Journal of Physics: Conference Series

PAPER • **OPEN ACCESS**

Incremental Rapidly Grouping Aggregation Method for Similar Web News Headline

To cite this article: Shengze Hu *et al* 2020 *J. Phys.: Conf. Ser.* **1453** 012153

View the article online for updates and enhancements.

IOP ebooks™

Bringing together innovative digital publishing with leading authors from the global scientific community.

Start exploring the collection–download the first chapter of every title for free.

This content was downloaded from IP address 220.202.210.195 on 31/03/2020 at 05:21

Incremental Rapidly Grouping Aggregation Method for Similar Web News Headline

Shengze Hu[1], Chunhui He[1*], Chong Zhang[1], Bin Ge[1], Huiming Zhu[1] and Wei Liu[1]

[1]Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology Changsha, Hunan, P.R., China, 410073.

*Corresponding author's e-mail: xtuhch@163.com

Abstract. With the popularity of the Internet, real-time web news has been extensively used in human daily life. Because of competition and repetitive reporting between the media, there may be multiple similar news reporting the same searing event, which may waste a lot of time for users to obtain and read the similar news. Hence, how to rapid aggregating similar news to help users obtain relevant news information accurately is a very meaningful and essential study. However, the accuracy of the existing aggregation method needs to be strengthened, and it not meet real-time incremental processing requirements.

In this paper, rapid incremental aggregating a lot of similar news headlines are our main study targets. We proposed a novel Rapidly Grouping Aggregation (RGA) method. It carries out similar news headline aggregation via two core aspects --- Improved Jaro-Winkler (IJW) similarity calculation, normalized mapping and rapidly grouping aggregation. The IJW using hierarchical matching windows to enhance the performance of similarity calculation, and using the MD5 normalization strategy and the Group By function of the MongoDB database to carry out the similarity news headline rapidly grouping. Compared with the state-of-the-art method, the actual system application results show that the RGA method achieves higher performance on similar news headline aggregation tasks.

1. Introduction

With the rapid development of the Internet, simple and efficient short texts have gradually become an important information exchange carrier in human's life. Short text main including Weibo content, web news headlines, entity and relation [1], etc. Faced with these short texts, how to aggregate and mine them is a very challenging task.

In addition, the aggregation of actual large-scale web news headline is not as simple as the above example due to a number of challenges: (a) the similar web news headlines often exists little word-using inconsistent problem, it is a common problem in multiple sources heterogeneous short text data sets; (b) Comparison with the long text, the value of the web news headlines often contains colloquial words, it causes the short text aggregation more difficult than the long text; (c) news headlines belong to short text, so, the headlines' available features are poor; (d) public opinion systems contain lots of news, usually the number of web news headlines under one topic may reach million-level. If we directly to calculate each similarity news headline, it's very inefficient. Thus, it's a tricky problem that needs to solve.

For challenges (a) and (b), it reflects the dilemma issue facing the current short-text aggregation study. That is the object name may differ. It should be aggregated, but it not be aggregated correctly, or different objects with a very similar name may be incorrectly aggregated together. However, in recent

Content from this work may be used under the terms of the Creative Commons Attribution 3.0 licence. Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.
Published under licence by IOP Publishing Ltd

years, with the vigorous development of big data technology, it's attracted lots of attention from researchers [2]. Text aggregation methods are summarized into two categories. First is based on the statistical method. And second is using the machine learning method [3]. Existing statistical and machine learning methods they are not supporting the challenges of similar web news headlines incremental aggregation tasks. To deal with the challenges (a)-(d), we propose a rapid group aggregation (RGA) model based on retrieval technology and short text similarity calculation algorithm IJW. By the RGA method, it can rapidly aggregate similar web news headlines. The method contains three core steps: At first, based on the BM25 algorithm to get all the most relevant news headlines. Second, using a character-based IJW similarity algorithm and specific normalization strategy, it can rapidly obtain the most similar news headline and efficient to finish the construction of the index. Finally, the aggregation results will be obtained by the Group By aggregation function of the MongoDB database. The flow chart of the RGA method is indicated in Figure 1.

