CONCEPT-BASED CLUSTERING FOR OPEN-SOURCED SOFTWARE(OSS) DEVELOPMENT FORUM THREADS

Jonathan Jason C. King Li
Ateneo de Manila University, Katipunan Avenue, Loyola Heights, Quezon City 1108, Philippines

Masanori Akiyoshi, Masaki Samejima, Norihisa Komoda
Osaka University, Yamadaoka 2-1, Suita, 565-0871, Osaka, Japan

Keywords: Clustering, Forum threads, Concept-based similarity.

Abstract: Open-Sourced Software Development depends on the Internet Forum for communication among its developers. However, a typical program would have related modules which are hard to express in the forum. Though human effort of reporting related modules is already being used, this technique is impractical due to human inaccuracy. Our approach uses the Concept-Based Document Similarity for its thorough analysis on the semantic value of a word or phrase on the sentence, document and corpus level for the purpose of measuring similarities between documents. Then we created a novel Clustering Algorithm that does not need any threshold values and it is able to stop clustering when the clusters are already correctly formed. This was first used on newspapers to test its effectiveness and then was used on a cluster of Bugzilla threads. The results from the newspapers proved the clustering process works but the results for the Bugzilla threads, where the comment content do not evidently reveal thread topic, reveals that other elements, aside from thread content, is needed to establish similarity. Future work will utilize other thread elements for clustering similar threads.

1 INTRODUCTION

Open-Sourced Software Development demands communication between its developers around the world to ensure the longevity and the usefulness of the product. To achieve such logistical feat, developers use Internet Forums to communicate with each other. However, given the complexities of any typical software design, developers would have to manually report related threads, which represent program modules, thus having these unreported related threads lost within the forum. However, developers may tend to forget to report some related threads or even report unrelated threads thus putting the trustworthiness of this technique into question. The aim of this research is to create an algorithm to find related threads so that the threads that are not reported by developers can be found and not lost within the forum. Our summarized approach employs text mining to analyze the comment content of each thread to find patterns and similarities between each thread. Then similar threads are clustered together, which would contain the related threads that the developers are looking for.

2 METHODOLOGY OVERVIEW

After data extraction from HTML format or from any format, the whole process can be summarized into two phases namely the Concept-Based Document Similarity and Document Clustering. After data cleansing, such as word stemming and stopword removal, the incoming document collection or corpus enters the Concept-Based Document Similarity phase where it will encounter three sub-phases namely Concept Extraction, Concept Counting and Similarity Calculation (Shehata et al., 2010). The following output will be a Similarity Matrix which will enter the Document Clustering phase where the similar documents are clustered together iteratively. Then the final output is clusters of similar documents. The whole process can be seen in Figure 1

2.1 Concept-based Document Similarity

The goal of this phase is to produce a similarity matrix where it contains all the similarity values between each document. As it was mentioned, the Concept-

690 Jason C. King Li J., Akiyoshi M., Samejima M. and Komoda N..
CONCEPT-BASED CLUSTERING FOR OPEN-SOURCED SOFTWARE(OSS) DEVELOPMENT FORUM THREADS.
DOI: 10.5220/0003478206900695
In *Proceedings of the 7th International Conference on Web Information Systems and Technologies* (WTM-2011), pages 690-695
ISBN: 978-989-8425-51-5
Copyright © 2011 SCITEPRESS (Science and Technology Publications, Lda.)

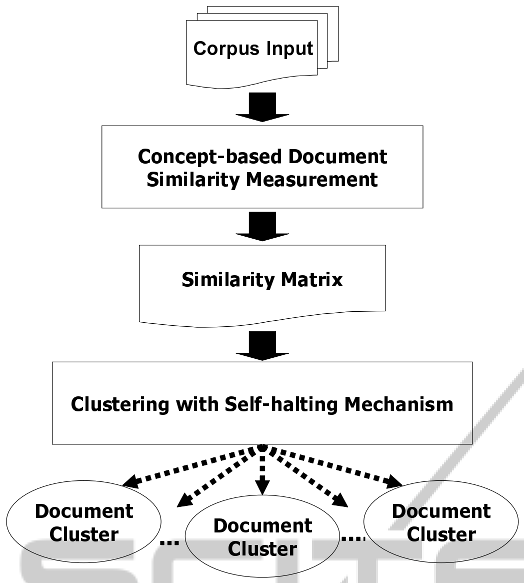
Corpus Input
Concept-based Document Similarity Measurement
Similarity Matrix
Clustering with Self-halting Mechanism
Document Cluster
Document Cluster
Document Cluster

Figure 1: Concept-based clustering process.

