ISSN:0975-9646

A Web Search Engine Application to Compare Semantic Equivalence between Two Words

Mallela Haribabu
M.Tech Student,
Dept of CSE,AIET

H.Devaraj
Asst. Professor,
Dept of CSE, AIET

Y.Ramesh Kumar
Asst. Professor,
Dept of CSE, AIET

Abstract: Measuring the semantic similarity between two words is an important component in various tasks on the web such as relation extraction, community mining, document clustering, and automatic meta-data extraction. Despite the usefulness of semantic similarity measures in these applications, accurately measuring semantic similarity between two words (or entities) remains a challenging task. We propose an empirical method to estimate semantic similarity using page counts and text snippets retrieved from a web search engine for two words. Specifically, we define various word co-occurrence measures using page counts and integrate those with lexical patterns extracted from text snippets. To identify the numerous semantic relations that exist between two given words, we propose a novel pattern extraction algorithm and a pattern clustering algorithm. The optimal combination of page counts-based co-occurrence measures and lexical pattern clusters is learned using support vector machines. The proposed method outperforms various baselines and previously proposed web-based semantic similarity measures on three benchmark data sets showing a high correlation with human ratings. Moreover, the proposed method significantly improves the accuracy in a community mining task.

Keywords: Web Mining, Information Extraction, Web Text Analysis

1. INTRODUCTION

Accurately measuring the semantic similarity between words is an important problem in web mining, information retrieval, and natural language processing. Web mining applications such as, community extraction, relation detection, and entity disambiguation, require the ability to accurately measure the semantic similarity between concepts or entities. In information retrieval, one of the main problems is to retrieve a set of documents that is semantically related to a given user query. Efficient estimation of semantic similarity between words is critical for various natural language processing tasks such as word sense disambiguation (WSD), textual entailment, and automatic text summarisation.

Semantically related words of a particular word are listed in manually created general-purpose lexical ontologies such as WordNet.1 In WordNet, a synset contains a set of synonymous words for a particular sense of a word. However, semantic similarity between entities changes over time and across domains. For example, apple is frequently associated with computers on the web. However, this sense of apple is not listed in most general-purpose thesauri or dictionaries. A user who searches for apple on the web, might be interested in this sense of apple and not apple as a fruit. New words are constantly being created as well as new senses are assigned to existing words. Manually maintaining ontology's to capture these new words and senses is costly if not impossible.

We propose an automatic method to estimate the semantic similarity between words or entities using web search engines. Because of the vastly numerous documents and the high growth rate of the web, it is time consuming to analyze each document separately. Web search engines provide an efficient interface to this vast information. Page counts and snippets are two useful information sources provided by most web search engines. Page count of a query is an estimate of the number of pages that contain the query words. In general, page count may not necessarily be equal to the word frequency because the queried word might appear many times on one page. Page count for the query P AND Q can be considered as a global measure of co occurrence of words P and Q. For example, the page count of the query "apple" AND "computer" in Google is 288,000,000, whereas the same for "banana" AND "computer" is only 3,590,000. The more than 80 times more numerous page counts for "apple" AND "computer" indicate that apple is more semantically similar to computer than is banana. Despite its simplicity, using page counts alone as a measure of co-occurrence of two words presents several drawbacks. First, page count analysis ignores the position of a wordin a page. Therefore, even though two words appear in a page, they might not be actually related. Second, page count of a polysemous word (a word with multiple senses) might contain a combination of all its senses. For example, page counts for apple contain page counts for apple as a fruit and apple as a company. Moreover, given the scale and noise on the web, some words might co-occur on some pages without being actually related [1]. For those reasons, page counts alone are unreliable when measuring semantic similarity.

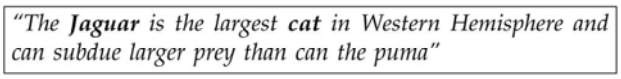

Fig. 1. A snippet retrieved for the query Jaguar AND cat.

