An Advanced Precision based Approach to String Transformation

B. Sankara Babu, K. Rajasekhar Rao, P. Satheesh

Abstract: ***Distinct obstacles occur in Natural language processing, Knowledge Engineering, Information Retrieval, Genetics Informatics, Computational molecular biology and Data Mining concerned to String Transformation. Consider an input string, the system automatically produces top k output strings referring to input string. Generally people perform various kinds of spelling errors such as misspell words accidentally while surfing the web. To circumvent such errors, this Paper propounds an advanced Precision based approach to string transformation which is very accurate. The proposed system comprises unique precision value allocated to each alphabet and these are aggregated to give the Total Precision of the particular word. Data sets are trained with the precision based approach by validating them to dictionary called the database. Misspell word precision is compared with the data sets precision and retrieves the top k nearest neighbour output strings relevant to input string. This is one of the best accurate Misspell word and sentence correction approach and experimentally proven on large data sets.***

Keywords: String Transformation, Precision based Approach, Misspell words, Total Precision.

I. INTRODUCTION

Required transformations that are necessary for the replacement of source string with different destination strings defined as String Transformation. In Natural language Processing, String Transformation is illustrated as Misspell word correction, word metamorphosis, word transition. In Information Retrieval, String transformation deals with retrieving the relevant records regarding the query. In Genetic Informatics, it is used to detect the disease in a person by using the standardized DNA patterns. In Data mining it is used as excavating the metonyms and opposite words. In String transformation, Input string is the string comprises the words, group of letters, tokens. Top k corrected output strings are produced by conducting various transformations on an input string which is misspell. Output strings are the strings which are corrected and generated by referring to the dictionary. Transformation is the several operations to replace the input string with different output strings. Correction of misspell word is done in two steps. In First step, it deals with the aspect of Keywords generation. In this step Misspelled words are corrected and distinct keywords are generated.

Manuscript Received on February 16, 2015.

B. Sankara Babu, Assoc. Prof., Department of CSE, Gokaraju Rangaraju Institute of Engineering and Technology Hyderabad, India.

Dr. K. Rajasekhar Rao, Prof., Department of CSE, Koneru Laxmaiah University Guntur, India.

Dr. P. Satheesh, Assoc. Prof., Department of CSE, Maharaj Vijayaram Gajapathiraj College of Engineering Vizianagaram, India.

For example consider a misspelled word “Thught”, here the original word is “Thought “, here the letter’ o’ should be placed to make the misspelled word as correct word. Similarly, many other words are generated related to the misspelled word. In second step it deals with the aspect of Best key word selection. In this step Best key word is selected from distinct keywords based on the priority. The words which are wrongly typed by one character or group of characters to which certain modifications are needed to make them as correct words are defined as Misspell words.Table1 describes the misspell words correction. In the table we considered wrong words which are corrected by operations such as insertion, deletion, substitution to make them into original words.

Table 1. Describes the misspell words correction.

Misspell word	Original Word	Wrongly typed character	Correction	Action Performed
Flowr	Flower	-	e	inserted
Mengo	Mango	e	-	inserted
Beauetiful	Beautiful	-	-	deleted
Technlgy	Technology	e	oo	inserted
Innovatiev e	Innovative	e	-	deleted
Approech	Approach	e	-	substituted

Word document contains misspell words are corrected syntactically and semantically by implicitly using ABC Spelling and Grammar.

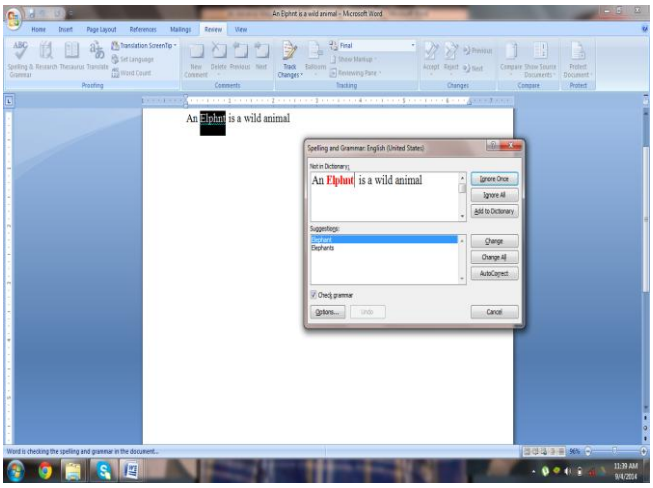

Figure1: Illustrates the correction of misspell word in word pad.