Based Document Similarity is composed of three sub-phases namely Concept Extraction, Concept Counting and Similarity Calculation. First, a document enters the Concept-Based Document Similarity phase and then during Concept Extraction, each sentence is extracted for verb-arguments. For every verb in a sentence, there is one verb-argument structure. The verb-argument structure contains the verb itself, the subject and the object of the verb. Using the Stanford parser (Dan and Christopher, 2003), made by the Stanford Natural Language Processing Group, it can distinguish the role of every word or group of words in a sentence. The noun phrase directly before the verb is the subject of the verb, and heuristically, the rest of the sentence after the verb is the object of the verb. Overlapping verb arguments are intended for Concept Counting. Once the subject, verb and object are extracted they are all considered as concepts and are added to the Concept List of the document. Also as a standard procedure, concepts are cleaned of stop words and are stemmed using the famous Porter Stemmer (Porter, 1980)

The second sub-phase is Concept Counting. Following the algorithm presented by Shehata et al., three measurements are introduced namely the Concept Term Frequency (CTF), Term Frequency (TF) and the Document Frequency (DF). The CTF counts how many times a concept repeats itself per verb-argument. The TF counts how many times a concept repeats itself per sentence. DF counts how many times a concept repeats itself per document so at this point, DF is always 1. This kind of counting is done for each concept registered in the Concept List of the current document. Afterwards, another document enters the Concept-Based Document Similarity phase and the first two sub-phases are executed. This process iterates until all documents in the corpus is exhausted and only then all documents enter the Similarity Calculation sub-phase.

In the Similarity Calculation sub-phase, we create a Union Concept List from the Concept Lists created. The Union Concept List is the Concept List of the corpus itself where it contains all concepts. First, all unique concepts are copied to the Union Concept List. Then if a duplicate is found, the CTF becomes the average between the current CTF value and the CTF value of the duplicate. Then, the current DF value is incremented. Afterwards, we calculate the similarities between documents using their Concept Lists. First a matched Concept List is created between two Concept Lists. A pair of concepts are considered a match if either an exact match or a partial match occurs (ex. w_1, w_2 and w3 are distinct words and Concept A contains w_1 and w_2, Concept B contains w_1 and w_2, and Concept C contains w_1, w_2 and w_3, Concept A and Concept B have an exact match while Concept A and C have a partial match). Finally, the following equations found in (Shehata et al., 2010) are used to measure the similarity between two documents.

$$sim_c(d_1,d_2) = \sum_{i=1}^{m} max\left(\frac{l_{i1}}{Lv_{i1}}, \frac{l_{i2}}{Lv_{i2}}\right) \times weight_{i1} \times weight_{i2}$$

The following explains the variables used in the formula:

- d_1 and d_2 are two documents.
- m is the number of matching concept pairs therefore i is the counter for the matching concept pairs.
- l is the length of the concept itself so l_{ik} is the length of the matching concept i in d_k.
- L_v is the length of the verb-argument where concept i belongs to.
- weight is the weight of the matching concept pairs.

The values that result from the similarity equation are recorded in the similarity matrix where each value can be traced to the two documents that were used. The matrix will be the input for our document clustering.

	A	B	C	D
A	5.4	2.3	1.5	0.7
B	2.3	6.6	1.0	0.5
C	1.5	1.0	5.9	1.1
D	0.7	0.5	1.1	6.2

Figure 2: Mutually relatedness with similarity matrix.

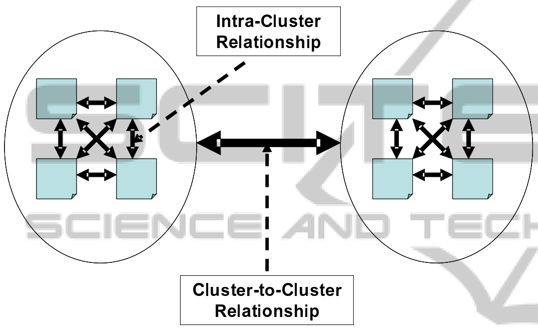

Figure 3: Intra-Cluster relationships and Cluster-to-Cluster relationships.

2.2 Document Clustering

When a similarity matrix enters the Document Clustering phase, the matrix is first saved. Aside from the values along the diagonal, we take the highest value per column, also known as the Top One of the column, and then we check for document pairs that has each other as their Top One. We call this kind of relationship as the "Top One Mutuality Relationship". Figure 2 clearly shows the definition of a Top One Mutuality (A and B have a Mutual Top One relationship while A and C, and C and D do not since the Top One of C is A, but A does not have C as its Top One. The same reason applies for C and D.).

