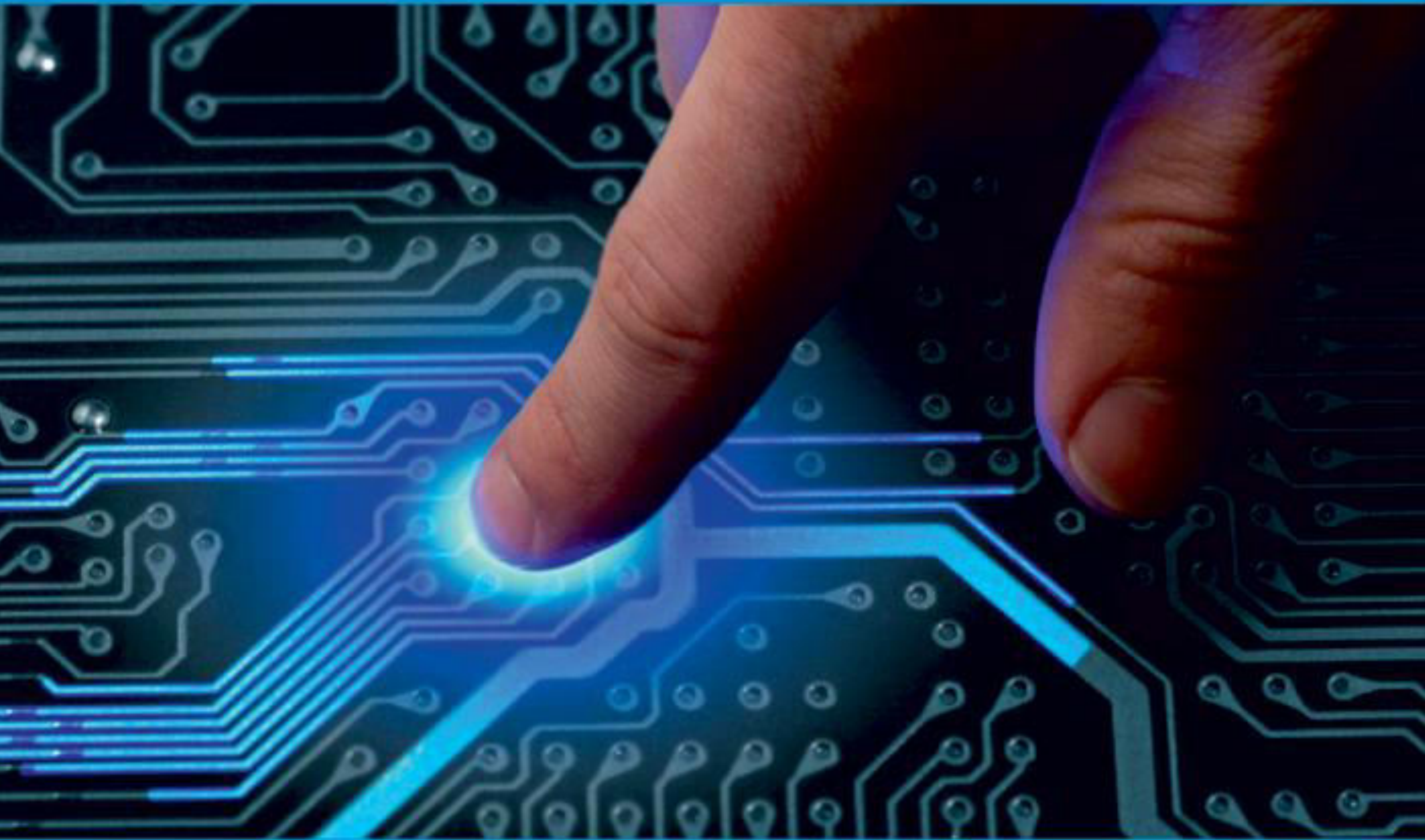
IJIRCCE

e-ISSN: 2320-9801 | p-ISSN: 2320-9798



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

Volume 11, Issue 2, February 2023

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 8.165

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165|

|| Volume 11, Issue 2, February 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1102018 |