String transformation is handled by assorted approaches but they are not excellent in terms of accuracy. Levenshtein distance measure and log linear model are the traditional approaches but have their own disadvantages in terms of accuracy. Levenshtein distance measure is called Edit distance in which it performs edit operations such as insertion, deletion, substitution. This measure calculates the distance with all the keywords in the dictionary which takes more time. The other method is Log linear model. In this method input string is compared with output string with the occurrences of characters. This method does not compare the indexes of the characters. Accuracy is the important task in our paper. There are three important issues in String Transformation.

1. How accurately an approach is defined.
2. How accurately it is trained.
3. How to generate top K nearest neighbour output strings and selecting the best from them.

So, we proposed an advanced Precision based approach in String Transformation. In this approach each alphabet is allocated with unique precision value. Precision is nothing but giving some value to each alphabet. The aggregation of precision of each letter in the misspelled word called the Total Precision (TP) is calculated. Data sets are trained by calculating the total precisions of the words and validating through the dictionary called database. Now the TP of misspelled word is compared with all the TP's of the words in the Database. The System generates the top k nearest neighbour output strings which are nearer to the TP of Misspell Word. Accuracy is very high when compared to existing methods because 1.Precision based approach for string transformation. 2. An Excellent algorithm for training.

Here the Accuracy is shown by comparing our proposed method with the existing method in the form of graph representation.

II. THE RESEARCH METHOD

String Transformation Plays a very prominent role in different applications such as Knowledge Engineering, Computational molecular biology, Information retrieval. In Natural language Processing, Misspell word correction, Word Metamorphosis and Word transition referred as String transformation. The main theme of this paper is to provide Accuracy when differentiated with traditional methods.

P.M.Malarvizhi et al[12] submitted a report by contemplating the distinct candidate generator methods of string transformation. Generative models, logistic regression models and Discriminative models are briefly explained. The models are compared with their advantages and disadvantages in a tabular format. The author suggested Discriminative models are good when contrasted with other models in terms of accuracy. Angel Wills et al[14] proposed string transformation by a probabilistic approach. The approach uses the log linear model. The author used the model to solve the Correction of spelling errors and reformulation of queries in the web. Feedback of search engine is given by user which is maintained in the query log called anchor text. Jeyalakshmi.S et al[11] put forwarded a model called log linear model applied on large data sets to improve the efficiency and accuracy in String transformation. Commentz Walter Algorithm is used here for generating the top K output strings.

Akshay Kumar N. et al[10] explained the string transformation by not given importance to a dictionary. Levenstein algorithm is used for the calculation of edit distance between the source and target string and training is given by using log linear model. Data structures are used for generation of strings. Top k pruning is applied to generate the output strings. Eric Brill et al[5] predicted a model which is improvement in noisy channel for the correction of spellings. The author considers generic string to string edits. Naoaki Okazaki et al[8] presented a Discriminative model for generating the candidates in String Transformations. They used L1-regularised logistic regression model in which substitution of substring is done. They also implemented a method to produce negative instances which influence the decision boundary. This author experimentally improved three aspects by using the model such as orthographic variants, derivation of nouns and lemmatization. Simon J.Greenhill et al[9] explained the failure of Levenshtein Distances. Method contains edit operations such as insertion, deletion, substitution which are performed on misspelled word in order to get correct spelling of the word. But the author predicted that edit distance was failed to determine language relationships accurately. It failed while applying at different languages.

RoyLowrance et al[1] extended the edit operations by deriving an algorithm for the String Correction . Their algorithm included new edit operation that is transposition of two adjacent characters under some restrictions. Andres Marzal et al[3] implemented a model named Normalized Edit distance. There are some problems with Levenshtein distance, Edit distance. To solve the problems Normalized Edit Distance came into existence. Graham Cormode et al[6] proposed the algorithm named deterministic near linear time for string edit distance matching problem with moves. The algorithm was known for the first significant sub quadratic for the above problem where it includes distance with nontrivial alignments. They used a parsing technique called Edit Sensitive Parsing (ESP) where strings are embedded into L1 vector space in order to get the results. William J.Masek et al[2] determined an algorithm for evaluating the shortest edit distance between two strings. The method has a disadvantage such that it would not work when alphabets are infinite.

