Journal of Computations & Modelling, vol.9, no.1, 2019, 55-64
ISSN: 1792-7625 (print), 1792-8850 (online)
Scienpress Ltd, 2019

Distributed Retrieval of English Similar Sentences Based on the Edit Distance

Yanyun Li[1] Yanqin Yan[2] and Jun Ye[1]

Abstract

The English similar sentences retrieval plays an important role in the text classification, text replication detection, text clustering, and the information retrieval. It deserves our in-depth research and exploration.Traditionally similar sentences retrieval is basically based on a single node, but the single node similar sentences retrieval is less efficiency when the sample space is large. Therefore, this paper proposes a distributed retrieval method of the English similar sentences based on the edit distance. The optimization of the edit distance algorithm makes the search efficiency greatly improved.

Keywords: Improve edit distance, Similar sentences retrieval, Distributed search

1 Introduction

Similar sentences retrieval has a very wide application background in the field of natural language processing, such as fuzzy matching of sentences in the information filtering technology, original language retrieval based on the example machine translation, retrieval of frequently asked questions sets and matching of questions and answers in the automatic question answering technology, English bilingual writing based on the bilingual corpus. Therefore, for a long time, researching the problems of the similar sentences retrieval has always been people's passion.

[1] Department of Applied Mathematics, Nanjing University of Science and Technology, China.
[2] Computer Science and Technology, Communications University of Zhejiang, China.

Article Info: *Received* : October 30, 2018. *Revised*: December 10, 2018
Published online: February 20, 2019

At present, the research methods for the sentence semantic similarity calculation mainly include: methods based on the same vocabulary [1], methods using the semantic lexicon [2], met hods using the edit distance [3], and methods based on the statistics [4]. The method based on the same vocabulary has obvious limitations, and it is powerless to replace the synonyms. However, the method of semantic dictionaries may solve this problem well, using the method of semantic dictionaries simply, however, doesn't consider the interactions between the internal structure of the sentences and the words, and is at low accuracy. And edit distance is usually used in the field of fast fuzzy matching of sentences, but its prescribed edit operation is not flexible enough and does not consider the synonymous substitution of words. What's more, as the methods based on the statistical require a large amount of training corpus, the workload is very large with the problem of data sparse.

The edit distance was proposed by A. Leeveshtein in 1966 [5] to verify the degree of similarity between strings or texts, or as the minimum cost which is required to change one string from one atom to another. It is widely used in the similar sentences retrieval. Edit distance algorithm (LD algorithm or Levenshtein algorithm) is always used in the field of fast fuzzy matching of input strings, English assisted writing, etc. It is a classic and widely used method. Recently, the common methods for improving the edit distance algorithm is the combination of edit distance and Jaccard. The edit distance adds “replace” atomic operations in the Chinese context; in the application field, edit distance approximates the combination of the string and space to support multiple queries, and approximates the combination of string matching and the keyword retrieval on the database.

Presently, there are many researches on the Chinese similar sentences retrieval based on the edit distance algorithm, which mainly focuses on the optimizing the accuracy of the edit distance algorithm. Few scholars have studied the similar sentences retrieval problem in English, so this paper proposes a distributed retrieval structure of English similar sentences based on the edit distance, which greatly increase the search efficiency for English similar sentences space with a large amount of data.

2 Sentence Edit Distance Calculation

The traditional edit distance [5] refers that the minimum times of edit operations from the source string S to the target string T, which is in order to calculate the similarity between S and T. The main edit operations include inserting, replacing, and deleting characters to the string. That means, the minimum times of edit operations which is required to convert between the character string S and the character string T is recorded as the edit distance.

The edit distance between sentences refers that the minimum times of edit operations which is required to change from a word-based sentence to another word-based sentence. There are three kinds of edit operations: inserting, replacing, and deleting.