A String Transformation with Edit Distance Algorithm Using Pruning Method

K.M.PRADEEPAN, DHINESHKUMAR D

Assistant Professor, Department of Computer Applications (MCA), K.S.R. College of Engineering (Autonomous), Tiruchengode, India

Department of Computer Applications (MCA), K.S.R. College of Engineering (Autonomous), Tiruchengode, India

ABSTRACT: String Transformation can be formalized such as given an input string, the system generates the k most likely output strings corresponding to the input string. The essential step for string transformation is to generate candidates to which the given string S is likely to be transformed. A novel approach is proposed to generate the candidates for string transformation which is more accurate and efficient. The process is initiated by the rule index with the help of the Aho-Corasick tree (A-C Tree) which frame the rules efficiently. Top candidate values are generated with the use of MDL (Minimum Description Length) method. This string generation algorithm based on MDL pruning is guaranteed to generate the optimal top k candidates. A Dictionary trie matching method is invoked for spelling error corrections in queries as well as reformulation of queries in web search. Thus a new statistical learning method which is novel and unique in its training model, learning method, pruning technique and the string generation algorithm is proposed.

I. INTRODUCTION

String Transformation can be generally considered as the natural language processing which includes pronunciation generation, spelling error correction, word transliteration, and word stemming, etc. String transformation is also used in query reformulation and query suggestion in web search. It can be employed in the mining of synonyms and database record matching in the area of Data Mining.

1.1 String Transformation

As many of the below explained methods are online applications, the transformation must be conducted not only accurately but also efficiently.

Spelling error Correction: Given the list of documents or queries, a spell checker implemented can find the potential candidates for a possibly misspelled word by performing a string search and comparison in its dictionary.

Pre-processing/ Data-Cleaning: Similar information's from different sources have always many difficulties. For example, the address line mentioned as "PO Box 14, East Street." and "P.O. Box 14, East St". Mistakes can be also introduced due to irregularities in the data-collection process, due to human errors, and some other causes. For all these reasons, it is essential for data cleaning to identify similar entities within a collection, or all similar pairs of entities around a number of collections.

Query Suggestion: A very recent important application is to provide answers to query results in real-time, as users are typing their query (e.g., a Google search box with a dropdown suggestion menu that updates as users type). Such interactive-search boxes are very helpful and have shown to be very important in practice, because they limit the number of errors made by users and also reduce the number of query reformulations submitted in order to find the one that will yield satisfying results.

Query Reformulation: Query reformulation in web search is aimed at dealing with the term mismatch problem. For example, if the query is "NYT" and the document only contains "New York Times", then the query and document do not match well and the document will not be ranked high. Query reformulation attempts to transform "NYT" to "New York Times" and thus make a better matching between the query and document. In the task, given a query (a string of words), one needs to generate all similar queries from the original query (strings of words).

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165|

|| Volume 11, Issue 2, February 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1102018 |

1.2 Candidate Generation

The essential and important step for string transformation is to generate candidates to which the given string ***S*** is likely to be transformed. Candidate generation can be used to find the most likely corrections of a misspelled word from the dictionary. In this case, a string of characters is input and the operators represent insertion, deletion, and substitution of characters with or without surrounding characters.

The candidate generator uses a set of transformations to judge the similarity between two objects, so that only the most similar candidate mappings between the sources are generated. The main function of the candidate generator is to produce an initial set of quality candidate mappings. The candidate generator keeps a record of the set of transformations that were applied for each mapping, which is essential for learning the transformation weights, and also calculates a set of similarity scores, necessary for learning the mapping rules. When comparing objects, the alignment of the attributes is determined by the user.

The values for each attribute are compared individually. Comparing the attributes individually is important in reducing the confusion that can arise when comparing the objects as a whole. Words can overlap between the attribute values. Comparing the attributes individually saves computation and also decreases mapping error by reducing the number of candidate mappings considered. Given the two sets of objects, the candidate generator is responsible for generating the set of candidate mappings by comparing the attribute values of the objects.

II. RELATED WORKS

The work conducted so far on string transformation can be divided into two categories. Some work mainly considered the 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.

2.1 Generative Models

Generative model is a full probabilistic model of all variables, which can be used, for example, to simulate (i.e. generate) values of any variable in the model.

