Survey Paper on Effective Web Search Through String Transformation

Prof. S. R. Durugkar[1], Miss. Dipika L Tidke[2]

Assistant Professor[1], P.G. student[2]
Department of Computer Engineering
Late G.N.Sapkal College of Engineering, Anjaneri, Nasik [1, 2]
University Of Pune

Abstract— as in today's world we are dealing with data mining, information retrieval and many more fields in which need arises to go for natural language processing. In which actually what we are doing is nothing but transforming the query or we can say the given string as a input.

Once the string is given as a input the system or application generates the 'n' most suitable output strings in response to that string or query. Meaning is that system will generate the suitable 'n' strings – combinations of probable strings.

In this survey paper we are focusing on the few aspects of the natural language processing and string or query transformation.

We will also try focus on the proposed system at the end of this paper.

In this approach we will show how a system a can generate the 'n' suitable combinations and how those will be perfect and efficient. Means we will try to focus on the systems accuracy and efficiency through this novel approach for retrieving the information.

We can go for pruning strategy which will guarantee to generate the 'n' suitable string or queries. The proposed method is applied to correction of spelling errors in queries as well as reformulation of queries in web search. Experimental results on large scale data show that the proposed approach is very accurate and efficient improving upon existing methods in terms of accuracy and efficiency in different settings.

Keywords—
String Transformation, Log Linear Model, Spelling Check, Query Reformulation

I. INTRODUCTION TO NATURAL LANGUAGE PROCESSING

Natural language processing is a field of computer science, artificial intelligence, and linguistics related with the connections between computers and natural languages. As such, NLP is related to the area of human–computer interaction. Main task is to understand the natural language first of all , then need arises to process it to achieve the desired output & enabling computer systems to obtain meaning from human or natural language input.

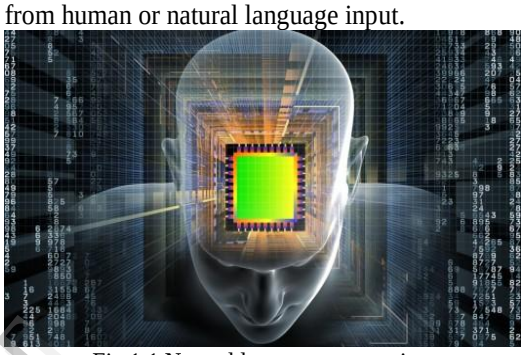

Fig 1.1 Natural language processing

Following are the few tasks which can be consider under this NLP (Natural language Processing):

1. **Automatic summarization**: as its name suggests it can produce a human readable outline of a given text. Frequently used to provide outline of text of a known type, such as articles in the sports section of a newspaper.
2. **Machine translation**: Automatically translate text from one human language to another. This is one of the most difficult problems, and is a member of a class of problems colloquially termed "AI-complete", i.e. requiring all of the different types of knowledge that humans possess (grammar, semantics, facts about the real world, etc.) in order to solve properly.
3. **Relationship extraction**: Given a of input text, recognize the relationships among named entities which will definitely helps to find the more relevant data or records immediately which will increases the effectiveness of system.
4. **Word segmentation**: Separating the continuous text into separate words. For a language like English, this is quite minor, since words are usually separated by spaces. However, some written languages like Chinese, Japanese do not mark word boundaries in such a fashion, and in those languages text segmentation is a significant task requiring

knowledge of the vocabulary and morphology of words in the language. Therefore need arises to focus on the word segmentation.

5. **Word sense disambiguation**: Many words have more than one meaning; therefore need arises to select the meaning which makes the most sense in context. For this problem, we are typically given a list of words and associated word senses; so we can prepare an index or a kind of dictionary which will help the end users.

II. SURVEY OF LITERATURE

2.1 String Transformation

It can be described as follows; if a given an input string is 'S' and a set of operators included in the string, we can transform the input string to the 'n' most probable output strings. Strings can be strings of words, characters, or any type of tokens.

Each operator is a transformation rule or semantics which can define the substitute of a substring with another substring. The probability of transformation can represent similarity, Relevance, and association between two strings in a specific application.

