Clustering-based fusion for medical information retrieval

Clustering-Based Fusion for Medical Information Retrieval

Qiuyu Xu[a], Yidong Huang[a], Shengli Wu[a,b,*], and Chris Nugent[b]

[a]School of Computer Science, Jiangsu University, Zhenjiang, China

[b]School of Computing, Ulster University, Belfast, UK

*Corresponding author {Email: swu@ujs.edu.cn}

Abstract

Medicine is a fast-moving field, and the number of medical publications has increased rapidly over recent years. How to find relevant information from this vast pool of research effectively and efficiently has therefore become highly challenges. Previous studies have demonstrated that data fusion can improve search performance if properly utilized. However, in most cases effectiveness is the only concern and efficiency is not considered. A fusion-based system is by nature more complicated and expensive computationally than other retrieval models such as BM25, because many component retrieval systems and an extra layer of fusion are required. The number of component retrieval systems involved is an important indicator of complexity of the fusion-based system. We aim to select the optimal *k*-subset of component retrieval systems for any given number *k*, to optimize both fusion performance and reduce the cost of data fusion. A clustering-based approach is proposed. First all the candidates are divided into clusters by the Chameleon clustering algorithm, then representatives from every cluster are chosen by Sequential Forward Selection for fusion. Evaluated with two datasets from TREC, the proposed method performs more effectively than the other baseline methods including the state-of-the-art subset selection method significantly. When either of the two typical fusion methods is used, an improvement rate of over 10% is observed for both measures Mean Average Precision and Recall-level Precision, and an improvement rate of over 5% is observed for both measures Precision at 10 document level and Mean Reciprocal Rank.

Keywords: medical information retrieval, data fusion, subset selection, clustering, efficiency and effectiveness

1. Introduction

With the continuous development of information technology, the prevalence of mobile devices and high-speed wireless networks, the Internet has become the primary channel for people to obtain information. As a specific example, online medical literature has grown rapidly in recent years [35]. Taking this into consideration, how to provide an effective and efficient search service for such a vast information resource is therefore timely. To date, a variety of technologies have been used to improve medical search performance. Query expansion supported by external resources such as ontologies has been widely investigated [6, 11, 24]. A range of other approaches have

also been considered including pseudo-relevance feedback [37], multiple queries in combination [14], name entity recognition [22], multiple retrieval models and their fusion [48, 7, 29, 28], semantic and linguistic features and rules [29], topic modelling and classification [48, 28]. Many of these technologies are close in performance and there is no all-time winner. Therefore, the data fusion approach has been used in one way or another for many of the retrieval systems to achieve better retrieval performance [14, 22, 48, 7, 28].

Fusion has been widely used in many applications areas including biomedical and healthcare systems [10]. Data fusion in information retrieval is an approach that combines the multiple ranked lists of documents from different retrieval systems into one [3, 2, 41]. Previous studies have demonstrated that it is an effective approach for retrieval performance improvement, especially when many competitive retrieval systems are available. Many experiments demonstrate that some typical data fusion methods such as CombSum, CombMNZ, linear combination, can beat the best component retrieval systems in a variety of retrieval tasks [19, 40, 44, 5].

According to previous investigation [39, 43], the number of component systems has positive impact on fusion performance. However, it does not necessarily mean that it is always a good policy to fuse as many systems as possible for practical use, because it also impacts overall efficiency of the fusion-based system significantly. For example, consider the scientific abstracts task in the TREC (Text Retrieval Evaluation Conference) 2017 precision medicine track, there were a total number of 125 runs submitted. We may get more effective result if fusing all those runs properly. However, each run is from a different retrieval system and to obtain those 125 runs needs a lot of resources in the first place. Rather than fusing all available solutions, fusing a subset of 5, or 10, or 20 may be a more feasible solution. The question then becomes how it is possible to select a small group to improve fusion effectiveness. This question is not trivial given the number of combinations involved. Consider taking 5 from all 125 runs, there are 125*124*123*122*121/5! (more than 10^8) different combinations. Much more combinations exist if we try to take 10 or 20 from 125 runs. As a matter of fact, subset selection is a NP hard problem. Therefore, an exhaustive search can be viewed as being not practical, from a computational perspective, to find the best possible solution. Instead, more efficient solutions are desirable.

The objective of this work is to investigate fusion strategies in which fusion performance and efficiency are considered in tandem. More specifically, the research problem is to select a given number of systems from all those available to achieve the best possible fusion performance in the context of performing searches. It is understandable that the number of component retrieval systems is a good indicator of the complexity and efficiency of the entire fusion-based retrieval system.

To our knowledge, very limited research has been conducted on this issue. The only work was done by Juárez-González et al. [16]. In that piece of work, a new rank-based metric was defined. All retrieval systems were evaluated using that metric and the selection was based on that metric value. The limitation of that method is it only considers performance, while it ignores diversity in those component systems. Therefore, there is room for improvement. In this paper, we propose a clustering-based

approach. First it categorizes all the systems into a number of clusters, and we can expect that the systems are in the same cluster are similar while systems in different clusters are very different. Then a representative from each cluster is chosen to form a fusion group. In this way both performance of component systems and diversity of the chosen component systems can be considered at the same time.

An empirical investigation has been conducted to validate the proposed approach with two data sets for the medical information retrieval task. Because the proposed approach is generic, we believe that it is appliable to many other information retrieval tasks as well.

The major contributions of this paper are as follows:

1. A clustering-based method is presented to select a given number of component systems from all candidates for good fusion performance.
2. Experiments with two groups of runs submitted to TREC show that the proposed method is better than the two baseline methods involved.
3. Analysis is given to explain why the proposed method can achieve better performance than the state-of-the-art technology.

The remainder of this article is presented as follows. Section 2 discusses related work. Section 3 presents the clustering-based method for selecting a subset of information retrieval systems (runs) from a collection of them for fusion. Section 4 describes the experimental setup and evaluation results. Finally, Conclusions are drawn in Section 5.

2. Related Work

In this paper, we investigate how to apply data fusion to medical information retrieval. Medical information retrieval and data fusion are two key topics. Therefore, we review them separately.

2.1 Medical information retrieval

Medical information retrieval has attracted a lot of attention recently in various ways. It has been the subject for some major information retrieval evaluation events such as TREC[1] and CLEF[2]. Quite a number of different retrieval tasks have been taken, such as Genomics, Medical Records, Clinic Decision Support, Precision Medicine tracks in TREC, eHealth in CLEF, Medical Case-based Retrieval track in ImageCLEF, to name but a few. In 2014, a workshop of medical information retrieval was co-located with the annual ACM SIGIR conference [12]. In 2016, the information retrieval journal had a special issue (Volume 19, Number 1-2) on medical information retrieval with nine papers [13].

Due to the richness of synonyms and closely related terms in the biomedical domain, synonyms and semantic relatedness between terms are key research topics of many

[1] *https://trec.nist.gov*

[2] *https://www.clef-campaign.org*

publications. The Unified Medical Language System (UMLS; https://www.nlm.nih.gov/research/umls) and Medical Subject Headings (MeSH: https://www.ncbi.nlm.nih.gov/mesh) are two commonly used resources [17, 23]. Some other resources including Wikipedia, The International Classification of Diseases (ICD) [1], and independently constructed resources [46] have been used. Word embedding methods (such as using word2vec, GloVe, BERT, and others) may also be useful for this purpose.

Due to the same reason as previously mentioned, query expansion has been widely investigated. The UMLS was used in [4, 9] and MeSH was used in [24]. Durão et al. investigated query expansion through finding and evaluating neighboring terms of manually tagged keywords [11]. Wang et al. attempted pseudo-relevance feedback to expand user queries, in which Tensor factorization and the UMLS were used [37]. Chen et al. expanded queries through not only individual terms but also term combinations as entities, in which convolutional neural networks and self-attention mechanisms were used to estimate weights of entities [6]. Clipa & Nunzio evaluated a variety of technologies including multiple retrieval models & fusion, relevance feedback, and query expansion [7]. Soni & Roberts evaluated two commercial deep learning-based information retrieval systems for COVID-19 literature [34]. Maree et al. argued that any individual external resource was not enough [25]. Therefore, they used a combination of external resources including the MetaMap tool, MRDEF relational table, and the UMLS SPECIALIST lexicon for indexing and query expansion. These different query expansion techniques have their own merits on some specific queries. Combining them to achieve more effective results is the objective of the data fusion technology. See next subsection for detailed discussions.

