Clustering and Classification Augmented with Semantic Similarity for Text Mining

S.Revathi[1], Dr.T.Nalini[2]

[1] PG Scholar, Department of CSE, Bharath University,
Chennai, Tamil Nadu, India.

[2] Professor, Department of CSE, Bharath University,
Chennai, Tamil Nadu, India.

Abstract

Semantic similarity is a way of analyzing the perfect synonym that exists between word-pairs. This measure is necessary to detect the degree of relationship that persists within word-pairs. To compute the semantic similarity that lies between a word-pair, clustering and classification augmented with semantic similarity (CCASS) was developed. CCASS is a novel method that uses page counts and text snippets returned by search engine. Several similarity measures are defined using the page counts of word-pairs. Lexical pattern clustering is applied on text snippets, obtained from search engine. These are fed to the support vector machine (SVM) which computes the semantic similarity that exists between word-pairs. Based on this value obtained from the support vector machine, Simple KMeans clustering algorithm is used to form clusters. Upcoming word-pairs can be classified, after computation of its semantic similarity measure. If it does match with the existing clusters, a new cluster may be created.

Keywords: *Semantic Similarity, Similarity measure, Clustering, Classification, Text mining.*

1. Introduction

Text mining refers to the knowledge extraction from textual databases or documents. This text mining [18] is different from mining the other types of databases because of its unstructured form and large number of dimensions. Each word in the document is a dimension.

To obtain useful information from the internet, we place the keywords or entities on the search engine, which in turn returns the corresponding results matching the searched keyword or entity. As time crosses, the technique of searching changes. When a word is being searched, much irrelevant information related to the word is also displayed. Semantic Similarity is a fast emerging technique where words are analyzed based on their meaning.

For example, when a user searches for a bank located near their home, may get unwanted results related to the river banks. Maintaining these new meanings together in a dictionary is difficult. It differs based on the usage of that word in that particular phrase.

A new empirical method is proposed to evaluate the semantic similarity that exists between words. The proposed method evaluates the semantic similarity among words based on page counts and lexical pattern clustering based on snippets.

In section II, the related work for CCASS is discussed. In section III, the detailed explanation of CCASS is given. In section IV, the experiment and results of CCASS are discussed. In section V, the conclusion of CCASS is analyzed, followed by the references used.

2. Related Work

Danushka Bollegala [2] has proposed similarity measures using page count returned by the search engine for the given word pair. These similarity measures are modified four popular co-occurrence measures; Jaccard, Overlap, Dice, and PMI (point-wise mutual information). Page-count-based metrics use association ratios between words that are computed using their co-occurrence frequency in documents. The basic assumption of this approach is that high co-occurrence frequencies indicate high association ratios and high association ratios indicate a semantic relation between words.

A technique to calculate similarity between two words is to find the length of the shortest path connecting the two words in the taxonomy [3]. If a word is polysemous then many paths might exist within the two words. Only the shortest path between any two senses of the words is considered for calculating similarity. A problem that it relies is all links in the taxonomy represent a uniform distance.

Resnik [4] proposed a similarity measure using data content. The similarity between two concepts $A1$ and $A2$ in the taxonomy as the maximum of the information content of all concepts A that subsume both $A1$ and $A2$. The similarity between two words is defined as the maximum of the similarity between any concepts that the words belong to.

Some work has been carried out on measuring semantic similarity using Web content. Matsuo et al., [5] proposed the application of Web hits for obtaining communities on the Web. They measured the relation between two personal names using the overlap (Simpson) coefficient. This can be decided based on the number of Web hits for each individual name and their conjunction (i.e., *AND* query of the two names).

Sahami et al., [6] measured semantic similarity between two queries using snippets returned for those queries by a search engine. For each query, they collect snippets from a search engine and represent each snippet as a TF-IDF weighted term vector. Each vector is *L*2 normalized and the centroid of the set of vectors is computed. Semantic similarity between two queries is then defined as the inner product between the corresponding centroid vectors. They did not compare their similarity measure with taxonomy-based similarity measures.

Chen et al., [7] proposed a double-checking model using text snippets returned by a Web search engine to compute semantic similarity between words. For two words *P* and *Q*, they collect snippets for each word from a Web search engine. Then they count the occurrences of word *P* in the snippets for word *Q* and the occurrences of word *Q* in the snippets for word *P*. These values are combined nonlinearly to compute the similarity between *P* and *Q*. This method depends heavily on the search engine's ranking algorithm. Although two words *P* and *Q* might be very similar, there is no reason to believe that one can find *Q* in the snippets for *P*, or vice versa.