Mohammad Ali Elmi et al[4] determined the extension of three way match algorithm which solves the word boundary failures and Abbreviations. Large lexicons can be solved by this algorithm. But they do not considered a dictionary for the database. Arvind Arasu et al[7] propounded a method in which transformation based framework had done to capture the string variations such as antonyms and Abbreviations. They are the first to put forward this Programmatic framework of matching the record in which input is user defined String Transformations. This is excellent with rich variations to handle the strings and solved many computational challenges. Xing Yi et al[13] developed an algorithm for spelling correction in which it integrates trusted domain knowledge and query log information in order

to correct the spelling of query. Their algorithm utilizes query reformulations from query log and bigram language taken from queries producing correct suggestions and given ranks for valid corrections.

III. METHODOLOGY

Users may type mistakes while searching data in the Web search. To prevent such problems we proposed an approach named Advanced Precision based Approach of String Transformation which is based on three steps.

1. Calculating the Total Precision of the misspell word.
2. Precision based nearest neighbour words Generation
3. Best nearest neighbour word selection.

3.1 Calculating the Total Precision of the misspell word

Precision is nothing but a unique value. In our Paper we consider the alphabets such that we give Precisions to the each letter. Now we calculate the Total Precision of the misspell word by taking the each letter Precision. Aggregated the each letter precision in order to get the Total Precision of the misspell word.

```
Input: Character c
Output: Precision
Step 1: Start
Step 2: return findKey(List,c);
Step 3: End.
```

Algorithm 1: Calculating the Total Precision of the Misspell word

Algorithm1 describes the calculation of the precision of the particular word. Here the input is character and the output is Precision of the letter.

3.2 Precision based nearest neighbour words Generation

Here, Input string is taken and set of operations are performed to get the top k nearest neighbour output strings. The input string comprises words, tokens, group of letters. Set of operations depends on Precision Summation. Precision is the unique value given to the alphabets. Each Alphabet is allotted with Precision value. Now Input word that is misspelled word is considered and each precision of alphabet is taken and we perform summation such that we get the Total Precision (TP). Then we consider data sets which are trained by the Precisions. These data sets are validated through dictionary. Now we compare the Misspelled word TP with the data sets referring to dictionary called our database. The System retrieves all the nearest neighbour words which are related to the Misspelled TP from the database in the linear order. These are the top K nearest neighbour output strings which are generated by using Precision. Similarly we calculate and retrieve the sentences. Sentence is the collection of words. So, here we check the words of the input sentence are error free or not. If it is not error free we calculate each precision of the word by precision approach and make them error free. We retrieve the sentences which are nearer to input sentence total precision value by validating the words of sentence through sentence.

```
Input: topK[],cp
Output: topK[]
Step 1: Start
Step2:
        lc1 =  cp+findPrecesion(x);
        lc2 =  cp+findPrecesion(x++);
         If(lc1<lc2){
            Return topK[] + x;
            }else{
            Return topK[] + (x++);
         }
Step 3: End.
```

Algorithm 2: Precision based linear keywords generation

Algorithm2 describes the Precision based nearest neighbour words generation. After finding the Precision of the misspell word, by using this Algorithm we compare the Precision of the misspell word with all the words in our database in which we trained them by precision approach. Now we retrieve the top k nearest neighbours words related to the misspell precision.

3.3 Best nearest neighbour word selection

After Generating all the Top K output strings, Best keyword selection is done by considering the Exactly nearest TP value of the misspelled word with the TP of the words in our database in which we have trained with the Precision approach.

```
Input: Random Size String S={c1, c2,c3, …….. cn}
Output: Transformed String
Step 1: Start
Step 2: topK=null;
Step3:
        foreach i in size(s)
                c = s[i];
                t = findPrecesion(c);
                cp = cp + t;
                topK[] = findKneighbours(topK[],cp);
        end.
Step 3: End.
```

Algorithm3: Generating the best nearest neighbour word.

Algorithm3 illustrates the best nearest neighbour form the top k nearest neighbour words. After generating the top k nearest neighbour words we take the word which is exactly nearest to the misspell word precision in linear order.

IV. EXPERIMENTAL RESULTS

Table2 illustrates the Outcomes of the Precision based keyword generation of misspell word. Here the input word is "sqmple" which is typed wrongly. We calculate the Total Precision of the input by taking the precision of each letter. By considering the Total Precision of the input as 82, we are retrieving the nearest neighbour words to misspell word precision 82 such as 'sample' as precision as '66', 'simple' as precision as '74' in linear order we get 'sample' word which has precision '66' for the misspell word 'sqmple'.

Table 2. Outcomes of precision based keywords generation of misspell word.