Since 2017, TREC introduced the Precision Medicine track, which includes two types of documents for the search task: literature articles and clinical trials. In the 2017 TREC Precision Medicine track, 32 research groups participated in the event and with a total number of 258 submissions. In the 2018 TREC Precision Medicine track, there were 27 participants and a total number of 193 submissions [30, 31]. In this study, we focus on the search task of literature articles.

2.2 Data fusion

According to the type of input data to be used, data fusion methods can be divided into two categories: score-based methods and rank-based methods. CombSum [3], CombMNZ [3] are score-based methods, while Borda Count [2], Condorcet Fusion [26] are rank-based methods. Data fusion methods, depending on how each component system is treated, may also be categorized into equally treated or biased methods. Usually, a biased method requires some training data to decide the weights for all the component systems involved. CombSum, CombMNZ, Borda Count, Condorcet Fusion are equally treated methods, while linear combination [41] and weighted Condorcet [42] are biased methods. The performance of equally treated methods depends on the component retrieval systems chosen, while linear combination can achieve better performance if adequate weights are assigned to those component retrieval systems.

Juárez-González et al. investigated data fusion by selecting a small number of component retrieval systems [16]. A new rank-based metric was defined for the purposes of selection. They found that with CombSum and CombMNZ, fusing fewer candidates achieved better performance rather than fusing all possible candidates as is the norm in many cases.

The geometric framework [43] demonstrates that the performance of data fusion methods is related to the performance of all component results involved and the diversity amongst them. It demonstrates that: for methods such as CombSum and Borda count, it is possible to obtain better results by selecting a smaller number of component systems; on the other hand, for linear combination, if the best weights can be found, then fusing more component systems offers improved results. It has also proved that multiple linear regression is a very effective approach for weights assignment of the linear combination method [41].

Data fusion has been used in a variety of applications [5]. It is also very popular in information retrieval-related tasks, for example, question answering [19], image retrieval [21], opinion retrieval [40], plagiarism detection [32], retrieval results diversification [44], classification [47], and others. It is also popular for medical information retrieval tasks [7, 15, 27]. Nevertheless, in these studies, some commonly used data fusion methods such as CombSum, CombMNZ, Borda count, RRF (Reciprocal Rank Fusion) [8] have been evaluated.

Different from all but one above-mentioned research work [16], in this paper we investigate how to select a subset of component systems from a large collection of them for fusion. Although initially Juárez-González et al. only considered improving effectiveness and did not consider efficiency of the whole retrieval system, the concept of their method is attractive. However, their method only considers the effectiveness of component systems/results, while the diversity of all component systems is ignored. In this paper, we propose a clustering-based method, which considers both factors at the same time and is able to achieve better effectiveness than the one proposed in [16].

3. Proposed Method

Before we present our clustering-based method, we will briefly introduce the general process of data fusion and the data fusion methods which will be used in the ensuing experiments.

3.1. Data fusion methods and retrieval evaluation

Fig. 1 presents the process of data fusion within the context of information retrieval.

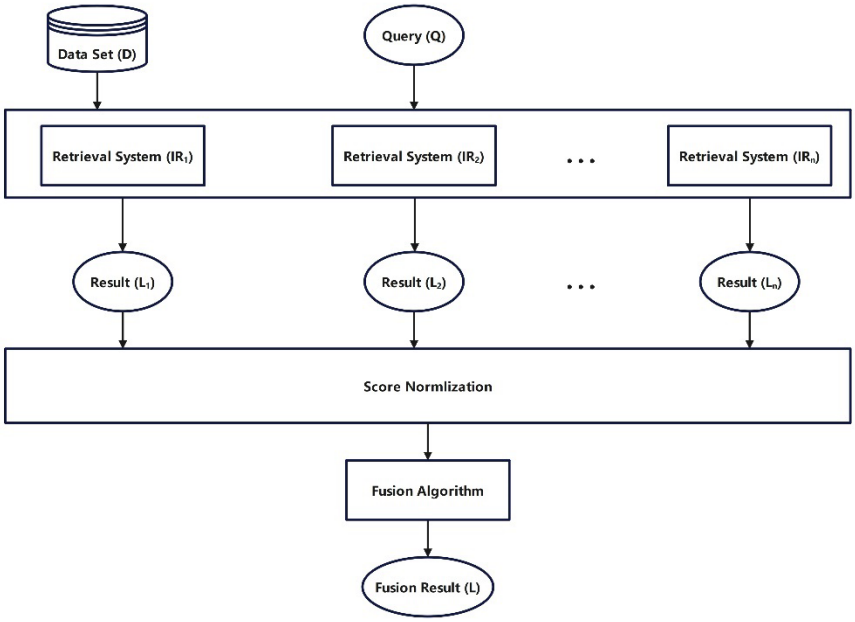


Fig. 1. The process of data fusion within the context of information retrieval

As shown in Fig. 1, for a collection of documents D and a user query q, there are n component systems IR_i ($1 \leqslant i \leqslant n$). Each component system performs a search and returns a list of documents. Thus we obtain n lists of results R_1, R_2, $\cdots$, R_n. Next, a fusion algorithm is used to merge the n lists to form the final outcome. In this work we consider three commonly used data fusion methods CombSUM, CombMNZ and linear combination whose weights are trained by multiple linear regression [41]. CombSUM uses the following formula to calculate the score of document d

$$g_{CombSum}(d) = \sum_{i=1}^{n} s_i(d) \tag{1}$$

where $s_i(d)$ is the score that component retrieval system IR_i gives to d, $g(d)$ is the global score that d obtains. Usually, $s_i(d)$ should be properly normalized to let $0 \leq s_i(d) \leq 1$. The final ranking of the documents is based on $g(d)$.

CombMNZ uses the following formula

$$g_{CombMNZ}(d) = m * \sum_{i=1}^{n} s_i(d) \tag{2}$$

where m is the number of component systems in which d is retrieved and obtains a positive score, or $s_i(d) > 0$.

Different from CombSUM and CombMNZ which treat all component systems equally, the linear combination method assigns variable weights to all the component systems involved. It uses the following formula to calculate score

$$g_{LN}(d) = \sum_{i=1}^{n} w_i * s_i(d) \quad (3)$$

where w_i is the weight assigned to component system IR_i. There are many ways of assigning weights. In this work, we take the multiple linear regression approach [41]. For the training data set, we require a collection D of l documents, a group of m queries Q, and n component systems. We have score (s_{jk}^i) for $i = (1, 2,\ldots, m)$, $j = (1, 2,\ldots, n)$, $k = (1, 2,\ldots, l)$. Here s_{jk}^i is the score assigned by retrieval system IR_j to documents d_k for query q^i. For the following equation

$$diff = \sum_{i=1}^{m}\sum_{k=1}^{l}\left[y_k^i - \left(w_0 + w_1 s_{1k}^i + w_2 s_{2k}^i + \cdots + \quad w_n s_{nk}^i\right)\right]^2 \quad (4)$$

where y_k^i is the judged relevance score of d_k for query q^i. The least squares optimization is to minimize the quantity of q, or the difference between the estimated score and the judged score of the documents. A group of weights $(w_1, w_2, \ldots, w_n)$ can be subsequently calculated.

Finally, let us consider the measure introduced in [16] for ranking and selecting results for fusion. It is referred to as J-measure in this paper. $w(rank)$, a weight for each rank position is defined as

$$w(rank) = 1 - \frac{\ln(rank)}{\ln|L|} \quad (5)$$

where $|L|$ is the length of the entire resultant list. L's J measure, $J(L)$, is defined as

$$J(L) = \sum_{i=1}^{|L|} w_i * rel(d_i) \quad (6)$$

where $rel(d_i)$=1 if the document at rank i is relevant, $rel(d_i)$=0 otherwise. For a group of resultant lists, each of them can be evaluated and ranked by its J measure, then those with higher J values will be selected for fusion. Their method is referred to as Top_J later in this paper.

In this study, we use MAP (Mean Average Precision over a group of queries), RP (Recall-level Precision), P@10 (Precision at 10 document level), and MRR (Mean Reciprocal Rank) to evaluate performance of retrieval results. All of them are commonly used metrics in retrieval evaluation. MAP and RP are system-oriented metrics, while P@10 and MRR are user-oriented metrics.