For two English sentences $Q_1Q_2\ldots Q_m$ and $P_1P_2\ldots P_n$, where m,n denote the number of English words contained in the sentences, Q_k denotes the k-th word in the English sentence $Q_1Q_2\ldots Q_m$, P_1 denotes the l-th word in the English sentence $P_1P_2\ldots P_n$ .Defined the function $edit(i,j)$, which represents the edit distance from the English sentence $Q_1\,Q_2\cdots Q_i(0\le i\le m)$ to the English sentence $P_1\,P_2\cdots P_j(0\le j\le n)$, draws the following dynamic programming formula:

(1)edit(i,j)=0, if i=j=0;

(2)edit(i,j)=j, if i =0 and 1≤j≤n;

(3)edit(i,j)=i,if j=0 and 1≤i≤m;

(4)edit(i,j)= $\min(edit(i\text{-}1,j)+1,edit(i,j\text{-}1)+1,edit(i\text{-}1,j\text{-}1)+f(i,j))$, if 1≤i ≤m and 1≤j≤n

,f(i ,j)= $\begin{cases}1, & \text{if } Q_i\neq P_j\\ 0, & \text{if } Q_i=P_j\end{cases}$.

Here is an example about how to calculate $edit(i,j)$. Assume that there are two English sentences “Each of us has a mooncake” and “We each have a mooncake”. It can be obtained from the formula above: edit(0,0)=0, edit(0,1)=1,…, edit(0,7)=7

, edit(1,0)=1,…,edit(6,0)=6, edit(1,1)=min(edit(1,0)+1,edit(0,1)+1,edit(0,0)+f(1,1)), because Q_1 is Each, P_1 is We, then we get f(1,1)=1, so edit(1,1)=1, calculate in turn ,then we get edit(6,7)=4, thus makes the following two-dimensional chart:

Table1: Example Edit Distance

		Each	of	us	has	a	mooncake
	0	1	2	3	4	5	6
We	1	1	2	3	4	5	6
each	2	1	2	3	4	5	6
have	3	2	2	3	4	5	6
a	4	3	3	3	4	4	5
mooncake	5	4	4	4	4	5	4

The following explains the rationality of the above dynamic programming formula:

(1) When i=j=0, edit(0,0) represents that the edit distance between two empty English sentences, is obviously edit(0,0)=0;

(2) When j=0,edit(i,j)=edit(i,0), indicates that the edit distance of the English sentence $Q_1Q_2…Q_i$ to the empty sentence. Obviously, only need to do I delete operation steps. So edit(i,j)=i;

(3) When i=0,edit(i,j)=edit(0,j), represents that the edit distance of the empty sentence to the English sentence $P_1P_2…P_j$. Obviously, it is only necessary to do j insert steps, so edit(i,j)=j.

(4) While$1\leq i \leq m$ and $1\leq j\leq n$, how to solve the edit distance from $Q_1Q_2…Q_i$ to, we $P_1P_2…P_j$ divided into three cases:

$Q_1Q_2…Q_{i-1}$ is converted to $P_1P_2…P_{j-1}$, then Q_i is converted to P_j, so edit(i,j)= f(i,j)+ edit(i-1,j-1);

$Q_1Q_2…Q_{i-1}$ is converted to $P_1P_2…P_j$, then we delete Q_i, so edit(i,j)=edit(i-1,j)+1;

$Q_1Q_2…Q_i$ is converted to $P_1P_2…P_{j-1}$, then we insert P_j, so edit(i,j)=edit(i,j-1)+1.

In summary, take the minimum of the three cases, which is edit(i,j)= min(edit(i-1,j)+1 ,edit(i,j-1)+1,edit(i-1,j-1)+f(i,j)),when $1\leq i \leq m$ and $1\leq j\leq n$,$f(i,j)=\begin{cases}1, & \text{if } Q_i\neq P_j\\ 0, & \text{if } Q_i=P_j\end{cases}$, As a result, we draw a general two-dimensional chart as follow:

Table2: Edit the calculation formula of distance matrix

		Q_1	Q_2	…	Q_m
	edit(0,0)	edit(0,1)	edit(0,2)	…	edit(0,m)
P_1	edit(1,0)	edit(1,1)	edit(1,2)	…	edit(1,m)
P_2	edit(2,0)	edit(2,1)	edit(2,2)	…	edit(2,m)
……	……	……	……	…	……
P_m	edit(n,0)	edit(n,1)	edit(n,2)	…	edit(n,m)