Snippets, a brief window of text extracted by a search engine around the query term in a document, provide useful information regarding the local context of the query term. Semantic similarity measures defined over snippets, have been used in query expansion [2], personal name disambiguation [3], and community mining [4]. Processing snippets is also efficient because it obviates the trouble of downloading webpages, which might be time consuming depending on the size of the pages. However, a widely acknowledged drawback of using snippets is that, because

of the huge scale of the web and the large number of documents in the result set, only those snippets for the top ranking results for a query can be processed efficiently. Ranking of search results, hence snippets, is determined by a complex combination of various factors unique to the underlying search engine. Therefore, no guarantee exists that all the information we need to measure semantic similarity between a given pair of words is contained in the top-ranking snippets. We propose a method that considers both page counts and lexical syntactic patterns extracted from snippets that we show experimentally to overcome the above mentioned problems. For example, let us consider the snippet shown in Fig. 1 retrieved from Google for the query Jaguar AND cat. Here, the phrase is the largest indicates a hypernymic relationship between Jaguar and cat. Phrases such as also known as, is a, part of, is an example of all indicate various semantic relations. Such indicative phrases have been applied to numerous tasks with good results, such as hypernym extraction [5] and fact extraction [6]. From the previous example, we form the pattern X is the largest Y, where we replace the two words Jaguar and cat by two variables X and Y.

Given taxonomy of words, a straightforward method to calculate similarity between two words is to find the length of the shortest path connecting the two words in the taxonomy [7]. If a word is polysemous, then multiple paths might exist between the two words. In such cases, only the shortest path between any two senses of the words is considered for calculating similarity. A problem that is frequently acknowledged with this approach is that it relies on the notion that all links in the taxonomy represent a uniform distance. Resnik [8] proposed a similarity measure using information content. He defined the similarity between two concepts C1 and C2 in the taxonomy as the maximum of the information content of all concepts C that subsume both C1 and C2. Then, the similarity between two words is defined as the maximum of the similarity between any concepts that the words belong to. He used WordNet as the taxonomy; information content is calculated using the Brown corpus.

Li et al. [9] combined structural semantic information from a lexical taxonomy and information content from a corpus in a nonlinear model. They proposed a similarity measure that uses shortest path length, depth, and local density in a taxonomy. Their experiments reported a high Pearson correlation coefficient of 0.8914 on the Miller and Charles [10] benchmark data set. They did not evaluate their method in terms of similarities among named entities. Lin [11] defined the similarity between two concepts as the information that is in common to both concepts and the information contained in each individual concept. Cilibrasi and Vitanyi [12] proposed a distance metric between words using only page counts retrieved from a web search engine. The proposed metric is named Normalized Google Distance (NGD) and is given by

$$NGD(P,Q) = \frac{\max\{\log H(P), \log H(Q)\} - \log H(P,Q)}{\log N - \min\{\log H(P), \log H(Q)\}}.$$

2. METHODOLOGY

Given two words P and Q, we model the problem of measuring the semantic similarity between P and Q, as a one of constructing a function simð(P;QÞ) that returns a value in range ½0; 1_. If P and Q are highly similar (e.g., synonyms), we expect simð(P;QÞ) to be closer to 1. On the other hand, if P and Q are not semantically similar, then we expect simð(P;QÞ) to be closer to 0. We define numerous features that express the similarity between P and Q using page counts and snippets retrieved from a web search engine for the two words. Using this feature representation of words, we train a two-class support vector machine to classify synonymous and non-synonymous word pairs. The function simð(P;QÞ) is then approximated by the confidence score of the trained SVM.