Let us assume for a collection of documents D and a group of queries $Q = \{q_1, q_2,\cdots, q_m\}$, an information retrieval system IR returns a ranked list L_i for each of the queries q_i. Average precision of L_i is defined as

$$AP(L_i) = \frac{1}{total_{r(q_k)}} \sum_{j=1}^{total_{r(q_k)}} \frac{j}{rank_{r(L_i,j)}} \quad (7)$$

where $rank\_r(L_i, j)$ is the ranking position of the j-th relevant document in L_i and $total\_r(q_k)$ is the total number of relevant documents in D for query q_k. MAP of L_1, L_2, $\cdots$, L_m is the mean AP values across all the queries. RP is defined as

$$RP(L_i) = \frac{num_{r\left(L_i, total_{r(q_k)}\right)}}{total_{r(q_k)}} \tag{8}$$

where $num\_r(L_i, total\_r(q_k))$ is the number of relevant ones in the top $total\_r(q_k)$ documents of L_i. RR is defined as

$$RR(L_i) = \frac{1}{rank_{r(L_i,1)}} \tag{9}$$

where $rank\_r(L_i, 1)$ is the ranking position of the first relevant document in L_i. $RR(L_i)$=0, if there are no relevant documents in L_i. MRR is the mean RR values over a group of queries.

3.2. Similarity measurement of two ranked lists

In information retrieval, in some cases, we need to measure the distance between two retrieval lists, or the degree of similarity between them [20]. In this work, we use rank-based overlap to measure the similarity between the two results lists [38]. It will be used for clustering resultant lists. According to [38], it is a good measure for our purpose.

For a ranked list of U, let U_i be its element at rank i, and $U_{i:j}$ be its elements from rank i to j. At depth d, the intersection of lists U and V is:

$$I_d = U_{1:d} \cap V_{1:d} \tag{10}$$

The agreement between U and V is the ratio of overlaps to depth d:

$$A_d = \frac{|I_d|}{d} = \frac{|U_{1:d} \cap V_{1:d}|}{d} \tag{11}$$

The RBO (rank-biased overlap) distance metric is defined as:

$$RBO(U, V, b) = (1 - b) \sum_{d=1}^{\infty} b^{d-1} \times A_d \tag{12}$$

Among them, b is an adjustable parameter, $0 < b < 1$. Finally, the similarity of two ranked lists is defined as

$$Sim(U, V) = \frac{1}{RBO(U,V,b)} \tag{13}$$

In this work, we set b to 0.9, the same as in [38]. $Sim(U, V)$ is used in the Chameleon clustering algorithm (See Algorithm 1 in the next subsection). It is also used as a component for the G metric (see Equation 17 in the next section), which aims to evaluates how good the clusters generated by the Chameleon clustering algorithm.

3.3. Chameleon hierarchical clustering

Chameleon clustering is a two-stage hierarchical clustering algorithm using dynamic models [38]. Considering the information between different clusters, it tries to generate

clusters in a way that maximizes the similarity within clusters and minimizes the similarity between two different clusters. The steps of the clustering algorithm are shown in Fig. 2.

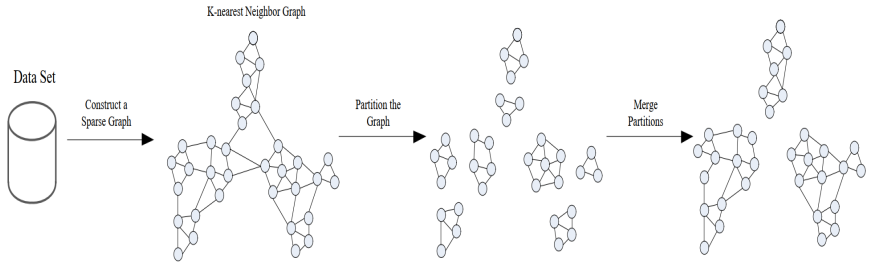


Fig. 2. The Process of Chameleon Hierarchical Clustering

In the first stage, for a group of objects (resultant lists in our case), we calculate their pairwise similarity according to Equation 13. By running k-NN (k-nearest neighbors), we obtain a sparse graph. Then hMetis, a graph partitioning algorithm, is used to divide the K-nearest neighbor graph into a large number of relatively small subclusters, so that the edge cuts are minimal.

In the second stage, sub-clusters are merged to larger clusters by agglomerative hierarchical clustering, in which two factors including inter-connectivity and their relative closeness are considered. The relative inter-connectivity between a pair of clusters C_i and C_j is defined as:

$$RI(C_i, C_j) = \frac{2\times|EC(C_i,C_j)|}{|EC(C_i)|+|EC(C_j)|} \tag{14}$$

where $|EC(C_i, C_j)|$ is the sum of the weights of all edges that connect vertices in C_i and C_j, and $EC(C_i)$ is the internal inter-connectivity of cluster C_i.

The relative closeness between a pair of clusters C_i and C_j is defined as:

$$RC(C_i, C_j) = \frac{(|C_i|+|C_j|)SEC(C_i,C_j)}{|C_i|SEC(C_i)+|C_j|SEC(C_j)} \tag{15}$$

where $|C_i|$ and $|C_j|$ are the number of elements in each cluster. $SEC(C_i)$ and $SEC(C_j)$ denote the average weights of all the edges that are of the minimal-cut bisector of each cluster. $SEC(C_i, C_j)$ denotes the average weight of the edges that connect both clusters. Taking into account *RI* and *RC* together, *F* is set as:

$$F = RI(C_i, C_j) \times RC(C_i, C_j)^{\alpha} \tag{16}$$

where α is a balance factor.

3.4. Sequential forward selection

Sequential forward selection is initially proposed to deal with the feature subset selection problem for various applications such as classification [36]. It is a greedy algorithm in nature for better efficiency, because in most cases to consider all

combinations is too costly. Now our problem is: for a given number of clusters, we would select one from each cluster to form a group for best possible fusion performance. This scenario is slightly different from the original one, the procedure involved also needs to adapt to the new situation.

Let us take an example to see how it works. Assume there are three clusters A, B, and C, each of which include 3 resultant lists, respectively. It is required to select one from each cluster to form a fusion group. First it selects the best performing one, say, a_1, in A. Then we fuse a_1 with each of lists in B. Assume that b_2 is the one in B that helps a_1 to achieve the best fusion performance. Both a_1 and b_2 are kept. Finally, we try to find one list in C that can help a_1 and b_2 to achieve the best fusion performance. This can be done by fusing a_1, b_2, and each of the lists in C. Assume that c_3 is the one to be found. As a result, we have a group including a_1, b_2, and c_3 as the solution. Fig. 3 illustrates the selection process for the above example. When trying to evaluate fusion performance of different combinations, we need the information of relevance judgment for the retrieved documents. Therefore, Sequential forward selection is a supervised method.

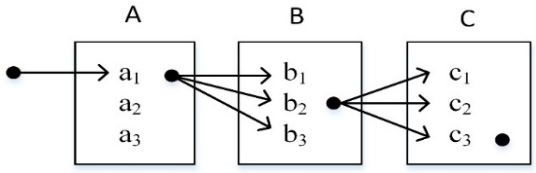


Fig. 3. An example of the sequential forward selection process

3.5. The proposed method for subset selection

The flowchart of the proposed method is shown in Fig. 4. It mainly includes two stages. For a group of n resultant lists ($L_1, L_2, \cdots, L_n$) as input, first we divide them into m clusters ($m \leq n$) using Chameleon. Each of them has t_i members ($1 \leq i \leq m$) and cluster i is composed of ($L_{i1}, L_{i2}, \ldots, L_{it_i}$). Then in the second stage we select one from each cluster to form a new group ($L_{1j_1}, L_{2j_2}, \ldots, L_{mj_m}$) for fusion using sequential forward selection. Note that clustering by Chameleon does not need any supervision while sequential forward selection does.

