International Journal of Engineering Research in Computer Science and Engineering (IJERCSE) Vol 3, Issue 4, April 2016

Enhancing Scalable Reverse Dictionary Using Text Rank

[1] Shailesh Raskar [2] Saurabh Nagpal [3] Shubham Das [4] Sharif Sayyed
[1][2][3][4] Department of Computer Engineering, Rajarshi Shahu College of Engineering, Pune.
Savitribai Phule Pune University
[1] shaileshraskar2008@gmail.com, [2] nagpals903@gmail.com [3] das.shubham011@gmail.com,
[4] sharifsayyad0610@gmail.com

Abstract: **The huge availability of words in usage is really becoming a challenging task in finding the correct meaning of words. When people think that they could have defined a sentence or a situation much more concisely by using a single word or a smaller phrase. Sometimes people go on describing the situation or a feeling in long sentences. In such situation, the tool reverse dictionary is useful that it gives an appropriate word to one's sentence. A reverse dictionary is a dictionary which is organized in a non-alphabetical order that provides the user with information that would be difficult to obtain from a traditional alphabetical ordered dictionary. A traditional dictionary accepts input as a word and gives one or more definitions in terms of result. Thus, the concept of reverse dictionary is exactly opposite to that of traditional dictionary. It takes input as a sentence or a phrase describing a concept or situation, and returns a set of precise and appropriate words that satisfy the meaning of the input sentence or phrase. In this project, an implementation of reverse dictionary is proposed which helps the user to get appropriate words to the input phrases.**

***Keywords*: Dictionary thesauruses, Search Processes, web-based services**

I. INTRODUCTION

The reverse dictionary uses the concept of reverse mapping i.e., given a phrase describing a desired concept, it provides words whose definitions match the entered definition phrase. This is not the case in regular forward dictionary. A regular dictionary maps the words to their definitions. For example, a regular dictionary helps the user to get the meaning of the word "regret". The regular dictionary shows the meaning of input word as "to feel sorry." On other hand, the reverse dictionary, offers the user an opportunity to enter the phrase "feeling of loss or longing for someone" as input, and can expect the word "regret" and possibly other words with similar meanings as output. Most of the techniques for the creation of reverse dictionary is based on the creation of multiple databases for synonyms, hyponyms, antonyms etc. In reverse dictionary, the user entered phrase need not necessarily be the same as in the definition, therefore the implementation is done in such a way that the concept of the user input will be considered and corresponding words will be obtained as the outcome. The output results will be ranked according to the perfect similarity of the word to the input phrase to the least possible similarity of the search concept. The reverse dictionary identifies a concept or idea hidden behind the input words or phrases.

In a reverse dictionary, the user input is unlikely to match the definition that is already present. For example: when a user wants to know the word for the concept

"Inability to sleep". Whereas, the forward dictionary definition would be rather "sleeplessness during night". Therefore the user input phrase should be conceptually similar to the definitions but need not be exactly the same. This issue is addressed by the proposed reverse dictionary application by using the concept called as building the RMS (Reverse Mapping Set). According to the concept of RMS, the word which is found in the definition of another word, the latter is mapped to the former. Example, the forward dictionary definition of "insomnia" may be "Sleeplessness during night". Therefore RMS of sleeplessness, i.e., R (Sleeplessness) will be "insomnia." This concept can be achieved by considering the key words in the user input. So it is necessary to negate some common words and consider the important words such as nouns and verbs from user input.

For some cases, there are no enough words or less than the target number of words. In such cases the search can be extended using synonyms and hyponyms. When there are not enough words, the synonym of each word needs to be considered. Example: the synonym of "sleep" will be "relax". Therefore for word Relax there will be n number of words by using the concept of RMS. In this way search result can be expanded. Consideration of Antonyms plays a very important part in the Reverse dictionary. For example,

when a user enters "Cannot sleep", the negation word "Cannot" needs to be considered. Therefore the word "cannot" has the same meaning as "ness" in "sleeplessness".

II. RELATED WORKS

In a reversible dictionary input is phrase given by the user and output received is certain number of arguments as graded with algorithm. The works related to this reversible dictionary are the reverse dictionaries [1] which are developed with some drawbacks. Vector computation is known to be quite computing intensive. Hence the vectorization which is used in the current system, uses the LDA (Latent Difficult Association) and PCA(Principal Component Analysis) both analyze the keywords of documents in amount to identify the central concepts in the document. Accordingly these foremost concepts are representing as vectors in the keyword space and are used as the basis of similarity association for classification. Such Vectorization is highly calculating exhaustive operation, as is the evaluation of the distance between two vectors (using techniques like cosine distances).

The other related work to a reverse dictionary is the other existing features of a reverse dictionary where certain constraints are to be addressed. The two main constraints needs to be addressed. The first constraint is that, the user input will not match the one in the forward dictionary. Next, the response time needs to be similar to that of a forward. In this paper, we report the creation of the Reversible Dictionary using the Text Rank Approach, a database-driven Reverse Dictionary system that challenges to address the core problem identified above.

III. PROPOSED WORK

Our reverse dictionary system is based on the notion that a phrase that conceptually describes a word should resemble the word's actual definition, if not matching the exact words, then at least conceptually similar. Example: The word "sleep" is found in 4 meanings belonging to 4 words. Therefore sleep will be "slumber", "torpor", "nap", "rest". These words must be manually entered for each word. Words can be found from the WorldNet dictionary.