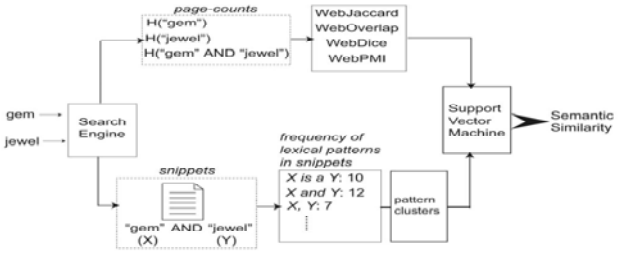

Fig. 2 illustrates an example of using the proposed method

Fig. 2 illustrates an example of using the proposed method to compute the semantic similarity between two words, gem and jewel. First, we query a web search engine and retrieve page counts for the two words and for their conjunctive (i.e., "gem," "jewel," and "gem AND jewel"). In Section 3.2, we define four similarity scores using page counts. Page counts-based similarity scores consider the global co-occurrences of two words on the web. However, they do not consider the local context in which two words co-occur. On the other hand, snippets returned by a search engine represent the local context in which two words co-occur on the web. Consequently, we find the frequency of numerous lexical syntactic patterns in snippets returned for the conjunctive query of the two words. The lexical patterns we utilize are extracted automatically using the method described in Section 3.3. However, it is noteworthy that a semantic relation can be expressed using more than one lexical pattern. Grouping the different lexical patterns that convey the same semantic relation, enables us to represent a semantic relation between two words accurately. For this purpose, we propose a sequential pattern clustering algorithm in Section 3.4. Both page counts-based similarity scores and lexical pattern clusters are used to define various features that represent the relation between two words.

PAGE COUNT-BASED CO-OCCURENCE MEASURES :

Page counts for the query P AND Q can be considered as an approximation of co-occurrence of two words (or multiword phrases) P and Q on the web. However, page counts for the query P AND Q alone do not accurately express semantic similarity. For example, Google returns 11,300,000 as the page count for "car" AND "automobile,"

whereas the same is 49,000,000 for "car" AND "apple." Although, automobile is more semantically similar to car than apple is, page counts for the query "car" AND "apple" are more than four times greater than those for the query "car" AND "automobile." One must consider the page counts not just for the query P AND Q, but also for the individual words P and Q to assess semantic similarity between P and Q. We compute four popular co-occurrence measures; Jaccard, Overlap Simpson), Dice, and Point wise mutual information (PMI), to compute semantic similarity using page counts.

For the remainder of this paper, we use the notation H(P) to denote the page counts for the query P in a search engine. The WebJaccard coefficient between words (or multiword phrases) P and Q, WebJaccard(P,Q), is defined as

$$\text{WebJaccard}(P,Q) = \begin{cases} 0, & \text{if } H(P \cap Q) \leq c, \\ \dfrac{H(P \cap Q)}{H(P) + H(Q) - H(P \cap Q)}, & \text{otherwise.} \end{cases}$$

Therein, $P \cap Q$ denotes the conjunction query P AND Q. Given the scale and noise in web data, it is possible that two words may appear on some pages even though they are not related. In order to reduce the adverse effects attributable to such co-occurrences, we set the WebJaccard coefficient to zero if the page count for the query $P \cap Q$ is less than a threshold $c.^2$ Similarly, we define WebOverlap, WebOverlap(P,Q), as

$$\text{WebOverlap}(P,Q) = \begin{cases} 0, & \text{if } H(P \cap Q) \leq c, \\ \dfrac{H(P \cap Q)}{\min(H(P), H(Q))}, & \text{otherwise.} \end{cases}$$

WebOverlap is a natural modification to the Overlap (Simpson) coefficient. We define the WebDice coefficient as a variant of the Dice coefficient. WebDice(P,Q) is defined as

$$\text{WebDice}(P,Q) = \begin{cases} 0, & \text{if } H(P \cap Q) \leq c, \\ \dfrac{2H(P \cap Q)}{H(P) + H(Q)}, & \text{otherwise.} \end{cases}$$

Point wise mutual information [20] is a measure that is motivated by information theory; it is intended to reflect the dependence between two probabilistic events. We define WebPMI as a variant form of point wise mutual information using page counts as

$$\text{WebPMI}(P,Q) = \begin{cases} 0, & \text{if } H(P \cap Q) \leq c, \\ \log_2 \left(\dfrac{\frac{H(P \cap Q)}{N}}{\frac{H(P)}{N} \frac{H(Q)}{N}} \right), & \text{otherwise.} \end{cases}$$

Here, N is the number of documents indexed by the search engine. Probabilities in (4) are estimated according to the maximum likelihood principle. To calculate PMI accurately using (4), we must know N, the number of documents indexed by the search engine. Although estimating the number of documents indexed by a search engine [21] is an interesting task itself, it is beyond the scope of this work. In the present work, we set N ¼ 1010 according to the number of indexed pages reported by Google. As previously discussed, page counts are mere approximations to actual word co-occurrences in the web. However, it has been shown empirically that there exists a high correlation between word counts obtained from a web search engine (e.g., Google and Altavista) and that from a corpus (e.g., British National corpus) [22]. Moreover, the approximated page counts have been successfully used to improve a variety of language modelling tasks.