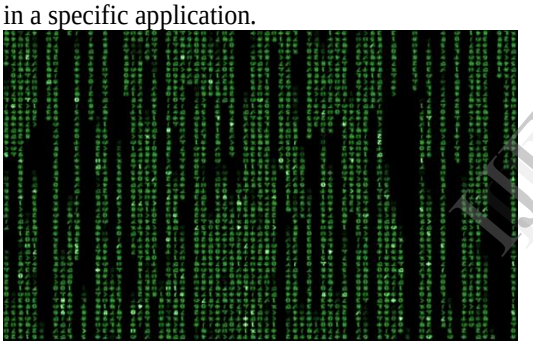

Fig 2.1 strings processed by computer system

Although advancement has been made in this area, further exploration of the task is still necessary nothing but future scope in this area is quite large & any one can do the advance research.

2.2 Approach towards string transformation

As we know we can prepare a dictionary for matching the given input string against it. When a dictionary is used, assumption is that output strings must exist in the given dictionary; it may happen that size of the dictionary may be large.

Fig 2.2 Dictionary

Another part which needs to be discussed is that we need to study correction of spelling errors in given first of all string which consists of characters need to be entered. Then next task to utilize a dictionary for finding the similar words or characters.

2.3 Spell Check

Correcting spelling errors in queries usually consists of two steps: candidate generation and candidate selection.

Fig 2.3 Spelling Check

Candidate generation is used to find the most likely corrections of a misspelled word from the dictionary.

In such a case, a string of characters is input and the operators represent insertion, deletion, and substitution of characters with or without surrounding characters, for example, "a"→ "e" and "lly" →"ly". Obviously candidate generation is an example of string transformation. Note that candidate generation is concerned with a single word; after candidate generation, the words in the context

2.4 Query Reformulation

Query reformulation in search is aimed at dealing with the term mismatch problem. For example, if the query is "TOI" 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 "TOI" to "Times of India" and thus make a better matching between the query and document. In the task, given a query one needs to generate all similar queries from the original query.

Query reformulation involves again writing the original query or string with its similar queries or words match with dictionary and enhancing the effectiveness of search. Most existing methods manage to mine transformation rules from pairs of queries in the search logs.

III. PROPOSED SYSTEM

After studying the various aspects of the string formation or even we can say the string reformation, proposed the following system which we feel will definitely give the effectiveness and accuracy.

This we can present with the help of following model.

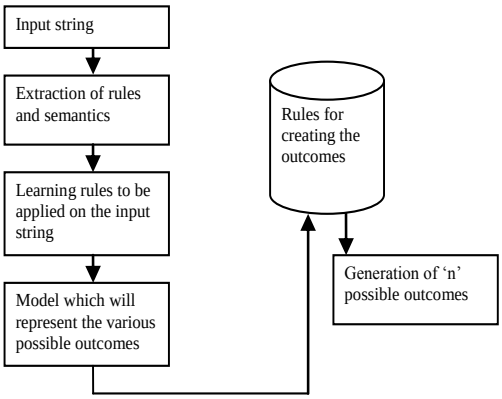

Fig 3.1 proposed basic model

Without beating of simplification, we presume that the maximum number of rules can be applicable to a input string predefined. As a result, the number of possible outcomes for an input string can be generated. 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.

3.1 Initial basic steps to be executed on the system

- Algorithm : Top 'n' Pruning
- Input: rule index which will specify the rules and semantics
- *Ir*, input string , candidate number 'n'
- Output: top 'n' output strings
- begin
- apply various rules applicable to input *string*
- calculate the *minscore*
- while check various possible paths
- do
- Pickup a possible
- if *score* < *minscore* then
- continue
- similarly check with all the possible outcomes
- if any candidate will have min score
- then Remove candidate with minimum score
- *Stop*

S**tring searching algorithms**, sometimes called **string matching algorithms**, are an important class of string algorithms those will try to find out a place where one or several strings are found within a larger string or text.

A search session while searching some information from web which comprises of a sequence of queries from the same user within a short time period. Many of search sessions in data consist of misspelled queries and their corrections.

When the algorithm reaches a node, it outputs all the dictionary entries that end at the current character position in the input text. This is done by printing every node reached by following the dictionary suffix links, starting from that node, and continuing until it reaches a node with no dictionary suffix link. In addition, the node itself is printed, if it is a dictionary entry.