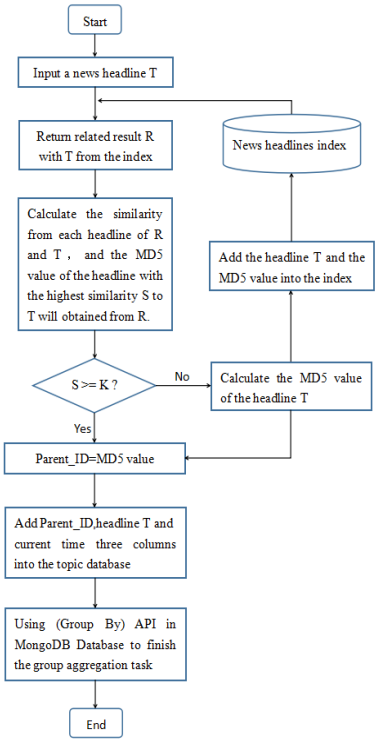


Figure 1. The flow charts of RGA method

In Figure 1, we use the inverted index technology to build the index on the web news headline. In our system, the similarity threshold K was set to 0.75, and it can get the best performance. Through the RGA method, we can map all similar news headlines to the same MD5 value and aggregate them together. In this paper, we mainly contributions as following:

1) As far as we know, it's the first time to utilize the integrated method built on a search engine, similarity calculation, and normalized mapping technology to solve the similar news headlines aggregation task.

2) We improved the JW algorithm by designing a hierarchical matching window and weight update method to get the IJW algorithm. It can enhance the similarity calculation accuracy for two strings.

3) We took place a normalization method founded on the MD5 Hash algorithm, BM25, and IJW algorithm to map all similar news headlines into the same MD5 value.

2. Related work

For the short text aggregation model, it is usually necessary to use a text similarity algorithm. The common string-based similarity algorithms generally divide into two categories [4]: Character-based

similarity algorithm and term-based similarity algorithm. The common character-based similarity algorithms include the Jaro-Winkler algorithm [5-9], Edit distance [10], Longest Common Sub-String (LCS) [11-12]. Common term-based similarity algorithms such as Cosine similarity [13-14] and Jaccard similarity [15]. For the above methods, the Jaro-Winkler algorithm is more suitable for calculating the short text similarity. Edit distance algorithms are usually used for fast fuzzy matching of short text [16]. Except for the similarity calculation with strings, and the LCS algorithm also has been used to measure the similarities between DNA molecules [17]. In addition to above the cases, some researchers have combined a semantic dictionary to propose a text similarity algorithm based on semantic similarities, such as HowNet [18] and WordNet [19].

According to the text-similarity algorithm, there are easy to calculate the similarity of two strings on the static small-scale dataset. But, in the public opinion system, one topic often contains lots of live update news headlines. To speed up the aggregation efficiency, indexing news headlines and using the index to improve performance is a feasible direction [20-21]. In document retrieval, the TF-IDF [22] or BM25 [23] algorithms are typically used.

For text aggregation tasks, studies find the relationship between text structure and aggregation through experiments [24]. With the development of the clustering method, using different clustering methods to discover a topic from the text also became a hot study [25]. Besides, a ranking aggregation method was proposed based on feature selection [26]. Particularly, for short text, and it's a very popular method to implement topic modeling through adaptive aggregation [27].

The large-scale text aggregation study results show that Simhash [28] algorithm performs well in the long text, but it's not suitable for short text aggregation tasks [29]. Consider the shortcomings of existing methods for solving the similar web news headline aggregation task, through combining search engine, text similarity calculation algorithms (IJW) and string normalization mapping and the Group By aggregate function of MongoDB database to carry out the rapidly grouping aggregation task.

3. Improved Jaro-Winkler similarity calculation method

To deal with the task of large-scale similar short text rapid aggregation, it can be better solved by combining BM25 [23] algorithm and string similarity calculation method.

According to the description of section 2, Jaro-Winkler and LCS algorithms are a common character-based short text similarity calculation method. Taking into account the particularity of the news headline, and to take advantage of them, we proposed an improved Jaro-Winkler (IJW) short text similarity calculation algorithm. First of all, the improved algorithm simultaneously considers the effect of the same Longest Common Sub-String prefix matching character and the same Longest Common Sub-String non-prefix matching the character on the similarity calculation. Besides because the design of the Jaro-Winkler algorithm matching window has some effect for calculating similarity. So, we using a hierarchical matching method to calculate the matching window. It includes a strong matching window and a weak matching window. If the character match occurs in the strong match window, it will set the weight to 1, but if the character match occurs within the weak match window, it will set the weight of 0.5. In this way, a more general matching window can be achieved. In any two strings s1 and s2, for any two characters distance less than the value of the matching window, it means the two characters are matched. According to the hierarchical matching method, the value of a strong matching window (SMW) and weak matching window (WMW) are calculated as follows:

$$SMW = \frac{\max(length(s_1), length(s_2))}{2} - 1 \tag{1}$$

$$WMW = \max(length(s_1), length(s_2)) - \text{index} \tag{2}$$

In formula (1) and (2), SMW is the value of strong matching window, WMW is the value of weak matching window, length(s1) is the length of String s1, length(s2) is the length of String s2, index is the value of current strong matching window end position, the WMW is a dynamic value. In the string matches window, we count the number of all matching characters and count as 1.0*m, and in the weak matching window, we count the number of all matching characters and count as 0.5*m. Then, the new

strings ms1 and ms2 are formed according to the order of each matching character in String s1 and s2. If the characters at the corresponding positions of ms1 and ms2 are different, this indicates that the two matched characters have undergone a transposition operation. The number of times of all transposition operations in the matching characters is counted and recorded as t_j, and the number of transposed characters t is half of the total number of transposition operations, recorded as $t = t_j/2$. Thence, the Jaro-Winkler similarity D_{jw} calculate formula is as follows:

$$D_{\mathrm{jw}} = \begin{cases} \frac{1}{3}\left(\frac{m}{length(s_1)} + \frac{m}{length(s_2)} + \frac{(m-t)}{m}\right), m > 0 \\ 0, m = 0 \end{cases} \tag{3}$$

It is worth noting that when the longest continuous matching characters exist in a string, the Jaro-Winkler similarity needs to finish fine-tuned, the formula is as follows:

$$D_{jw} = D_{jw} + (L * P(1 - D_{jw})) \tag{4}$$

Where p is a weight, p=min (0.1, 1.0/(max(length(s1), length(s2)))), L is the length of the longest common Sub-String in the two strings s1 and s2. The length of the LCS is calculated as follows:

$$C[i, j] = \begin{cases} 0, i = 0 \; or \; j = 0 \\ C[i-1, j, j-1] + 1, x_i = y_j \\ 0, x_i \neq y_j \end{cases} \tag{5}$$

Where C[i,j] recorded all the result of common Sub-String, the value of L is max(C[i,j]), and *i* represents *i-th* character index in s1, *j* is *j-th* character index in s2, x_i is *i-th* character of s1, and y_j is *j-th* character of s2. Using formula (1) to (5), it can be easy to calculate two Strings improved Jaro-Winkler similarity. In this section, matching windows' design step (formula (1), (2)), and weight update step (formula (4), (5)) are improved relative to the original Jaro-Winkler algorithm.

4. Data processing and rapid grouping aggregation

From flowchart of Fig 1., it is obvious that the RGA method main contains three modules: (1) Use retrieval ranking algorithm to get the most relevant web news headlines, and based on the retrieval result to finish the normalization of the input news headline; (2) Adopt improved Jaro-Winkler short text similarity algorithm to calculate the similarity of news headline, and through the contrast result of similarity to build index with the input news headline and MD5 value; (3) Utilize rapidly aggregation method to deal with the similar news headline. The data processing step and news headline rapid grouping aggregation step will be shown in Section 4.1 and 4.2.

4.1. News headline normalize and index construction

Considering that the web news in the public opinion system is dynamically updated in real-time, when the news has pushed to the topic, the similar news headlines of the topic may appear to change. To rapidly aggregate new pushed news headlines into similar collections, it is necessary to normalize and build an index for all the news headlines within the topic.

The normalization criterion is to drive similar web news headlines to have the same unique parent ID. This makes it easy to finish aggregation by the same Parent_ID with all news headline. In reality, usually using the MD5 value of news headlines as the unique Parent_ID, it is an effective way to distinguish between similar and non-similar news headlines. The normalization processing is as follows：

Step 1: Push a new web news headline into the specific topic;

Step 2: Use the headline to build a query, and utilize BM25 retrieve ranking algorithm to retrieval relevant news headlines already existing under the topic;

Step 3: According to step 2 return retrieval results, using IJW algorithm to calculate the max similarity S between the pushed news headline and the retrieval results list, at the same time, it can get the MD5 value of max similarity corresponding news headline from the results list;

Step 4: If the value of max similarity S >= K, it implies pushed news headline is existing a similar news headline in the topic, using the max similarity corresponding headline MD5 value as the Parent_ID

of the newly news headline;

Step 5: If S<K, it means the pushed new news headline is not existing similar headline in this topic, calculate MD5 value of the newly news headline as it's Parent_ID, and build an index with it.

By step 1 to 5, we can rapidly normalize for any newly pushed news headline. When placed in a split-new news headline, use a real-time incremental method to build an index. Depending to step 3, the max similarity will be obtained. When the max similarity is more than K, it implies index existing the similar news headline, and in this case, the pushed newly news headline no need to be added to the index. But, if the max similarity is lower than K, it means the pushed newly news headline should be to build an index, at the same time, add the Parent_ID in step 5 and web news headline at two different fields into the index. By this mechanism, it is guaranteed that there are not any similar headlines in the index. During the build index, we selecting the distributed index tool Elasticsearch [30] clusters with two nodes.