Lemma 1 *The edit distance from $Q_1Q_2…Q_i$ to $P_1P_2…P_j$ is edit(i,j)*

$\text{edit}(i\text{-}k,j\text{-}k) \le \text{edit}(i\text{-}k+1,i\text{-}k+1)$, where $1 \le k \le \max(i,j)$ (1)

Proof: From the dynamic programming formula we can see that

$\text{edit}(i,j)=\min(\text{edit}(i\text{-}1,j)+1,\text{edit}(i,j\text{-}1)+1,\text{edit}(i\text{-}1,j\text{-}1)+f(i,j))$, where $f(i,j)=\begin{cases}1, & \text{if } Q_i \neq P_j \\ 0, & \text{if } Q_i = P_j\end{cases}$,

When $Q_i=P_j$, f (i, j) =0, edit(i-1,j-1)=edit(i,j);

When k=1, $\text{edit}(i\text{-}1,j\text{-}1) \le \text{edit}(i,j)$, we use antithesis method, assuming edit(i-1,j-1)> edit(i,j), due to the edit distance is a non-negative number, $\text{edit}(i\text{-}1,j\text{-}1)\text{-edit}(i,j) \ge 1$, regardless of Q_i and P_j, is equal or not, edit(i-1, j-1) and edit(i,j) cannot be equal, and the assumption is not true, then $\text{edit}(i\text{-}1,j\text{-}1) \le \text{edit}(i,j)$ is proved.

Assume that when k=t, $\text{edit}(i\text{-}t,j\text{-}t) \le \text{edit}(i\text{-}t+1,j\text{-}t+1)$, when k=t+1, we still use the antithesis method, assume that edit(i-t-1,j-t-1)>edit(i-t,j-t),when $Q_{i-t}=P_{j-t}$, edit(i-t,j-t)= edit(i-t-1,j-t-1) and this contradicts the hypothesis, then when k=t+1, $\text{edit}(i\text{-}t\text{-}1,j\text{-}t\text{-}1) \le \text{edit}(i\text{-}t,j\text{-}t)$ is proved.From the Lemma above, we can see that the value of the diagonal of edit(i,j) from $Q_1Q_2…Q_i$ to $P_1P_2…P_j$ is an increasing number.

3 Sentence similarity calculation

After the edit distance calculation formula in the second part, we derive the sentence similarity formula. Assume that the English sentences $Q_1Q_2…Q_m$ and

$P_1P_2…P_n$, where m and n denote the number of English words contained in the sentence, edit(m,n) represents the edit distance between the above two sentences, then the similarity S(m,n) between the above two English sentences is calculated as follow:

$$S(m,n)=\frac{(\max(m,n)-\text{edit}(m,n))\times 100}{\max(m,n)}$$

From the edit distance calculation formula of the sentences, we known that $0\leq S(m,n)\leq 100$, for example of S(m,n), assuming that $Q_1Q_2…Q_m$ is "Each of us has a mooncake" and $P_1P_2…P_n$ is "We each have a Mooncake", from the second part we can see that edit(m,n)=4, and m=7,n=6, so

$$S(m,n)=\frac{(\max(7,6)-4)\times 100}{\max(7,6)}\cong 42.86$$

Lemma 2 *Given the English sentence $Q_1Q_2…Q_m$, for any two English sentences $P_1P_2…P_n$ and $T_1T_2…T_k$, suppose that the similarity between $Q_1Q_2…Q_m$ and $P_1P_2…P_n$ is S(m, n), if*

$$\max(m.k)\times\frac{(100-S(m,n))}{100}<\text{edit}(m-1,k-1) \qquad (2)$$

then there must be S(m,n)>S(m.k).