3. Clustering and Classification Augmented with Semantic Similarity (CCASS)

Semantic similarity [16] is calculated based on page count and text snippets. Four similarity measures are defined for any 2 words say P and Q based on their page counts namely WebJaccard, WebOverlap, WebDice and WebPMI. The lexical patterns are extracted from the text snippets obtained from the search engine. The patterns are ranked based on their ability to analyze the Semantic similarity. The Support vector machine (SVM) is used to express the semantic similarity value.

After detecting the semantic similarity value, Simple KMeans Clustering algorithm is used to cluster the semantically related words. Thereafter when a new word enters, its semantic similarity value is calculated and it can be easily classified into any one of the existing clusters. If it does not belong to any of the existing clusters, then a new cluster can be created by placing this one member into the new cluster.

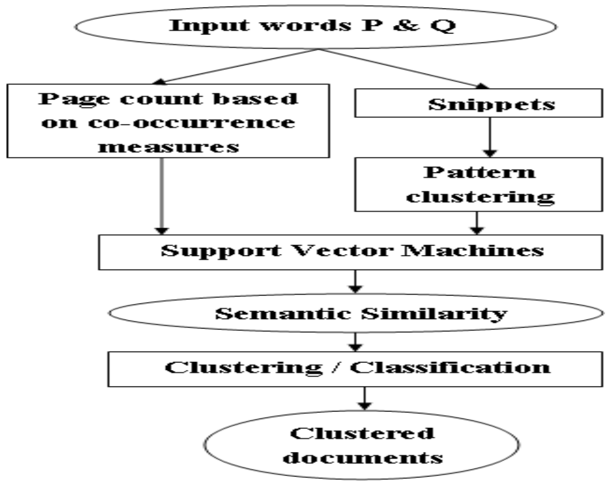

Figure 1: CCASS (Clustering and Classification Augmented with Semantic Similarity)

3.1 Similarity measure based on Page Count

Semantic similarity [17] between two words can be calculated based on their page counts. To find the similarity between 2 words say P and Q, the page count measure of P AND Q is not sufficient. For example, the page count for “doctor” AND “car” in GOOGLE is 643,000,000. Whereas the page count for “doctor” AND “profession” is 51,200,000. The page count of P AND Q is not sufficient and so the individual page count has also become necessary.

The similarity measures WebJaccard, WebOverlap, WebDice and WebPMI based on the page counts can be defines as shown below:

$$\mathrm{WebJaccard}(P,Q) = \begin{cases} 0 & \text{if } H(P\cap Q) \leq c \\ \frac{H(P\cap Q)}{H(P)+H(Q)-H(P\cap Q)} & \text{otherwise.} \end{cases} \quad (1)$$

$$\mathrm{WebOverlap}(P,Q) = \begin{cases} 0 & \text{if } H(P\cap Q) \leq c \\ \frac{H(P\cap Q)}{\min(H(P),H(Q))} & \text{otherwise.} \end{cases} \quad (2)$$

$$\mathrm{WebDice}(P,Q) = \begin{cases} 0 & \text{if } H(P\cap Q) \leq c \\ \frac{2H(P\cap Q)}{H(P)+H(Q)} & \text{otherwise.} \end{cases} \quad (3)$$

$$\text{WebPMI}(P,Q) = \begin{cases} 0 & \text{if } H(P \cap Q) \le c \\ \log_2\left(\frac{\frac{H(P\cap Q)}{N}}{\frac{H(P)}{N}\frac{H(Q)}{N}}\right) & \text{otherwise.} \end{cases} \quad (4)$$

Here H(P) is the page count of word P. In our example, the value of C = 5. And N is the total number of documents indexed by GOOGLE, for example 10^{10}.

3.2 Extracting Lexical Patterns from Snippets

The search engine returns text snippets on the result of searching a word. Lexical patterns are extracted from the text snippets. For example, the following is the snippet returned by GOOGLE for the keywords “doctor” and “profession”.

Doctor is a noble profession who serves our country to save human life.

A Lexical pattern extraction algorithm is proposed to extract the patterns from the text snippets obtained from the search engine. Given a set W of synonymous word –pairs, GetSnippets function returns a list of text snippets for the query “P” AND “Q”. For each “a” returned by GetSnippets function, GetNgrams extract word n-grams, n=2, 3, 4, 5. CountFreq counts the frequency of each pattern extracted.