4.2. Rapid group aggregate method

Based on Section 4.1, for any newly pushed web news headline through the normalization processing, it will generate a unique Parent_ID of the newly pushed news headline. To implement a rapidly grouping aggregate target, it is necessary to store the information of the headline content, Parent_ID and current time into the Elasticsearch index database under this topic. Due to the task need to rapidly aggregate large-scale news headlines, one possible way is using MongoDB [31] to build distributed database clusters to store the data. Fortunately, there is a friendly API in the MongoDB database that can help us easily complete grouping aggregation tasks. The aggregate method is as follows:

Step 1: Put all the news headlines, Parent_ID and current time three columns into the same table in MongoDB database with the same topic;

Step 2: Call the Group By grouping aggregation function in MongoDB according to all Parent_ID, it will get the grouping aggregation result with the same Parent_ID;

Step 3: According to current time column do descend sort for the grouping results of Step 2;

Step 4: Return all the grouping aggregation results and the size of each group.

Step 1 to 4, it can be efficient to finish large-scale similar news headline rapidly aggregate task.

5. Experimental results and analysis

To effectively verify the performance of the method, we selected the public short-text similarity algorithm verification dataset of SemEval-STS[1] as the experimental dataset. The entire experiment was divided into two parts: (1) Extract some representative headline pairs from the SemEval-STS dataset to evaluate the performance of improved Jaro-Winkler (IJW) algorithm; (2) Use the SemEval-STS 2013 to 2016 dataset to evaluate the RGA method. The experimental results are shown in Section 5.1 and 5.2.

5.1. The evaluation of IJW algorithm

To evaluate the performance of the improved Jaro-Winkler algorithm, a set of comparative experiments was designed based on sub dataset with SemEval-STS. In the experiment, Jaro-Winkler (JW) and improved Jaro-Winkler (IJW) similarity algorithms were used to calculate the similarity of the same pair on selected sentence, and then the performance of the algorithm was measured by the difference between the calculated similarity value and the baseline similarity (baseline_Sim) values, they are linked above with gold human experts annotated scores[2]. Some samples of the sentence pair in SemEval-STS dataset are shown in Table 1.

Table 1. The Samples of sentence pair in the SemEval-STS dataset

id	sentence_A	sentence_B	baseline_Sim
1	coerce by violence, fill with terror	coerce by violence or with threats	4
2	behave or move in a certain manner	behave in a certain manner	4.4

[1] https://github.com/brmson/dataset-sts/tree/master/data/sts/semeval-sts

[2] http://alt.qcri.org/semeval2016/task1/index.php?id=data-and-tools

3	the person who is assigned to a non-elective position	a person who is appointed to a job or position	4.6
4	Give an incentive or reason for action	give an incentive for action	4.2
5	Preserve in a tin or jar	preserve in a can or tin	4.4
6	get involved with or mixed up	get involved or mixed-up with	4.6
7	remove by pinching or snipping	sever or remove by pinching or snipping	4.4
8	the act of providing incentive to someone	the act of motivating; providing incentive	4
9	read or conform to the metrical pattern of	conform to a metrical pattern	4.4
10	restrict or limit the range of	limit the range or extent of	4.8

Here, id is sentenced pair mark number, sentence_A is the first sentence, sentence_B is the second sentence, baseline_Sim is the baseline similarity with sentence_A and sentence_B that annotated by human experts. The baseline_Sim scores are between 0 and 5. It was given by the dataset of SemEval-STS. In Table 1, using the JW and IJW algorithm to compute the similarity of all samples. The results will be as shown in Figure 2.

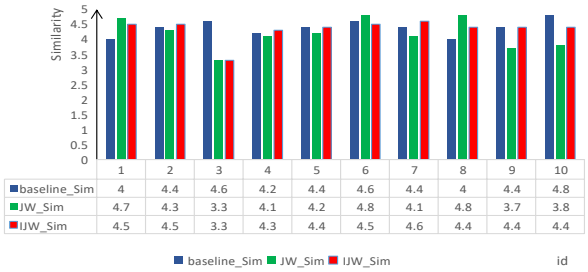


Figure 2. Similarity calculation results with a different similarity algorithm

According to the similarity calculate the results in Figure 2., we can find some differences between JW and IJW algorithm. To quantify the difference between the two algorithms, the offset is used to evaluate them. Centered on baseline_Sim, Using the absolute value of the discrepancy between baseline_Sim and JW_Sim or IJW_Sim as the offset value. According to the definition of the offset, it is easy to know they are an inverse relationship between offset and algorithm performance. The result of the offset value with the samples in Table 1, will show in Figure 3.