LEXICAL PATTERN EXTRACTION:

Page counts-based co-occurrence measures described in Section 3.2 do not consider the local context in which those words co-occur. This can be problematic if one or both words are polysemous, or when page counts are unreliable. On the other hand, the snippets returned by a search engine for the conjunctive query of two words provide useful clues related to the semantic relations that exist between two words. A snippet contains a window of text selected from a document that includes the queried words. Snippets are useful for search because, most of the time, a user can read the snippet and decide whether a particular search result is relevant, without even opening the url. Using snippets as contexts is also computationally efficient because it obviates the need to download the source documents from the web, which can be time consuming if a document is large. For example, consider the snippet in Fig. 3. Here, the phrase is a indicates a semantic relationship between cricket and sport. Many such phrases indicate semantic relationships. For example, also known as, is a, part of, is an example of all indicate semantic relations of different types. In the example given above, words indicating the semantic relation between cricket and sport appear between the query words. Replacing the query words by variables X and Y , we can form the pattern X is a Y from the example given above. Despite the efficiency of using snippets, they pose two main challenges: first, a snippet can be a fragmented sentence, second, a search engine might produce a snippet by selecting multiple text fragments from different portions in a document. Because most syntactic or dependency parsers assume complete sentences as the input, deep parsing of snippets produces incorrect results. Consequently, we propose a shallow lexical pattern extraction algorithm using web snippets, to recognize the semantic relations that exist between two words. Lexical syntactic patterns have been used in various natural language processing tasks such as extracting hypernyms [5], [24], or meronyms [25], question answering [26], and paraphrase extraction [27]. Although a search engine might produce a snippet by selecting multiple text fragments from different portions in a document, a predefined delimiter is used to separate the different fragments. For example, in Google, the delimiter "..." is used to separate different fragments in a snippet. We use such delimiters to split a snippet before we run the proposed lexical pattern extraction algorithm on each fragment

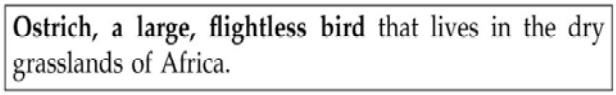

Fig.3. A snippet retrieved for the query "ostrich _ _ bird."

Given two words P and Q, we query a web search engine using the wildcard query "P***** Q" and download snippets. The "_" operator matches one word or none in a webpage. Therefore, our wildcard query retrieves snippets in which P and Q appear within a window of seven words. Because a search engine snippet contains ca. 20 words on average, and includes two fragments of texts selected from a document, we assume that the seven word window is sufficient to cover most relations between two words in snippets. In fact, over 95 percent of the lexical patterns extracted by the proposed method contain less than five words. We attempt to approximate the local context of two words using wildcard queries. For example, Fig. 4 shows a snippet retrieved for the query "ostrich***** bird." For a snippet _, retrieved for a word pair H(P,Q), first, we replace the two words P and Q, respectively, with two variables X and Y . We replace all numeric values by D, a marker for digits. Next, we generate all subsequences of words from that satisfy all of the following conditions:

- A subsequence must contain exactly one occurrence of each X and Y .
- The maximum length of a subsequence is L words.
- A subsequence is allowed to skip one or more words. However, we do not skip more than g number of words consecutively. Moreover, the total number of words skipped in a subsequence should not exceed G.
- We expand all negation contractions in a context. For example, didn't is expanded to did not. We do not skip the word not when generating subsequences. For example, this condition ensures that from the snippet X is not a Y, we do not produce the subsequence X is a Y.
- Finally, we count the frequency of all generated subsequences and only use subsequences that occur more than T times as lexical patterns.