Proof: Lemma 1 shows that edit(m,k)≥edit(m-1,k-1), so

$$\frac{(\max(m,k)-\text{edit}(m,k))\times 100}{\max(m,k)}\leq\frac{(\max(m,k)-\text{edit}(m-1,k-1))\times 100}{\max(m,k)}$$

Due to,

$$\max(m,k)\times\frac{(100-S(m,n))}{100}<\text{edit}(m-1,k-1)$$

We get,

$$S(m,n)>\frac{(\max(m,k)-\text{edit}(m-1,k-1))\times 100}{\max(m,k)}$$

So,

$$S(m,n)>\frac{(\max(m,k)-\text{edit}(m,k))\times 100}{\max(m,k)}=S(m,k)$$

4 Similar sentences distributed retrieval

The traditional similar sentences retrieval refers that we calculated the similarity of each English sentence in the retrieval space to a fixed English sentence, and sorts according to the magnitude of the similarity, and outputs the sentence with the highest

magnitude of similarity, or output the first few sentences with high similarities.

However, peoples tended to focus on the algorithm itself and ignored the problem of efficiency of similar sentences retrieval. Therefore, this paper proposes a distributed retrieval scheme for the efficiency of similar sentences retrieval. Given the search space S, it contains K English sentences, the number of the node is N. The English sentences in the search space is divided into N equal parts so that the number of English sentences per node is $m_1, m_2, ... m_N$, and the subsearch space of each node is S_{m_i}, where i=1…N, and $\sum_{i=1}^{N} m_i$=K,The distributed search algorithm is as follow:

(1) Enter the English sentence L, that is, $L_1\ L_2 \cdots L_{|L|}$, where |L| represents the number of English words;

(2) For each subsearch space S_{m_i}, where i=1…N, max(S(|L|, S_{m_i})) is the maximum similarity degree between each subspace S_{m_i} and L, and note that the English sentences of the subspace is indexed as index for q=1,⋯, m_i:

Note that $P_{(q)_1}\ P_{(q)_2} \cdots P_{(q)_{|P_q|}}$ is the q-th English sentence in S_{m_i}, $|P_q|$ represents the number of English words. $\max\left(S\left(L,\ S_{m_i}\right)\right)$=S(L, $|P_q|$), index=q, S(|L|, $|P_q|$) is the similarity of L to $P_{(q)_1}\ P_{(q)_2} \cdots P_{(q)_{|P_q|}}$,

$$T=\max\left(|L|,|P_{q+1}|\right) \times \frac{\left(100\text{-}S(|L|,\ |P_q|)\right)}{100}$$

If,

edit(|L|-k, $|P_{q+1}|$-k)>T, while $0 \le K \le \max\left(|L|,|P_{q+1}|\right)$

As the lemma 2, S(|L|, $|P_{q+1}|$)< $\max\left(S\left(L,\ S_{m_i}\right)\right)$=S(|L|, $|P_q|$), continue

else

$$\text{if } \max\left(S\left(|L|,\ S_{m_i}\right)\right) < S(|L|,\ |P_{q+1}|)$$

$$\max\left(S\left(|L|,\ S_{m_i}\right)\right)=S(L,\ |P_{q+1}|),\ \text{index}=q+1$$

(3) max(S(|L|, S_{m_1}),S(|L|, S_{m_2}),⋯,S(|L|, S_{m_N}))) is the maximum similarity between

sentence L and search space S, thus the English sentence is P_t.

(4) Output $\max(S(L, S_{m_1}), S(L, S_{m_2}), \cdots, S(L, S_{m_N}))$ and P_t.

5 Experimental results and analysis

In this project, the number of English sentences in our search space is respectively one million, two million, four million, and forty million. The program is written in JAVA language. The distributed architecture is dubbo + zookeeper + tomcat. The specific results are shown in the figure below:

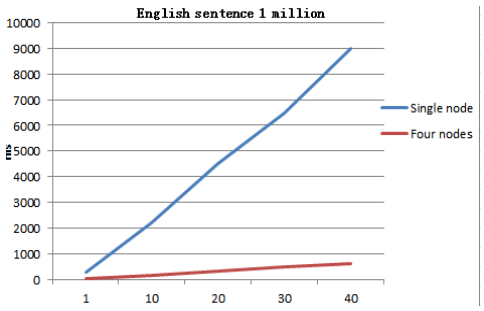

Figure 1: One million English sentences

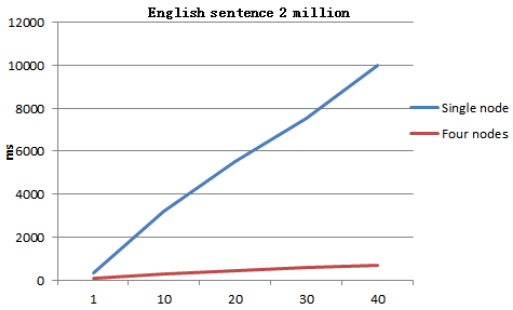

Figure 2: Two million English sentences

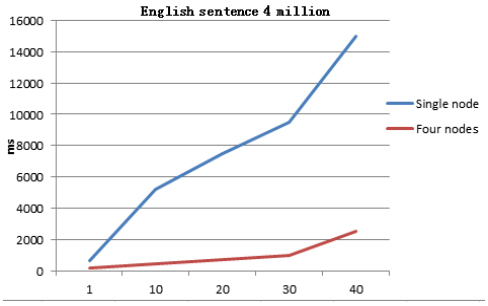

Figure 3: Four million English sentences

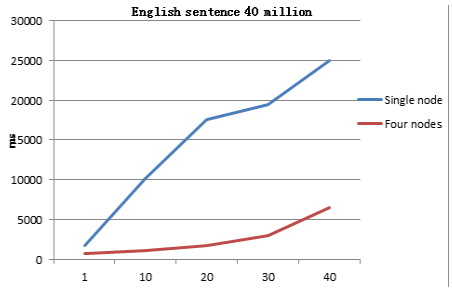

Figure 4: Forty million English sentences

From the results obtained from the experiments in Figure 1, Figure 2, Figure 3, and Figure 4, it can be concluded that the distributed search algorithm has more performance in the case of a certain data size, and the more English sentences it extracts, the better the performance is. It proved that the algorithm that was improved is feasible and efficient.

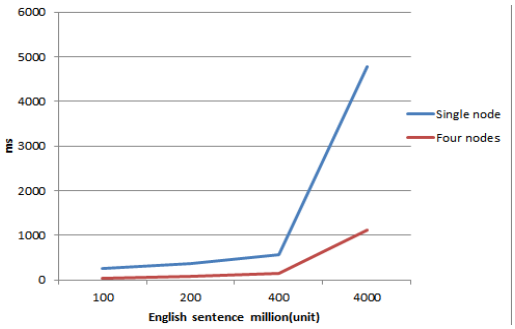

Figure 5: time consuming of optimal similarity sentences under different retrieval space scale

From the results obtained from the experiment in Figure 5, with the continuous expansion of the size of the search space, the distributed search algorithm is more and more effective when it retrieves the optimal similarity sentences.

In summary, the distributed search algorithm proposed in this paper is becoming more and more obvious as the search space becomes larger. Thus, the feasibility and effectiveness of the distributed search algorithm are obtained.

6 Conclusion

In this paper, a distance calculation algorithm for string edit is introduced to calculate the edit distance between English sentences, and the similarity calculation formula between English sentences is given. It is proved that when only the English string is considered while calculating the similarity and the normalization of exchange and other issues is not taken into consideration when calculate the similarity. An approximate matching algorithm for Chinese character strings based on improved edit distance and similarity is proposed. The improved edit distance algorithm improves the recognition accuracy and makes the approximate matching algorithm more practically applicable. At the same time, the experimental results of similarity comparison are given. The accuracy of the algorithm is verified by three evaluation indicators. Experimental results show that compared with the traditional algorithms, the improved algorithm has obvious advantages in precision, recall and average time-consuming, and improves the performance of the recommendation algorithm.

References

[1] Nirenburg S. Two approaches of matching in exampIe-based machine transIation. In: Proc TMI-93. Kyoto, Japan, 1993.

[2] Li S，Zhang J, et aI. Journal of Computer Science and Technology, 2002，17(6):933.

[3] Ristad E S, YianiIos P N. IEEE PAMI, 1998, 20(5): 522.

[4] Chatterjee N. A StatisticaI approach for simiIarity measurement between sentences for EBMT. 1999.

[5] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals,1966,163[4]: 707-710.