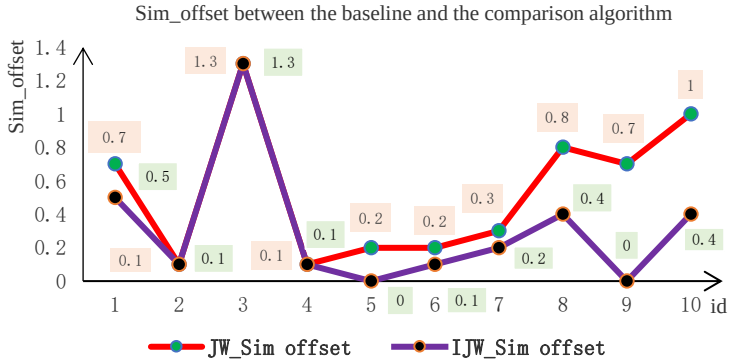


Figure 3. The Sim_offset between baseline algorithm and comparison algorithm

Figure 3 shows that the IJW algorithm has a smaller offset on the 10 samples pairs than the JW algorithm compared with baseline_Sim, which fully reveals the effectiveness of the IJW algorithm.

5.2 The evaluation of the RGA method

All above of the examples in section 5.1, it shows that the performance advantages of the IJW algorithm.

For the overall performance of the rapid grouping aggregation method, the further experiment is very necessary. By deeply analysis of dataset SemEval-STS, we select all sentence pairs from SemEval-STS 2013 to 2016, while the baseline_Sim value is not less than 4. Using all the selected sentence pairs after deduplication to build test data set. The test dataset statistic information is provided in Table 2.

Table 2. The statistics information of test dataset

Interval division of b_Sim	b_Sim=4	4<b_Sim<=4.5	4.5<Sim<5	Sim=5	Total
Number of sentence pairs	140	181	149	155	626

In Table 2, b_Sim means baseline_Sim. It means there are 626 different sentence pairs in the test dataset, and each sentence pair equivalent to one group. For each sentence pair, if the RGA method aggregates them into one group, it means the aggregation is succeeding; else it means no aggregation. For this case, we need statistics the total number of aggregated succeeded and using the accuracy to evaluate the performance of the aggregation algorithm. The accuracy calculation formula is as follows:

$$\text{Accuracy} = \frac{\text{total number of correct aggregation sentence pairs}}{\text{total number of sentence pairs}} \times 100\% \quad (6)$$

Considering that the rapid aggregation algorithm needs to use an index for retrieval and quickly complete normalization and aggregation. Therefore, the number of related documents returned in Hits will directly affect the performance of the aggregation. During the query process, when the return documents number is too small for Hits, it may filter the related documents, and causing the aggregation failure. But, if the return documents number is too big for Hits@M, it will increase the total time-consuming of similarity calculation. In summary, choosing a reasonable number of return documents has a great impact on the performance of the algorithm. In the test data set, we select different M (M=1,3,5,10,20,30) as the number of return documents for Hits to evaluate the effectiveness of the aggregation algorithm. The accuracy and average time-consuming of experiment with Hits@M as shown in Table 3.

Table 3. The experiment accuracy and average time-consuming of RGA with Hits@M

RGA Hits@M	Hits@1	Hits@3	Hits@5	Hits@10	Hits@20	Hits@30
Accuracy (%)	84.82	91.21	92.57	93.01	93.01	93.01
Average time-consuming (S)	16	17	17	18	21	23

According to Table 3, the average time-consuming means the average time spent in three times experiments. For the test data set, when M=1, it can get the lowest average time-consuming and the accuracy is 84.82%; When M=3 or M=5, the time-consuming rise was steady, and the accuracy increases to 92.57%. When M=10, the average time-consuming is 18 seconds and the accuracy received high level 93.01%; and when M= 20 or 30, the accuracy is changeless, but the average time-consuming is increased to 21 and 23 seconds. The experimental results show that the average time-consuming will increase as the M value increases, one possibility of increase is due to the increased time of similarity calculation. Besides, the accuracy will rise first as the M value increases and then keep stable when M to get to 10, for this case, we think that it is the effect of the BM25 scoring algorithm. It means that M between 5 and 10 is a better choice for the method because it works better and takes the least of time.

Here, we designed a comparative experiment to assess the method, using CNN [32] and glove[3] pre-trained word vectors and K-means[33] on short text clustering [34-37] tasks to compare with Hits@10 on the experimental dataset. The accuracy and average time-consuming is given in Table 4.