Algorithm 1 : EXTRACTPATTERN

```
Given a set W of word-pairs, extract patterns.
For each word-pair ( P, Q ) Ɛ W
        do A ← GetSnippets (“P Q”)
 N ← null
 For each snippet a Ɛ A
        do N ← N + GetNgrams ( a, P, Q )
Pats ← CountFreq ( N )
return ( Pats )
```

3.3 Integration of page counts and lexical patterns

Similarity measures were used to calculate the similarity between two words based on their page counts retrieved from a search engine [19]. Lexical patterns are extracted from the snippets, which are obtained from the search on the search engine.

Algorithm 2 :GETFEATUREVECTOR(P,Q)

```
Given a word-pair P,Q get its feature vector F.
A ← GetSnippets(“P Q”)
N ← null
for each snippet a Ɛ A
      do N ← N + GetNgrams ( a, P, Q )
SelPats ← SelectPatterns ( N, GoodPats )
PF ← Normalize (SelPats)
F ← [ PF, WebJaccard, WebOverlap, WebDice, WebPMI ]
return ( F )
```

A feature vector is created for each word-pair. The snippets for words “P” AND “Q” are extracted. Words “P” and “Q” are replaced with X and Y. GetNgrams extracts n-grams, n=2, 3, 4, 5. n-grams with perfectly one X and one Y are selected. Let the patterns be selected based on X^2 values be GoodPats. Normalizing refers to the process of dividing the count of each pattern by the total number of counts of observed patterns. The similarity measures that are calculated earlier based on the page count of the word-pairs are also uploaded. The final feature vector F is created.

The feature vectors are calculated for each word-pair. The SVM is trained to calculate the semantic similarity for each word-pair. Consider another word-pair (P', Q') and calculate its feature vector F'. The semantic similarity SemSim(P', Q') is defined as,

$$\text{SemSim}(P', Q') = \text{Prob}\,(F' | \text{synonymous})$$

which is a posterior probability, that says the feature vector F' belongs to synonymous-word class.

3.4 Simple KMeans Clustering

The semantic similarity values obtained from the support vector machine are clustered based on their probability values. Here, Simple KMeans clustering [20] is used to form clusters. The numbers of clusters are independent of the number of word-pair.

The simple KMeans is a type of partitional clustering. KMeans is an iterative clustering [21] algorithm in which items are moved among sets of clusters until the desired set is reached.

3.5 Classification

After formation of clusters, when a new word enters, it can be easily classified based on the existing clusters. If it does not match with the existing clusters, a new cluster is created and this new member is placed into the cluster.

Algorithm 3 : For Classification

```
Input: set of files D, cluster id C, clusters
Output: allotting cluster id
Procedure:
Begin
Let C be array of cluster ids
Let M be the array size equals to clusters
for each file fd in D
for each cluster
M[i]= get_similarity(fd,Ci )
```

```
end for
cid=find_max(M)
output cid as cluster id for file fd
update clusters // assign fd to cluster
with cid as cluter id if cid is new
form new cluster
end for
end
find_max(M)
{
Let cid=0;
for i 1 to size(M)
if m[i]>m[cid]
cid=i;
end for
if(cid=0)
cid=size(M)+1; // new cluster id is formed
return cid;
}
```

4. Experiment and Results

4.1 Experimental setup

CCASS is experimented against the Miller-Charles dataset, with 28 word-pair. Their similarity is rated from 0 (no similarity) to 4 (perfect synonym). Miller-Charles data set is a subset of Rubenstein Goodenough's original data set of 65 word pairs. This dataset contains words as pair which are quite similar in meaning. They are formed with a human rating is associated with this data set. Works are also performed with the Rubenstein Goodenough's original data set.

Weka is a collection of open source ML algorithms used for pre-processing, classifiers, clustering, and association rule. Weka is created by researchers at the University of Waikato in New Zealand. It is a Java based tool used in the field of data mining. It uses flat text files to describe the data. It can work with a wide variety of data files including its own ".arff" format and C4.5 file formats.

Sample Word-pair

Cord - smile
Rooster - voyage
Noon - string
Glass - magician
Monk - slave
Coast – forest
Monk – oracle
Lad – wizard
Forest -graveyard
Food - rooster

4.2 Similarity measures based on Page count

Based on the dataset, we consider each word-pair as "P" and "Q" to calculate the various similarity measure such as WebJaccard, WebOverlap, WebDice and WebPMI. The following is the result against the Miller-Charles dataset that contains 28 word-pairs. Each word-pair is considered individually and its similarity measure is calculated for all the 4 methods.