Document pairs with a Top One Mutuality Relationship will be merged together where unique concepts will be added to one of the Concept Lists, thus the remaining one will be discarded. Duplicate concepts will update its CTF and TF by adding their values together but the DF will remain as 1. This merged cluster is now considered as one cluster. After this iteration is finished, a new Union Concept List is recreated and a new similarity matrix, which will not be saved, is created. The following iterations will have a different procedure. Before we continue, we need to introduce some definitions. All relationships based on new similarity matrices are "Cluster-to-Cluster relationships" regardless if there is only one document in a cluster and all relationships based on the initial similarity matrix are "Document-to-Document relationships". Furthermore, the Intra-Cluster Document-to-Document relationship is the similarity value between two documents within the same cluster based on the initial similarity matrix. Figure 3 shows the difference between Intra-Cluster relationships and Cluster-to-Cluster relationships. Going back to the process, we again look for Top One Mutuality Relationships based on the new similarity matrix. Document pairs with Top One Mutuality Relationships based on this new similarity matrix will be merged but with an additional condition. If the current Cluster-to-Cluster relationship is greater than half the average of all Intra-Cluster Document-to-Document relationships of the two clusters involved, then the two clusters should be merged, otherwise they should not be merged.

This algorithm closely matches our intuitive way of clustering documents together. After reading the documents, we try to cluster the closest documents together based on what we read from each document. However, our own clustering system stops on its own without clustering all the documents into one cluster. This is because when try to decide to merge two clusters, we actually compare the relationship between the two clusters itself against the relationship of documents within the involved clusters. Thus we came up with our clustering algorithm that tries to match this intuitive process. The advantage of this process is that it no longer needs threshold values and the clustering process stops on its own.

3 EXPERIMENTAL RESULTS

We first applied the algorithm to newspaper articles. The newspapers were taken from the Manila Bulletin, the Philippine Daily Inquirer, BBC, the Associated Press and the ABC News for a variety of writers and issues. The articles are assigned a number from 1 to 16 and then we manually created the clusters from the 16 newspaper articles that we gathered. Then we let the algorithm create the clusters. The results are found in Table 1.

The algorithm generated clusters are equal to the manually generated clusters. This same algorithm is also applied to 31 Bugzilla threads and results are in Table 2. Furthermore, the algorithm was applied to 25 threads in Bugzilla concerning Firefox, an open sourced web browser, and 40 threads in Bugzilla concerning Thunderbird, an open sourced email client,

Table 1: Applied result to Newspapers.

Manually generated clusters	Concept-based algorithm clusters
Truth commission topic cluster: 01, 02, 03, 08, 09	Cluster 1: 01, 02, 08, 09, 03
Traffic topic cluster: 04, 11, 12, 13	Cluster 2: 04, 11, 12, 13
Wikileaks topic cluster: 05, 06, 07, 14, 15, 16	Cluster 3: 05, 15, 06, 16, 07, 14
Christmas topic cluster: 10	Cluster 4: 10

and the results are in Table 3 and in Table 4 respectively.

Simply put, the clusters that were generated by the algorithm for both Table 2, 3 and 4 are not equal. Apparently the content of the comments do not capture the very topic of the thread itself. However, it is possible to regard the single thread clusters found in the results as insignificant. To actually recreate the results of the manually created clusters, other elements found in the forums should be found and utilized to measure similarity.

4 RELATED WORKS

Text mining is generally used in the field of information retrieval especially in this age of the Internet (Arimura et al., 2000). Because of the vast amount of information that we receive in this era, many researches have been geared for "teaching" computers how to read. This technology has been applied to marketing strategies to create profiles for retail items which may contain irregular words thus needing to measure the semantic value of the words that are found in their transactions (Ghani and Fano, 2002). This has also been applied to find patterns and trends among documents to discern the popularity of a document (Terachi et al., 2006). Some are also used to find patterns in error reports to ameliorate their standard procedures (Malin et al., 2009).

Traditional procedures usually include the use of Term Frequency and Inverse Document Frequency or Recency/Frequency/Monetary Analysis (Terachi et al., 2006) (Clifton et al., 2004) or other data mining techniques such as Naive Bayes or Expectation-Maximization (Ghani and Fano, 2002). These procedures are often used to measure the semantic value of a word in their given domain. However, this makes the algorithm too specialized and not reusable to other fields not to mention that they still depend on one-word statistics.