Input	s	q	M	P	L	e	Tot
Precession	19	17	13	16	12	5	82
KN1	S	a	M	P	L	e	
	19	1	13	16	12	5	66
KN2	S	i	M	P	L	e	
	19	9	13	16	12	5	74

Table3 describes the Outcomes of the Precision based sentence generation. Here the input is sentence contains the words. We calculate each and every word's Precision of the sentence and aggregate all the words to get the total precision. We calculate the total precision of the input sentence and we retrieve all the sentences based on the total precision of input sentence and validated through the dictionary. For example we consider a sentences "Why you come late" and "why you came late". For 1st sentence we got Precision as 203 and second sentence we got precision as 217. The System displays the first sentence.

Table 3.Outcomes of Precision based sentence generation

Sentence												**TP**
w	h	Y		y	O	u		C	a	m	e	
23	8	25	32	25	15	21	32	3	1	13	5	203
W	h	y		Y	o	U		c	O	m	e	
23	8	25	32	25	15	21	32	3	15	13	5	217

Table4 shows the Top k Outcomes of string transformation by precision approach. Considered the input as length of the word , input word and retrieved all top k nearest neighbour strings relevant to Input string along with their precisions and validated through dictionary. Here we consider the input as Abas with length 4 and input as Car with length 5. Retrieved the top k nearest neighbour output strings for the inputs with their precisions.

Table 4. Top k Outcomes of string transformation by precision approach

S. no	**Length**	**Input**	**Top K Strings**	**TOP K Nearest Neighbours**
1	4	Abas	Abase Abash Abasia	1,2,1,19,5 1,2,1,19,8 1,2,1,19,9,1
2	3	Car	Car Carbonado Carmine Carnage Carnivorous	3,1,18 3,1,18,2,15,14,1,4,15 3,1,18,13,9,14,5 3,1,18,14,1,7,5 3,1.18,14,9,22,15,18, 15,21,19

V. CONCLUSION

String Transformation plays a major role in many applications. Users may perform spelling mistakes while surfing the web. To prevent such problems we are proposing an advanced based precision approach to String Transformation. Misspell word and sentence correction is done by this method. Data sets are trained by using this approach. Accuracy is very high when compared to traditional methods. Accuracy of our approach is shown in the form of graph representation.

REFERENCES

1. Wagner, Robert A., and Roy Lowrance. "An extension of the string-to-string correction problem." *Journal of the ACM (JACM)* 22.2 (1975): 177-183.
2. Masek, William J., and Michael S. Paterson. "A faster algorithm computing string edit distances." *Journal of Computer and System sciences* 20.1 (1980): 18-31.
3. Marzal, Andres, and Enrique Vidal. "Computation of normalized edit distance and applications." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 15.9 (1993): 926-932.
4. Elmi, Mohammad Ali, and Martha Evens. "Spelling correction using context."*Proceedings of the 17th international conference on Computational linguistics-Volume 1*. Association for Computational Linguistics, 1998.
5. Brill, Eric, and Robert C. Moore. "An improved error model for noisy channel spelling correction." *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2000.
6. Cormode, Graham, and S. Muthukrishnan. "The string edit distance matching problem with moves." *ACM Transactions on Algorithms (TALG)* 3.1 (2007): 2.
7. Arasu, Arvind, Surajit Chaudhuri, and Raghav Kaushik. "Transformation-based framework for record matching." *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*. IEEE, 2008.
8. Okazaki, Naoaki, et al. "A discriminative candidate generator for string transformations." *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2008.
9. Greenhill, Simon J. "Levenshtein distances fail to identify language relationships accurately." *Computational Linguistics* 37.4 (2011): 689-698.
10. Kumar, Akshay. "A Log Linear Probabilistic Model for String Transformation Using Non Dictionary Approach." *International Journal of Innovative Research and Development* 3.5 (2014).
11. Jeyalakshmi.S et al. "Improving Efficiency and Accuracy in String Transformation on Large Data Sets."*International Journal of Computer Science and Mobile Applications Vol.2 Issue.3, March-2014, pg.55-65 ISSN:2321-8363*
12. Malarvizhi, Mrs P., and Mrs S. Mohana. "A Survey on Various Candidate Generator Methods for Efficient String Transformation."
13. Yi, Xing, Henry Feild, and James Allan. "Spelling Correction Based on User Search Contextual Analysis and Domain Knowledge."
14. Angel Wills and D.F.Jenolin Flora. "Approach For Query Reformulation Using Log Linear Model."*International Journal of Research in Engineering and Bioscience*