TABLE I SIMILARITY MEASURES

Word pair	Web Jaccard	Web Overlap	Web Dice	Web PMI
Cord – smile	0.102	0.108	0.036	0.207
Rooster-voyage	0.011	0.012	0.021	0.228
Noon - string	0.126	0.133	0.060	0.101
Glass – magician	0.117	0.124	0.408	0.598
Monk - slave	0.181	0.191	0.067	0.610
Coast – forest	0.862	0.870	0.310	0.417
Monk – oracle	0.016	0.017	0.023	0
Lad – wizard	0.072	0.077	0.070	0.426
Forest - graveyard	0.068	0.072	0.246	0.494
Food - rooster	0.012	0.013	0.425	0.207
Coast – hill	0.963	0.965	0.279	0.350
Car – journey	0.444	0.460	0.378	0.204
Crane – implement	0.071	0.076	0.119	0.193
Brother – lad	0.189	0.199	0.369	0.644
Bird – crane	0.235	0.247	0.226	0.515
Bird – cock	0.153	0.162	0.162	0.428
Food – fruit	0.753	0.765	0	0.448
Brother-monk	0.261	0.274	0.340	0.622
Asylum – madhouse	0.024	0.025	0.102	0.813
Furnace-stove	0.401	0.417	0.118	1
Magician – wizard	0.295	0.309	0.383	0.863
Journey – voyage	0.415	0.431	0.182	0.467
Coast – shore	0.786	0.796	0.521	0.561
Implement – tool	1	1	0.517	0.296
Boy - lad	0.186	0.196	0.601	0.631
Automobile – car	0.654	0.668	0.834	0.427
Midday-noon	0.106	0.112	0.135	0.586
Gem-jewel	0.295	0.309	0.094	0.687
Correlation	**0.259**	**0.267**	**0.382**	**0.548**

4.3 Semantic Similarity

TABLE II SEMANTIC SIMILARITY

Word pair	Semantic Similarity
Cord – smile	0
Rooster-voyage	0.017
Noon - string	0.018
Glass - magician	0.180
Monk - slave	0.375
Coast – forest	0.405
Monk – oracle	0.328
Lad – wizard	0.220
Forest -graveyard	0.547
Food - rooster	0.060
Coast – hill	0.874
Car – journey	0.286
Crane – implement	0.133
Brother – lad	0.344
Bird – crane	0.879
Bird – cock	0.593
Food – fruit	0.998
Brother-monk	0.377
Asylum – madhouse	0.773
Furnace-stove	0.889
Magician – wizard	1
Journey – voyage	0.996
Coast – shore	0.945
Implement – tool	0.684
Boy – lad	0.974
Automobile – car	0.980
Midday-noon	0.819
Gem-jewel	0.686
Correlation	**0.834**

4.4 Simple KMeans Clustering

The clustering algorithm takes the above calculated semantic similarity value as input and separates it into 2 different clusters. 13 word pairs are clustered separately and 15 word pairs are clustered separately.

TABLE III SIMPLE KMEANS CLUSTERING

Cluster ID	No. of entities	Percentage
1	13	46
2	15	54

4.5 Classification

When a new word-pair enters CCASS, its semantic similarity value is calculated. Based on that, it will be placed in the existing cluster or else a new cluster will be created.

TABLE IV CLASSIFICATION

Word - pair	Semantic Similarity	Cluster ID
Company– stock	0.867	2
Stock - life	0.006	3

According to the results in Table IV, the word-pair "company-stock" yields a semantic similarity value of 0.0867 and will be assigned to the cluster 2. But the word-pair "stock-life" produces a semantic similarity of 0.006. Therefore a new cluster id 3 is generated and this word-pair is assigned to the new cluster.

5. Conclusion

CCASS is a novel method that calculates the semantic similarity between a word-pair. The similarity measures are calculated based on the page counts of the word-pair. Text snippets are extracted from the search engine for the word-pair. Their lexical patterns are extracted and clustered together. These are fed to the SVM to obtain the semantic similarity value as a posterior probability.

Based on the semantic similarity value, clusters are formed to group the semantically similar words. When a new word-pair enters CCASS, it can be easily classified into the existing clusters. If it does not match with the existing cluster, a new cluster can be created for that word-pair.

Acknowledgements:
The authors would thank the management of Bharath University for their support and the Department of Computer Science and Engineering for their encouragement towards the research work.