LEXICAL FREQUENCY PATTERN CLUSTERING :

The basic idea of Lexical Frequency Pattern Clustering (LFPC) consists in finding all the sequential patterns in a data structure, which is built from the document database (DDB). The data structure stores all the different pairs of contiguous words that appear in the documents, without losing their sequential order. Given a threshold β specified by the user, LFPC reviews if a pair is β - frequent. In this case, LFPC grows the sequence in order to determine all the possible maximal sequential patterns containing such pair as a prefix. A possible maximal sequential pattern (PMSP) will be a maximal sequential pattern (MSP) if it is not a subsequence of any previous MSP. This implies that all the stored MSP which are subsequence of the new PMSP must be deleted. The proposed algorithm is composed of three steps described as follows:

In the first step, for each different word (item) in the DDB, LFPC assigns an integer number as identifier. Also, for each identifier, the frequency is stored, i.e., the number of documents where it appears. These identifiers are used in the algorithm instead of the words. Table 1 shows an example for a DDB containing 4 documents.

In the second step (Fig. 1), LFPC builds a data structure from the DDB storing all the pairs of contiguous words <wi,wi+1> that appear in a document and some additional information to preserve the sequential order. The data structure is an array which contains in each cell a pair of words C=<wi,wi+1>, the frequency of the pair (C_f), a Boolean mark and a list Δ of nodes.

We present an automatically extracted lexical syntactic patterns-based approach to compute the semantic similarity between words or entities using text snippets retrieved from a web search engine. We propose a lexical pattern extraction algorithm that considers word subsequences in text snippets. Moreover, the extracted set of patterns are clustered to identify the different patterns that describe the same semantic relation.

We integrate different web-based similarity measures using a machine learning approach. We extract synonymous word pairs from WordNet synsets as positive training instances and automatically generate negative training instances. We then train a two-class support vector machine (SVM) to classify synonymous and no synonymous word pairs. The integrated measure outperforms all existing web based semantic similarity measures on a benchmark data set.

We apply the proposed semantic similarity measure to identify relations between entities, in particular people, in a community extraction task. In this experiment, the proposed method outperforms the baselines with statistically significant precision and recall values. The results of the community mining task show the ability of the proposed method to measure the semantic similarity between not only words, but also between named entities, for which manually created lexical ontologies do not exist or incomplete.

3. DATA ANALYSIS

First we were considered "apple" word for analysis as shown in figure4.

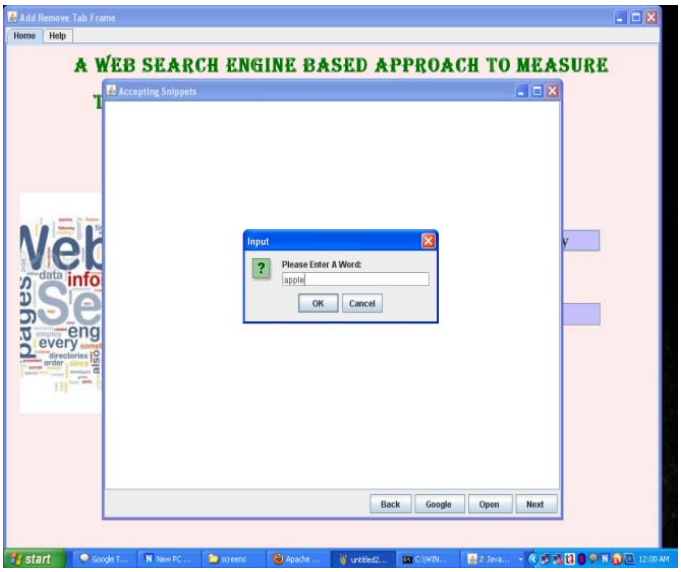

Figure4

We were inserted the a sample snipplet regarding apple as shown in fig5.

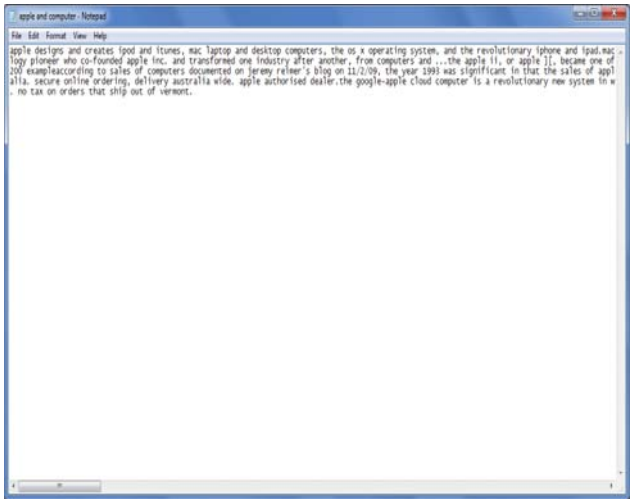

Figure5

We are accepted the snipplets for apple,computer and apple as shown in fig6&7.