Examples of generative models include:

- Gaussian mixture model and other types of mixture model
- Hidden Markov model
- Probabilistic context-free grammar
- Naive Bayes
- Averaged one-dependence estimators
- Latent Dirichlet allocation
- Restricted Boltzmann machine

If the observed data are truly sampled from the generative model, then fitting the parameters of the generative model to maximize the data likelihood is a common method. However, since most statistical models are only approximations to the true distribution, if the model's application is to infer about a subset of variables conditional on known values of others, then it can be argued that the approximation makes more assumptions than are necessary to solve the problem at hand. In such cases, it can be more accurate to model the conditional density functions directly using a discriminative model, although application specific details will ultimately dictate which approach is most suitable in any particular case.

2.2 Discriminative Models

Discriminative model provides a model only for the target variable(s) conditional on the observed variables which allow only sampling of the target variables conditional on the observed quantities. Despite the fact that discriminative models do not need to model the distribution of the observed variables, they cannot generally express more complex relationships between the observed and target variables.

A discriminative model requires a training set in which each instance (pair of strings) is annotated with a positive or negative label. Even though some existing resources (e.g., inflection table and query log) are available for positive instances, such resources rarely contain negative instances. Therefore, we must generate negative instances that are effective for discriminative training. Various models are used under the discriminative approach.

Logistic Regression Model

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165|

|| Volume 11, Issue 2, February 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1102018 |

Logistic regression measures the relationship between a categorical dependent variable and one or more independent variables, which are continuous, by using probability scores as the predicted values of the dependent variable. Frequently logistic regression is used to refer specifically to the problem in which the dependent variable is binary (either 1 or 0).

Okazaki's [2] method incorporated rules into an L1-regularized logistic regression model and utilized the model for string transformation. Okazaki's model is a discriminative model. Their model is defined as a logistic regression model (classification model) P(t/ s), where s and t denote input string and output string respectively, which utilizes all the rules that can convert s to t and it is assumed only one rule can be applied each time.

Log Linear Model

A key advantage of log-linear models is their flexibility. They allow a very rich set of features to be used in a model, arguably much richer representations than the simple estimation techniques.

The goal of log-linear analysis is to determine which model components are necessary to retain in order to best account for the data. Model components are the number of main effects and interactions in the model. For example, if examined the relationship between three variables—variable A, variable B, and variable C—there are seven model components in the saturated model. The three main effects (A, B, C), the three two-way interactions (AB, AC, BC), and the one three-way interaction (ABC) gives the seven model components. The log-linear models can be thought of to be on a continuum with the two extremes being the simplest model and the saturated model. The simplest model is the model where all the expected frequencies are equal. This is true when the variables are not related.

Dreyer's [3] method proposed a log linear model for string transformation, with features representing latent alignments between the input and output strings. Finite-state transducers are employed to generate the candidates.

Wang [1] also proposed a log linear method but in a probabilistic approach. Wang's learning method is based on maximum likelihood estimation. Thus, the model is trained toward the objective of generating strings with the largest likelihood given input strings. The generation algorithm efficiently performs the top k candidate's generation using top k pruning. It is guaranteed to find the best k candidates without enumerating all the possibilities.

2.3 n-Gram Based Models

An n-gram is a contiguous sequence of n items from a given sequence of text or speech. The items can be phonemes, syllables, letters, words or base pairs according to the application. The n-grams typically are collected from a text or speech corpus. An n-gram of size 1 is referred to as a "unigram"; size 2 is a "bigram" (or, less commonly, a "digram"); size 3 is a "trigram". Larger sizes are sometimes referred to by the value of n, e.g., "four-gram", "five-gram", and so on.

An n-gram model [4] is a type of probabilistic language model for predicting the next item in such a sequence in the form of a (n - 1) order Markov model. The n-gram models are now widely used in probability, communication theory, computational linguistics (for instance, statistical natural language processing), computational biology (for instance, biological sequence analysis), and data compression. The two core advantages of n-gram models (and algorithms that use them) are relative simplicity and the ability to scale up by simply increasing n value. When used for language modeling, independence assumptions are made so that each word depends only on the last n-1 words. This assumption is important because it massively simplifies the problem of learning the language model from data. In addition, because of the open nature of language, it is common to group words unknown to the language model together.