References

[1] SaiSindhu Bandaru, Dr. K B Madhuri, "**An Efficient Semantic Model For Concept Based Clustering And Classification**", International Journal on Computer Science and Engineering (IJCSE), Vol. 4 No. 03, PP.340-347, March 2012.

[2] Danushka Bollegala, Yutaka Matsuo, Mitsuru Ishizuka, "**Measuring Semantic Similarity between Words Using Web Search Engines**", International World Wide Web Conference Committee (IW3C2), May 8–12, 2007.

[3] R. Rada, H. Mili, E. Bichnell, and M. Blettner. "**Development and application of a metric on semantic nets**",IEEE Transactions on Systems, Man and Cybernetics, 9(1):17-30, 1989.

[4] P. Resnik."**Using information content to evaluate semantic similarity in a taxonomy**", Proc. of 14th International Joint Conference on Aritificial Intelligence, 1995.

[5] Y. Matsuo, J. Mori, M. Hamasaki, K. Ishida, T. Nishimura, H. Takeda, K. Hasida, and M. Ishizuka. Polyphonet**: "An advanced**

social network extraction system", Proc. of 15th International World Wide Web Conference, 2006.

[6] M. Sahami and T. Heilman**, "A web-based kernel function for measuring the similarity of short text snippets"**, Proc. of 15th International World Wide Web Conference, 2006.

[7] H. Chen, M. Lin, and Y. Wei. **"Novel association measures using web search with double checking"**, Proc. of the COLING/ACL 2006, pages 1009-1016, 2006.

[8] Shady Shehata, , Fakhri Karray, and Mohamed S. Kamel, **"An Efficient Concept-Based Mining Model for Enhancing Text Clustering,"** IEEE Transactions on Knowledge and Data Engineering , vol. 22, no. 10, October 2010.

[9] Z. Bar-Yossef and M. Gurevich. "**Random sampling from a search engine's index**.*",* Proceedings of 15th International World Wide Web Conference, 2006.

[10] Rudi L. Cilibrasi and Paul M.B. Vita´ nyi, **—" The Google SimilarityDistance"**, IEEE Transactions On Knowledge And Data Engineering, Vol.19, No. 3, March 2007.

[11] Ann Gledson and John Keane**," Using Web-Search Results to Measure Word-Group Similarity"** Proceedings of the 22nd International Conference on Computational Linguistics, pages 281–288 Manchester, August 2008.

[12] Elias Iosif and Alexandros Potamianos, **"Unsupervised Semantic Similarity Computation between Terms Using Web Documents"** IEEE transactions on knowledge and data engineering, vol. 22, no.11, November 2010.

[13] J. Gracia, R. Trillo, M. Espinoza, and E. Mena, **—"Querying the web: A multiontology disambiguation method"**, in Proc. of International Conference on Web Engineering, 2006, pp. 241–248.

[14] M. Lapata and F. Keller. **"Web-based models for natural language processing"** ACM Transactions on Speech and Language Processing, 2(1):1-31, 2005.

[15] M. Pasca, D. Lin, J. Bigham, A. Lifchits, and A. Jain. **"Organizing and searching the world wide web of facts - step one: the one-million fact extraction challenge"**, In Proc. of AAAI-2006, 2006.

[16] Kishor Wagh, Satish Kolhe **"Information Retrieval based on Semantic Similarity Using Information Content"**, IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 4, No 2, July 2011, pp.364-370.

[17] K.Saruladha, Dr.G.Aghila, Sajina Raj, **"A new Semantic Similarity metric for solving sparse data problem in ontology based Information Retrieval System"**, IJCSI International Journal of Computer Science Issues, Vol. 7, Issue 3, No 11, May 2010, pp.40-48.

[18] Shaidah Jusoh, Hejab M.Alfawareh, **"Techniques, Applications and Challenging Issue in Text Mining"**, IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 6, No 2, November 2012,pp 431-436.

[19] Sumit Vashishtha, Dr.Yogendra Kumar Jain, **"Efficient retrieval of Text for Biomedical domain using Expectation maximization algorithm"**, IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 6, No 1, November 2011, pp 366-370.

[20] M.Yasodha, Dr.P.Ponmuthuramalingam, **"An advanced concept based mining model to enrich text clustering",** IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 4, No 1, July 2012,pp 417-422.

[21] V.M.NavaneethaKumar, Dr.C.Chandrasekar,, "**A consistent web documents based text clustering using concept based mining model"**, IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 4, No 1, July 2012, pp 365-370.