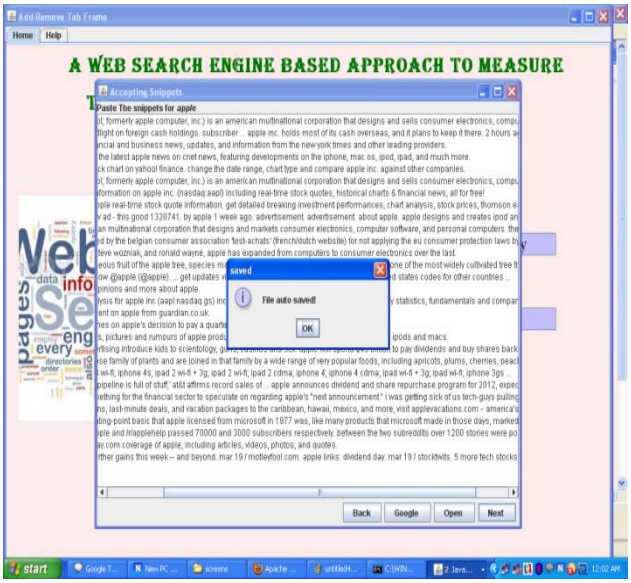

Figure6

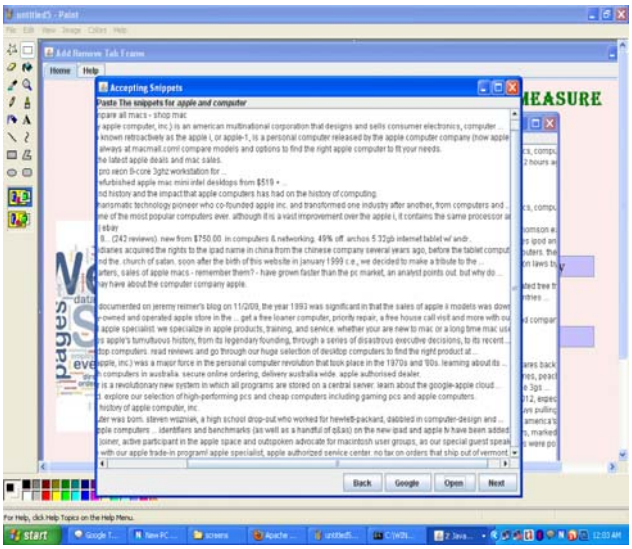

Figure 7

Page count measures are shown in fig8.

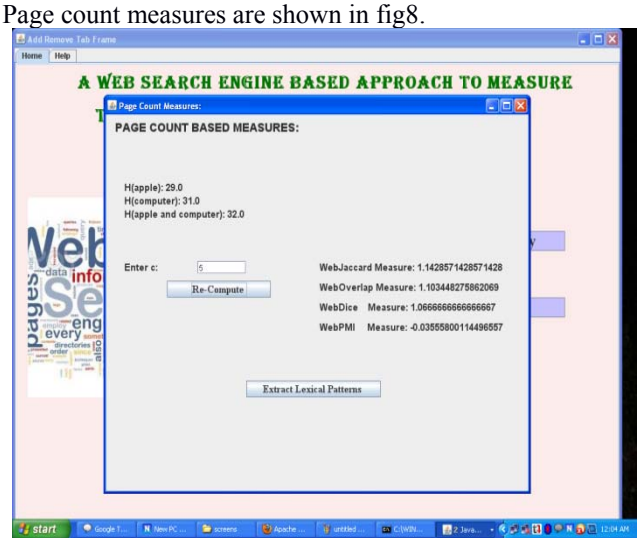

Figure8:page count measures

The sorted data according to page count as displayed in figure9.