2.4 Pruning

Pruning is a technique in machine learning that reduces the size of decision trees by removing sections of the tree that provide little power to classify instances. The dual goal of pruning is reduced complexity of the final classifier as well as better predictive accuracy by the reduction of over fitting and removal of sections of a classifier that may be based on noisy or erroneous data. One of the questions that arise in a decision tree algorithm is the optimal size of the final tree. A tree that is too large risks over fitting the training data and poorly generalizing to new samples. A small tree might not capture important structural information about the sample space. However, it is hard to tell when a tree algorithm should stop because it is impossible to tell if the addition of a single extra node will dramatically decrease error. This problem is known as the horizon effect. A common strategy is to grow the tree until each node contains a small number of instances then use pruning to remove nodes that do not provide additional information.

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165|

|| Volume 11, Issue 2, February 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1102018 |

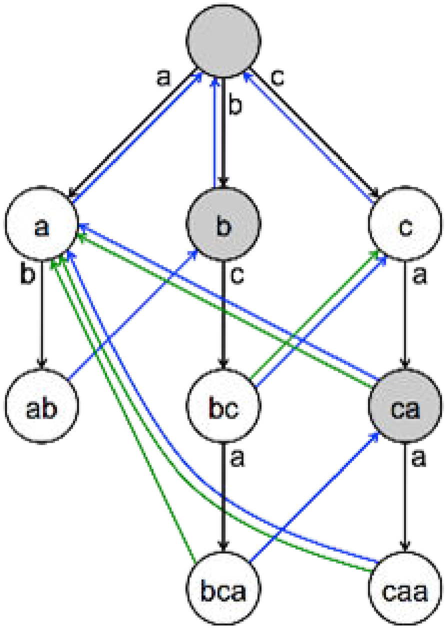


Figure 1 Aho-Corasick Data Structure

Pruning should reduce the size of a learning tree without reducing predictive accuracy as measured by a test set or using cross-validation. There are many techniques for tree pruning that differ in the measurement that is used to optimize performance. Pruning can occur in a top down or bottom up fashion. A top down pruning will traverse nodes and trim sub-trees starting at the root, while a bottom up pruning will start at the leaf nodes.

Wang [1] uses a top k pruning algorithm to generate the optimal top k candidates.

2.5 Dictionary Trie Matching

Using the dictionary trie, the misspelled word or string is corrected. Sometimes a dictionary is utilized in string transformation in which the output strings must exist in the dictionary, such as spelling error correction, database record matching, and synonym mining. Specifically, the dictionary is indexed in a trie, such that each string in the dictionary corresponds to the path from the root node to a leaf node. When a path (substring) is expanded in candidate generation, it is matched against the trie, and checked for whether the expansions from it are legitimate paths or not. If not, the expansions are discarded and avoided generating unlikely candidates. In other words, candidate generation is guided by the traversal of the trie. Finally, the identified word pairs are aggregated across sessions and discarded the pairs with low frequencies which improves the accuracy and consumes less running time.

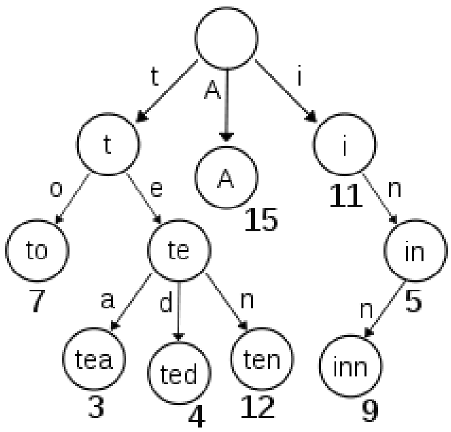


Figure 2 An Example for Trie

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165|

|| Volume 11, Issue 2, February 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1102018 |