Table 4. The accuracy and average time-consuming compared with other methods

Method and evaluation index	CNN (glove)	RGA(IJW) Hits@10	RGA(JW) Hits@10	K-means (K=626)
Aggregation Accuracy (%)	87.4	**93.01**	91.14	85.78
Average time-consuming (S)	119	18	17.8	**10**
Support real-time incremental aggregation	Not	**Yes**	**Yes**	Not

According to Table 4, the experimental results show that the RGA method average time-consuming about 18 seconds, it is between K-means and CNN (glove). But, for the perspective of accuracy, the

[3] https://nlp.stanford.edu/projects/glove/

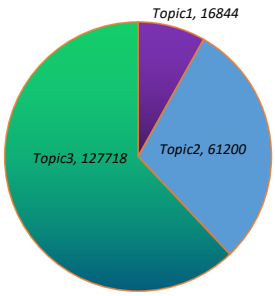
CNN and K-means with Squared Euclidean distance are 87.4% and 85.78%. However, our method is better than both of them, and it achieves 93.01% accuracy on RGA(IJW) Hits@10. Besides, the RGA(JW) Hits@10 also get 91.14% accuracy, it is slightly lower than RGA(IJW) Hits@10. For the adaptability of real-time incremental aggregation, because RGA method no model training process, so it is friendly supports real-time incremental operation. But, CNN and K-means methods rely on annotation data to complete model training, so it is impossible to meet the requirements of real-time incremental aggregation. Table 5 shows the value of core hyper-parameters in the CNN model.

Table 5. The value of core hyper-parameters in the CNN model

Embedding dim =300	Conv1D (100, 5, activation='tanh')	Dropout=0.5	lr=1e-3	Epoch = 25
optimizer='Adam'	loss='crossentropy'	epsilon=1e-08	beta_1=0.9	beta_2=0.999

6. The application of RGA method

To display the performance of the RGA. We applied the RGA method to real large-scale Chinese news datasets in a public opinion system. For the application test, three topics were selected from the public opinion system to complete the experimental evaluation of RGA. Topic1 was about the Middle East War, and it includes the total number of 16,844 documents; Topic2 was about the Aircraft Carrier, and it contains the total number of 61,200 documents; Topic3 was a topic about Korean Peninsula Nuclear Experiment, and it involves the total number of 127,718 documents. The statistical information for the documents of the three topics is as shown in Figure 4. RGA method aggregates the number of similar news groups under the three topics that will show in Fig 5.



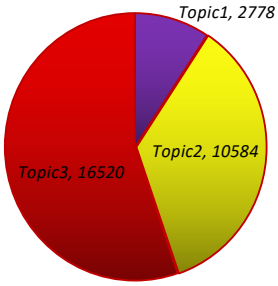


Figure 4. Number of documents in three topics Figure 5. Number of groups in three topics

In Figure 5, the groups it means more than one similar news headline under the topic. According to Fig 5, It can be seen that the rapid grouping aggregation method found 2778 similar web news headlines groups in Topic1, and 10584 similar web news headlines groups in Topic2; It also found 16520 similar news headlines groups in Topic3. The experiment results fully demonstrate the superior performance of the rapid grouping aggregation method in the real dataset application.

Parts samples of discovered similar news headlines under the three topics are as shown in Table 6.

Table 6. Parts of Chinese similar news headlines' aggregate results with RGA(IJW)

Topic	Similar news headline	Total Numbers	Publish date
Topic1	Iranian President Rohani responds to the US increased in the Middle East: no war will be waged against any country.	3	2019-06-18
Topic2	US aircraft carrier commander: coming to channel is not snoring just for deterring Iran.	2	2019-08-14
Topic3	US Defense Secretary Espour and Abe talks, and striving to achieve complete denuclearization of the Korean Peninsula.	2	2019-08-07

Table 6 shows that part of Chinese web news headline aggregation results on three different topics. It contains Chinese news headline content, the total number of similar news, and publication date 3 fields. According to Table 6, we can find that the RGA(IJW) method has achieved high performance on Chinese large-scale news headlines aggregation task. Also, the aggregation time-consuming on three topics is as shown in Figure 6:

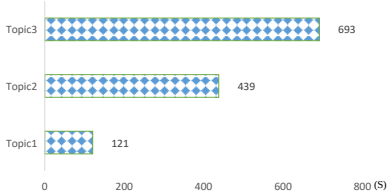


Figure 6. Aggregate time-consuming information statistics on three topics

Figure 6 shows that time-consuming will increase as the size of the data growth, but the growth trend will become more and more gradual.