Semantic determination for a word is a growing research area and has many applications in computational linguistics, such as data clustering, and artificial intelligence. Many approaches have been made by many research works over the past decade. One method involves using WordNet to create a hierarchy of words to depict semantic relatedness (Shen and Angryk, 2007). Another extends this idea to use multiple sources such as WordNet and the Brown Corpus (Li et al., 2003). These approaches would help in mining small bodies of text such as sentences (Li et al., 2006) or short documents (Jing et al., 2007) where the meaning of each word needs through analysis due to the size of the text. However, application of this method in large bodies of texts would become too tedious.

A few recent development of text mining shows a paradigm shift from one-word statistics to phrasal statistics, a precursor to concept-based mining. Such a movement started with Hammouda et al. (Hammouda and Kamel, 2004) with the use of the Document Index Graph. Phrasal computation is then followed by Hung Chim and Xiaotie Deng (Hung and Xiaotie, 2008) where they used phrasal computation with TF-IDF similarity measures. These researches have proved themselves more superior to single-word statistics analysis but they still depend on traditional procedures used for single-word statistics and thus these procedures are not thorough enough.

The most recent development in text mining is Concept-Based Mining (Shehata et al., 2010). The algorithm argues phrasal statistics are still insufficient in discerning the semantic value of a certain phrase. Therefore, they introduced concepts which can be single-words or phrases but they measure similarity in a different way as it will be explained later. They propose counting concepts on the sentence level, document level and corpus level and then they base similarity according to concept matches instead of word or phrase matches. Furthermore, regardless of the domain, this is applicable since there is no storage of certain words and the importance of words are measured depending on the corpus input. As we know that Bugzilla may use technical jargon, using Concept-Based Mining may be the best choice. However, they did not present a clustering algorithm to handle the results from the Concept-Based Mining. Therefore, we created our own document clustering without a threshold value and it stops when the correct clusters are already formed.

Table 2: Applied result to Bugzilla threads.

Manually generated clusters	Concept-based algorithm clusters
Cluster 1: 302121, 303043, 303105, 303332, 303398, 303115, 303121, 304553, 303065, 303412, 303206, 303335, 303444, 303509, 303684, 303737	Cluster 1: 302121, 303043, 303105, 303332, 304622, 303848, 304962, 304426, 302749, 303398, 303115, 303121, 304553, 304862
Cluster 2: 304622, 303848, 304962, 304426	Cluster 2: 303065, 303412, 303644, 303647 302749, 304862, 303644, 304436, 304457, 304597
Cluster 3: 303647	Cluster 3: 303195
Cluster 4: 303195	Cluster 4: 303206
Cluster 5: 304705, 305928, 306019	Cluster 5: 303335, 304436
	Cluster 6: 303444
	Cluster 7: 303509
	Cluster 8: 303684
	Cluster 9: 303737
	Cluster 10: 304457, 304597
	Cluster 11: 304705, 305928, 306019

Table 3: Applied result to Firefox threads.

Manually generated clusters	Concept-based algorithm clusters
Cluster 1: 300412, 305216, 304861, 304558	Cluster 1: 300001 304748, 304405, 305955, 306132
Cluster 2: 300001, 300074	Cluster 2: 300074
Cluster 3: 300565, 300814, 300794, 300740	Cluster 3: 300412, 300706, 300696, 300651, 300615
Cluster 4: 305267, 308398, 318168, 307877	Cluster 4: 300565, 300706, 300814, 300651 300696, 300740
Cluster 5: 305998, 319861, 306409	Cluster 5: 300615
	Cluster 6: 300794, 305995, 306132, 304748, 318168
	Cluster 7: 304405
	Cluster 8: 304558
	Cluster 9: 304861
	Cluster 10: 305216
	Cluster 11: 305267
	Cluster 12: 305998, 306409
	Cluster 13: 307877
	Cluster 14: 308396
	Cluster 15: 319861

5 FUTURE WORK

The algorithm works for clustering documents with cohesive content. However, for media such as the Bugzilla Forums, other elements may be needed to match implicit relationships between threads. It is possible to insert the idea by (Hammouda and Kamel, 2004) where they processed different parts of a Web document into HIGH, MEDIUM and LOW portions. Then it is also possible to use in-depth word analysis introduced by (Shen and Angryk, 2007) (Jing et al., 2007) (Li et al., 2003) to process small bodies of text such as titles. Furthermore, the manually created clusters may already help in trying to find patterns between each thread within each cluster or comparative experiments with other competing text clustering algorithms maybe used to clearly measure the efficacy of this algorithm.