The detailed steps of the proposed method, C_S (Chameleon plus Sequential forward selection), are presented in Algorithm 1. It includes three parts. The first part (lines 1-6) constructs the sparse graph by using k-NN. It prepares some arrays required as input by Chameleon. The second part (line 7-22) generates a group of clusters by using Chameleon. Both parts 1 and 2 have been discussed in detail in Section 3.3; the third part (line 23-31) selects a given number of lists by sequential forward search, which has been discussed in Section 3.4. The time complexity of part one, two, and three is $O(|Q|mn^2)$, $O(n^2)$, $O(|C||Q|m)$, respectively. Here n is the number of runs, $|C|$ is the number of clusters generated, $|Q|$ is the number of queries, and m is the number of

documents in each resultant list for a query. Note that $|C|<n$; therefore, the time complexity of the entire algorithm is $O(|Q|mn^2)$. The actual time required for executing this algorithm and three data fusion methods will be given in next section.

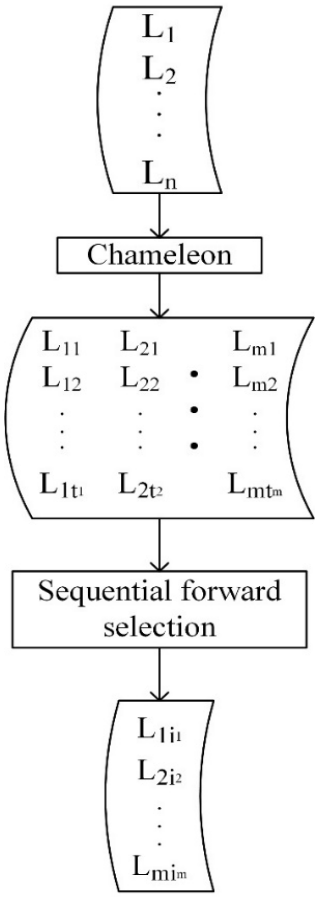


Fig. 4. Flowchart of the proposed method C_S

In the proposed method C_S, clustering is a major component, which impacts the performance of the whole algorithm significantly. If we can obtain good quality clustering results, then those component retrieval systems in the same cluster are similar, and retrieval systems in different clusters are different. It is very helpful for us to select a subset of retrieval systems with more diversity, which is a favorable condition for data fusion methods to achieve more effective results. Many clustering algorithms have been proposed [45]. Different clustering methods have different characteristics. However, for many of them, there is strong positive correlation between the complexity of the clustering algorithm and the quality of its clustering results. In our case, there are only several dozens of component systems. It is affordable to use a complex and expansive clustering algorithm to obtain very good results. That is why we choose Chameleon in this study.

Algorithm 1: Chameleon-based component system selection C_S

Input: *a set of lists $L=\{L_1, L_2, \ldots, L_n\}$, parameters k, α, p (k=1, α=1.5, p=2-20)*
Output: *S, a subset of L, is selected for fusion*
/ Part one: construct a sparse graph E for Chameleon */*
1: ***for every*** $i \in [1,n]$ ***do***
2: ***for every*** $j \in [1,n]$ *and* $j \neq i$ ***do***
3: $W(i,j) \leftarrow SIM(L_i, L_j)$ *// See Equation 13*
4: ***end for***
5: ***end for***
6: $E \leftarrow KNN(W,k)$ *//Calculate E using k-NN*
/ Part two: Clustering by Chameleon */*
7. $C_{init} \leftarrow \emptyset$; $C \leftarrow \emptyset$
8: ***for every*** $i \in [1,n]$ ***do***
9: ***if*** $L_i \in L$ ***then***
10: $C_{sub} \leftarrow hMetis(L_i, L, E)$ *//Generate a group of sub-clusters by hMetis*
11: $L \leftarrow L \backslash L_i$
12: ***end if***
13: $C_{init} \leftarrow C_{init} + C_{sub}$
14: ***end for***
15: ***while*** $|C_{sub}| > p$ ***do***
16: ***for every*** $j \in [i+1, |C_{sub}|]$ ***do*** *// Generate final clusters*
17: $F \leftarrow RI(C_{sub}^i, C_{sub}^j) \times RC(C_{sub}^i, C_{sub}^j)^\alpha$ *// See Equation 16*
18: ***end for***
19: $i,j \leftarrow arg\ min\{F\}$
20: $C_{temp} \leftarrow C_{sub}^i \cup C_{sub}^j$, $C \leftarrow C \cup C_{temp}$
21: $C_{sub} \leftarrow C_{sum} \backslash C_{sub}^j$, $C_{sub} \leftarrow C_{sub} \backslash C_{sub}^j$
22: ***end do***
/ Part three: select a given number of lists by sequential forward selection */*
23: $S \leftarrow \emptyset$
24: ***for every*** $i \in [1, |C|]$ ***do***
25: ***for every*** $L_{C_i}^j \in C_i$ ***do***
26: $tempS_j^i \leftarrow S \cup L_{C_i}^j$
27: $MAP_j^i \leftarrow Cal\ MAP(tempS_j^i)$ *//Calculate MAP value for fusion*
28: $m,n \leftarrow arg\ max\ \{MAP\}$ *//$L_{C_m}^n$ leads to the maximal fusion performance*
29: $S \leftarrow S \cup L_{C_m}^n$ *//add a new list to S*
30: ***end for***
31: ***end for***

4. Experiments

In this section we present experimental results to demonstrate the validity of the proposed method. First the experimental setting is presented, then the results are presented. Some discussion about the proposed method and some baseline methods are given. The time requirement for the implemented methods is also presented.

4.1. Setting and evaluation metrics

Experiments were conducted with two data sets in TREC to evaluate the proposed method [30, 31]. They used the same corpus, which is composed of 26,679,399 MEDLINE abstracts and 37,007 abstracts from the proceedings of the American Society of Clinical Oncology (ASCO), and 33,018 abstracts from the proceedings of the American Association for Cancer Research (AACR). 30 and 50 queries were used in the 2017 and 2018 track, respectively. All these queries were created by some experienced oncologists in the United States. In 2017, 29 participants submitted 125 runs, and a total number of 22,624 documents were judged by human experts. In 2018, 24 participants submitted 103 runs, and a total number of 22429 documents were judged by human experts for their relevance to the specific query.

To ensure that score normalization and data fusion methods work properly, we removed some of the runs with much fewer documents than the others. For the 2017 year group, most runs have 30,000 retrieved documents for 30 queries, or 1000 documents for each query. Those runs that have fewer than 20,000 documents are removed. For the 2018 year group, most runs have 50,000 retrieved documents. Those runs that have fewer than 30,000 documents are removed. Thus 108 out of 125, and 86 out of 103 submissions (runs) are selected for the year group of 2017 and 2018, respectively. The information of the two data sets is presented in Table 1.

Table 1. Data set used for the experiment (both are submissions to the scientific abstracts task in the precision medicine track)

Data set	Number of runs	Number of queries	MAP mean	MAP Variance	MAP min	MAP max
TREC 2017	109 (125)	30	0.1158	0.0030	0.0019	0.2327
TREC 2018	86 (103)	50	0.2079	0.0101	0.0001	0.3296

In each year, participants tried numerous techniques to boost retrieval performance. The difference exists in many aspects from the retrieval model used to some other components such as document representation, topic construction, query expansion, named entity expansion, boosting of some specific terms, and the like. Even for the same component, there are various ways of implementing it. For example, in 2017, The Medical University of Graz team used an open-source search engine Elasticsearch. Some other relevant strategies included query structuring, topic-oriented and document-oriented boosting, medical knowledge-based query expansion. The

University of Manchester team applied some methods of named entity recognition, topic representation, query construction, ontological expansion, demographic eligibility, and others. The Philips Research North America team took both strict rule match-based and Ontology-based approach. The team from the University of Chinese Academy of Sciences used both language modelling and BM25 search engines, pseudo relevance feedback, and word embedding. Through these examples we can see that all the runs were implemented with diversified technologies and the collection of them provide us a very good platform to carry out our investigation on data fusion.

Five different component result selection methods are generated. They are:

1. Selecting top-n results by their J value (refer to Equation 6, referred to as Top_J). It is a state-of-the-art subset selection method, proposed for data fusion in information retrieval [16].
2. Selecting top-n results by their MAP values (referred to as Top_MAP). It is a variant of TOP_J, proposed in this paper as a baseline method for comparison.
3. Clustering by Chameleon and then selecting one result from each cluster by sequential forward search (referred to as C_S). This is the main method proposed in this paper.
4. Clustering by Chameleon and then selecting one result with the best MAP value from each cluster (referred to as Cha). A naive variant of C_S, served as a baseline method.
5. A given number of component results are chosen randomly for fusion. It is served as a baseline. Given that its performance varies from one time to another. It is repeated 50 times and the average is taken. It is referred to as Average.

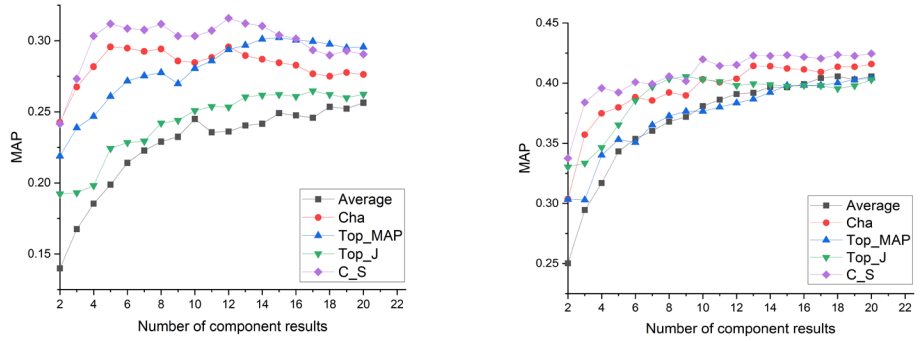
Three commonly used fusion methods, including CombSum, CombMZN, and linear combination are tested. Four metrics, including MAP (Mean Average Precision), RP (Recall-level Precision), P@10 (Precision at 10 document level), and MRR (Mean Reciprocal Rank) are used for evaluation. For all the documents in a resultant list, their scores are normalized by 1/(doc(*rank*)+60) [8]. According to [8] and our observation, it is a good method for converting ranking into scores. For Chameleon, k is set to 1, α is set to 1.5. 1 is the smallest possible value for k, which makes Chameleon fast. 1.5 is a typical value for α.

Apart from the baseline method "Average", all other methods require some training data to select a subset of results. Therefore, we divide all the queries in a year group (30 for TREC 2017 and 50 for TREC 2018) into two sub-groups: odd-numbered queries and even-numbered queries. Then two-fold cross-validation is used to test all the methods involved: one sub-group is used for training and the other sub-group for testing and vice versa. Besides, linear combination also needs training data to decide suitable weights. The same two-fold cross-validation methodology applies. Therefore, we use one group of queries for selecting a subset of systems/results and also for assigning weights of those selected systems/results, and the other group for testing all fusion methods including linear combination. Note in this study, it is not possible to use the data in one year group for training and the data in another year for testing, because those runs in different years were submitted by different retrieval systems. Two-fold cross-validation is an appropriate machine learning methodology in such a situation, and it

has been used by many researchers to evaluate data fusion methods in information retrieval before [2, 26, 33, 40, 41, 49]. It demonstrates that data fusion models are able to reach strong levels of performance when training on half the data (two-fold cross-validation), rather than larger subsets (e.g. 90% with 10-fold cross-validation).

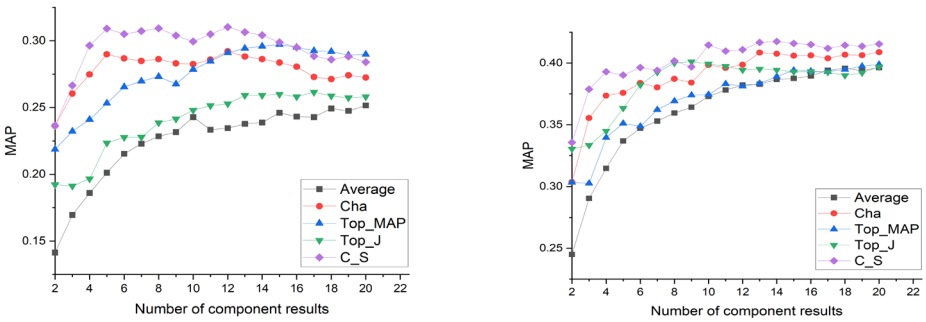
4.2. Results presentation and discussion

Figs 5-7 presents the experimental results (on MAP) of five subset selection methods with three different data fusion methods for various number of component systems. Average performance of fusing 2-20 component systems on MAP, RP, P@10, and MRR are presented in Tables 2-3. On average, C_S is the best performing approach and better than the others in most cases. From Tables 2-3, we can see that C_S performs consistently better than all the others apart from one case, in which TOP_MAP is better (2017, CombSum, MRR). Cha is the second best approach in many cases. All four methods are better than the Average baseline in most cases. The difference between C_S and Cha is similar across three different methods and two data sets. Average is the worst in most of the cases with a few exceptions.



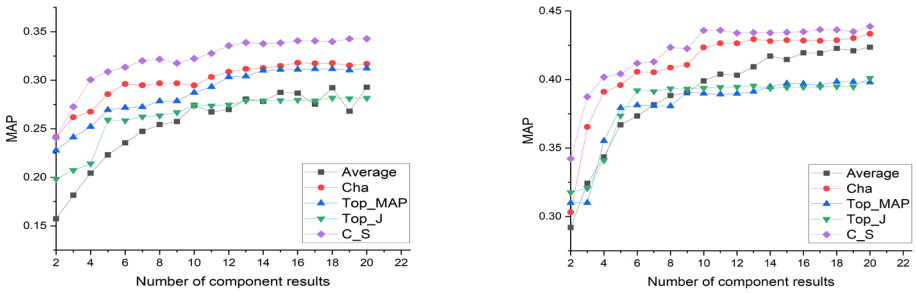
a. TREC 2017 b. TREC 2018

Fig. 5. Fusion performance of different component system selection algorithms (CombSum)



a. TREC 2017 b. TREC 2018

Fig. 6. Fusion performance of different component system selection algorithms (CombMNZ)



a. TREC 2017 b. TREC 2018

Fig. 7. Fusion performance of different component system selection algorithms (Linear Combination)

Table 2. Average performance of a group of methods (TREC 2017)

Method	CombSum				Linear Combination			
	MAP	RP	P@10	MRR	MAP	RP	P@10	MRR
Ave	0.2260	0.2774	0.5065	0.7648	0.2544	0.3055	0.5555	0.7784
Cha	0.2828	0.3290	0.6004	0.8304	0.2985	0.3512	0.6221	0.8412
Top_MAP	0.2794	0.3319	0.6181	**0.8592**	0.2873	0.3397	0.6236	0.8811
Top_J	0.2423	0.2902	0.5462	0.7716	0.2628	0.3153	0.5817	0.8326
C_S	**0.2991**	**0.3408**	**0.6207**	0.8529	**0.3212**	**0.3632**	**0.6395**	**0.8953**

Table 3. Average performance of a group of methods (TREC 2018)

Method	CombSum				Linear Combination			
	MAP	RP	P@10	MRR	MAP	RP	P@10	MRR
Ave	0.3695	0.4028	0.6375	0.8294	0.3902	0.4185	0.6685	0.8549
Cha	0.3938	0.4175	0.6559	0.8749	0.4104	0.4225	0.6991	0.8771
Top_MAP	0.3729	0.4014	0.6499	0.8616	0.3804	0.4088	0.6768	0.8545
Top_J	0.3872	0.4135	0.6676	0.8629	0.3825	0.4127	0.6834	0.8639
C_S	**0.4078**	**0.4264**	**0.6825**	**0.8851**	**0.4208**	**0.4367**	**0.7166**	**0.8914**

The best system in TREC 2017 is UTDHLTFF, whose MAP is 0.2327 for 30 queries; the best system in TREC 2018 is eth_a_ws_q, whose MAP is 0.3296 for 50 queries. Using C_S with CombSum, we can reach 0.3158 and 0.4246 in MAP. It indicates a considerable improvement of 35.71% and 28.82% over the best component result/system, respectively. More improvement is achievable for linear combination. When fusing 20 component systems, we can reach 0.3428 and 0.4383 in MAP. It

indicates a considerable improvement of 47.31% and 32.98% over the best component result/system, respectively. In this research, we have only explored a very small number of combinations. If more combinations are considered, it is possible to obtain even more effective fusion results at a higher computational cost. See Table 7 later for more details. Nevertheless, this current approach demonstrates the strength of data fusion methods.

1) Comparison between C_S and Cha

Comparing C_S and Cha, C_S performs better than Cha for all three fusion methods in both datasets. Although the difference between them is not big and varying from 1.17% (MRR, TREC 2018) to 6.43% (MRR, TREC 2017), the difference is significant at the .01 level (two-tailed T test) in all the cases. This is not surprising because the first step of C_S is the same as Cha. The second step, sequential forward search, is applied in C_S to further improve fusion performance. Without this second step, Cha does not have the chance to win over C_S.

2) Comparison between TOP_MAP and TOP_J

Comparing TOP_MAP with TOP_J, we can observe that they are close. TOP_MAP performs better than TOP_J for the TREC 2017 group, but it performs slightly worse than TOP_J for the TREC 2018 group. It is not surprising because both are based on the same underlying concepts: they attempt to select the most effective results. The difference is the way of measurement to decide the effectiveness of results: MAP or J measure. This can be confirmed by an alternative measure. For all component results in a dataset (either the 2017 or 2018 group), we calculate their MAP and J values and rank all of them by their MAP and J values, separately. Then we compare the two rankings. The Pearson correlation coefficient is 0.800 for the 2017 group and 0.953 for the 2018 group. In both cases, the correlation is strong. It indicates that the two rankings generated by MAP and J values are very similar, and a large percentage of the results selected by TOP_MAP and TOP_J are the same. This explains why the fusion results are similar for them.

3) Comparison between C_S and TOP_J and further analysis

Although it was published in 2010, TOP_J [16] is the state-of-the-art method for this problem. It is desirable to make a comparison between the proposed method C_S and TOP_J.

We observe that the former is better than the latter in all the cases. Refer to Table 4 for the improvement of C_S over TOP_J for each of the three methods over the two datasets. The differences (from 2.23% to 23.44%) are significant at the .05 level (Two-tailed T test).

Table 4. Improvement of C_S over TOP_J

Method/	CombSum				Linear Combination			
Data set	MAP	RP	P@10	MRR	MAP	RP	P@10	MRR
2017	23.44%	17.44%	13.64%	10.54%	22.22%	15.19%	9.94%	7.53%
2018	5.32%	3.21%	2.23%	2.57%	10.01%	5.82%	4.86%	3.18%

It has been found that the performance of component results, dissimilarity among all component results are two major factors that affect fusion performance (Wu & Crestani,

2015). Both TOP_MAP and TOP_J only consider the factor of performance of component results, however, do not consider the factor of dissimilarity among component results. This is why they are not as good as C_S. In the following we perform a quantitative analysis. Slightly different from the setting before, we use all 30 or 50 queries in TREC 2017 or 2018 for the training and test of both C_S and TOP_J. Such a setting is better for the comparison. Information about both aspects of all the component results involved are presented in Tables 5-6.

In Tables 5-6, MAP gives the average performance of all chosen component results in MAP for C_S or TOP_J, while Distance gives the average of all different pair-wise Euclidean distances of all component results. From Tables 5-6 we can see, in all but two cases, TOP-J selects those component results with higher MAP values than C_S does. On the other hand, in all the cases the average distance in C_S is longer than their counterpart in TOP_J. Putting these two factors together, we can see that C_S has larger MAP*Distance values than TOP_J in all the cases considered. This explains why C_S is a more effective method than TOP_J.

Table 5. Analysis of component results in C_S and TOP_J (TREC 2017)

Number of Results	C_S			TOP_J		
	MAP	Distance	MAP*Dis	MAP	Distance	MAP*Dis
2	0.2327	4.0625	0.9453	0.1877	3.8757	0.7273
3	0.2036	4.3372	0.8831	0.1989	2.7386	0.5447
4	0.1817	4.1470	0.7535	0.1994	3.0614	0.6103
5	0.1870	4.2206	0.7893	0.2000	2.8998	0.5801
6	0.1794	4.3404	0.7787	0.1964	3.0660	0.6022
7	0.1651	4.3283	0.7146	0.2016	3.1070	0.6264
8	0.1666	4.3649	0.7272	0.1964	3.1468	0.6180
9	0.1578	4.4118	0.6962	0.1922	3.1263	0.6009
10	0.1509	4.3981	0.6637	0.1954	3.1261	0.6109
11	0.1489	4.4873	0.6682	0.1954	3.1969	0.6246
12	0.1470	4.4593	0.6555	0.1954	3.1368	0.6128
13	0.1485	4.5116	0.6700	0.1923	3.1533	0.6063
14	0.1508	4.4687	0.6739	0.1923	3.1011	0.5963
15	0.1445	4.5185	0.6529	0.1895	3.0918	0.5859
16	0.1435	4.5943	0.6593	0.1887	3.1343	0.5913

Table 6. Analysis of component results in C_S and TOP_J (TREC 2018)

Number of Results	C_S			TOP_J		
	MAP	Distance	MAP*Dis	MAP	Distance	MAP*Dis
2	0.3246	3.3291	1.0806	0.3266	0.0733	0.0239
3	0.2996	3.6660	1.0983	0.3276	0.5003	0.1639
4	0.2893	3.6958	1.0692	0.3255	1.9107	0.6220
5	0.2556	4.0070	1.0242	0.3242	2.2448	0.7278
6	0.2658	3.8181	1.0149	0.3216	2.3504	0.7559
7	0.2738	3.6916	1.0108	0.3208	2.3350	0.7491
8	0.2630	3.7540	0.9873	0.3182	2.4238	0.7713
9	0.2498	3.9166	0.9784	0.3162	2.4534	0.7758
10	0.2489	3.9142	0.9742	0.3168	2.5798	0.8172
11	0.2457	3.9366	0.9672	0.3168	2.6330	0.8342
12	0.2343	4.0449	0.9477	0.3166	2.6308	0.8330
13	0.2365	4.0215	0.9512	0.3169	2.6534	0.8409
14	0.2387	3.9613	0.9456	0.3170	2.6708	0.8465
15	0.2379	3.9573	0.9414	0.3170	2.6611	0.8434
16	0.2418	3.9029	0.9437	0.3167	2.6400	0.8362

Table 7. The best performance for a given fusion method (TREC 2017)

Fusion / Selection	CombSUM		CombMNZ		Linear Combination	
	$N_{\max}$	MAP	N_{max}	MAP	N_{max}	MAP
C_S	12	**0.3158**	12	**0.3103**	20	0.3428
Cha	12	0.2957	12	0.2921	16	0.3180
Top_MAP	15	0.3023	15	0.2972	20	0.3124
TOP_J	17	0.2647	17	0.2615	20	0.2836
Full_List	108	0.2656	108	0.2590	108	**0.3605**

Table 8. The best performance for a given fusion method (TREC 2018)

Fusion / Selection	CombSUM		CombMNZ		Linear Combination	
	$N_{\max}$	MAP	N_{max}	MAP	N_{max}	MAP
C_S	20	**0.4246**	14	**0.4147**	20	0.4383
Cha	20	0.4158	20	0.4088	20	0.4333
Top_MAP	20	0.4042	20	0.3989	18	0.3984
TOP_J	9	0.4055	9	0.4012	20	0.4010
Full_List	86	0.4212	86	0.4107	86	**0.4463**

Table 9. Time required for training of data fusion methods in the 2018 dataset (unit: milliseconds)

Number of resultant lists	CombSum	CombMNZ	Linear Combination
2	7,516	7,570	21,952
3	9,001	9,025	29,282
4	9,941	9,929	33,266
5	10,913	10,933	37,910
10	22,069	22,093	121,474
15	29,721	29,729	185,310
20	40,295	40,531	303,898

Table 10. Time required for testing of data fusion methods in the 2018 dataset (unit: milliseconds)

Number of resultant lists	CombSum	CombMNZ	Linear Combination
2	74	76	96
3	99	108	114
4	114	120	135
5	158	164	174
10	244	251	268
15	313	316	339
20	376	381	422

Specifically, it is noticeable that in TREC 2018, when two or three results are fused, TOP_J has large MAP values, but it has very small Distance values (0.0733 and 0.5003) and therefore very small MAP*Distance values (0.0239 and 0.1639). The corresponding values for C_S are much larger (3.3291, 3.6660, 1.0806 and 1.0983), especially 3.3291/0.0733=45.42 indicates a large difference. It is interesting to see what really happened. For fusing two results, TOP_J chooses hpipubnone and hpipubcommon, both are very effective (0.3266 and 0.3265 in MAP), but they are very similar with a Euclidean distance of 0.0733. In TREC, each research group is allowed to submit multiple runs for the same task. Those runs submitted by the same research group is usually generated by the same information retrieval system with some difference in parameter setting, using optional/alternative components, and so on. The difference between them may be smaller than that of the runs submitted from different research groups. In fact, both hpipubnone and hpipubcommon are submitted by the same participant -- Hasso Plattner Institute in Germany (Oleynik et al., 2018). C_S chooses hpipubboost and UCASSA2. Their MAP values are 0.3296 and 0.3195, respectively. Hpipubboost is as good as hpipubnone and hpipubcommon, also from Hasso Plattner Institute. UCASSA2's MAP value is slightly lower than hpipubnone, hpipubcommon, and hpipubboost. Nevertheless, because it is from another participant -- University of Chinese Academy of Sciences, the difference between it and

hpipubboost is much larger than that of hpipubnone and hpipubcommon. Therefore, we can expect an effective fusion result from C_S (0.3966, CombSum) than TOP_J (0.3267, CombSum). The situation is very similar for fusing three results. TOP_J chooses hpipubnone, hpipubcommon, and hpipubboost, all of which are very effective, however, very similar because they are from the same participant; C_S takes hpipubboost, UCASSA2 and MSIIP_PBL (0.2497 in MAP), which are less effective (especially MSIIP_PBL) but all are from different participants. Their difference is larger. The fusion performance of CombSum is 0.3306 for TOP_J and 0.4155 for C_S. From these two examples, we can observe more clearly why C_S is better than TOP_J.

4) The effect of number of component results

Now we consider how the number of component results affect fusion performance. Tables 7-8 show when the best fusion performance is achieved for a given result selection method with a fusion method. The fusion performance for all available component results is also shown as Full_list. It is noticeable that the number of component results and fusion performance has a strong positive correlation. For example, the Pearson coefficient is 0.854 for C_S with CombSum in TREC 2017. On the other hand, it also means that more component results do not always lead to better fusion performance for both CombSum and CombMNZ. This is because fusion performance is decided by multiple factors and the number of component results is just one of them. Therefore, carefully choosing a subset of component systems is a meaningful undertaking for us to obtain more effective fusion performance efficiently, especially when CombSum and CombMNZ are used for fusion.

The situation is different for linear combination. According to the geometric framework [43], better fusion results are achievable if we add more component results using linear combination with the optimal weights. We can observe that linear combination achieves the best fusion result with all the component results. This can be used as a target for data fusion methods to achieve, but with fewer component results. In this experiment, for the 2017 year group, we reach 95% of the target value (0.3428/0.3605) by using less than 1/5 of the systems (20/108); for the 2018 year group, we reach 98% of the target value (0.4383/0.4463) by using less than 1/4 of the systems (20/86). Such a finding is useful for making judicious decisions in practical use.

5) Time Required

In this subsection we look at the time required for the training and test of three data fusion methods. A personal computer with an i7-10700 CPU and 16G main memory is used for it. Tables 9-10 show the training and testing time of data fusion methods with the 2018 dataset, respectively.

For both training (mainly using Algorithm 1) and testing, 25 queries are processed. The time shown is for all the queries together. We can see that CombSum takes slightly less time than CombMNZ and Linear combination takes slightly more time than CombMNZ for the testing, but the difference between them is small. The same for the training but the difference is bigger. This is because for linear combination, it needs to calculate optimal weights for each new combination found in the stage of sequential forward selection. The time for training includes read and write operations of files on hard discs as input and output, which accounts for a large percentage of the whole time

required. The time required for testing is much less than the time for training. Take CombSum as an example, it uses 376 milliseconds for fusing 20 resultant lists of 25 queries, which is equivalent to 15.04 milliseconds per query. Such a cost should be acceptable in many cases.

6) Comments

In this paper we have demonstrated that in the context of medical information retrieval, the proposed method C_S is able to achieve improved retrieval performance. It is better than the state-of-the-art technology TOP_J by a clear margin. However, for supervised subset selection methods such as C_S and TOP_J, considerable effort is required to make them workable. Mainly we need:

A training data set includes a collection of documents, a group of queries, and relevance judgment (it indicates which document is relevant to which query).

Search results from a relatively large number of information retrieval systems which are implemented using different technologies ideally.

Fortunately, TREC provides all the information needed and two above conditions are satisfied. However, how to meet these two requirements may be a challenging issue under some other circumstances.

In both TREC 2017 and 2018 precision medicine tracks, dozens of research groups submitted over 100 runs to the literature articles task and diversified technologies were used in their search systems. Many competitive ones represent the up-to-date technologies in the domain of medical information retrieval. The proposed method, C_S, performs better than the best run by a clear margin in both data sets. This is good evidence that data fusion, especially the proposed method, can be very useful for medical information retrieval if affordable.

5. Conclusions

We have proposed a method employing clustering-based component system selection to efficiently optimize fusion performance. Coupled with sequential forward search, we can choose a given number of component systems to achieve improved fusion performance. The approach simultaneously optimizes for both the performance of component systems and their dissimilarity (or diversity). Four commonly used metrics show the proposed method C_S significantly outperforms TOP-J, a state-of-the-art subset selection method, based on two TREC data sets in the precision medicine track. When CombSum is used for fusion, C_S is better than TOP-J by 14.38%, 10.33%, 7.94%, and 6.56% in MAP, RP, P@10, and MRR, respectively; when linear combination is used for fusion, C_S is better than TOP-J by 16.12%, 10.51%, 7.40%, and 5.36% in MAP, RP, P@10, and MRR, respectively. When fusing 20 component systems using linear combination, a considerable improvement of 40.15% over the best component system is achieved in MAP. Such results demonstrate the potential of using data fusion to improve medical information retrieval.

In our future work, we plan to further investigate the relationship between component system performance and dissimilarity among component results. A more precise relationship would enable more efficient and effective system selection methods for

fusion. The geometric framework [43] provides a suitable platform for us to do this. Another approach would be to design an unsupervised version of C_S. At present, generating a usable training dataset can be very costly because relevance judgment by human referees is required for those retrieved documents. If some automatic performance estimation methods can be applied instead, then its usefulness can be improved considerably.

References

[1] Amini, I., Martínez, D., Li, X. & Sanderson, M. (2016). Improving patient record search: A meta-data based approach. Information Processing & Management, 52(2), 258-272.

[2] Aslam, J. A. & Montague, M. H. (2001). Models for Metasearch. In Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 275-284).

[3] Bartell, B. T., Cottrell, G. W. & Belew, R. K. (1994). Automatic Combination of Multiple Ranked Retrieval Systems. In Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 173-181).

[4] Bhatt, M., Rahayu, J. W., Soni, S. P. & Wouters, C. (2009). Ontology driven semantic profiling and retrieval in medical information systems. Journal of Web Semantics, 7(4), 317-331.

[5] Canalle, G. K., Salgado, A.C. & Loscio, B. F. (2021). A survey on data fusion: what for? in what form? what is next? Journal of Intelligent Information Systems, 57(1), 25-50.

[6] Chen, S., Hu, Q. V., Song, Y., He, Y., Wu, H. & He, L. (2019). Self-Attention based Network for Medical Query Expansion. In proceedings of IEEE International Joint Conference on Neural Network, pp. 1-9.

[7] Clipa, T. & Nunzio, G. M. D. (2020). A Study on Ranking Fusion Approaches for the Retrieval of Medical Publications. Information, 11(2), 103.

[8] Cormack, G. V., Clarke, C. L. A. & Büttcher, S. (2009). Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 758-759).

[9] Díaz-Galiano, M. C., Martín-Valdivia, M. & López, L. A. U. (2009). Query expansion with a medical ontology to improve a multimodal information retrieval system. Computers in Biology and Medicine, 39(4), 396-403.

[10] Domingues, I., Müller, H., Ortiz, A., Dasarathy, B. V., Abreu, P. H. & Calhoun, V. D. (2020). Guest Editorial: Information Fusion for Medical Data: Early, Late, and Deep Fusion Methods for Multimodal Data. IEEE Journal of Biomedical and Health Informatics, 24(1), 14-16.

[11] Durão, F. A., Bayyapu, K., Xu, G., Dolog, P. & Lage, R. (2014). Expanding user's query with tag-neighbors for effective medical information retrieval. Multimedia Tools and Applications, 71(2), 905-929.

[12] Goeuriot, L., Jones, G. J. F., Kelly, L., Müller, H. & Zobel, J. (2014). Proceedings of the Medical Information Retrieval Workshop at SIGIR co-located with the 37th annual international ACM SIGIR conference (ACM SIGIR 2014). In CEUR Workshop Proceedings 1276.
[13] Goeuriot, L., Jones, G. J., Kelly, L., Müller, H. & Zobel, J. (2016) Medical information retrieval: introduction to the special issue. Information Retrieval Journal, 19(1/2), 1–5.
[14] Goodwin, T. R., Skinner, M. A. & Harabagiu, S. M. (2017). UTD HLTRI at TREC 2017: Precision Medicine Track. In Proceedings of The Twenty-Sixth Text REtrieval Conference, Gaithersburg, Maryland, USA.
[15] Herrera, A. G. S., Schaer, R., Markonis, D. & Müller, H. (2015). Comparing fusion techniques for the ImageCLEF 2013 medical case retrieval task. Computerized Medical Imaging and Graphics, 39, 46-54.
[16] Juárez-González, A., Montes-y-Gómez, M., Pineda, L. V., Avendaño, D. P. & Pérez- Coutiño, M. A. (2010). Selecting the N-Top Retrieval Result Lists for an Effective Data Fusion. In Proceedings of the 11th International Conference of Computational Linguistics and Intelligent Text Processing (pp. 580-589).
[17] Kang, T., Perotte, A. J., Tang, Y., Ta, C. N. & Weng, C. (2021). UMLS-based data augmentation for natural language processing of clinical research literature. Journal of the American Medical Informatics Association, 28(4), 812-823.
[18] Karypis, G., Han, E. & Kumar, V. (1999). Chameleon: Hierarchical Clustering Using Dynamic Modeling. IEEE Computer, 32(8), 68-75.
[19] Kato, S., Shimizu, T., Fujita, S. & Sakai, T. (2019). Unsupervised Answer Retrieval with Data Fusion for Community Question Answering. In Proceedings of the 15th Asia Information Retrieval Societies Conference (pp. 10-21).
[20] Kumar, R. & Vassilvitskii, S. (2010). Generalized distances between rankings. In Proceedings of the 19th International Conference on World Wide Web (pp. 571-580).
[21] Li, Y., Kong, X., Fu, H. & Tian, Q. (2020). Node-Sensitive Graph Fusion via Topo-Correlation for Image Retrieval. IEEE Transactions on Circuits and Systems for Video Technology, 30(10), 3777-3787.
[22] Ling Y. et al. (2017). A Hybrid Approach to Precision Medicine-related Biomedical Article Retrieval and Clinical Trial Matching. In Proceedings of The Twenty-Sixth Text REtrieval Conference, Gaithersburg, Maryland, USA.
[23] Liu, Y. & Wacholder, N. (2017). Evaluating the impact of MeSH (Medical Subject Headings) terms on different types of searchers. Information Processing and Management, 53(4), 851-870.
[24] Lu, Z., Kim, W. & Wilbur, W. J. (2009). Evaluation of query expansion using MeSH in PubMed. Information Retrieval, 12(1), 69-80.
[25] Maree, M., Noor, I., Rabayah, K., Belkhatir, M. & Alhashmi, S. M. (2020). On the Combined Use of Extrinsic Semantic Resources for Medical Information Search. CoRR, abs/2005.08259.
[26] Montague, M. H. & Aslam, J. A. (2002). Condorcet fusion for improved retrieval. In Proceedings of the 2002 ACM CIKM International Conference on Information and Knowledge Management (pp. 538-548).

[27] Mourão, A., Martins, F. & Magalhães, J. (2015). Multimodal medical information retrieval with unsupervised rank fusion. Computerized Medical Imaging and Graphics, 39, 35-45.
[28] Oleynik, M. et al. (2018). HPI-DHC at TREC 2018 Precision Medicine Track. In Proceedings of The Twenty-seventh Text REtrieval Conference, Gaithersburg, Maryland, USA.
[29] Pasche, E. et al. (2017). Customizing a Variant Annotation-Support Tool: an Inquiry into Probability Ranking Principles for TREC Precision Medicine. In Proceedings of The Twenty-Sixth Text REtrieval Conference, Gaithersburg, Maryland, USA.
[30] Roberts, K. et al. (2017). Overview of the TREC 2017 Precision Medicine Track. In Proceedings of The Twenty-Sixth Text REtrieval Conference, Gaithersburg, Maryland, USA.
[31] Roberts, K., Demner-Fushman, D., Voorhees, E. M., Hersh, W. R., Bedrick, S. & Lazar, A. J. (2018). Overview of the TREC 2018 Precision Medicine Track. In Proceedings of the Twenty-Seventh Text REtrieval Conference, Gaithersburg, Maryland, USA.
[32] Roostaee, M., Sadreddini, M. H. & Fakhrahmad, S. M. (2020). An effective approach to candidate retrieval for cross-language plagiarism detection: A fusion of conceptual and keyword-based schemes. Information Processing and Management, 57(2), 102150.
[33] Shokouhi, M. (2007) Segmentation of Search Engine Results for Effective DataFusion. In Proceedings of the 29th European Conference on Information Retrieval Research (ECIR ’07). Rome, Italy, 185–197
[34] Soni, S. & Roberts, K. (2021). An evaluation of two commercial deep learning-based information retrieval systems for COVID-19 literature. Journal of the American Medical Informatics Association, 28(1), 132-137.
[35] Vardakas, K. Z., Tsopanakis, G., Poulopoulou, A. & Falagas, M. E. (2015). An analysis of factors contributing to PubMed's growth. Journal of Informetrics, 9(3), 592-617.
[36] Wang, L., Shen, C. & Hartley, R. I. (2011). On the Optimality of Sequential Forward Feature Selection Using Class Separability Measure. In Proceedings of the 2011 International Conference on Digital Image Computing: Techniques and Applications (DICTA) (pp. 203-208).
[37] Wang, H., Zhang, Q. & Yuan, J. (2017). Semantically Enhanced Medical Information Retrieval System: A Tensor Factorization Based Approach. IEEE Access, 5, 7584-7593.
[38] Webber, W., Moffat, A. & Zobel, J. (2010). A similarity measure for indefinite rankings. ACM Transactions on Information Systems, 28(4), 20:1-20:38.
[39] Wu, S. & McClean, S. (2006) Performance prediction of data fusion for information retrieval. Information Processing and Management, 42(4): 899-915.
[40] Wu, S. (2012a). Applying the data fusion technique to blog opinion retrieval. Expert Systems with Applications, 39(1), 1346-1353.
[41] Wu, S. (2012b). Linear combination of component results in information retrieval.

Data & Knowledge Engineering, 71(1), 14-126.
[42] Wu, S. (2013). The weighted Condorcet fusion in information retrieval. Information Processing & Management, 49(1), 108-122.
[43] Wu, S. & Crestani, F. (2015). A geometric framework for data fusion in information retrieval. Information Systems, 50, 20-35.
[44] Wu, S., Huang, C., Li, L. & Crestani, F. (2019). Fusion-based methods for result diversification in web search. Information Fusion, 45, 16-26.
[45] Xu, D & Tian, Y. (2015) A comprehensive survey of clustering algorithms. Annals of Data Science, 2, 165-193.
[46] Yeganova, L., Kim, S., Chen, Q., Balasanov, G., Wilbur, W. J. & Lu, Z. (2020). Better synonyms for enriching biomedical search. Journal of the American Medical Informatics Association, 27(12), 1894-1902.
[47] Zhang, Y., Satapathy, S. C., Guttery, D. S., Górriz, J. M. & Wang, S. (2021). Improved Breast Cancer Classification Through Combining Graph Convolutional Network and Convolutional Neural Network. Information Processing and Management, 58(2), 102439.
[48] Zhou, X., Chen, X., Song, J., Zhao, G. & Wu, J. (2018). Team Cat-Garfield at TREC 2018 Precision Medicine Track. In Proceedings of The Twenty-seventh Text REtrieval Conference, Gaithersburg, Maryland, USA.
[49] Zhou, X., Depeursinge, A., & Muller, H. (2010). Information Fusion for Combining Visual and Textual Image Retrieval. In 2010 20th International Conference on Pattern Recognition. IEEE, 1590–1593.