7. Discussion

We recommend that the maximum similarity threshold K is better between 0.7 and 1.0 in our system. We define it to 0.75. Besides, the number of documents M returned by the query result Hits it is allowed to fine-tune according to different usage scenarios to optimize the performance of the RGA method. In this paper, why we propose the IJW algorithm? Mainly considering that the JW algorithm can't accurately calculate the same character beyond the matching window, and it also does not fully consider the length of common sub-string when doing the weight update. So, in IJW, we design the hierarchical matching windows and combined the length of strings' LCS to update weight. We use the Improved Jaro-Winkler algorithm to calculate the similarity, not the edit distance or other text similarity algorithm, mainly because IJW has a higher performance, and we also encourage readers to use other similarity algorithms to improve the performance of the RGA. Faced super large-scale aggregation tasks, by a time-sliced method to finish aggregation task may be a feasible approach. The RGA method not only suitable for the aggregation of large-scale web news headlines but also can apply it to entity disambiguation.

8. Conclusion and future work

According to the experimental results, the following main conclusions are drawn: (1) The performance of the IJW algorithm achieved better performance than the JW algorithm on the test dataset; (2) It is a feasible method by using MongoDB Group By aggregation function and adopting indexing-query strategy to rapidly aggregation a lot of similar news headline under one topic; (3) Normalization strategy it shows that blocky advantage in speed up grouping procedure; (4) The RGA method is ability to adapt to cross-lingual aggregation task. The RGA method shows superior performance in similar news headline aggregation tasks. In the future, we will use a more effective method to reduce the time-consuming on building an index. Finally, it's one of the directions to apply the RGA method to the question-answering system, recommendation system and the knowledge graph system.

Acknowledgment

This research was supported by the National Natural Science Foundation of China (NSFC) via grant No.61872446 and No. 61902417. Also included Natural Science Foundation of Hunan Province, China via grant No.2018JJ2475 and No.2018JJ2476. Special thanks to Dr. Zhen Tan, Associate Professor Yanli Hu, and Professor Weidong Xiao for their valuable suggestions during the article writing.

References

[1] Lin, Y., Liu, Z., Sun, M., Liu, Y., & Zhu, X.: (2015) Learning Entity and Relation Embeddings for Knowledge Graph Completion. Twenty-ninth AAAI Conference on Artificial Intelligence. AAAI Press.

[2] Shan, W., Ju, W. H., Pai, Q. X., Xuan, Z., & Beijing.: (2011) Architecting big data: challenges, studies and forecasts. Chinese Journal of Computers, 34(10), 1741-1752.
[3] Islam, A., & Inkpen, D.: (2008) Semantic text similarity using corpus-based word similarity and string similarity. Acm Transactions on Knowledge Discovery from Data, 2(2), 1-25.
[4] Gomaa, W., & Fahmy, (2013) A. A.: A survey of text similarity approaches. International Journal of Computer Applications,68(13).
[5] Jaro, M.: (1989) Advances in record-linkage methodology as applied to matching the 1985 census of tampa, florida. Publications of the American Statistical Association, 84(406), 414-420.
[6] Jaro, M. A.: (1995) Probabilistic linkage of large public health data files. Statistics in Medicine, 14(5-7), 8.
[7] Winkler, W. E.: (1990) String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage. Proceedings of the ERIC, 8.
[8] Nikolov, A., Uren, V., Motta, E., & De Roeck, A.: (2008) Integration of semantically annotated data by the knofuss architecture. Knowledge Engineering Practice & Patterns, 5268, 265-274.
[9] Agbehadji, Israel Edem, et al. (2018) The comparative analysis of Smith-Waterman algorithm with Jaro-Winkler algorithm for the detection of duplicate health related records. 2018 International Conference on Advances in Big Data, Computing and Data Communication Systems (icABCD). IEEE.
[10] Hall, P. A. V., & Dowling, G. R.: (1980) Approximate string matching. Acm Computing Surveys, 12(4), 381-402.
[11] Tasi, C. S., Huang, Y. M., Liu, C. H., & Huang, Y. M.: (2012) Applying vsm and lcs to develop an integrated text retrieval mechanism. Expert Systems with Applications, 39(4), 3974-3982.
[12] Hunt, J. W.: (1977) A fast algorithm for computing longest common subsequences. Communications of the Acm, 20(5), 350-353.
[13] Ye, J.: (2011) Cosine similarity measures for intuitionistic fuzzy sets and their applications. Mathematical & Computer Modelling,53(1), 91-97.
[14] Shrivastava, S. K., Rana, J. L. & Jain, R. C. (2013) Text document clustering based on phrase similarity using affinity propagation. International Journal of Computer Applications,61(18), 38-44.
[15] Deng, D., Li, G., Wen, H., & Feng, J.: (2015) An efficient partition based method for exact set similarity joins. Proceedings of the Vldb Endowment, 9(4), 360-371.
[16] Jupin, J., Shi, J. Y., & Obradovic, Z.: (2012) Understanding Cloud Data Using Approximate String Matching and Edit Distance. Sc Companion: High Performance Computing, Networking Storage & Analysis. IEEE Computer Society.
[17] Namiki, Y., Ishida, T, & Akiyama, Y.: (2012) Fast DNA Sequence Clustering Based on Longest Common Subsequence. Emerging Intelligent Computing Technology and Applications. Springer Berlin Heidelberg.
[18] Bin, Y., Yue-Song, Y. Ying-Ge, S., & Jing, L.: (2013) Method of information content evaluating semantic similarity on hownet. Computer Systems & Applications.
[19] Pedersen T, Patwardhan S, Michelizzi J.: (2004) WordNet: Similarity: Measuring the relatedness of concepts. In: Demonstration Papers at HLT-NAACL. Palo Alto: Association for Computational Linguistics, 38–41.
[20] Zhen, L., Jing, C., Jian-Bin, Z., etc.: (2017) Research on aggregation model for chinese short texts. Journal of Software.
[21] Cutting, D., & Pedersen, J.: (1989) Optimization for dynamic inverted index maintenance. Enformatika, 405-411.
[22] Zhai, C., & Lafferty, J.: (2004) A study of smoothing methods for language models applied to information retrieval. Acm Transactions on Information Systems, 22(2), 179-214.
[23] Chenxi, F., Lican H., Xueli L.: (2013) Research on scoring mechanism of bm25 model based on lucene. Industrial Control Computer.
[24] Konrad, A. M., & Harris, C.: (2000) Experimenting with the Interaction between Aggregation and