6 CONCLUSIONS

This paper proposed a novel document clustering based on concept-based similarity, with automatic clustering stopping process and without creating one

Table 4: Applied result to Thunderbird threads.

Manually generated clusters	Concept-based algorithm clusters
Cluster 1: 216223, 216225, 227883	Cluster 1: 216223, 216225, 227883, 232432, 259951, 259959, 259331, 229879
Cluster 2: 218774, 240476, 240138, 243631	Cluster 2: 218774
Cluster 3: 232967, 289375, 284030	Cluster 3: 220173, 232967, 284030, 289375
Cluster 4: 229179, 229740, 230241	Cluster 4: 227841
Cluster 5: 220173, 231959, 234811, 234707	Cluster 5: 228300, 233944, 232433, 232432, 231960, 230925
Cluster 6: 227841, 230700, 232699, 231552, 229879, 228300	Cluster 6: 229179, 230241, 259289, 243631, 234707, 240476, 240138
Cluster 7: 257378, 259227, 273422, 259958, 259951, 259331, 259321, 259317, 259959, 259289, 258897, 258447	Cluster 7: 229740
	Cluster 8: 230700
	Cluster 9: 230925, 231552, 231960, 233944, 257378
	Cluster 10: 231959, 234811
	Cluster 11: 232433, 258897, 259317, 259321, 273422
	Cluster 12: 232699, 258447
	Cluster 13: 259227
	Cluster 14: 259958

huge cluster containing all the documents in a corpus. However, further testing of this clustering procedure is probably needed to test its trustworthiness.

REFERENCES

Arimura, H., Abe, J., Fujino, R., Sakamoto, H., Shimozono, S., and Arikawa, S. (2000). Text data mining: Discovery of important keywords in the cyberspace. In *International Conference on Digital Libraries: Research and Practice, pp.220-226*.

Clifton, C., Cooley, R., and Rennie, J. (2004). Topcat: Data mining for topic identification in a text corpus. *IEEE Transactions on Knowledge and Data Engineering, Vol.16, No.8, pp.949-964*.

Dan, K. and Christopher, D. M. (2003). Accurate unlexicalized parsing. In *the 41st Meeting of the Association for Computational Linguistics, pp. 423-430*.

Ghani, R. and Fano, A. (2002). Using text mining to infer semantic attributes for retail data mining. In *2002 IEEE International Conference on Data Mining(ICDM 2002), pp.195-202*.

Hammouda, K. and Kamel, M. (2004). Efficient phrase-based document indexing for web document clustering. *IEEE Transactions on Knowledge and Data Engineering, Vol.16, No.10, pp.1279-1296*.

Hung, C. and Xiaotie, D. (2008). Efficient phrase-based document similarity for clustering. *IEEE Transactions on Knowledge and Data Engineering, Vol.20, No.9, pp.1217-1229*.

Jing, P., Dong-qing, Y., Jian-wei, W., Meng-qing, W., and Jun-gang, W. (2007). A clustering algorithm for short documents based on concept similarity. In *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, pp.42-45*.

Li, Y., Bandar, Z., and Mclean, D. (2003). An approach for measuring semantic similarity between words using multiple information sources. *IEEE Transactions on Knowledge and Data Engineering, Vol.15, No.4, pp.871-882*.

Li, Y., McLean, D., Bandar, Z., O'Shea, J., and Crockett, K. (2006). Sentence similarity based on semantic nets and corpus statistics. *IEEE Transactions on Knowledge and Data Engineering, Vol.18, No.8, pp.1138-1150*.

Malin, J., Millward, C., Schwarz, H., Gomez, F., Throop, D., and Thronesbery, C. (2009). Linguistic text mining for problem reports. In *IEEE International Conference on Systems, Man and Cybernetics(SMC 2009), pp.1578-1583*.

Porter, M. (1980). An algorithm for suffix stripping. *Program, Vol.14, No.3, pp.130-137*.

Shehata, S., Karray, F., and Kamel, M. (2010). An efficient concept-based mining model for enhancing text clustering. *IEEE Transactions on Knowledge and Data Engineering, Vol.22, No.10, pp.1360-1370*.

Shen, W. and Angryk, R. (2007). Measuring semantic similarity using wordnet-based context vectors. In *IEEE International Conference on Systems, Man and Cybernetics(SMC 2007), pp.908-913*.

Terachi, M., Saga, R., and Tsuji, H. (2006). Trends recognition in journal papers by text mining. In *IEEE International Conference on Systems, Man and Cybernetics 2006(SMC2006), pp.4784-4789*.