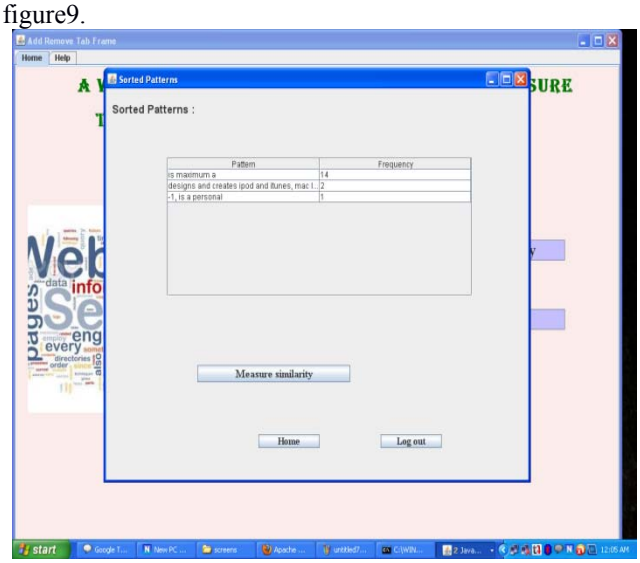

Figure9:sorted data.

The similarity between computer and apple are as shown in figure10

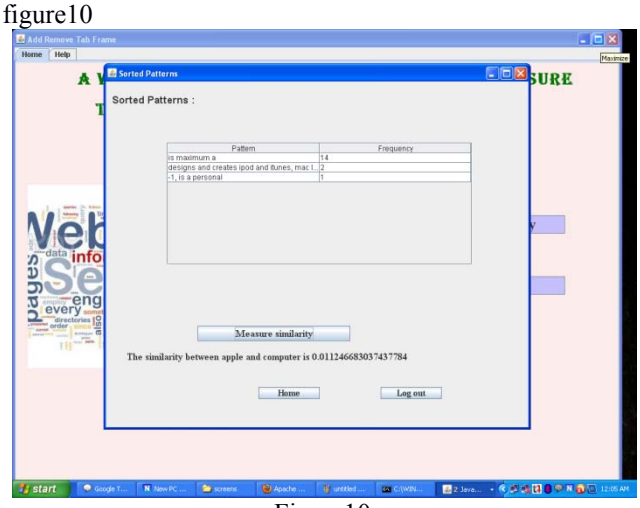

Figure10

4. CONCLUSION

We proposed a semantic similarity measure using both page counts and snippets retrieved from a web search engine for two words. Four word co-occurrence measures were computed using page counts. We proposed a lexical pattern extraction algorithm to extract numerous semantic relations that exist between two words. Moreover, a sequential pattern clustering algorithm was proposed to identify different lexical patterns that describe the same semantic relation. Both page counts-based co-occurrence measures and lexical pattern clusters were used to define features for a word pair. A two-class SVM was trained using those features extracted for synonymous and no synonymous word pairs selected from WordNet synsets. Experimental results on three benchmark data sets showed that the proposed method outperforms various baselines as well as previously proposed web-based semantic similarity measures, achieving a high correlation with human ratings. Moreover, the proposed method improved the F-score in a community mining task, thereby underlining its usefulness in real-world tasks, that include named entities not adequately covered by manually created resources.

REFERENCES

[1]. Web Search Engine-based Approach toMeasure Semantic Similarity between Words by Danushka Bollegala in IEEE 2011.

[2]. [1] A. Kilgarriff, "Googleology is bad science," Computational Linguistics,vol. 33, pp. 147–151, 2007.

[3]. [2] M. Sahami and T. Heilman, "A web-based kernel function formeasuring the similarity of short text snippets," in Proc. of 15thInternational World Wide Web Conference, 2006.

[4]. [3] D. Bollegala, Y. Matsuo, and M. Ishizuka, "Disambiguatingpersonal names on the web using automatically extracted keyphrases," in Proc. of the 17th European Conference on ArtificialIntelligence, 2006, pp. 553–557.

[5]. [4] H. Chen, M. Lin, and Y. Wei, "Novel association measures usingweb search with double checking," in Proc. of the COLING/ACL2006, 2006, pp. 1009–1016.