3. Model for String Transformation

3.1 Rule Extraction (Training Data)

All the possible rules are derived from the training data based on string alignment. First the characters are aligned in the input string and the output strings based on edit-distance, and then rules are derived from the alignment. Next the derived rules are expanded with surrounding contexts. Without loss of generality, considering expanding 0 - 2 characters from left and right sides as contexts in this paper. Derivation of word-level rules can be performed similarly. If a set of rules can be utilized to transform the input string si to the output target string so, then the rule set is said to form a "transformation" for the string pair si and so. Note that for a given string pair, there might be multiple possible transformations. For example, both ("n"!"m","tt"!"t") and ("ni"!"mi","t$"! $) can transform "nicrosoftt" to "microsoft". Without loss of generality, the maximum number of rules is assumed as applicable to a string pair which is predefined. As a result, the number of possible transformations for a string pair is also limited. This is reasonable because the difference between an input string and output string should not be so large. In spelling error correction, for example, the number of possible spelling errors in a word should be rather small. Let (si, so) denote a string pair, and R(si, so) denote a transformation that can rewrite si to so. Considering that there is a probabilistic mapping between the input string si and output string so with transformation R(si, so) and also considering a conditional probability distribution of so and R(si, so) given si and take it as model for string transformation. The log linear model actually uses binary features to indicate whether or not rules are applied.

In general, the weights can be any real numbers. To improve generation efficiency, it is assumed that all the weights are non-positive. It introduces monotonicity in rule application and implies that applying additional rules cannot lead to generation of better candidates. For example, in spelling error correction, both "office" and "officer" are correct candidates of "ofice". Viewing "office" a better candidate than "officer" (higher probability), as it needs one less rule. The assumption is reasonable because the chance of making more errors should be lower than that of making fewer errors.

3.1 String Matching Using MDL Pruning

A top MDL pruning algorithm is employed, which can guarantee to find the optimal output strings. Minimum Description Length (MDL) is an information theoretic model selection principle. MDL assumes that the simplest, most compact representation of data is the best and most probable explanation of the data. It is well known that the most compact encoding of a sequence is the encoding that best matches the probability of the symbols. The algorithm proceeds bottom up from leaves towards the root of the complete tree and for each internal node makes a decision whether to prune the sub-tree or not by the MDL node criterion. The decision is based on the difference between the description length of the classification of instances that fall into the given node treated as a leaf and the description length of the structure of the sub-tree plus the description lengths of the classifications of the subsets of instances in all the leaves of the sub tree. If the difference is negative or equal zero the sub-tree should be pruned. Restricting the set of allowed codes in such a way that it becomes possible (computable) to find the shortest code length of the data, relative to the allowed codes. Whereas behind the pruning it is necessary to perform the negotiation of stop words, identify the stem words.

Algorithm 1: Minimum Description Length (MDL) Pruning

Input: rule index I_r, input string s, candidate number k

Output: top k output strings (candidates) in S_{topk}

begin

Find all rules applicable to s from I_r with Aho-Corasick algorithm

$minscore = -\infty$

$Q_{path} = S_{topk} = \{\}$

Add $(1, \hat{}, 0)$ into Q_{path}

while Q_{path} *is not empty* **do**

Pickup a path $(pos, string, score)$ from Q_{path} with heuristics

if $score \leq minscore$ **then**

continue

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165|

|| Volume 11, Issue 2, February 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1102018 |

if $pos == |s|$ **AND** $string\ reaches\ \$$ **then**
if $|S_k| \geq k$ **then**
Remove candidate with minimum score from S_{topk}
Add candidate $(string, score)$ into S_{topk}
Update $minscore$ with minimum score in S_{topk}
foreach $next\ substring\ c\ at\ pos$ **do**
$\alpha \rightarrow \beta$ = corresponding rule of c
$pos' = pos + |\alpha|$
$string' = string + \beta$
$score' = score + \lambda_{\alpha \rightarrow \beta}$
Add $(pos , string , score)$ into Q_{path}
if $(pos', string', score'')$ in Q_{path} **then**
Drop the path with smaller score
return S_k

Process in Algorithm:

Input: rule index, input string, candidate number k.

Output: Top k output strings (candidates) in list.

1. Find all rules applicable to s with Aho-Corasick algorithm.
2. Find rule with minimum score.
3. Find path with trie.
4. Remove candidate with minimum score from list.
5. Add candidate (string; score) into list.
6. Update minimum score in list.
7. Drop the path with smaller score.
8. Return valid path.

