matching queries select the top query which user wants. When select that query the system shows the details in a tabular format. Here accuracy can be calculated by the information which is retrieved. For that given some example queries and then takes the matching query. When the query is selected information is obtained in a table form. These queries are related to books and authors. If the result obtained is correct then it is 99% accurate.

Table 6 Dataset 4 (Query and result accuracy)

QUERY	MATCH QUERY	RESULT
apjabdulkalam	Famous books by Dr.APJ Abdul Kalam	Books and its details are shown
muhammadbasheer	Famous books by Vaikom Muhammad Basheer	Books and its details are shown
m t vasudevan	Famous books by M T Vasudavan Nair	Books and its details are shown

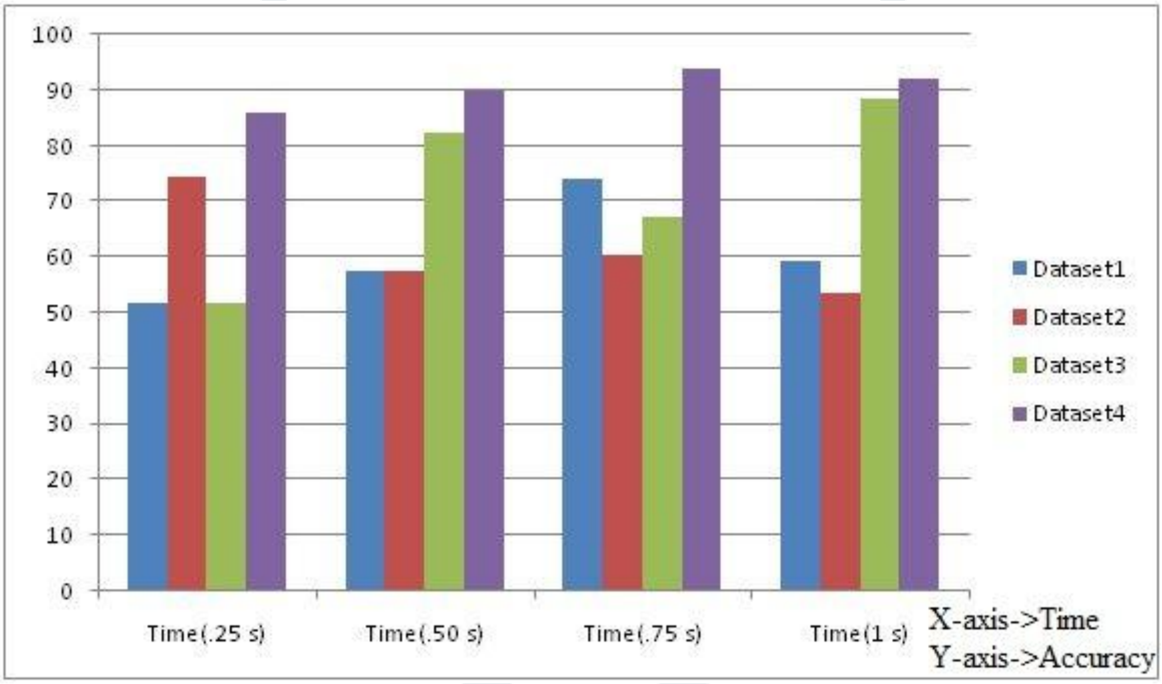

Fig. 4 Overall performance of proposed system

V. CONCLUSION AND FUTURE WORK.

Proposed method is novel and unique in its model, learning algorithm, and string generation algorithm. Two specific applications are addressed with this method, namely spelling error correction of queries and query reformulation in web search. Experimental results on two large data sets and Microsoft Speller Challenge show that the proposed method improves upon the baselines in terms of accuracy and efficiency. Proposed method is particularly useful when the problem occurs on a large scale. No another cost factors need to get knowledge. Another big factor is that there is less need of time for the retrieval of data and also for string transformation. The work described in this thesis has been concerned with efficient algorithms for reducing time delay and to boost accuracy. We can use proposed algorithms more efficiently than the implemented system. Log linear model with conditional probability boost the accuracy of string generation. It can transform an input string to output string more accurately. Predefined datasets can also improve the efficiency of the implemented system.

VI. ACKNOWLEDGMENT

I am thankful to my guide Mrs. Diliya M Khan, Assistant Professor of Computer Science and Engineering, for her guidance and encouragement for the paper work.

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. 26, no. 5, May 2014.

[2] Vishal Gupta, Gurpreet S. Lehal, “A Survey of Text Mining Techniques and Applications,” Journal of emerging technologies in web intelligence, Vol. 1, No. 1,August 2009

[3] H. Duan and B.-J. P. Hsu, Online spelling correction for query completion, in Proc. 20th Int. Conf. World Wide Web, New York, NY, USA, 2011, pp. 117126.

[4] J. Guo, G. Xu, H. Li, and X. Cheng, A unified and discriminative model for query refinement, in Proc. 31st Annu. Int. ACM SIGIR Conf. Research Development Information Retrieval, New York, NY, USA, 2008, pp. 379386.

[5] A. Behm, S. Ji, C. Li, and J. Lu, Space-constrained gram-based indexing for efficient approximate string search, in Proc. 2009 IEEE Int. Conf. Data Engineering, Washington, DC, USA, pp. 604 615.

[6] M. Li, Y. Zhang, M. Zhu, and M. Zhou, Exploring distributional similarity based models for query spelling correction, in Proc. 21st Int. Conf. Computational Linguistics and the 44th Annu. Meeting Association for Computational Linguistics, Morristown, NJ, USA, 2006, pp. 10251032.

[7] N. Okazaki, Y. Tsuruoka, S. Ananiadou, and J. Tsujii, A discriminative candidate generator for string transformations, in Proc. Conf. Empirical Methods Natural Language Processing, Morristown, NJ, USA, 2008, pp. 447456.

[8] A. Arasu, S. Chaudhuri, and R. Kaushik, Learning string transformations from examples, Proc. VLDB Endow., vol. 2, no. 1, pp. 514525, Aug. 2009.

[9] S. Tejada, C. A. Knoblock, and S. Minton, Learning domain independent string transformation weights for high accuracy object identification, in Proc. 8th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, New York, NY, USA, 2002, pp. 35035

[10] A. V. Aho and M. J. Corasick, E_cient string matching: an aid to bibliographic search, Commun. ACM, vol. 18, no. 6, pp. 333340, Jun. 1975.

[11] N. Okazaki, Y. Tsuruoka, S. Ananiadou, and J. Tsujii, A candidate generator for string transformations, in Proc. Conf. Empirical Methods Natural Language Processing, Morristown, NJ, USA, 2008, pp. 447456.

[12] M. Dreyer, J. R. Smith, and J. Eisner, Latent-variable modeling of string transductions with finite-state methods, in Proc. Conf. Empirical Methods Natural Language Processing, Stroudsburg, PA, USA, 2008, pp. 10801089.

[13] J. Xu and G. Xu, Learning similarity function for rare queries, in Proc. 4th ACM Int. Conf. Web Search and Data Mining, NewYork, NY, USA, 2011, pp. 615624.

[14] M. Hadjieleftheriou and C. Li, E_cient approximate search on string collections, Proc. VLDB Endow., vol. 2, no. 2, pp. 16601661, Aug. 2009.

[15] Jess Vilares, Manuel Vilares, Juan Otero, Managing misspelled queries in IR applications,Information Processing and Management 47 (2011) 263286.

[16] A. R. Golding and D. Roth, A winnow-based approach to context-sensitive spelling correction, Mach. Learn., vol. 34, no. 13, pp. 107130, Feb. 1999.