[6]. [5] M. Hearst, "Automatic acquisition of hyponyms from large textcorpora," in Proc. of 14th COLING, 1992, pp. 539–545

[7]. [6] M. Pasca, D. Lin, J. Bigham, A. Lifchits, and A. Jain, "Organizingand searching the world wide web of facts - step one: the onemillionfact extraction challenge," in Proc. of AAAI-2006, 2006.

[8]. [7] R. Rada, H. Mili, E. Bichnell, and M. Blettner, "Development andapplication of a metric on semantic nets," IEEE Transactions onSystems, Man and Cybernetics, vol. 9(1), pp. 17–30, 1989.

[9]. [8] P. Resnik, "Using information content to evaluate semantic similarityin a taxonomy," in Proc. of 14th International Joint Conferenceon Aritificial Intelligence, 1995.

[10]. [9] D. M. Y. Li, Zuhair A. Bandar, "An approch for measuringsemantic similarity between words using multiple informationsources," IEEE Transactions on Knowledge and Data Engineering, vol.15(4), pp. 871–882, 2003.14

[11]. [10] G. Miller and W. Charles, "Contextual correlates of semanticsimilarity," Language and Cognitive Processes, vol. 6(1), pp. 1–28,1998.

[12]. [11] D. Lin, "An information-theoretic definition of similarity," in Proc.of the 15th ICML, 1998, pp. 296–304.

[13]. [12] R. Cilibrasi and P. Vitanyi, "The google similarity distance," IEEETransactions on Knowledge and Data Engineering, vol. 19, no. 3, pp.370–383, 2007.

[14]. [13] M. Li, X. Chen, X. Li, B. Ma, and P. Vitanyi, "The similaritymetric," IEEE Transactions on Information Theory, vol. 50, no. 12,pp. 3250–3264, 2004.

[15]. [14] P. Resnik, "Semantic similarity in a taxonomy: An informationbased measure and its application to problems of ambiguity innatural language," Journal of Aritificial Intelligence Research, vol. 11,pp. 95–130, 1999.

[16]. [15] R. Rosenfield, "A maximum entropy approach to adaptive statisticalmodelling," Computer Speech and Language, vol. 10, pp. 187–228, 1996.

[17]. [16] D. Lin, "Automatic retrieival and clustering of similar words," inProc. of the 17th COLING, 1998, pp. 768–774.

[18]. [17] J. Curran, "Ensemble menthods for automatic thesaurus extraction,"in Proc. of EMNLP, 2002.

[19]. [18] C. Buckley, G. Salton, J. Allan, and A. Singhal, "Automatic queryexpansion using smart: Trec 3," in Proc. of 3rd Text REtreivalConference, 1994, pp. 69–80.

[20]. [19] V. Vapnik, Statistical Learning Theory. Wiley, Chichester, GB, 1998.

[21]. [20] K. Church and P. Hanks, "Word association norms, mutual informationand lexicography," Computational Linguistics, vol. 16, pp.22–29, 1991.

[22]. [21] Z. Bar-Yossef and M. Gurevich, "Random sampling from a searchengine's index," in Proceedings of 15th International World Wide Web Conference, 2006.

[23]. [22] F. Keller and M. Lapata, "Using the web to obtain frequencies forunseen bigrams," Computational Linguistics, vol. 29(3), pp. 459–484,2003.

[24]. [23] M. Lapata and F. Keller, "Web-based models ofr natural languageprocessing," ACM Transactions on Speech and Language Processing,vol. 2(1), pp. 1–31, 2005.

[25]. [24] R. Snow, D. Jurafsky, and A. Ng, "Learning syntactic patternsfor automatic hypernym discovery," in Proc. of Advances in NeuralInformation Processing Systems (NIPS) 17, 2005, pp. 1297–1304.

[26]. [25] M. Berland and E. Charniak, "Finding parts in very large corpora,"in Proc. of ACL'99, 1999, pp. 57–64.

[27]. [26] D. Ravichandran and E. Hovy, "Learning surface text patterns fora question answering system," in Proc. of ACL '02, 2001, pp. 41–47.

[28]. [27] R. Bhagat and D. Ravichandran, "Large scale acquisition of paraphrasesfor learning surface patterns," in Proc. of ACL'08: HLT,2008, pp. 674–682.