In our Reverse Dictionary, a user might input a phrase describing an unknown term of interest. Since an input phrase might potentially satisfy the definition of multiple words, a Reverse Dictionary should return a set of possible matches from which a user may select his/her choice of terms. However, the meaning of the phrase the user entered should be conceptually similar enough to an actual dictionary definition to generate a set of possible matches. For matching the words we are using Text Rank Algorithm

A. System Architecture

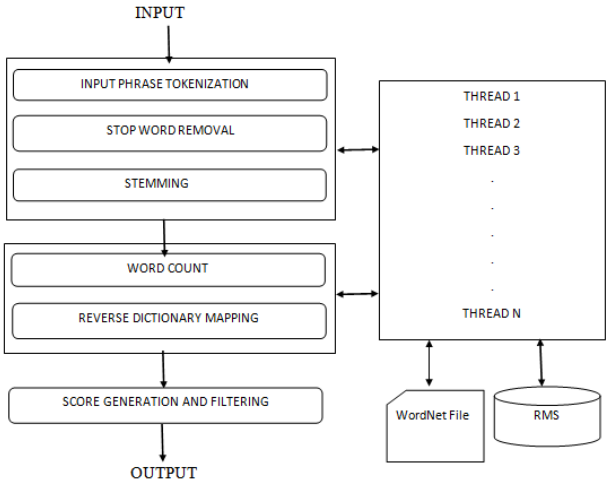

Fig 1. System Architecture

Fig. 1 presents the architecture of the system. The Reverse Dictionary Application (RDA) is a software module that takes a user phrase as input, and returns a set of conceptually related words as output.

RDA needs access to stored information in the databases:

- The RMS DB contains mapping between words and their meanings;
- The Synonym DB contains the synonym set for each word;
- The Hyponym/Hypernym DB contains the Hyponym and hypernym relationship between words;
- The Antonym DB, contains the antonym set for each word; and
- The actual dictionary definitions.

The mappings for the RMS, synonyms, hyponyms, Hyponyms and antonyms are stored as integer mappings, where each word in the Word Net dictionary is represented by a unique integer. This both solves the size of the mapping sets, and allows for fast processing of comparison for similarity, as compared to string processing.

When the user input is given, it is tokenized and preprocessing is done. In preprocessing, the stop words are removed using a standard lexical parser. Then stemming [2]

is performed on the words to improve precision for information retrieval. Antonyms are considered for the words starting with 'un', 'not', etc. for removing negation. Word count for the word occurrences, reverse mapping, and similarity checking of the sentence comes under Text Rank Algorithm [6].

B. Text Rank Algorithm

In proposed system steps for text Rank are:

- Separate the text into sentences based on trained model.
- Build a sparse matrix of words and count it. Sparse matrix is the matrix which is considered between the sentences and the words of the sentences.
- Normalize each word with tf-idf. This weightage is provided to each occurred word in the matrix.
- Construct similarity matrix. Obtained from the product of matrix in step 3 and its transpose, to get individual score for similarity.
- Use page rank for ranking.

After the text rank the words are indexed with a particular score. A specific threshold value is provided to the system which is used for filtering of words from the result set. The top five words are shown to the user. For providing better user satisfaction we are going to provide a link where user can find the proper meaning of the result word from the forward dictionary, making it user friendly and efficient.

IV. MATHEMATICAL MODEL

Let the system be x

x={I, O, S, F, C}

Where,

I= set of Inputs
O= set of outputs
S= set of outputs in success cases
F= set of outputs in failure cases
C= set of constraints

I= {W, P}
W= set of words
p=set of phrases

O= {W0}
W0=set of words

S={s}
s=set of words and synonyms relevant to input

F= {NULL, Wof}
NULL=no output
Wof=set of words not relevant to input

C={C1,C2}
C1="input containing only 1 word"
C2="input containing incorrect spelling"

V. CONCLUSION

Thus, the proposed approaches are to create a reversible dictionary from an existing forward dictionary using WordNet. Thus, the system works in developing a meaning-to-word dictionary. Depending on the phrase input there may be variation of the results shown. The system becomes efficient with the use of Text Rank instead of vector based system. With our feature of finding words from an abstract string given as an input empowers the concept of reverse dictionary. Parts of Speech classification feature also add quality. Improving the efficiency by holding the results makes the access fast. Adding new features like words searching enhances our work from previously available reverse dictionary.

REFERENCES

1. Ryan Shaw, Member, IEEE, Anindya Datta, Member, IEEE, Debra Vander Meer, Member, IEEE, and Kaushik Dutta, Member, IEEE, "Building a Scalable Database-Driven Reverse Dictionary, VOL. 25, NO. 3, MARCH 2013

2. J. Carlberger, H. Dalianis, M. Hassel, and O. Knutsson, "Improving Precision in Information Retrieval for Swedish Using Stemming," ,Aug. 2001.

3. H. Cui, R. Sun, K. Li, M.-Y. Kan, and T.-S. Chua, "Question Answering Passage Retrieval Using Dependency Relations", pp. 400-407, 2005.

4. J. Kim and K. Candan, "Cp/cv: Concept Similarity Mining without Frequency Information from Domain Describing Taxonomies," Proc. ACM Conf. Information and Knowledge Management,2006.

5. E. Gabrilovich and S. Markovitch, "Wikipedia-Based Semantic Interpretation for Natural Language Processing," J. Artificial Intelligence Research, vol. 34, no. 1, pp. 443-498, 2009

6. Rada Mihalcea and Paul Tarau, "Text Rank: Bringing Order into Texts".