Text Structuring. The Workshop on Student Research (pp.1-6). Morgan Kaufmann Publishers Inc.
[25] Ayad, H., & Kamel, M. S.: (2002) Topic Discovery from Text Using Aggregation of Different Clustering Methods. Conference of the Canadian Society for Computational Studies of Intelligence on Advances in Artificial Intelligence (Vol.2338, pp.161-175). Springer-Verlag.
[26] Wu, O., Zuo, H., Zhu, M., Hu, W., Gao, J., & Wang, H.: (2009) Rank Aggregation Based Text Feature Selection. Ieee/wic/acm International Joint Conferences on Web Intelligence and Intelligent Agent Technologies. Wi-Iat (Vol.1, pp.165-172). IEEE.
[27] Quan, X., Ge, Y., & Pan, S. J.: (2015) Short and sparse text topic modeling via self-aggregation. International Conference on Artificial Intelligence (pp.2270-2276). AAAI Press.
[28] Charikar, M. S.: (2002) Similarity estimation techniques from rounding algorithms. Thiry-fourth Acm Symposium on Theory of Computing. ACM.
[29] Fu, X., Gao, M., Liu, Q., & Liu, H.: (2017) Group Detection for Specific Topic on Micro-Blog. Web Information Systems and Applications Conference, 2016 (pp.134-137). IEEE.
[30] Gormley, C., & Tong, Z.: (2015) Elasticsearch: the definitive guide. O'Reilly Media.
[31] Singh, Ajit. (2019) Data Migration from Relational Database to MongoDB. Global Journal of Computer Science and Technology.
[32] Xu, J., Xu, B., Wang, P., Zheng, S., Tian, G, & Zhao, J. et al.: (2017) Self-taught convolutional neural networks for short text clustering. Neural Networks, 88, 22-31.
[33] Bafna, P. Pramod, D., & Vaidya, A.: (2016) Document clustering: TF-IDF approach. 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), IEEE.
[34] Zheng, C. T., Qian, S., Cao, W. M., & Wong, H. S.: (2017) Locality-Sensitive Term Weighting for Short Text Clustering. In International Conference on Neural Information Processing (pp. 434-444). Springer, Cham.
[35] Zheng, C. T., Liu, C., & San Wong, H.: (2018) Corpus-based topic diffusion for short text clustering. Neurocomputing, 275, 2444-2458.
[36] Ji, Lei, et al. (2019) Microsoft Concept Graph: Mining Semantic Concepts for Short Text Understanding. Data Intelligence: 238-270.
[37] Chen, Yong, et al. (2019) Experimental explorations on short text topic mining between LDA and NMF based Schemes. Knowledge-Based Systems 163: 1-13.