Informally, the algorithm builds a finite state machine that look like a try 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.

When the pattern dictionary is known in advance (e.g. entries for detecting of virus in a computer), the construction of the machine can be performed once off-line and the compiled automaton stored for later use.

The complexity of the algorithm to be used will be linear in the length of the patterns which are present plus the length of the searched text or string plus the number of possible outcomes.

Here one thing to be noted is that as all matches are found, there can be a quadratic number of matches if every substring matches (e.g. dictionary = b, bb, bbb, bbbb and input string is bbbb).

3.2 Spell Check as an Obstacle

We need to focused on this definitely as it is related with the input string.

- **Ambiguity:** Like other languages, English spelling has never been systematically updated and therefore only partly holds to the alphabetic principle. As an result, in spelling there is a need to concentrate on weak rules with many exceptions and ambiguities.
- **Redundant letters**: alphabet may have several letters whose characteristic are already represented elsewhere in the alphabet. These include X (as in *ks*, C (hard as in K) and many more examples exists.
- **The spellings of some words** – such as *tongue* and *stomach* – are so unindicative of their pronunciation that changing the spelling would noticeably change the shape of the word.
- Likewise, the **irregular spelling** of very common words such as *is*, *are*, *have*, *done* and *of* makes it difficult to fix them without introducing a noticeable change to the

appearance of English text. This would create acceptance issues.

Misspelled	Correct	Misspelled	Correct
Reteive	Retrieve	Tabel	Table
Infomation	Informati on	Chevrole	Chevrolet
liyerature	liyerature	newpape	newspaper

in this way system should detect the various misspelled words and correct them accordingly.

3.3 Pruning on strings

Pruning is a method in machine learning that decreases the size of decision trees by removing sections of the tree that provide little power to classify instances.

The other objective of pruning is reducing the complexity of the final classifier as well as better predictive accuracy by the reduction of overfitting and removal of sections of a classifier that may be based on noisy or erroneous data.

Pruning of noisy data is important to fully automate the process of data collection and learning for object recognition.

The efforts in this way have been focused on creating complicated features, particularly with respect to the targeted area.

Our key idea is that automatic elimination of non relevant and possible outcomes can be done without incorporating very complicated steps. Therefore in this way we can reduce the 'n' possible outcomes at a certain level so that what will remain will be the perfect one.

3.4 Predicted Modules in proposed system

First module which handle the input string to be entered by the end user.

Second module which will immediately check whether the entered string is correct with respect to syntax and semantics.

Third module will suggest corrected spellings with respect to user's query or string.

Fourth module will retrieve the 'n' possible outcomes of user's query or string.

Fifth module will retrieve the relevant documents from the database which will satisfy the user's request.

These above mentioned modules will handle the overall system smoothly and effectively.

CONCLUSION

After discussing above mentioned points now we proposed our contribution what we wish to do in our proposed system.

As almost information retrieval application will require an internet connection in live state; what happens if sometimes that connection will not be available?

So our one proposed approach is to store those input strings and possible outcome with respect to that query.

Means even in offline mode an end user can find out the possible outcomes.

If that query or string is already executed on the system then our system will prompt a message to user that query or string is already executed.

Even for reducing the time required to form the possible outcomes we will mainly focus on the string entered by user. Meaning is that our system within a less time should check the spelling of that string. If it wrongs instead of warning user about the same , system should give correct spellings in drop down list immediately.

If the input string is large enough then also we are thinking on applying the algorithms to remove the frequently occurring word in that string.

So what will remain will be the important words or tokens. Then by using those tokens we will form 'n' possible outcomes.

REFERNCES

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. http://en.wikipedia.org/wiki/Pruning
3. http://en.wikipedia.org/wiki/English-language_spelling_reform
4. Anelia Angelova, Yaser Abu-Mostafa, Pietro Perona, "Pruning Training Sets for Learning of Object Categories"
5. A. R. Golding and D. Roth, "A winnow-based approach to context-sensitive spelling correction," *Mach. Learn.*, vol. 34, pp. 107–130, February 1999.
6. 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.