3.3 String Transformation

Using the dictionary trie method, the misspelled word string is corrected. In the setting of using a dictionary, the efficiency is further enhanced. Sometimes a dictionary is utilized in string transformation in which the output strings must exist in the dictionary, such as spelling error correction, database record matching, and synonym mining. Specifically, the dictionary is indexed in a trie, such that each string in the dictionary corresponds to the path from the root node to a leaf node. When a path (substring) is expanded in candidate generation, it is matched against the trie, and checked for whether the expansions from it are legitimate paths or not. If not, the expansions are discarded and avoided generating unlikely candidates. In other words, candidate generation is guided by the traversal of the trie. Finally, the identified word pairs are aggregated across sessions and discarded the pairs with low frequencies which improves the accuracy and consumes less running time.

IV. CONCLUSION

The proposed paper addresses the problem of generating the string candidates effectively and accurately by adopting a discriminative log-linear model for query correction. To efficiently retrieve the query corrections with the highest probability, this method is novel and unique in its model, learning model, pruning technique and string generation algorithm. The proposed MDL decision tree pruning technique in machine learning that reduces the size of decision trees by removing sections of the tree that provide little power to classify instances. The dual goal of pruning is reduced complexity of the final classifier as well as better predictive accuracy by the reduction of over fitting and removal of sections of a classifier that may be based on noisy or erroneous data. Searching for an efficient code reduces to searching for a good probability distribution, and vice versa, which produces better result in identifying the top values

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165|

|| Volume 11, Issue 2, February 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1102018 |

for the pruning. They also provide the unique values in result for the string present in the given query. Thus, a new statistical learning approach is proposed for string transformation which is more accurate and efficient.

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, 2013.

[2] Behm .A, S. Ji, C. Li, and J. Lu, “Space-constrained gram-based indexing for efficient approximate string search,” ICDE ’09. IEEE Computer Society, pp. 604–615, 2009.

[3] Brill E. and R. C. Moore, “An improved error

[4] model for noisy channel spelling correction,” ACL ’00. Morristown, NJ, USA: Association for Computational Linguistics, pp. 286–293, 2000.

[5] Chen Q, M. Li, and M. Zhou, “Improving query spelling correction using web search results,” EMNLP ’07, pp. 181–189, 2007.

[6] Dreyer .M, J. R. Smith, and J. Eisner, “Latent-variable modeling of string transductions with finite-state methods,” EMNLP ’08, Association for Computational Linguistics, pp. 1080–1089, 2008.

[7] Duan.H and B.-J. P. Hsu, “Online spelling correction for query completion,” WWW’11. ACM, pp. 117–126, 2011.

[8] Guo .J, G. Xu, H. Li, and X. Cheng, “A unified and discriminative model for query refinement,” SIGIR ’08. ACM, pp. 379–386, 2008.

[9] Hadjieleftheriou.M and C. Li, “Efficient approximate search on string collections,” Proc. VLDB Endow., vol. 2, pp. 1660–1661, August 2009.

[10] Igor Kononenko, "The minimum description length based decision tree pruning",PRICAI’98: Topics in Artificial Intelligence, Lecture Notes in Computer Science Volume 1531, pp 228-237, 1998.

[11] McCallum A., K. Bellare, and F. Pereira, “A conditional random field for discriminatively-trained finite-state string edit distance,” in Proceedings of the 21st Conference on Uncertainty in Artifical Intelligence, ser. UAI ’05, pp. 388–395, 2005.

[12] Okazaki .N, Y. Tsuruoka, S. Ananiadou, and J. Tsujii, “A discriminative candidate generator for string transformations” EMNLP ’08, Association for Computational Linguistics, pp. 447–456, 2008.

[13] Tejada .S, C. A. Knoblock, and S. Minton, “Learning domain independent string transformation weights for high accuracy object identification,” KDD ’02. ACM, pp. 350–359, 2008.

[14] Yang Y., J. Yu, and M. Kitsuregawa, “Fast algorithms for top-k approximate string matching,” in Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, ser. AAAI ’10, pp.1467–1473.

INNO SPACE
SJIF Scientific Journal Impact Factor
Impact Factor: 8.165

INTERNATIONAL
STANDARD
SERIAL
NUMBER
INDIA

निस्केयर
NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING