Utilizing Ant Colony Optimization for Result Merging in Federated Search

Adamu Garba
School of Computer Science and Communication Engineering, Jiangsu University, China
Adamugarba84@gmail.com

Shah Khalid
Department of Computing, National University of Sciences and Technology (NUST), Pakistan
shah.khalid@seecs.edu.pk (corresponding author)

Aliya Aleryni
Department and College of Computer Science, King Khalid University, Abha, Saudi Arabia
amalaryani@kku.edu.sa

Irfan Ullah
Department of Computer Science, Shaheed Benazir Bhutto University, Sheringal, 18050, Pakistan
irfan@sbbu.edu.pk

Nasser Mansoor Tairan
Department and College of Computer Science, King Khalid University, Abha, Saudi Arabia
mmtairan@kku.edu.sa

Habib Shah
Department and College of Computer Science, King Khalid University, Abha, Saudi Arabia
hurrahman@kku.edu.sa

Diyawu Mumin
Department of Computer Science, Tamale Technical University, Tamale, Ghana
mdiyawu@tatu.edu.gh

Received: 20 March 2024 | Revised: 16 April 2024 | Accepted: 18 April 2024

 DOI: https://doi.org/10.48084/etasr.7302

ABSTRACT

Federated search or distributed information retrieval routes the user's search query to multiple component collections and presents a merged result list in ranked order by comparing the relevance score of each returned result. However, the heterogeneity of the component collections makes it challenging for the central broker to compare these relevance scores while fusing the results into a single ranked list. To address this issue, most existing approaches merge the returned results by converting the document ranks to their ranking scores or downloading the documents and computing their relevance score. However, these approaches are not efficient enough, because the former methods suffer from limited efficacy of result merging due to the negligible number of overlapping documents and the latter are resource intensive. The current paper addresses this problem by proposing a new method that extracts features of both documents and component collections from the available information provided by the collections at query time. Each document and its collection features are exploited together to establish the document relevance score. The ant colony optimization is used for information retrieval to create a merged result list. The experimental results with the TREC 2013 FedWeb dataset demonstrate that the proposed method significantly outperforms the baseline approaches.

Keywords-information retrieval; distributed information retrieval; federated search; result merging; ant colony optimization

I. INTRODUCTION

Internet users rely on search engines like Google, Bing, and Yahoo to find information. However, some information sources are not fully accessible through these traditional search engines for commercial or security reasons [1]. In addition, understanding the semantics of a search query can be challenging to traditional search engines [2]. As such, users relying on search engines miss valuable collections of information sources that may be relevant, but not fully accessible. Federated search, also known as Distributed Information Retrieval (DIR) [3], connects users directly to those information sources through a unified search interface. This interface can perform multiple searches across distributed collections and combine their results into a single list [4]. However, with multiple collections involved, users cannot know beforehand which collections will most likely be relevant to their query. This is where the central broker acts as an intermediary between the component collections and users [3]. This way a user issues the query to the search interface, which is transmitted directly to the broker, and the broker selects a few collections that may contain relevant documents and routes the query toward them (Figure 1). Each collection processes the received query, returns a ranked result list to the broker, and the broker merges them into a single ranked list and presents it to the user.

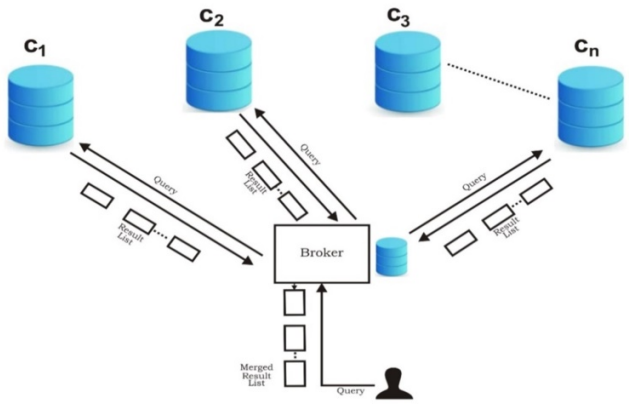

Fig. 1. User interaction with a federated search system.

The literature divides the domain of federated search into three separate research areas: resource description, collection selection, and result merging [3]. The resource description [5], also called collection representation [3], regards obtaining information concerning each collection's content and size. The collection selection [6, 7] uses the sampled information to select a few collections that may contain the most relevant documents for a given user query. Finally, the result merging [8, 9] combines the results returned by the component collections into a single ranked list. This paper focuses on the result merging problem as it plays a key role in satisfying users' information needs, because user satisfaction is compromised if the most relevant collections are selected and their result lists are merged poorly, where the most relevant documents may not make it to the top of the merged result list. In addition, merging multiple result lists is a challenging task, especially in non-cooperative environments [3] due to the differences in the collections corpus statistics, the use of varying indexing schemes and retrieval models, and the non-availability of the documents' full-text at the merging time. Moreover, some collections may have non-text content, such as images or videos. Learning to Rank (L2R) techniques are considered an alternative to traditional document-ranking approaches. Several research works [8, 10-14] used L2R for various retrieval tasks. In these techniques, a set of queries, each associated with the ranked list of documents and their relevance judgments, is exploited as training data [15]. These training data are employed to extract query-document relevance features and then to learn a ranking function that predicts the documents' relevance for each given query. For example, in federated search, authors in [14] considered the result merging problem as a classification problem. They extracted some features from the returned documents by the collections and learned a ranking function that merged the multiple result list. This method's drawback is the need for a huge amount of training data when learning the ranking function.

Swarm intelligence and genetic programming are widely deployed in various domains to solve optimization problems. When applying these techniques, multiple rounds of iterative processes of selections, variations, and mutation are performed on the extracted features to obtain the optimum solution. These techniques have been reported to provide better solutions than the algorithms defined by experts [16]. Recently, authors in [8] used genetic programming to solve the result merging problem in which returned documents by the collections are downloaded, some likely relevance features are extracted from the documents at each iterative process, and then the documents are re-ranked based on their relevance scores. However, the drawbacks of this approach are the increases in bandwidth usage, computational resources, and latency time. The proposed work addresses this issue by extracting similar features, like in [14], but with a few modifications: First, the mentioned studies consider only the document's features in obtaining their relevance score, whereas the present study considers the quality of the collection as an important factor in addition to these features. Second, the current study departs from learning a ranking function because it requires colossal training data. Instead, the former utilizes the Ant Colony Optimization (ACO), a swarm intelligence algorithm, to generate the merged result list. The novelty of the proposed method lies in the usage of the only available information in a way that neither overwhelms the broker nor increases its computational resources or response time. Key points to this research are:

- A method that uses only the information provided by the component collections at query time is proposed.
- The proposed method applies the ACO algorithm and considers collection quality and document features in computing the document results merging score.

- The effectiveness of the proposed approach is tested by conducting experiments with the TREC 2013 FedWeb dataset. The experimental results demonstrate the robustness of the recommended approach as it outperforms both the learning and non-learning models utilized as the baselines.

II. BACKGROUND AND RELATED WORKS

Result merging combines the results returned by multiple collections into a unified list before serving to the search user. Many factors make result merging challenging, the most prominent among them is the incomplete information about collections' corpus statistics and the use of different retrieval models [3]. The simplest scenario to assume is that the component collections return their documents' relevance score or run the same retrieval model. Even with this assumption, the document's returned relevance score cannot be compared directly due to the differences in their documents index and processing method [9]. For these reasons, only a few models have been proposed for the result merging problem. The first models assumed that the component collections share their documents' index information with the broker [17] or that they should return their results with their index terms statistics [18]. Neither of these assumptions is achievable in a realistic web environment because most component collections are uncooperative and treat the broker as an ordinary search user. As such, the only response to the broker request for each query received is by returning a ranked result list without sharing information about the documents index [19].

Owing to the uncooperative nature of the component collections, several studies [9, 20, 21] engaged query-based sampling to sample representative documents from each component collection. They built a Centralized Sample Index (CSI) at the broker site. When the broker receives a user query, it forwards it to the most relevant collections and runs it on the CSI. The final merging score of the documents is estimated by mapping the documents' ranks returned in the collections result lists to their corresponding relevance scores obtained from the CSI. The drawback of these models is that their effectiveness hinges on the high number of overlapping documents between the collections' result lists and the CSI list [3]. Various approaches [5, 22] were proposed in the TREC FedWeb search track for the result merging task. Authors in [23] merged the results in a round-robin manner, where the first document in the merged result list is the first document of the most relevant collection selected in the resource selection phase, the second document is the first document in the second most relevant collection, and so on. The models proposed in [24, 25] estimated the document merging scores by converting their ranks into a ranking score. Specifically, authors in [24] converted the ranks of the documents into a ranking score deploying the modified reciprocal rank fusion [26]. They estimated the merging score for each document by taking the log of the reciprocal of the document rank and then multiplying it with the relevance score of the collection that returned it [20]. Similarly, the ICTNET [27] computes each document merging score by computing the BM25 score of the document parts (i.e. title, URL, heading, etc.) and combining them with a linear weighting method. The L2R approaches have displayed competitiveness in generating effective document ranking compared to non-learning methods. Recently, authors in [13, 14, 28, 29] used L2R for result merging. Authors in [14] utilized the information in the collections' results list to extract some likely relevant features, which include the occurrences of query terms in the titles, the order of occurrences of the terms, and the presence or absence of URLs for a document. These features were fed to the boosting algorithm [30] to learn the ranking function. Authors in [23] applied a gradient boost algorithm to learn the composition of the final merged result list when different verticals returned it. However, the downside of machine learning algorithms is the need for huge training data and their computational cost.

The meta-heuristic approach has recently gained much attention due to its ability to solve hard combinatorial optimization problems [31]. Most meta-heuristic algorithms study the behavior of social insects and design algorithms that mimic their behavior to solve real-world problems. Authors in [32] focused on the Ant Colony Optimization (ACO) algorithm to see how the ants communicate and perform complex tasks. Interestingly, ants can locate the shortest path between their nest and food sources by depositing a chemical called pheromone on the ground. One of the characteristics of this pheromone is that it evaporates with time. Because of this evaporating nature, the pheromone deposit of the path followed by most ants is continuously updated, and their trail is sustained as the shortest path. On the other hand, the pheromone deposit of the path followed by fewer ants is quickly evaporated, their trail is lost, and the ants abandon that path. In solving optimization problems, artificial ants are created to mimic natural ones with additional capabilities. The number of ants created equals the number of data points for solving the problem. Each ant is placed on a data point with some pheromone value. The selection bias is avoided by initializing the same pheromone value to all the data points. The ants select the next data point based on its importance in solving the problem. The pheromone value is updated after the first iteration (i.e. the ants transverse the data points). The update is performed by increasing the pheromone values of the data points the ants selected, whereas the pheromone value of the data points the ants did not select remains unchanged. This pheromone update is performed multiple times until the algorithm converges. The data points with the highest frequency after the convergence are considered the optimal solution to the problem.

In the literature on federated search, the genetic algorithm has been employed to fuse multiple result lists into a single list [4]. This approach extracts the BM25 and language modeling scores of features like titles and URLs from the downloaded collection to rank the merged results. A better alternative is using document features only [4]. However, it is not explicitly explained how the component collections are created from the dataset implemented for their experiment. The problem with this and the above mentioned studies is that they consider document features but cannot consider collection-level features. Therefore, one possible solution could be to not only consider the document and collection features in estimating the document merging scores, but to also use the foraging behavior of ants. This method does not require any training data and

relies only on the click-through data or relevance judgments of the documents. The following section explains how the proposed approach can be materialized.

III. MATERIALS AND METHODS

A. Preliminaries

To explain the proposed method, a CSI, an accumulation of sample documents from all the collections involved, must be created. The sample documents are obtained employing the most widely used Query-Based Sampling (QBS) method [5]. In QBS, a random query is issued to all component collections and the top five documents are downloaded to the CSI. Querying and downloading documents to the CSI continues until 300 documents are sampled from each collection. In this paper, the CSI is deployed only for collection selection purposes. For the experiment, let us suppose a search query q is issued to the broker, and the broker selects C_i collections, with $1 < i \leq m$ as probably the most relevant to the search for query q in the collection selection phase. The broker routes q to those collections and each one processes the query and responds by returning a ranked result list $C_i = \{c_{i1}, c_{i2}, \dots, c_{in}\}$ to the broker. The main goal in this paper is to fuse these result lists into a single ranked list $C = \{c_1, c_2, \dots, c_n\}$ similar to what is obtained. The query is issued to a centralized search system.

B. The Proposed Approach

Unlike most existing approaches, the proposed method uses only the available information that the collection returned to the broker at query time in merging the result list. The detailed description of the processes involved in the proposed method follows:

Step 1: Assume that for a query q issued to the broker, the broker runs the query on the CSI and selects the most relevant collection to search.

Step 2: The query is routed to the collections selected in Step 1. Each collection processes the query and returns a ranked result list. This results list is expected to contain snippets of documents similar to those attained in real-world search systems.

Step 3: For each snippet returned in Step 2, its query-dependent features are extracted, as shown in Table I.

Step 4: For each collection that returns a result list, its quality concerning query q is represented by the features depicted in Table II.

Step 5: The number of ants is initialized to be equal to the number of features in Tables I and II. The pheromone value is also initialized to a unit value and assigned to all features. Each ant is assigned a random feature for the first iteration to begin its search. From the second iteration to convergence, the ants select a feature based on its importance in determining the document's relevance, which is obtained based on the probability specified by:

$$f_i = \alpha_k \Delta\alpha_k \quad (1)$$

where α_k is the pheromone value, which is initiated at a unit value, $\Delta\alpha_k$ is the proportion of the ants that select that particular feature. Whenever an ant selects a feature, the pheromone value of the feature is updated by:

$$\alpha_{k+1} = (\alpha_k + exp^r) \quad (2)$$

where α_{k+1} is the next iteration pheromone value, and r is the relevance label of the document d_i provided in the relevance judgment file.

Step 6: At each iteration of feature selection, the relevance score of each document is obtained by aggregating its features and that of the collection that returned it, as observed in (3):

$$\mathrm{S(d}_i) = \sum_{i=1}^{m} f_i * \sum_{j=1}^{n} (f_j) \quad (3)$$

where f_i and f_j represent the document and the collection quality features, respectively. The feature selection, pheromone update, and relevance score estimation continue until the algorithm converges.

TABLE I. LIST OF EXTRACTED FEATURES OF THE DOCUMENT

Feature (f_i)	**Description**
Document rank	Convert each document's rank in a collection result list into a ranking score: $\mathcal{R}(s) = exp - (\frac{r}{n})$ where *r* is the document ranking position in a particular collection's returned results list, and *n* is the number of documents in the results list.
Query terms in the URL	Count the occurrences of query terms of each document URL: $\sum_{i=1}^{\lvert q\rvert} C(q_i, U)$
Query terms in the title	Count the number of query terms in the title: $\sum_{i=1}^{\lvert q\rvert} C(q_i, T)$
Query terms in the description	Count the number of query terms in the description: $\sum_{i=1}^{\lvert q\rvert} C(q_i, D)$
BM25 of the title	Relevance score of the title using BM25 as a retrieval model
BM25 of the description	Relevance score of the description using BM25 as a retrieval model
LM of the title	Relevance score of the title using a Language Model (LM) as a retrieval model with Dirichlet prior
LM of the description	Relevance score of the description using an LM as a retrieval model with Dirichlet prior
Average query terms in the description	The average number of query terms in the description is: $avg(q, D) = \frac{\sum_{i=1}^{\lvert q\rvert} C(q_i,\ D)}{\lvert D\rvert}$ where $\lvert D\rvert$ is the length of the description.

TABLE II. THE LIST OF EXTRACTED FEATURES OF THE COLLECTION QUALITY

Feature (f_j)	**Description**
Relevance score	The relevance score of the collection, obtained in the collection selection phase.
No. of query terms	Count the total number of query terms in the collection result list $TF(t, C_i) = \sum_{t_q \in C_i} TF(t_q, C_i)$
No. of documents	The total number of documents a collection returned in its result list.
Average query terms	The average number of query terms in a collection

C. Experimental Evaluation

For the experiments carried out, this study utilized search engine snippets sampled from the FedWeb Greatest Hits dataset [22, 33] of the TREC 2013 FedWeb compilation as the collections. This is the first standard dataset created to promote federated search research while discouraging the artificial creation of collections using TREC web track datasets [34]. The dataset contains the results downloaded from 157 real-world search engines in 24 vertical categories (i.e. academics, blogs, entertainment, jobs, kids, etc.). In its creation, each search engine received 2000 queries, resulting in a total of 1,973,591 snippets extracted. On average, each search engine returned 12,570.6 results, stored in XML format. This dataset is more suitable for the current work because each search engine uses its proprietary retrieval method to retrieve its result, and each result is separated in the dataset [35]. The present work used 50 queries released with the dataset to represent the user information needs. The queries are judged employing the five levels of graded relevance judgment, i.e. not relevant (NRel), relevant (Rel), highly relevant (HRel), top relevant (Key), and navigational (Nav) by the team of experts.

ALGORITHM 1: RESULT MERGING USING ACO

Input: Initial pheromone value α_k, number of ants, number of features, and number of iterations.
Output:
A merged results list of documents $C = \{c_1, c_2, \ldots, c_n\}$
Repeat
Create the number of ants equal to the number of features extracted in Tables 1 and 2.
Initialize the pheromone values α_k to 1
Allow ants to select the features based on their pheromone value using (1).
Perform pheromone update using (2).
For each iteration, rank the documents based on their score computed using (3).
Check if the number of iterations is not reached, then repeat steps 3 to 5
Return the final merged results list

Regarding the collection selection, it was observed that routing a query to all the relevant and non-relevant collections would waste resources. Therefore, utilizing a modified version of the collection selection algorithm proposed in [36], only the top five most relevant collections were put into service for each query. In the model, a collection is considered relevant based on the number of relevant documents in the top *N*-ranked list and the documents ranking positions.

The collection selection experiments were performed using Apache Solr version 8.2 with Python 3.6 on an Intel core i7-based system running Linux and 8 GB memory. The default vector space model of the Apache Solr was implemented for indexing and retrieval of the results snippets. This study queried the title and description of each indexed snippet and then summed up their scores as the snippet relevance score. The required parameters for the ACO algorithm were set as number of ants = 40, number of iterations = 20, α_k = 1.0, β = 0.45, and γ = 0.55. Regarding the evaluation of the proposed approach, the selected baselines can be categorized into learning [4, 10] and non-learning [20, 23] methods. The official evaluation metric for the TREC 2013 FedWeb results merging task is nDCG@k [37]. The nDCG metric measures the goodness of the retrieved result list compared to the best/ideal ordering of the result list. The results are reported at the top k cutoff ranks, where the value of k is 1, 3, 5, and 10, respectively.

IV. RESULTS AND DISCUSSION

Table III showcases the experimental results. For each cutoff rank, the best-performing method is boldfaced. The introduced approach is represented as ACO-RM. From the result, it can be observed that the recommended approach has the highest performance compared to the baselines across all the cutoff ranks. The highest performance of over 94.75% on nDCG@1 was achieved compared to the baselines, and the lowest was 61.97% on nDCG@10. It can be further noticed from the results that except for TF-RF, which achieved its highest performance on nDCG@3, all the other methods obtained their highest performance on nDCG@1. Among the baseline methods, nsRRF has shown the strongest performance, followed by TF-RF. The poor performance of the BTM method can be attributed to insufficient training data. That is, for fairness, only the result of the top five selected collections is utilized for the model's training and testing. This result has once again proved that without enough training data, the non-learning methods, in some cases, can outperform machine learning models. The following two subsections further elaborate these findings.

TABLE III. PERFORMANCE COMPARISON OF THE PROPOSED METHOD WITH THE BASELINES

Works	nDCG@1	nDCG@3	nDCG@5	nDCG@10
nsRRF	0.5075	0.4515	0.4294	0.4253
ICTNET	0.3790	0.3579	0.3401	0.3397
BTM	0.3499	0.3556	0.3499	0.3452
TF-RF	0.3353	0.3892	0.3852	0.3559
ACO-RM	0.9475	0.8037	0.7411	0.6197

A. Ranking Construction at Each Iteration

Figures 2 and 3 are the sample of the first iteration generated ranking and the relevance label of the top 5 documents extracted from the TREC relevance judgment file, respectively. Looking at the top 5 documents highlighted in red in Figure 2 and their corresponding relevance label in Figure 3, it can be seen that only the top document is marginally relevant, whereas all the remaining are non-relevant. At this stage, the ants explore features by selecting them based on the initialized pheromone value. However, as the iteration progresses, the pheromone update increases the merging score of highly relevant documents while decreasing the ones of non-relevant documents. This can be observed in the fifth iteration generated ranking portrayed in Figure 4. Comparing the first and fifth iterations generated ranking as illustrated in Figures 2 and 4, it can be noted that the top ranking document of Figure 2

is now on the ranking position 3 in Figure 4 and the document that was on position 3 is now on ranking position 5, while, the remaining three documents have lost their ranking spot. Finally, Figures 5 and 6 manifest the top 5 documents of the proposed model after its convergences and their corresponding extracted relevance label. The extracted relevance label exhibited in Figure 6 indicates that the suggested model could rank most of the highly relevant documents at top-ranking positions, as shown in Figure 5.

```
Query   Search Engines ID Rank  Score
7001 Q0 FW13-e185-7001-09 1 90056.401929943
7001 Q0 FW13-e185-7001-08 2 55805.864970911
7001 Q0 FW13-e176-7001-04 3 53906.629568599245
7001 Q0 FW13-e176-7001-10 4 49898.56567783001
7001 Q0 FW13-e176-7001-07 5 46445.967103958734
7001 Q0 FW13-e176-7001-08 6 43440.8474175728
7001 Q0 FW13-e185-7001-03 7 41467.55791728804
7001 Q0 FW13-e185-7001-02 8 41456.80942692804
7001 Q0 FW13-e185-7001-10 9 39432.796884270545
7001 Q0 FW13-e175-7001-07 10 37565.01871162625
```

Fig. 2. First iteration generated ranking. Query is the query number, Search Engines ID is the search engine identity created by the FedWeb organizers. Here e185 is the search engine number, 7001 is the query number, and 09 is the document rank in the search engine result list. Rank is the ranking positions generated by our model, and Score is the document score used in generating the ranking.

```
7001 0 FW13-e185-7001-09 1
7001 0 FW13-e185-7001-08 0
7001 0 FW13-e176-7001-04 0
7001 0 FW13-e176-7001-10 0
7001 0 FW13-e176-7001-07 0
```

Fig. 3. First iteration top documents relevance label extracted from the TREC relevance file.

```
Query   Search Engines ID Rank  Score
7001 Q0 FW13-e185-7001-03 1 889853.0868863908
7001 Q0 FW13-e185-7001-06 2 831480.0284284129
7001 Q0 FW13-e185-7001-09 3 90860.88476110343
7001 Q0 FW13-e185-7001-08 4 56143.11038431412
7001 Q0 FW13-e176-7001-04 5 54238.66510238029
```

Fig. 4. Fifth iteration generated ranking.

```
Query   Search Engines ID Rank  Score
7001 Q0 FW13-e185-7001-03 1 49547461.593474954
7001 Q0 FW13-e185-7001-06 2 48820417.69238441
7001 Q0 FW13-e185-7001-09 3 94227.02725169474
7001 Q0 FW13-e185-7001-02 4 80492.0046515333
7001 Q0 FW13-e175-7001-10 5 72467.60226749117
```

Fig. 5. Convergence-generated ranking.

```
7001 0 FW13-e185-7001-03 7
7001 0 FW13-e185-7001-06 7
7001 0 FW13-e185-7001-09 1
7001 0 FW13-e185-7001-02 3
7001 0 FW13-e175-7001-10 3
```

Fig. 6. Convergence top document relevance label extracted from the TREC relevance file.

B. The Contribution of Collection Quality to Relevance

This study attributed the superior performance achieved by the adopted approach to including collection quality features in computing the merging score of the documents. However, to provide more insight into how much collection quality features contribute to identifying the most relevant documents, a further experiment was performed by removing the collection quality features in computing the merging score and comparing the results with ACO-RM. Table IV depicts the experimental results in which we denoted the result of merging scores without collection quality features as ACO-RMN. From the results, it can be seen that by removing the collection quality features in computing the merging score, the effectiveness of the merged results list has decreased by 18.2% on nDCG@1 and 3.67% on nDCG@10. These findings allow the assumption that including collection features in computing the documents merging score increases the effectiveness of the merged results list in a federated search environment.

TABLE IV. CONTRIBUTION OF COLLECTION QUALITY IN IDENTIFYING RELEVANT DOCUMENTS

Works	nDCG@1	nDCG@3	nDCG@5	nDCG@10
ACO-RM	0.9475	0.8037	0.7411	0.6197
ACO-RMN	0.7755	0.7354	0.6835	0.5969

V. SUMMARY

Most of the existing approaches consider only the documents relevance in merging results into a single ranking list. However, considering only document relevance in federated search result merging neglects the contextual understanding provided by the collections. Some collections may specialize in certain domains or types of information and ignoring their relevance can result in overlooking valuable resources that may contribute significantly to the user's information needs. Considering this, this work presented the proposed method that utilizes the collection quality in calculating the documents merging score. Specifically, the proposed model draws inspiration from ACO's foraging behavior. It leverages information returned by collections to the central interface, akin to ants depositing pheromones along their paths and then extracts both documents' features and collection quality features. Through this decentralized approach, the model dynamically merges the returned results to improve retrieval efficiency.

The contribution of this work to the literature of federated search is three-fold. First, it introduces a bio-inspired approach based on ACO for improved retrieval effectiveness of the federated search. Second, it promotes efficient resource utilization by deploying only the information returned by collections to the central interface, minimizing redundancy and optimizing relevance. Third, unlike most of the previous model which considered only the documents relevance, this work considers both the documents' as well as the collection's quality in calculating the documents merging score.

VI. CONCLUSIONS AND FUTURE WORK

This paper proposeS a result merging method for federated search that employed the foraging behavior of ACO in merging

the multiple result lists. Unlike the existing methods, the suggested method uses document and component collection features to compute the result merging score. The features are first extracted and then summed as the relevance score of the documents. Then a pheromone value is assigned to each relevance score and it is updated at each iteration of the result until it converges. This study experimented with the TREC 2013 FedWeb dataset. The experimental results demonstrated the effectiveness of the proposed method, as it significantly outperformed both the learning and non-learning methods employed as baselines.

Even though the recommended approach exhibits remarkable performance compared to the baselines, its limitation lies in the fact that it is not fully automating the learning process, as the model parameters are selected based on trial and error instead of learning. A possible direction for future work should be to utilize a learning method in selecting the parameters. In addition, there is an intention for this work to be extended to scholarly search [38, 39] and book retrieval [40], especially in federated search settings.

ACKNOWLEDGMENT

The authors extend their appreciation to the Deanship of Scientific Research at King Khalid University for funding this work through the Small Groups Project under grant number RGP.1/369/44

COMPETING INTERESTS

The authors explicitly declare that "No Competing Interests are at stake and there is No Conflict of Interest" with other people or organizations that could inappropriately influence or bias the article's content. A pre-print of this article can be found at [41].

REFERENCES

[1] A. Garba and S. Wu, "Snippet-based result merging in federated search," *Journal of Information Science*, Jan. 2023, Art. no. 01655515221144864, https://doi.org/10.1177/01655515221144864.

[2] B. Nethravathi, G. Amitha, A. Saruka, T. P. Bharath, and S. Suyagya, "Structuring Natural Language to Query Language: A Review," *Engineering, Technology & Applied Science Research*, vol. 10, no. 6, pp. 6521–6525, Dec. 2020, https://doi.org/10.48084/etasr.3873.

[3] M. Shokouhi and L. Si, "Federated Search," *Foundations and Trends® in Information Retrieval*, vol. 5, no. 1, pp. 1–102, Mar. 2011, https://doi.org/10.1561/1500000010.

[4] V. Stamatis, M. Salampasis, and K. Diamantaras, "Machine learning methods for results merging in patent retrieval," *Data Technologies and Applications*, Jan. 2023, https://doi.org/10.1108/DTA-06-2021-0156.

[5] J. Callan and M. Connell, "Query-based sampling of text databases," *ACM Transactions on Information Systems*, vol. 19, no. 2, pp. 97–130, Dec. 2001, https://doi.org/10.1145/382979.383040.

[6] A. Garba, S. Khalid, I. Ullah, S. Khusro, and D. Mumin, "Embedding based learning for collection selection in federated search," *Data Technologies and Applications*, vol. 54, no. 5, pp. 703–717, Jan. 2020, https://doi.org/10.1108/DTA-01-2019-0005.

[7] L. Li, Z. Zhang, and S. Wu, "LDA-Based Resource Selection for Results Diversification in Federated Search," in *15th International Conference on Web Information Systems and Applications*, Taiyuan, China, 2018, pp. 147–156, https://doi.org/10.1007/978-3-030-02934-0_14.

[8] H. T. Vo, "New Re-ranking Approach in Merging Search Results," *Informatica*, vol. 43, no. 2, pp. 235–242, Jun. 2019, https://doi.org/10.31449/inf.v43i2.2132.

[9] D. Hong and L. Si, "Mixture model with multiple centralized retrieval algorithms for result merging in federated search," in *35th international ACM SIGIR conference on Research and development in information retrieval*, Portland, OR, USA, Aug. 2012, pp. 821–830, https://doi.org/10.1145/2348283.2348393.

[10] T. Wu, X. Liu, and S. Dong, "LTRRS: A Learning to Rank Based Algorithm for Resource Selection in Distributed Information Retrieval," in *25th China Conference on Information Retrieval*, Fuzhou, China, Sep. 2019, pp. 52–63, https://doi.org/10.1007/978-3-030-31624-2_5.

[11] M. Ibrahim and M. Carman, "Comparing Pointwise and Listwise Objective Functions for Random-Forest-Based Learning-to-Rank," *ACM Transactions on Information Systems*, vol. 34, no. 4, Dec. 2016, Art. no. 20, https://doi.org/10.1145/2866571.

[12] P. Mohapatra, M. Rolinek, C. V. Jawahar, V. Kolmogorov, and M. P. Kumar, "Efficient Optimization for Rank-Based Loss Functions," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, Jun. 2018, pp. 3693–3701, https://doi.org/10.1109/CVPR.2018.00389.

[13] K. Tjin-Kam-Jet and D. Hiemstra, "Learning to merge search results for efficient Distributed Information Retrieval," in *10th Dutch-Belgian Information Retrieval Workshop, DIR 2010*, Nijmegen, Netherlands, Jan. 2010.

[14] B. Ghansah, S. Wu, and N. Ghansah, "Rankboost-Based Result Merging," in *IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*, Liverpool, UK, Oct. 2015, pp. 907–914, https://doi.org/10.1109/CIT/IUCC/DASC/PICOM.2015.136.

[15] H. Li, "A Short Introduction to Learning to Rank," *IEICE TRANSACTIONS on Information and Systems*, vol. E94-D, no. 10, pp. 1854–1862, Oct. 2011.

[16] J. R. Koza, *Genetic programming: on the programming of computers by means of natural selection*. Cambridge, MA, USA: MIT Press, 1992.

[17] Y. Rasolofo, D. Hawking, and J. Savoy, "Result merging strategies for a current news metasearcher," *Information Processing & Management*, vol. 39, no. 4, pp. 581–609, Jul. 2003, https://doi.org/10.1016/S0306-4573(02)00122-X.

[18] S. T. Kirsch, "Document retrieval over networks wherein ranking and relevance scores are computed at the client for multiple database documents," US5659732A, Aug. 19, 1997.

[19] P. Ogilvie and J. Callan, "The effectiveness of query expansion for distributed information retrieval," in *10th international conference on Information and knowledge management*, Atlanta, GA, USA, Oct. 2001, pp. 183–190, https://doi.org/10.1145/502585.502617.

[20] M. Shokouhi and J. Zobel, "Robust result merging using sample-based score estimates," *ACM Transactions on Information Systems*, vol. 27, no. 3, Feb. 2009, Art. no. 14, https://doi.org/10.1145/1508850.1508852.

[21] C. He, D. Hong, and L. Si, "A weighted curve fitting method for result merging in federated search," in *34th international ACM SIGIR conference on Research and development in Information Retrieval*, Beijing, China, Jul. 2011, pp. 1177–1178, https://doi.org/10.1145/2009916.2010107.

[22] T. Demeester, D. Trieschnigg, D. Nguyen, D. Hiemstra, and K. Zhou, "Overview of the TREC 2014 Federated Web Search Track," in *Twenty-Third Text REtrieval Conference*, Gaithersburg, MD, USA, Nov. 2014, pp. 1–14.

[23] E. Di Buccio and M. Melucci, "University of Padua at TREC 2014: Federated Web Search Track," in *Twenty-Third Text REtrieval Conference (TREC 2014)*, Gaithersburg, MD, USA, Nov. 2014.

[24] [24] A. Mourao, F. Martins, and J. Magalhaes, "NovaSearch at TREC 2013 Federated Web Search Track: Experiments with rank fusion," in *The Twenty-Second Text REtrieval Conference*, Gaithersburg, MD, USA, Nov. 2013, pp. 1–8.

[25] D. Pal and M. Mitra, "ISI at the TREC 2013 Federated task," in *Proceedings of the Twenty-Second Text Retrieval Conference, Trec 2013*, Gaithersburg, MD, USA, 2013.

[26] G. V. Cormack, C. L. A. Clarke, and S. Buettcher, "Reciprocal rank fusion outperforms condorcet and individual rank learning methods," in

32nd International ACM SIGIR conference on research and development in Information Retrieval, Boston, MA, USA, Jul. 2009, pp. 758–759, https://doi.org/10.1145/1571941.1572114.

[27] F. Guan, Y. Xue, X. Yu, Y. Liu, and X. Cheng, "ICTNET at Federated Web Search Track 2013," in *Twenty-Third Text REtrieval Conference (TREC 2014)*, Gaithersburg, MD, USA, Nov. 2014.

[28] A. K. Ponnuswami, K. Pattabiraman, Q. Wu, R. Gilad-Bachrach, and T. Kanungo, "On composition of a federated web search result page: using online users to provide pairwise preference for heterogeneous verticals," in *Fourth ACM International Conference on Web Search and Data Mining*, Hong Kong, China, Feb. 2011, pp. 715–724, https://doi.org/10.1145/1935826.1935922.

[29] R. Takanobu, T. Zhuang, M. Huang, J. Feng, H. Tang, and B. Zheng, "Aggregating E-commerce Search Results from Heterogeneous Sources via Hierarchical Reinforcement Learning," in *The World Wide Web Conference*, San Francisco, CA, USA, Dec. 2019, pp. 1771–1781, https://doi.org/10.1145/3308558.3313455.

[30] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer, "An Efficient Boosting Algorithm for Combining Preferences," *Journal of Machine Learning Research*, vol. 4, pp. 933–969, 2003.

[31] P. Shunmugapriya and S. Kanmani, "A hybrid algorithm using ant and bee colony optimization for feature selection and classification (AC-ABC Hybrid)," *Swarm and Evolutionary Computation*, vol. 36, pp. 27–36, Oct. 2017, https://doi.org/10.1016/j.swevo.2017.04.002.

[32] M. Dorigo and G. Di Caro, "Ant colony optimization: a new meta-heuristic," in *Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, Washington, DC, USA, Jul. 1999, vol. 2, pp. 1470-1477 Vol. 2, https://doi.org/10.1109/CEC.1999.782657.

[33] T. Demeester, D. Trieschnigg, D. Nguyen, D. Hiemstra, and K. Zhou, "FedWeb Greatest Hits: Presenting the New Test Collection for Federated Web Search," in *24th International World Wide Web Conference*, Florence, Italy, Dec. 2015, pp. 27–28, https://doi.org/10.1145/2740908.2742755.

[34] D. Trieschnigg, T. Demeester, A. Zhou, D. Nguyen, and D. Hiemstra, "FedWeb Greatest Hits." [Online]. Available: https://fedwebgh.intec.ugent.be/.

[35] C.-J. Lee, Q. Ai, W. B. Croft, and D. Sheldon, "An Optimization Framework for Merging Multiple Result Lists," in *24th ACM International on Conference on Information and Knowledge Management*, Melbourne, VIC, Australia, Oct. 2015, pp. 303–312, https://doi.org/10.1145/2806416.2806489.

[36] M. Shokouhi, "Central-Rank-Based Collection Selection in Uncooperative Distributed Information Retrieval," in *29th European Conference on IR Research*, Rome, Italy, Apr. 2007, pp. 160–172, https://doi.org/10.1007/978-3-540-71496-5_17.

[37] C. L. A. Clarke *et al.*, "Novelty and diversity in information retrieval evaluation," in *31st Annual International ACM SIGIR Conference*, Singapore, Asia, Jul. 2008, pp. 659–666, https://doi.org/10.1145/1390334.1390446.

[38] S. Khalid, S. Khusro, I. Ullah, and G. Dawson-Amoah, "On The Current State of Scholarly Retrieval Systems," *Engineering, Technology & Applied Science Research*, vol. 9, no. 1, pp. 3863–3870, Feb. 2019, https://doi.org/10.48084/etasr.2448.

[39] S. Khalid and S. Wu, "Supporting Scholarly Search by Query Expansion and Citation Analysis," *Engineering, Technology & Applied Science Research*, vol. 10, no. 4, pp. 6102–6108, Aug. 2020, https://doi.org/10.48084/etasr.3655.

[40] I. Ullah, S. Alam, Z. Ali, M. Khan, F. Jabeen, and S. Khusro, "On the current state of query formulation for book search," *Artificial Intelligence Review*, vol. 56, no. 10, pp. 12085–12130, Oct. 2023, https://doi.org/10.1007/s10462-023-10483-7.

[41] A. Garba, S. Khalid, H. Shah, I. Ullah, N. M. Tairan, and D. Mumin, "Using Ant Colony Optimization for Results Merging in Federated Search." researchsquare, Jul. 17, 2023, https://doi.org/10.21203/rs.3.rs-3115769/v1.