Federated Query Processing Using Ontology Structure and Ranking in a Service Oriented Environment
Technical Report UTDCS-12-11
Department of Computer Science
The University of Texas at Dallas
March 2011
Neda Alipanah, Pallabi Parven, Latifur Khan and Bhavani Thuraisingham
Federated Query Processing Using Ontology Structure and Ranking in a Service Oriented Environment

Neda Alipanah, Pallabi Parven, Latifur Khan, and Bhavani Thuraisingham
Department of Computer Science
University of Texas at Dallas
Richardson, U.S.A.
Email: na061000, pxp013300, lkhan, bxt043000@utdallas.edu

Abstract—In view of the need for service-based query processing in a federated environment, ranking data sources (ontologies) and robust query expansion in a specific domain have great impact on the performance and accuracy of web applications. Since robust query expansion exploits multiple data sources (ontologies) instead of a single ontology, ontology ranking is considered as a precursor for robust query expansion. Ontology ranking determines quality of an ontology and commonality of overlapping entities across different ontologies of the same domain. For this, first, we calculate the similarity of ontologies by an Entropy Based Distribution (EBD) measurement based on commonality of overlapping entities. Next, we determine robust expansion terms by a number of semantic measures. We consider each individual ontology and user query keywords to determine the Basic Expansion Terms (BET) based on the structure of ontology. We use Density Measure (DM), Betweenness Measure (BM), Semantic Similarity Measure (SSM) and Weight of Semantic Path (WSP) to calculate BET. Then, we specify New Expansion Terms (NET) using the ontology alignment (OA). Further, we determine the Robust Expansion Term (RET) using ontology rank as a dynamic threshold. Finally, we show the effectiveness of our novel ontology-driven expansion approach in compare to wordnet.

Index Terms—Federated Query, Query Expansion, Ontology, Ontology Matching, Expansion Terms.

1 INTRODUCTION

The world wide web continues to point out the benefits of highly distributed, federated architectures to solve many web problems. However, there are always several related data sources for simultaneous consideration by web applications that causes performance and accuracy problems for federated queries. To improve the information retrieval performance and accuracy, it is required to rank data sources (ontologies) and enrich the original user query. The process of adding terms to an original user query to cover the gap between the user query and required information is called query expansion. For example, consider several bibliography ontologies which present description of publication like Author, Organization, Product, Event and Topic. They include several classes and their relationships between them. When a user searches for publications and Academic Staff, it is desired to retrieve the URIs for different kinds of publications (i.e. Article, Book, Proceedings) and the URIs for all Academic Staff in a correct order. But this knowledge stores in different ontologies. Different query expansion approaches use dependent or independent knowledge models [4] for their expansion purpose. In the dependent knowledge model, the system uses term co-occurrence, concept node structure or distribution analysis of terms in collections. In the independent knowledge model, the system uses structured knowledge such as domain ontologies or general ontologies (i.e. WordNet) [4],[16]. In our bibliography example, several expansion terms can result such as Thesis, Tech Report, Soft Copy, Association, Collection and Writer. The major problem with different available expansion methods is that they may worsen the query result in some cases [3]. Thus, the goal of many researchers is to discriminate between different expansion terms and improve the robustness of query expansion.

In this paper, we propose a novel weighting mechanism to find the expansion terms based on a combination of independent and dependent knowledge benefiting from both methods. We use ontologies as independent knowledge and find expansion terms based on their structure. We determine the Basic Expansion Terms (BET) in each ontology based on their semantic similarity, density and betweenness to user query. Then, we use the idea of co-occurrence across ontologies as dependent knowledge. We find the New Expansion Terms (NET) by aligning ontologies. Similar entities are determined by their name and structural similarity in each ontology. At the end of expansion we generate a set of terms along with weight as a query vector. Weights are calculated based on different criteria including Semantic Similar-
ity Measure (SSM), Density Measure (DM), Alignment Confidence Measure (ACM), and Weight of Semantic Path (WSP). Then, we rank ontologies and use their rank as a heuristic to find the set of Robust Expansion Terms (RET) with the largest weights. Finally, we use Blackbook [8] to facilitate federated queries and consider a number of data sources. Blackbook exploits Lucene indexing to find the matching of the query vector based on cosine similarities.

The idea of ranking ontologies is not only considered as a solution for query expansion scenarios, but also it is a solution for a wider scope of knowledge searching in ontology-driven searches. We rank ontologies based on the frequency of their Common Subset of Entities (CSE). To the best of our knowledge, all of the available ontology ranking techniques consider the structure of a single ontology and rank it accordingly by focusing on its own. In other words, current state of the art ranking does not discriminate entities across ontologies. Therefore, ontologies having shared entities may not get high ranking. We propose a novel strategy to rank ontologies by their CSE. CSE is considered as verified entities across different ontologies in the same domain. In our work, we extract all entities and relations between them using RDF graph of ontologies. We find out the CSE between two ontologies using the name similarity and the structural similarity of RDF graph with an efficient graph-based matching algorithm. CSE between a pair of ontologies indicates the frequency of repetition of entities. Based on the Zipf law, highly ranked entities occur more frequently in different ontologies [5]. Furthermore, we use the Entropy Based Distribution (EBD) technique to find the distance between each pair of ontologies. Once we calculate EBD values between two ontologies, then we determine average EBD values for each ontology by considering the rest of ontologies. Finally we rank ontologies based on two approaches. In the first approach, ranking is purely applied by average EBD values in ascending order. In the second approach, we apply bisecting clustering algorithm to group similar ontologies together. Then we rank each group based on the number of ontologies appearing in the group and EBD values.

The contribution of our paper is as follows:

- Determining BET based on structure of individual ontologies including the density, betweenness and semantic similarity of terms in each ontology.
- Finding the Common Set of Entities (CSE) across multiple ontologies and specifying the set of candidate terms that are semantically similar to the original query among different ontologies. This is used for both ranking ontologies and finding additional terms across ontologies calling NET.
- Defining a similarity measurement between ontologies based on Entropy Based Distribution (EBD) of entities for ranking ontologies using the CSE.
- Defining a novel weighting measurement for extracting Robust Expansion Terms (RET) using heuristic. Weight of expansion terms are calculated based on betweenness and centrality, density, semantic similarity, weight of semantic paths measurements and alignment confidence. Ontology rank is used as heuristic to calculate RET.
- Analyzing the improvement of our query expansion algorithm and comparing it to other existing methods [16].

The rest of this paper is organized as follows. In section 2, we present a survey on related works. In section 3, we provide the architecture for query processing in a service oriented and federated environment to lay the context of our research. We will also describe the problem statement. In section 4, we explain details of our proposed solution. First, we describe our aligning algorithm to calculate CSE. Second, we explain information theory measurement to calculate distance between ontologies and rank ontologies. Finally, we present details of our robust query expansion mechanism in section 5. In section 6, we present experimental results of aligning and ranking OAEI [9] and I3CON [10] benchmark by comparing traditional approaches. Also, we present the experimental results our expansion for series of queries. Finally, in section 7, we summarize conclusions and potential future work.

2 Related Works

Various approaches exist for conducting query expansion using ontological information. Perez-Aguera et al. [3] compare and combine different methods for query expansions. They consider co-occurrence of terms in different documents using Tanimoto, Dice and Cosine coefficients to weight expansion terms. Also, they analyze the distribution of expansion terms in the top ranked documents and the whole collection of documents using Kullback-Liebler Divergence. Bhogal et al. [4] review ontology based query expansion and explain various query expansion approaches including corpus dependent and independent knowledge models. Finally, they present different case studies on query expansion using domain-specific and general ontologies. They focus on a single ontology for query expansion and do not consider more than one ontology and related issues to ontology alignment. Our work relies on a combination of the co-occurrence method and knowledge models (ontologies). That is, we define BET in each individual ontology using user query keywords and then align each ontology's BET to other ontologies entities. We propose a novel weighting approach based on the combination of ontology structure analysis and the weight of semantic path between expansion terms.

Alani et al. [1] use relevance of multiple query keywords for ontology ranking. The ranking of on-
ologies reflects the importance of each relevant vocabulary which is defined by centrality and density of the vocabularies. Centrality is the distance of a vocabulary to the central level of its hierarchy and density is a weighted summation of the number of its subclasses, siblings, and relations. We use the idea of centrality and density of classes as an important factor in our weighting mechanism for class structure analysis. Also, we calculate the weight of semantic path using an information theoretic approach. This enables us to find NET based on entity weights.

Lee et al. [2] propose an effective semantic search using ontologies. They consider the number of meaningful and distinguishable semantic relationships between keywords and resources to find the most relevant k-top resources. Our work is different from [2] in that they find the most relevant terms using only a single ontology. We present the idea of finding the BET in each individual ontology. Then, we align different ontologies with each other and find NET in other ontologies.

Our previous work [11] presents ranking ontologies using the CSE algorithm. We find the Common Subset of Entities (CSE) between two ontologies using the lexical and structural similarity of terms and finding the similarity between ontologies by an Entropy Based Distribution (EBD) measurement. Further, we rank ontologies based on information retrieval authority score. Also, we propose ontology driven query expansion using different weighting techniques in [12]. We choose the k-top rank entities as the most semantically relevant terms to query keywords and good candidates for expanding the user query. K-top rank entities are determined by different weighting criteria including betweenness and density, semantic similarity, ontology alignment and weights of semantic paths. Then, we use a static threshold to distinguish between expanded terms and robust expansion terms. In our current work, we use the same weighting techniques to determine expansion terms as [12], but we use the rank of ontologies as a heuristic for calculating robust expansion terms. That is, we choose robust expansion terms from the most common entities among different ontologies. Rank of ontologies is a very good indicator of common entities in each ontology. Therefore, we choose more expansion terms from the ontologies with higher rank and less expansion terms from the ontologies with lower rank.

Our approach for aligning ontologies relies on an efficient algorithm for determining the Common Subset of Entities (CSE) in the semantic network of ontologies. Wang et al. [5] produce matching pairs for all nodes of two input graphs and create sorted lists for every node. They determine the similarity of two nodes based on degree differences of nodes and their direct neighbors, node attribute similarity and edge attribute similarity. However, no query expansion has been addressed in this method. On the other hand, we use the same structure for matching entities from the semantic of two ontologies. We compute the edge attribute similarity by finding the cosine similarity of edge types and first neighbors. Finally, we use a specific threshold applied to the average of all similarities and determine the common subset of entities.

In the next section, the details of our query processing service are explained.

3. Service Oriented Query Processing Using Ontology Structure and Rank

In this section we describe the concept of operation of our system and state the problem we are tackling. Figure 1 presents our architecture for federated query processing as a service for web applications. In our system, the user queries (keywords) are received by web application. The web application invokes the federated query processing service. This service will then invoke a service to compute the ontology ranks and query expansion terms. In particular, this latter service computes the BET, NET and RET along with dynamic threshold for the query. The results are then returned to the federated query service. Our research improves the performance and accuracy of web application retrieval. Details of BET, NET and RET calculations are explained in section 4.

Fig. 1. Federated Query Service.

As we have mentioned in Section 1, ranking ontologies accurately and efficiently is crucial for robust query expansion which in turn is needed for high performance service based query processing. In the remainder of this section we will give a formal definition of the problem. Our solutions to the problem will be given in details in Sections 4 and 5.

Given a query Q with m keywords and different ontologies \( \{O_1, \ldots, O_n\} \), all describing semantic information on a particular domain with different level of coverage, the goal is to determine robust query expansion for Q based on the structure of ontologies. That is, a query expansion for Q is a set of weights \( w_1 = (w_{11}, w_{12}, \ldots, w_{1m}) \) over entities in a vocabulary V of several ontologies, where \( w_i \in [0,1] \). Each \( w_i \) represents a weight for term \( v_i \in V \). Using a rounding soft threshold, intuitively \( w_i \) is an indicator to include or exclude the term from expansion to retrieve the most relevant top-k terms. The key idea is to find robust expansion terms including the best semantically related terms that improve the query retrieval. For instance, considering Bibliography domain, suppose there are four bibliography ontologies including MIT, UMBC, Karlsruhe and INRIA ontologies [10] and a user searches for individuals and corresponding web URLs for Academic Staff and Publication. As depicted in
Figure 2, possible query expansion in Karlsruhe ontology is defined as \( \text{Publication} = \{ \text{Article, Book, Booklet, InBook, InCollection, InProceedings, Manual, Misc, Proceedings, Report, Technical Report, Project Report, Thesis, Master Thesis, PhD Thesis, Unpublished} \} \) and \( \text{Academic Staff} = \{ \text{Faculty Member, Lecturer} \} \), while in UMBC ontology, possible expansion is specified as \( \text{Publication} = \{ \text{Article, Book, InBook, InCollection, InProceedings, Master Thesis, PhD Thesis, Misc, Proceedings, Tech Reports} \} \) and \( \text{Academic Staff} = \{ \} \) as shown in Figure 3. Thus, it is required to determine a mechanism to discriminate between the related terms that improves the query retrieval. For this, we concentrate on expanding terms using ontology alignment, ontology rank and the structure of ontology.

![Fig. 2. Karlsruhe Bibliography Ontology.](image)

![Fig. 3. UMBC Bibliography Ontology.](image)

4 **Ranking Ontologies Techniques**

4.1 **Ontology Alignment**

In our ontology alignment method, an RDF graph is used to exhibit the structure of ontologies for the purpose of entity verification. The RDF graph of OWL DL ontology presents necessary elements for describing the semantic structure of ontologies.

**Definition 1 (Ontology Graph):** Ontology graph is a directed, cyclic and connected graph \( G = (V, E) \), where \( V \) include all the entities of ontology and \( E \) is a set of all properties between entities.

**Definition 2 (Ontology Vocabulary):** Ontology vocabulary of an RDF graph is all subjects and objects that are RDF URI references and defined in RDF sentences. They cover all entities of an ontology including concepts, object properties, data properties and individuals. They do not belong to the built-ins provided by Ontology Languages.

**Definition 3 (Zipf Law):** Given some corpus of natural language utterances, the frequency of a word is inversely proportional to its statistical rank such that \( P(r) \approx \frac{1}{r^\beta} \) (1) where \( R \) is the number of different words.

We use Zipf Law for entity verification. In a set of domain-specific ontologies, the most frequent entity will occur approximately twice as often as the second most frequent entity. This property holds for other entity ranks as well.

4.1.1 **Common Subset of Entities (CSE) between Each Pair of Ontologies**

In this section, we present our algorithm to find CSE between ontology graphs. The algorithm determines the CSE of two ontologies using similarity bounds in the graph of ontologies as follows.

**Step 1:** Given \( O_1 \) and \( O_2 \), the algorithm produces the matching pairs of vocabularies for input ontologies such as concepts \( C_i \in O_1 \) and \( C_j \in O_2 \) along with a similarity measurement between them. Matching pairs are placed in a sorted linked list according to the similarity measurement, so that given a particular entity from \( O_1 \), the most similar corresponding concept in \( O_2 \) can be retrieved as the first element of the list. In our algorithm, we define a similarity function as a combination of Name and Structural similarities. The algorithm works in the following way: First, we examine all the entities from the first ontology and create a list for them. Second, we compare each of the entities of the first ontology with the second ontology and find out separately their name similarity and structural similarities. Name similarity between entities is measured by the Jaro Winkler distance method. Structural similarity between entities is estimated by difference of edge types (properties) between corresponding entities in sorted linked lists and first neighbor similarity. Finally, the corresponding sorted list for each entity is stored in CSE data structure between two ontologies. Details of structural similarity measurement is explained below.

- **Neighbors Similarity (NS):** We define a procedure to calculate similarity of entities from two different ontologies based on their direct neighbors in an ontology graph. The average of similarities between neighbors of two entities is calculated as a Neighbor Similarity measurement. First, we consider all the neighbors of corresponding entities from \( O_1 \) and \( O_2 \). Next, we check if the direct neighbors are the same using name match; we increase total similarity value; and finally, we normalize the total similarity value.

- **Similarity of Relation Types (SRT):** Considering each pair of similar entities from linked list like \( O_1 - C_1 \) and \( O_2 - C_2 \), we define two separate Relation Type Vectors (RTV) corresponding to \( C_1 \) and \( C_2 \). Each vector has "n" entries according to the relation types of entity and its neighbors. By checking each relation type for concepts
and neighbors, corresponding RTV entries are increased as an indicator for the number of relation types. Using RTV ($C_1$) and RTV ($C_2$), we are able to find the cosine similarity of edges for $C_1$ and $C_2$. Algorithm 2 presents the detailed steps for calculating SRT as follows. In line 2 on the algorithm, we find all the relation types for concept $C_1$ and create the type array for it. In line 3 and 7, we find out the type array for $C_1$ from $O_1$ and $C_2$ from $O_2$, respectively. In line 10, cosine similarity between two arrays is found and the sorted link list for each concept is updated based on the corresponding total structural similarity calculated by NS and SRT.

**Algorithm 1 Similarity of Relation Types (SRT)**

**Require:** Entity $C_i \in O_i$, Entity $C_j \in O_j$, Relation Types ($C_i$, Neighbors), Relation Types ($C_j$, Neighbors)

**Ensure:** Overall Neighbor Similarity

1. $RTVC_{C_1} = \{0, 0, ... , 0\}$, $RTVC_{C_2} = \{0, 0, ... , 0\}$
2. for all neighbors of $(O_1 - C_1)$, $n_1$
3. 3. Check relation Type i ($n_1$, $O_1$, $C_1$), ($n_1$, neighbor))
4. 4. $RTVC_{C_1}[i] = RTVC_{C_1}[i] + 1$
5. 5. end for
6. for all neighbors of $(O_2 - C_2)$, $n_2$
7. 7. Check relation Type j ($n_2$, $O_2$, $C_2$), ($n_2$, neighbor))
8. 8. $RTVC_{C_2}[j] = RTVC_{C_2}[j] + 1$
9. 9. end for
10. 10. SRT ($C_1$, $C_2$) = Cosine Similarity ($RTVC_{C_1}$, $RTVC_{C_2}$)
11. 11. return (SRT ($C_1$, $C_2$))

**Step 2:** In this step, CSE is built through a series of linked lists by choosing a pair of similar entities from each linked list. The algorithm defines a priority queue. Whenever an entity is selected from the ontology, all of its children are added to the priority queue. The algorithm uses a similarity threshold to perform pruning. If similarity value is less than threshold, that branch is ignored and the algorithm keeps track of classes with the remaining children in the priority queue. Each entity is checked to ensure that it doesn’t already appear in the path.

**Step 3:** The procedure of finding CSE is continued until all paths have been considered. The procedure stops when the threshold requirement satisfies, if a threshold size was specified.

CSE algorithm determines similar entities between each pair of ontologies as explained in Algorithm 1. It uses a fixed threshold for approximate calculation of final similar entities between ontologies. Considering the Bibliography example, we use threshold = 0.9 that results in the following matches.

- $O_{K}r\text{tars}e\text{h}c_{Author, O_{UM}BC:Author, 1}$
- $O_{K}r\text{tars}e\text{h}c_{TechnicalReport, O_{UM}BC:TechnicalReport, 0.92}$
- $O_{K}r\text{tars}e\text{h}c_{Author, O_{UM}BC:Author, 1}$
- $O_{K}r\text{tars}e\text{h}c_{MasterThesis, O_{UM}BC:MastersThesis, 0.95}$
- $O_{K}r\text{tars}e\text{h}c_{PhDThesis, O_{UM}BC:PhDThesis, 0.94}$
- $O_{K}r\text{tars}e\text{h}c_{Note, O_{UM}BC:Note, 0.90}$
- $O_{K}r\text{tars}e\text{h}c_{Misc, O_{UM}BC:Misc, 0.90}$

**4.2 EBD Ranking Mechanism**

Considering two ontologies, CSE algorithm determines a set of similar entities between them. In this section, we introduce a distance measurement that uses the set of similar entities from CSE algorithm and specifies the distance between two original ontologies. Furthermore, we explain the details of two different ranking mechanisms that use the distance between each pair of ontologies and rank them accordingly.

**Definition 4:** Let $X$ be a discrete random variable. The entropy $H(X)$ of the distribution $P_i = P(x_i) = P(X = x_i)$ is computed as:

$$H(X) = \Sigma_i P(x_i) \cdot \log P(x_i)$$

**Definition 5:** Conditional entropy of a random variable $X$ given $Y$ is denoted by $H(X | Y)$ and is computed as:

$$H(X | Y) = H(X,Y) - H(Y) = \Sigma_i P(x_i, y_i) \cdot \log P(x_i | y_i)$$

**Definition 6:** The mutual information between two discrete random variables $X$ and $Y$ is defined as:

$$I(X; Y) = H(X) - H(X | Y) = \Sigma_i \Sigma_j P(x_i, y_j) \cdot \log \frac{P(x_i, y_j)}{P(x_i)P(y_j)}$$

**Definition 7:** Relative Entropy (Kullback-Leibler) distance is a measure for comparing two distributions $P$ and $Q$ and is denoted by $KL(P, Q)$. It is computed as follows:

$$KL(P, Q) = \Sigma_k P_k \cdot \log \frac{P_k}{Q_k}$$

**Definition 8:** Jensen-Shannon distance is a standard measure for comparing distributions $P$ and $Q$. It is computed as:

$$JS(P, Q) = \alpha * KL(P, Q) + (1 - \alpha) * KL(Q, P)$$

Where:

$$M = \alpha + \beta * Q, 0 < \alpha, \beta < 1, \alpha + \beta = 1$$

We define an information theory measurement to estimate the similarity between ontologies. We calculate the similarity of ontologies using Entropy Based Distribution (EBD) of each type across different ontologies. Types are defined as groups of similar entities between ontologies that are calculated by CSE algorithm [16]. EBD is calculated as

$$EBD = \frac{H(X,Y)}{H(Y)}$$

In this Equation, O and T are random variables where O indicates the union of entities from two ontologies (i.e. $O_1$, $O_2$) and T indicates different groups of entities that are calculated from ontology alignment. EBD is a normalized value with a range from 0 to 1, where 0 indicates the lowest EBD, or no similarity whatsoever between compared ontologies, and 1 indicates the highest EBD. Intuitively, EBD is a comparison of the ratio of entity types for each distinct value type (conditional entropy) with entity types in O (entropy). An ontology O contains high entropy if it is impure; that is, the ratios of entity types making up O are similar to one another. On the other hand, low entropy in O exists when one entity type exists at a much higher ratio than any other type. Conditional entropy is similar to entropy in the sense that ratios of entity types are being compared. However, the difference is that before computing this ratio, CSE gives us a subset of entity types that all are associated with a given value type. Figure 4.a provides an example to help visualize the concept. In this example, crosses indicate value types originating from $O_1$, while squares indicate value types originating from $O_2$. The entity types are represented as clusters (larger circles), each of which is associated with a number of entity values from $O_1$ and $O_2$. In Figure 4.a, the total number of crosses is 10 and the total number of squares is 11, which implies that entropy is very high. The conditional entropy is also quite high, since the ratios of crosses to squares within 2 of the clusters are equal and nearly equal within the other. Thus, the ratio of conditional entropy to entropy will be very close to 1, since the ratio of crosses to squares is nearly the same from an overall perspective and from an individual cluster perspective. Considering the EBD calculation, we now have a value representing the distance between each pair of ontologies. Thus, we rank ontologies using an authority score.
Authority scores are calculated in two methods including the naive and clustering approach as discussed below.

(a) Distribution of Entity Types (b) Ranking Ontologies using EBD Scores

Fig. 4. EBD Calculation and EBD Ranking

Derivation of EBD

Let \( O_1 \) and \( O_2 \) be two ontologies, that we would like to calculate their distance based on their similar entities. Each set of similar entities are considered as a type. Using the soft clustering of two ontologies, it is required to associate conditional probabilities \( P(t \mid x) \) for each entity in \( O_1 \cup O_2 \) and \( t \in T \), the set of types. Type distribution of each entity \( x \) in \( O_1 \cup O_2 \) will be computed as:

\[
P(t \mid O_1) = \sum_{x \in O_1} P(t \mid x) * \frac{P(x)}{P(O_1)} \tag{8}
\]

For this, we will find the distance between two ontologies using Equation 6. Our goal is to determine similarity between two distributions of each type that came from \( O_1 \mid O_2 \). We have:

\[
J_S(P(t \mid O_1), P(t \mid O_2), P(O_1), P(O_2)) = P(O_1) * K_L(P(t \mid O_1), P(t)) + P(O_2) * K_L(P(t \mid O_2), P(t)) \tag{9}
\]

Using Equation 5 for ontology \( O_1 \) and type \( t \), we have:

\[
K_L(P(t \mid O_1), P(t)) = \sum_t P(t \mid O_1) * \frac{P(O_1)}{P(t)} \tag{10}
\]

\[
P(t \mid O_1) = \frac{P(t)}{P(O_1)} \tag{11}
\]

Using (10), (11) in Equation (9), will result in the following:

\[
J_S(P(t \mid O_1), P(t \mid O_2), P(O_1), P(O_2)) = P(O_1) * \sum_t \frac{P(t)}{P(O_1)} * \log \frac{P(O_1)}{P(t)} + P(O_2) * \sum_t \frac{P(t)}{P(O_2)} * \log \frac{P(O_2)}{P(t)} \tag{12}
\]

After some simplification, we have:

\[
J_S(P(t \mid O_1), P(t \mid O_2), P(O_1), P(O_2)) = \sum_t P(t, O_1) * \log \frac{P(O_1)}{P(t)} = I(O_1; T) \tag{13}
\]

The above Equation proves that for more ontologies \( O_1 \) and \( O_2 \) (as indicated by value types), \( I(O_1; T) \) will be lower. For example, for two identical ontologies \( O_1 \) and \( O_2 \), \( I(O_1; T) = 0 \). On the other hand, in the case when \( O_1 \) and \( O_2 \) are dissimilar, \( I(O_1; T) \) will be higher.

Based on definition 6:

\[
I(O_1; T) = H(O_1) - H(O_2 \mid T) \tag{14}
\]

Therefore:

\[
\frac{I(O_1; T)}{H(O_1)} = 1 - \frac{H(O_2 \mid T)}{H(O_2)} \tag{15}
\]

We define Entropy Based Distribution (EBD) as follows:

\[
\frac{I(O_1; T)}{H(O_1)} = 1 - EBD \tag{16}
\]

The above Equation proves that EBD and mutual information are inversely proportional. The higher the EBD between \( O_1 \) and \( O_2 \), the lower the mutual information, and the opposite is also true.

4.3 Ranking Mechanisms

Considering the EBD values between each pair of ontologies, now we present two different methods to rank ontologies based on the concept of hub and authority scores in web information retrieval.

Definition 9 (Authority Score): Given a set of web pages in the internet, counting the number of links to a page can give us a general estimate of its prominence on the Web.

A page that is particularly popular and linked by many different directories will typically have a higher authority score than a page that is unpopular. We calculate authority score for each ontology and rank them. Ontology with more relevant (similar) entities than all the other ontologies in a specific domain has higher authority score. In our methods, we use EBD measurement as an important factor to determine the authority score. For a given set of ontologies \( \{O_1, O_2, \ldots, O_n\} \) in the same domain, first CSE determines the similar entities between each pair of them. Second, we calculated the EBD between each pair using the common entities. Finally, ontologies are ranked using the authority score. Below is detail of our ranking methods.

4.3.1 Naive Method for Ranking Ontologies

In this method, we rank ontologies based on the average of EBD distance with other ontologies. Figure 4.b portrays our naive ranking process using the authority score. For each \( O_i \) in a set of ontologies, authority score is calculated by finding the average of EBDs between \( O_i \) and all other ontologies. As Figure 4.b presents, \( O_1 \) has similarity to six other ontologies, while \( O_2 \) has similarity only with three other ontologies; meaning that \( O_2 \) doesn’t have any similarity with ontologies \( O_4, O_5 \), and \( O_6 \). Thus, \( O_1 \) has a better rank and higher authority score than \( O_2 \). Intuitively, higher authority score for \( O_1 \) shows that distance between \( O_1 \) and all other ontologies is lower than the distance between \( O_2 \) and all other ontologies; therefore, this ontology, \( O_1 \), has better quality among the set of ontologies. Thus, it has the highest rank.

4.3.2 Bisecting K-Medoids Clustering (BKMC) Algorithm

In this method, we use BKMC to rank ontologies using the EBD as distances between ontologies. The main idea of BKMC is similar to that of the centroid-based bisecting k-means clustering. BKMC starts by considering the whole dataset to be one cluster. At each step, one cluster is selected and bisected further into two sub-clusters using the basic k-median. The clustering solution of the BKMC is mainly based on selection criterion for splitting. Usually, the larger cluster or clusters with least overall similarity are considered as a candidate for splitting. In our paper, the average cluster compactness as a measure of homogeneity of each cluster is defined as:

\[
\text{Average Compactness} (C_i) = \frac{\sum_{o_j \in C_i} EBD(o_j, \text{medoid}_{o_j})}{|C_i|} \tag{17}
\]

Where \( C_i \) is the \( i^{th} \) cluster and \( \text{medoid}_{o_j} \) is the medoid of \( C_i \). \( O_j \) is the ontology in the \( i^{th} \) cluster. A small value of this measure indicates a high homogeneity of ontologies, meaning more similar ontologies. The clustering step of BKMC continues until the compactness of all clusters becomes almost equal. After calculating the instances of each cluster, we specify authority score for each cluster by Equation 18.

\[
\text{Authority Score}(C_i) = \frac{|C_i|}{\sum_{o_j} EBD(o_j, O_k) + \sum_{o_j} EBD(o_j, O_k)} \tag{18}
\]

Where \( |C_i| \) represents the cardinality of cluster \( C_i \), \( |O| \) represents total number of ontologies. EBD represents the distance of an ontology with other ontologies in the cluster \( C_i \). Algorithm 2 presents the sequence of operation of BKMC clustering and authority score as follows. In line 1, the algorithm finds clusters by bisecting the set of all ontologies based on the number specified by user. It chooses two random medoids and creates two sub-clusters using medoids. Then the algorithm chooses the cluster for further partitioning based on selection criterion for splitting. Usually, the larger cluster (or clusters) with least overall
Algorithm 2 BKMC Algorithm for Ontology Ranking

Require: Set of ontologies \( O = \{O_1, O_2, \ldots, O_N\} \).
The number of clusters \( k \).
Ensure: The Authority Score for Clusters \( C = \{C_1, C_2, \ldots, C_k\} \).
1: \( \{C_1, \ldots, C_k\} = \text{Bisecting K-medoids} (O = \{O_1, O_2, \ldots, O_N\}) 
2: \text{for all clusters} \ i = 1 \to k \ \text{do} 
3: \text{if} \ (\text{Cardinality}(C_i) > 1) \ \text{then} 
4: \quad \text{AuthorityScore}(C_i) = \frac{|C_i|}{|O|} \cdot \sum_{i \in C_i, \text{EBD}_i} \text{EBD}_i 
5: \text{end if} 
6: \text{if} \ (\text{Cardinality}(C_i) = 1) \ \text{then} 
7: \quad \text{AuthorityScore}(C_i) = \frac{|C_i|}{|O|} \cdot \text{AverageEBD} 
8: \text{end if} 
9: \text{end for} 
10: \text{return} (\text{AuthorityScore}(C_i)) 

similarity is (are) considered as a candidate for splitting.
Algorithm repeats clustering until we have as many clusters as specified by the user. Then, the algorithm finds the
authority score for each cluster using Equation 18 between pair \( i \) of ontologies in cluster \( C_i \) in line 3 and line 4. In
the case that a cluster has a single ontology, the algorithm
uses the average EBD between that ontology and all the
other ontologies in the set \( O \), as stated in line 6. Because
there is no summation of EBDs for a single ontology, we
use the average EBD of all the other ontologies to have
a uniform weight. Finally, in line 7 algorithm returns all
Authority scores.

Using Naive or BKMC algorithms, we are able to
determine the rank of ontologies. Rank of ontologies is
very important for calculating weights in query expansion
process. Next, we discuss all steps of our ontology ranking
approach by an example of three sample ontologies.

Example of Ontology Ranking

Three sample ontologies as subgraphs of Karlsruhe, UMBC,
MIT Bibliography Ontologies are depicted in Figure 1,
Figure 2 and Figure 5. We explain the ranking process
for these three sample ontologies based on the common
subset of entities between them. First, we find CSE for
every possible pair of ontologies. Second, we determine
distance between ontologies by calculating EBD for each
compared pair. Finally, we use naive and BKMC clustering
to rank ontologies. In order to rank ontologies, we need to
determine CSE and calculate EBD three times as follows.
EBD (Karlsruhe, UMBC), EBD (MIT, UMBC) and finally
EBD (Karlsruhe, MIT).

Considering the two ontologies, MIT and UMBC, following mappings are determined as similar entities using CSE.

\[
\begin{align*}
(O_{\text{MIT}}, \text{Proceedings}, O_{\text{UMBC}}, \text{Proceedings}, 0.99), \\
(O_{\text{MIT}}, \text{Proceedings}, O_{\text{UMBC}}, \text{InProceedings}, 0.96), \\
(O_{\text{MIT}}, \text{TechReport}, O_{\text{UMBC}}, \text{TechReport}, 0.99), \\
(O_{\text{MIT}}, \text{MasterThesis}, O_{\text{UMBC}}, \text{MastersThesis}, 0.95), \\
(O_{\text{MIT}}, \text{PhDThesis}, O_{\text{UMBC}}, \text{PhDThesis}, 0.94), \\
(O_{\text{MIT}}, \text{Article}, O_{\text{UMBC}}, \text{Article}, 0.90), \\
(O_{\text{MIT}}, \text{InCollection}, O_{\text{UMBC}}, \text{InCollection}, 0.90), \\
(O_{\text{MIT}}, \text{InBook}, O_{\text{UMBC}}, \text{InBook}, 0.90), \\
(O_{\text{MIT}}, \text{Misc}, O_{\text{UMBC}}, \text{Misc}, 0.97).
\end{align*}
\]

Using the above aligned classes, eight different clusters are specified as types and finally EBD (MIT, UMBC) is calcul-
ated. Following the same procedure we calculate similarity
between other subgraphs. EBD values are as follows.

EBD (MIT, UMBC) = 0.60
EBD (Karlsruhe, UMBC) = 0.51
EBD (Karlsruhe, MIT) = 0.66

For ranking these three sample ontologies, average EBD
value for all of them is calculated. MIT = 0.65, UMBC = 0.55
and Karlsruhe = 0.58. Thus, we give higher authority scores
to ontologies with higher EBD values, that is MIT is ranked
as first, Karlsruhe as second and UMBC as third ontology.
In BKCM Clustering also C1 = {MIT}, C2 = {Karlsruhe,
UMBC} are the clusters while Authority Score (C1) = 0.21
and Authority Score (C2) = 0.34. Thus C2 has the first rank.
In the next section, we concentrate on robust query expansion
using ontology structure and rank.

Fig. 5. MIT Bibliography Ontology.

5 Robust Query Expansion

Our strategy toward determining Robust Expansion Terms
(RET) concerns producing a top-\( k \) candidate list as alter-
native terms to user query terms. To do this, we use different
factors that help to discriminate between terms in ontologies
for the expansion purpose. Any term expanded within
the same ontology is named BET. On the other hand, the
terms that are expanded using inter-ontology are called
NET. First we focus on BET strategy within intra-ontology.
That is, first we find the matching classes to query terms
within each ontology. Second, we determine the central class
using Betweenness Measure (BM). Third, we expand those
matching terms using the concept of Semantic Similarity
Measure (SSM) and Density Measure (DM) as discussed
below.

5.1 Basic Query Expansion Measures in Each Indi-
vidual Ontology for BET

Considering lexical matches to the query keywords in each
ontology, now we present different measures for BET in each
ontology as follows.

Density Measure (DM): This measurement expands
some details of each query keyword in an ontology graph.
That is, it evaluates properties and neighbors for each entity
similar to the query keyword including the subclasses,
superclasses, etc. It is limited to first relations [1]. Suppose
\( q_i \in O_i, c \in O_i \), and \( c = \text{NameMatch}(q_i) \) in \( O_i \). BET
includes all entities that have a relation with \( c \) that is \( \{c_1, c_2, \ldots, c_n\} \). DM(c) is the Density Measure of entity \( c \), which
represents the number of possible expansions for \( c \). Let \( RT \) = Set of all related types (properties) for \( c \) including
subclasses, superclasses, and relations.

\[
DM(c) = \sum_{\text{all relation types}} w_k \ast |RT| \quad (19)
\]

\( w_k \) is the weight factor for different relation types. The
weight factor enables us to emphasize some special relation
types like subclasses.

Betweenness Measure (BM) and Central Entity: This is
the method that determines the centrality of entities in
a single ontology. It assigns the number of shortest paths
that pass through each node in the graph when calculating
expansion terms. The node that occurs on many shortest
paths for expanding user terms is considered the central
keyword in each ontology [1]. Let \( c_1, c_2 \in O_k \). BM(c) is the
betweenness Measure of entity \( c \).

\[
BM(c) = \sum_{c_1 \neq c \neq c_2} \frac{(\text{shortestpath}(c_1, c_2) \text{passes})}{(\text{shortestpath}(c_1, c_2))} \quad (20)
\]

BM determines the central keyword that is used in SSM for
finding BET. Central keyword has the highest BM value.
Semantic Similarity Measurement (SSM): SSM uses ontology graph as a semantic presentation of a domain to determine weights for all expansion terms in every ontology. Entities that are closer to the central node have more weights. SSM is calculated using the shortest-path measure. The more relationships entities have in common, the closer they will be in the ontology [1]. If any entity is positioned relatively far from the central, then it has smaller weight. Therefore, we use the shortest path measure as weights for the ontology vocabulary. Let entities $c_j, c \in O_t$ and there is a path between $c$ (central) and $c_j$.

$$SSM(c, c_j) = \begin{cases} \frac{1}{\text{length(minpath}(c, c_j))} & i \neq j \\ 1 & i = j \end{cases} \quad (21)$$

BET in this method, are all entities in the shortest-path from central keyword to $c_j$.

Back to our bibliography example on Karlsruhe and UMBC ontologies in section 4.1, and a query on Academic Staff and Publication, first we find the lexical matches in both ontologies. That is, \{Academic Staff, Publication\} in Karlsruhe ontology and the \{Publication\} in UMBC ontology. Next, we find the central keyword in each ontology. For this, we list all the entities of each ontology and calculate the number of times query keywords are visited in the shortest path between the entities. In both ontologies Publication is the central keyword because it has more relations with other entities in compare to Academic Staff. Next, we complete the BET in each ontology using DEM and SSM. In Karlsruhe ontology BET={TechnicalReport, Proceedings, Report, Inbook, Projectreport, Thesis, InCollection, PhdThesis, Associateprofessor, Lecturer, Booklet, Misc, Book, MasterThesis, Facultymember, In-proceedings, Manual, Article, Unpublished, Manager, Technic-staff, Fullprofessor, Assistantprofessor, Employee, Academicstaff}, while in UMBC BET={Masterthesis, Techreport, Inproceedings, Inbook, Thesis, TechnicalReport, Misc, Person, Article, Incollection, Book, Phdthesis}. Further, we determine different measures for each term $c \in$ BET. For example, in Karlsruhe ontology, BM(Report)=2 because it is the shortest path from Technical report and Project Report to Publication. SSM(Report)=1 because the minimum path from Report to Publication has the length 1. DEM(Report)=3, because it has 1 subclass and 1 superclass.

Considering the BET in each ontology, now we focus on NET strategy using inter-ontology. For this, we align BET to semantically similar entities in other ontologies using an ontology alignment algorithm as discussed in the next section. This enables us to find all appropriate expansion terms in relevant ontologies. This operation continues between all possible pairs of ontologies. During the alignment process of two entities from different ontologies, if one of the entities does not belong to the BET, we consider that entity as the NET.

5.2 Query Expansion using Ontology Alignment for NET

In this section, we determine NET using the CSE algorithm explained in section 4.1. CSE aligns ontologies using the name and structural similarity of entities. Each entity $c_t \in O_t$ is compared with all other entities of the other ontology ($O_2$) and the similar pairs are determined [11]. Similar pair of entities along with their confidence measure (ACM) are used to specify the NET between each pair of ontologies.

For instance in our bibliography example, CSE algorithm finds a set of similar pairs as follows:

\begin{align*}
(\text{O}_\text{Karlsruhe:Author}, \text{O}_\text{UMBC:Author}), \ldots, \\
(\text{O}_\text{Karlsruhe:MasterThesis}, \text{O}_\text{UMBC:MasterThesis}, 0.95)
\end{align*}

As explained in section 4.1, Karlsruhe-BET does not include TechnicalReport, while UMBC-BET includes TechnicalReport. CSE algorithm aligned \{TechnicalReport\} to \{TechReport\} with confidence value of 0.92. Thus, \(\text{NET}_{UMBC} = \{\text{TechnicalReport}, \text{MasterThesis}\}\).

Next, we weight the NET that evolved as a result of ontology alignment across different ontologies. There are two ways to evaluate the weight of the NET. On the one hand, we exploit matching across ontologies by considering alignment confidence value; on the other hand, we solely rely on its own ontology for the weight. For the former, we calculate the weight of NET as the original weight of the referenced entity multiplied by ontology alignment confidence value. For the later, we measure the weight of NET from the central entity within its own ontology using BM, SSM and DM. Once we get weights of NET by BM on these two approaches, we decide the final weight of the NET by choosing either the minimum, maximum, or average values of these two weights (details can be found in Section 5.4). In the next section, we determine another weighting measure to calculate weights for semantic paths.

5.3 Weighting the Semantic Path (WSP)

We determine the weight of semantic path for all terms $c \in \text{BET} \cup \text{NET}$ using the amount of information contained in the properties of the path and characterizability of properties between entities including classes and object/data properties. We use information theory to measure the information content of each property in the semantic path. Content of a property is computed based on the occurrence probability of the property in the ontology vocabulary [2]. For a semantic path from central keyword $c$ to expanded $e_i \in \text{BET} \cup \text{NET}$, weight of path $(c, e_i)$ is calculated based on weight of the sequence of properties from $c$ to $e_i$. In our work, for every property $p(a,b)$, $a$ is a class (subject) and $b$ is either a class or object/data properties in ontology graph. Thus content property of $p(a,b)$, is computed as

$$I(p(a,b)) = -\log_2 \text{Pr}(p(a,b)) \quad (22)$$

where $\text{Pr}(p(a,b))$ is the probability that $a$ is the source (subject) of property $p(a,b)$ in ontology graph. If $a$ occurs rarely as the subject in the RDF graph, it has more information. Characterizability between entities (subject-object) of a property $p(a,b)$ in an ontology graph is measured by mutual information. Mutual information, MI represents the amount of information that a property subject has about the property object.

$$\text{MI}(p(a,b)) = \text{Pr}(p(a,b)) \cdot \log_2 \frac{\text{Pr}(p(a,b))}{\text{Pr}(p(a)) \cdot \text{Pr}(b)} \quad (23)$$

where $\text{Pr}(a)$ is the probability that $a$ is the subject for the property $p$ in the graph and $\text{Pr}(b)$ is the probability that $b$ is the object for the property $p$. $\text{Pr}(p(a,b))$ is the probability that property $p$ has $a$ as its subject and $b$ as its object at the same time.

Using the above Equations, we compute the weight of property $p(a,b)$ as follows.

$$W(p(a,b)) = \alpha \cdot I(p(a,b)) + \beta \cdot \text{MI}(p(a,b)) \quad 0 < \alpha, \beta < 1$$

and $\alpha + \beta = 1 \quad (24)$

As the length of the semantic path gets longer, the semantic relevance of terms decreases. Therefore for a path in the graph from a central entity (c) to expansion terms $e_i$, the weight of semantic path is as follows:

$$\text{WSP}(c, e_i) = \prod_{p(a,b) \in \text{sp}(c, e_i)} W(p(a,b)) \cdot \delta^{(\text{lengthpath}(e_i, c) - 1)} \quad (26)$$

where $\text{lengthpath}(e_i, c)$ indicates the number of properties in the semantic path $\text{sp}(c, e_i)$ and $\delta$ is an attenuation parameter $0 < \delta < 1$.

For example, in Karlsruhe bibliography ontology,

$$\text{WSP}(\text{Technical Report}, \text{Publication}) = W(p(\text{Technical Report}, \text{Report})) \cdot W(p(\text{Report}, \text{Publication})) \cdot \delta^{(2-1)}$$
Then, we need to calculate weight of each single length path as follows:
\[
W(p, \text{Technical Report, Report}) = \alpha \times I(p(a, b)) + \beta \times M(p(a, b))
\]
where \( p(a, b) = \) subclass and \( a = \) TechnicalReport and \( b = \) Report.

In Karlsruhe example, there are 19 subclass relations and 1 domain and 1 range relations. Thus \( I(p(a, b)) = 0.043 \) and \( M(p(a, b)) = 2.080 \). Using \( \alpha = \beta = \delta = 0.5 \), we have:
\[
W(p, \text{Technical Report, Report}) = 1.061
\]
\[
W(p, \text{Report, Publication}) = 0.82
\]

Thus, WSP(Technical Report, Publication) = 0.435

5.4 Combining Different Weighting Methods

Considering different weighting criteria for candidate terms including the SSM, BM, DM, WSP and ACM we are able to determine the overall weight for each term as follows.

If \( e_i \in \text{BET on an ontology } i \)
\[
W(e_i) = [\alpha \times SSM(e_i, c_i) + \beta \times BM(e_i) + \gamma \times DM(e_i) + \delta \times WSP(e_i, c_i)]
\]

SSM presents the semantic similarity between the central keyword and expansion term, BM represents the betweenness of expansion term and WSP shows the weight of semantic path between central and expansion terms. \( \alpha, \beta, \gamma, \delta \) are coefficients to put weights on some of the weighting methods and \( 0 < \alpha, \beta, \gamma, \delta < 1 \) and \( \alpha + \beta + \gamma + \delta = 1 \).

If \( e_j \in \text{NET on an ontology } j \)
\[
W(e_j) = \text{Optimize} [\text{ACM}_j, \alpha \times SSM(e_j, c_j) + \beta \times BM(e_j) + \gamma \times DM(e_j) + \delta \times WSP(e_j, c_j)]
\]
where ACM_j represents the alignment confidence measure of \( e_j \) related to \( e_i \) from ontology \( i \). For optimizing values in Equation 28, a number of options are available such as choosing a maximum value between these two, minimum value or take average values. Note that maximum may overestimate weights of terms; on the other hand, minimum may underestimate weight.

Considering the weights of terms \( e_i \in \text{BET} \cup \text{NET} \), in the next section we explain about dynamic threshold calculation to determine the k-top most relevant expansion terms namely robust expansion terms.

5.5 Robust Expansion Terms using Dynamic Interval (DI)

Determining the optimal number of expansion terms, is one of the fundamental questions in query expansion. In this section, we explain about the heuristic metric that we use to calculate the Robust Expansion Terms (RET). We determine Dynamic Interval (DI) based on the Rank of Ontologies (RO). Given \( S = \{O_1, O_2, \ldots, O_n\} \), a set of ontologies and their corresponding rank, for each ontology we determine DI in the way that we have more expansion terms from the ontologies with higher rank. That is, for the ontology with the highest rank (i.e. \( O_1 \)), we use the lowest threshold interval DI_1 = 1. Further for other ontologies (i.e. O_j), we calculate the ratio between the highest rank and rank of other ontologies to calculate Dynamic Interval as follows.
\[
DI_j = DI_1 \times \frac{RO_j}{RO_1} \quad (29)
\]

\[
\text{threshold}_j = \lfloor DI_j \times |\text{BET} \cup |\text{NET}| \rfloor \quad (30)
\]

Where |BET| \( \cup \) |NET| presents the number of expansion terms \( e_i \in \text{BET} \cup \text{NET} \).

Suppose \( \text{threshold}_j = k \) for \( O_j \), We calculate RET as
\[
\text{RET} = \Sigma_{j=1}^{n} \text{RET}_j \quad (31)
\]

\[
\text{RET}_j = \lfloor \text{Weight}(e_j) \in k\text{-top weights} \rfloor \quad (32)
\]

For each ontology, we find the ratio of current ontology rank over highest rank and dynamically calculate intervals (\( DI_j \)). Next, we find k-top rank terms based on the term weights. Intuitively, we accept all the expansion terms from the ontology with highest rank (\( O_1 \)) but discard lowest weights candidate from other ontologies (\( O_j \)) based on the ratio between the value of highest rank and current ontology rank (\( \frac{RO_j}{RO_1} \)). The number of RET depends on the query keywords and the structure of the ontology. The more user keywords have density and semantic relations, the more we get expansion terms. That is, if there are more relations or semantic paths between user keywords, we get more expansion terms out of them in each ontology.

5.6 Expand Federated Query (EFQ) Algorithm

Considering different steps in query expansion, Figure 6 presents the flow of operation and algorithm 3 explains the detail of operations as follows.

![Flow of Expansion Technique](Fig. 6. Flow of Expansion Technique)

Algorithm 3 presents the details of our expansion technique. In lines 4 and 5, we determine the similar entities to query terms in every ontology. Then, we find the central entity by calculating the maximum number of shortest paths between all user keywords in each ontology in line 9. Furthermore, in line 10 and 11, we expand and weight terms using BM, DM, SSM, WSP and OR in each ontology. In lines 12 and 13, we find the BET and their corresponding weights for all the ontologies by taking the union from each ontology. In line 17, we consider every possible pair of ontologies to find the similar pair of entities and alignment confidence measure between them. Similar entities are calculated only once and are used for every query expansion. In line 18, we consider every pair of BET in \( O_1 \) and \( O_n \) and check if there is any mapping between them by ontology alignment in lines 19 and 28. If there is no mapping between them we expand them using OA in line 21 and 25 and add them to NET. Otherwise, we do nothing. In line 22 and 26 we calculate weights of NET by Equation (28) and add the new weights to the weight vector \( W_{NET} \). In line 33, we get the final expansion terms by taking union between BET and NET. In line 34, we find the corresponding weights. In line 35, we find RET using DI. Finally, in line 36 we return RET elements along with their weights.

6 EXPERIMENTS

In this section, we present the result of our experiments on benchmark ontologies of I3CON [9] and OAEI [10]. We run our experiments on ontologies of different domains. In each domain, we classify our experiments into two sections. First, we rank ontologies in each domain. Second, we expand queries on each domain and show the effectiveness of our query expansion method. For the ontology ranking part, first we extract the RDF graph of ontologies by Jena API. Second, we use CSE algorithm to find the verified entities between each pair of ontologies and calculate EBD measurement between all pair of ontologies in the same domain. Third, we determine authority score for each ontology using naive and BKMC algorithms. Finally, we show the effectiveness of our EBD ranking results by comparing them with AktiveRank results. For the query expansion part, we determine corresponding terms \( e_i \in \text{BET} \cup \text{NET} \) for different user queries and calculate weights for terms. Based on the calculated weights, we determine RET with DI values. Then, we use Blackbook environment to check the efficiency of our weighting mechanism. Blackbook is a graph analytic platform for semantic web data. It provides the facility of

Algorithm 3 Expand Federated Query (EFQ)

Require: User Query (Q) with length m, Threshold th
Ensure: Set of Expanded Terms
1: for all Ontology i ∈ SO do
2: for all Terms j ∈ Q do
3: for all Entities k ∈ Oi do
4: if NameMatch(Termj, Entityk ∈ Oi) ≥ th then
5: QSj = {(qi, ejk) : qi = Termj, ejk = Entityk}
6: end if
7: end for
8: end for
9: Ci = Central entity between all terms in QSj
10: BETj = Basic expansion terms (Oi, Ci, QSj)
11: Wi = Calculate weights (BETj, Ci) using Equation (27)
12: BET = BET ∪ BETj
13: W BET = W BETU Wi
14: end for
15: NET = {}
16: for all possible pair (Ωa, Ωb) ∈ SO do
17: SSE = CSE (Ωa, Ωb)
18: for all possible pair of (a, b) where a ∈ BETv, b ∈ BETv do
19: if (a, b) ∈ SSE then
20: if ((a, c) ∈ SSE and c ∉ BETv then
21: NET = NET ∪ c
22: W BET = W NETU w, w = weight of term c by (28)
23: end if
24: if (b, d) ∈ SSE and d ∉ BETv then
25: NET = NET ∪ d
26: W BET = W NETU w, w = weight of term d by (9)
27: end if
28: else if ((a, b) ∈ SSE) then
29: Do not Expand
30: end if
31: end for
32: end for
33: BET = BET ∪ NET
34: W = W BETU W NET
35: RET = Di(W)
36: return (RET and their associated weights)

retrieving data from different ontologies (RDB/SDB database stores) in federated architecture [8]. Blackbook federates the keywords query across all the data sources and retrieve all the relevant instances (ontology individuals) in different data sources. It searches for keywords using the Lucene Index algorithm.

To assign the weight for expansion terms, we choose the average values in Equation 28. Also we use α = 0.25, β = 0.25, γ = 0.25 and δ = 0.25 for their parameter setting.

6.1 Datasets
We test our ranking and expansion mechanism in OAEI and I3CON benchmarks.
For ontology ranking purpose, we create some synthetic ontologies in Russia, Animals and Tourisms domains by appending and editing entities in the original Russia A, Animals B and Tourisms A ontologies. We append/edit different number of entities in each ontology based on the structure of original ontologies to analyze the weakness and strenght of our ranking algorithm against AKtive rank technique. In Animals domain, we create synthetic data Animals B12, Animals B50, Animals B70, and Animals B90 with 12%, 50%, 70% and 90% difference from original ontology Animals B by appending some new entities to the original ontology. In Russia domain, we create new ontologies including Russia A10, Russia A50, Russia A70 and Russia A85 with respectively 10%, 50%, 70% and 85% difference from the original Russia A by removing and editing some entities. In Tourisms domain, we apply the same operations as Animals domain and create Tourisms A10%, Tourisms A50%, Tourisms A70% and Tourisms A90% ontologies. In Food domain, we create new synthetic Food 90% by removing and editing some entities from the original Food ontologies.

Different metrics of ontologies in each domain are presented in Table 1, 2, 3, 4 and 5. For query expansion purpose, we concentrate on both original and synthetic Ontologies of Bibliography, Russia, Animals, Tourisms and Food domains. We have added the average of 20 instances to Bibliography and the average of 25 instances to Animals B ontologies, since they do not have any instances.

6.2 Results
Our experiments are performed in two different phases. In the first phase, we match I3CON and OAEI benchmarks ontologies using the CSE algorithm. Then, we rank ontologies in each domain using EBD measure and authority score. In the second phase, we expand queries in each domain separately using ontology structure and rank. Further, we analyze the improvement of query expansion in each domain.

6.2.1 Rank of Different Ontologies
We ranked ontologies in Animals, Russia, Tourisms domains from I3CON benchmark as well as ontologies in Bibliography and Food domains from OAEI benchmark. We used our method, EBD and AktiveRank method [1] to rank them. We show EBD values of a number of synthetic Animals, Tourisms and Russia Ontologies in Figure 7.

In EBD method, as Figure 7 presents, for synthetic Animals, Russia and Tourisms Ontologies, EBD values smoothly decrease while similarities between ontologies decrease. In this Figure, X-axis represents a number of synthetic ontologies and Y-axis indicates EBD values between synthetic ontology and the original ontology. For instance, in Russia domain, O10 represents Russia A10, O50 represents Russia A50, O70 represents Russia A70 and O85
represents Russia A85 for X-axis. While Y-axis represents the similarity between Russia (A10 and A), Russia (A50 and A), Russia (A70 and A), Russia (A85 and A). The same pattern also exists for Animals and Tourisms ontologies.

In the AktiveRank method, we use all entities of original ontology as keywords for ranking purpose, since AktiveRank relies only on a list of keywords to rank ontologies. Therefore, AktiveRank is not able to find the distance between two ontologies, which is a restriction for this method. Hence, we use a modified version of AktiveRank method in Animals, Tourisms and Russia domains and rank ontologies accordingly. In Animals and Tourisms domain, AktiveRank value remains almost constant, which is a problem with this method. The reason is that these two synthetic ontologies (Animals B, Tourisms A) preserve their original entities and have been modified by appending some additional entities. When we rank synthetic Animals B and Tourisms A ontologies, we use a constant set of keywords to compare with all corresponding Animals B and Tourisms A ontologies. We get almost the same AktiveRank values for all synthetic Animals B (4.65) and Tourisms A (251) ontologies because AktiveRank uses Class Match Measure, Density Measure, Semantic Similarity Measure and Betweenness Measure [1] which are all the same in these test cases. In Russia domain, AktiveRank values decrease when similarities between synthetic Russia ontologies and original ontology decrease. Therefore, while in Russia domain, synthetic ontologies have less entities or relations than the original ontology; both EBD and AktiveRank values decrease properly as shown in Table 6.b. The second and third columns of Table 6.a show EBD between Animals B and Tourisms A with their corresponding synthetic ontologies. The first value denotes our method EBD value and the second value denotes AktiveRank value. For example, EBD value between Animals B and Animals B12 is 0.87, while AktiveRank value is 4.65 respectively. Note that AktiveRank produces the same value between Animals and its 4 counterparts. On the other hand, our method EBD distinguishes difference between synthetic Animals ontologies. Therefore, this demonstrates the shortcomings of the AktiveRank method and the effectiveness of our method.

In the next step, we rank Animals and Russia ontologies by EBD ranking algorithm. Table 7.a presents the rank of Animals and Russia Ontologies in EBD naive and EBD BKMC. EBD BKMC Clustering results in three clusters including C1 = {AnimalsA}, C2 = {AnimalsB}, C3 = {AnimalsB12, AnimalsB50, AnimalsB70, AnimalsB90}. In Russia domain also, EBD BKMC finds five different clusters including C1 = {Russia1, RussiaA}, C2 = {RussiaC}, C3 = {RussiaD}, C4 = {RussiaB} and C5 = {Russia2}.

We expand our ranking mechanism in OAEI benchmark as well. Table 8 present the EBD rank of Bibliography and Food ontologies. In Bibliography domain, three clusters exist including C1 = {UMBC, MIT}, C2 = {INRIA} and C3 = {Karlsruhe}. While, in Food domain two clusters exist including C1 = {Food} and C2 = {Wine, Food90}. As we can see in both EBD Naive and BKMC methods, highest and lowest ranks are compatible in both methods. For instance, in Animals domain, highest rank ontology is AnimalsB70 and the lowest rank ontology is AnimalsA. In Russia domain, Russia1 and RussiaA has the highest rank values while Russia2 has the lowest ranks.

### 6.2.2 Query Expansion on Different Ontologies

We expand several user queries on different ontologies of each domain from both ECON and OAEI benchmarks. For this, first we expand queries using OAEI and NET for each query. Next, we calculate RET by a heuristic metric based on the rank of original ontologies. Finally, we analyze the effectiveness of our query expansion and dynamic threshold by presenting experiment results.

Considering the query keywords={Academic Staff, Publication} from section 3, we present BET and NET in the Table 11. Based on the structure of each ontology and query keywords, different query expansions may occurs. In our example three different cases exist for this query.

**Case I:** query is expanded by BET (i.e. in Karlsruhe Ontology first row of table).

**Case II:** query is expanded by NET (i.e. in MIT Ontology second row of table).

**Case III:** query is expanded by both NET and BET (i.e. in UMBC Ontology third row of table). For each ontology, first we apply the RET calculation to all terms ∈ RET ∪ NET and further, we find the union of all expanded terms from different ontologies. Considering the terms∈ RET ∪ NET, we use rank of ontologies as a heuristic metric to calculate Dynamic Interval (DI) and find the set of highly weighted terms in each ontology. In our previous work [12], we calculated the RET without using any heuristic. We use different SI values and the maximum
weight of expansion term for RET calculation. In different ontologies and domains, we can have different optimal values for SI, therefore here for each query we calculate dynamic intervals and find the k-top highest rank expansion terms. For example, in Bibliography domain using different SI, we found SI=90% as optimal value for RET calculation, that eliminates unrelated term (smallest weights) \{Person\} from UMBC and \{Manager,TechnicalStaff,Employee\} from Karlsruhe-BET. While in our new heuristic method, we eliminates \{Person,AcademicStaff\} from UMBC and \{Manager,TechnicalStaff,Employee,Unpublished,AdministrativeStaff,TechnicalReport,ProjectReport\} from Karlsruhe-BET. While we calculate BET for different queries, some related expansion terms may be eliminated in our RET calculation like in Karlsruhe-BET. But finding RET from different data sources finally results in only adding related terms to our final RET set from some other data sources (ontologies). This helps to improve the precision calculation, when we are using DI heuristic. For instance, in above example, \{TechnicalReport,Unpublished\} are related terms and will be added to RET set later by \textit{UMBC_{NET}} and \textit{MIT_{NET}} calculation.

Table 9 presents details of dynamic query threshold calculation using Ontology Naive Ranks, while Table 10 presents details of dynamic query threshold calculation using BKMC Ranks. Results of using different DBs vs. SI are depicted in


cellular

<table>
<thead>
<tr>
<th>EBD</th>
<th>Active</th>
<th>Animals B</th>
<th>Tourisms A</th>
</tr>
</thead>
<tbody>
<tr>
<td>O12</td>
<td>0.87</td>
<td>4.65</td>
<td>0.80</td>
</tr>
<tr>
<td>O50</td>
<td>0.74</td>
<td>4.65</td>
<td>0.70</td>
</tr>
<tr>
<td>O70</td>
<td>0.70</td>
<td>4.65</td>
<td>0.67</td>
</tr>
<tr>
<td>O90</td>
<td>0.69</td>
<td>4.65</td>
<td>0.65</td>
</tr>
</tbody>
</table>

(b) Editing

<table>
<thead>
<tr>
<th>EBD</th>
<th>Active</th>
<th>Russia A</th>
</tr>
</thead>
<tbody>
<tr>
<td>RussiaA10</td>
<td>0.79</td>
<td>96.02</td>
</tr>
<tr>
<td>RussiaA50</td>
<td>0.63</td>
<td>60.07</td>
</tr>
<tr>
<td>RussiaA70</td>
<td>0.69</td>
<td>24.78</td>
</tr>
<tr>
<td>RussiaA85</td>
<td>0.13</td>
<td>21.43</td>
</tr>
</tbody>
</table>

TABLE 6
Scoring for Synthetic Ontologies (ISICON)

(a) Animals and Russia

<table>
<thead>
<tr>
<th>Animals Rank</th>
<th>Naive</th>
<th>BKMC</th>
<th>Russia Rank</th>
<th>Naive</th>
<th>BKMC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Animals B90</td>
<td>0.78</td>
<td>0.25</td>
<td>Russia A</td>
<td>0.73</td>
<td>0.36</td>
</tr>
<tr>
<td>Animals B50</td>
<td>0.78</td>
<td>0.24</td>
<td>Russia 1</td>
<td>0.7</td>
<td>0.36</td>
</tr>
<tr>
<td>Animals B90</td>
<td>0.77</td>
<td>0.24</td>
<td>Russia D</td>
<td>0.71</td>
<td>0.075</td>
</tr>
<tr>
<td>Animals B12</td>
<td>0.78</td>
<td>0.22</td>
<td>Russia B</td>
<td>0.71</td>
<td>0.075</td>
</tr>
<tr>
<td>Animals B</td>
<td>0.74</td>
<td>0.017</td>
<td>Russia C</td>
<td>0.66</td>
<td>0.069</td>
</tr>
<tr>
<td>Animals A</td>
<td>0.52</td>
<td>0.012</td>
<td>Russia 2</td>
<td>0.49</td>
<td>0.050</td>
</tr>
</tbody>
</table>

(b) Tourisms

<table>
<thead>
<tr>
<th>Tourisms Rank</th>
<th>Naive</th>
<th>BKMC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tourisms A</td>
<td>0.56</td>
<td>0.73</td>
</tr>
<tr>
<td>Tourisms A90</td>
<td>0.5</td>
<td>0.73</td>
</tr>
<tr>
<td>Tourisms B</td>
<td>0.41</td>
<td>0.13</td>
</tr>
</tbody>
</table>

TABLE 7
EBD Ranks of Ontologies (ISICON)

Table 8 presents details of dynamic query threshold calculation using Ontology Naive Ranks, while Table 9 presents details of dynamic query threshold calculation using BKMC Ranks. Results of using different DBs vs. SI are depicted in Figure 9.f. As we can see, DI-Naive ranks have very good results in compare to SI and DI-BKMC ranks. Intuitively, when we use BKMC Clustering to rank ontologies, we ignore real distance between ontologies, that is not desirable for dynamic interval calculation. Therefore, in our further examples, we use naive ranks to calculate query thresholds. Also, we compare our method with WordNet as the baseline [5]. WN expands our query to \{Issue, Academician, Faculty Member\}.

<table>
<thead>
<tr>
<th>Food Rank</th>
<th>Naive</th>
<th>BKMC</th>
<th>Bibliography Rank</th>
<th>Naive</th>
<th>BKMC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wine</td>
<td>0.92</td>
<td>0.55</td>
<td>UMBC</td>
<td>0.414</td>
<td>0.413</td>
</tr>
<tr>
<td>Food 90</td>
<td>0.90</td>
<td>0.55</td>
<td>MIT</td>
<td>0.412</td>
<td>0.413</td>
</tr>
<tr>
<td>Food</td>
<td>0.94</td>
<td>0.30</td>
<td>INRIA</td>
<td>0.52</td>
<td>0.13</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Karlsruhe</td>
<td>0.38</td>
<td>0.095</td>
</tr>
</tbody>
</table>

TABLE 8
EBD Ranks of Ontologies (OAEI)

We also compare the result of BET calculation on synthetic vs. non-synthetic ontologies in Figure 8. In this Figure, X-axis represents different synthetic ontologies and Y-axis indicates the Ratio of BET Cardinality in non-synthetic ontology versus synthetic ontology. In synthetic ontologies which preserve original structure of ontologies such as Animals B synthetic ontologies, BET calculation results in slightly increasing number of expansion terms. For example, Ratio of BET cardinality is 1.05 for Animals B 50 vs. Animals B. While in Russia synthetic ontologies that the original structure of Russia A ontology is not preserved, Ratio of BET decreases drastically. (i.e. Ratio of BET cardinality is 0.1 for Russia A 50 vs. Russia A). We run user Query and ExpandedTerms in blackbook for both non-synthetic and synthetic ontologies in different domains. For non-synthetic (original) ontologies, we analyze Precision, Recall and F-measure for user Query, RET and WN. While, we analyze user Query and BET for synthetic ontologies. Blackbook returns all the individuals related to query keywords in different ontologies. Each individual indicates a corresponding URI reference and presents the related web document to query keywords. In synthetic ontologies, Table 12 presents the Precision, Recall and F-measure for Animals B and Table 13 presents different measures for Russia A synthetic ontologies. F-measure calculation is affected by the number

<table>
<thead>
<tr>
<th>Ontology</th>
<th>BKMC Rank</th>
<th>DI</th>
<th>BET Value</th>
<th>Query Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>INRIA</td>
<td>0.413</td>
<td>0.79</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>UMBC</td>
<td>0.413</td>
<td>0.79</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>MIT</td>
<td>0.13</td>
<td>0.314</td>
<td>16</td>
<td>6</td>
</tr>
<tr>
<td>Karlsruhe</td>
<td>0.095</td>
<td>0.23</td>
<td>27</td>
<td>7</td>
</tr>
</tbody>
</table>

TABLE 9
Threshold for different Bibliographic Ontologies (OAEI)
TABLE 11
Query Expansion on Different Bibliography Ontologies

<table>
<thead>
<tr>
<th>Ontology</th>
<th>BET</th>
<th>NET</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Manual, Article, Unpublished, Manager, Administrative staff</td>
<td>Techreport, Misc</td>
</tr>
<tr>
<td></td>
<td>Fullprofessor, Assistantprofessor, Employee, Technical Staff</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Academicstaff</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Person, Academic Staff, Article, Incollection, Book, Phdthesis</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 8. Ratio of BET changes between Synthetic and non-synthetic Ontologies

of appended individuals (instances) to the original Animals B and Russia A ontologies. For appending synthetic ontologies (Animals B), Precision and Recall are both increased by BET calculation. While for non-appending synthetic ontologies (Russia A), Precision increases and Recall decreases.

TABLE 12
Animals B Synthetic Ontologies F-measure

<table>
<thead>
<tr>
<th>Ontology Animals</th>
<th>B</th>
<th>B12</th>
<th>B50</th>
<th>B70</th>
<th>B90</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precision</td>
<td>0.875</td>
<td>0.875</td>
<td>0.875</td>
<td>0.92</td>
<td>0.93</td>
</tr>
<tr>
<td>Recall</td>
<td>0.67</td>
<td>0.76</td>
<td>0.83</td>
<td>0.71</td>
<td>0.71</td>
</tr>
<tr>
<td>F-measure</td>
<td>0.75</td>
<td>0.81</td>
<td>0.85</td>
<td>0.8</td>
<td>0.8</td>
</tr>
</tbody>
</table>

TABLE 13
Russia A Synthetic Ontologies F-measure

<table>
<thead>
<tr>
<th>Ontology Animals</th>
<th>A</th>
<th>A10</th>
<th>A50</th>
<th>A70</th>
<th>A90</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precision</td>
<td>0.59</td>
<td>0.58</td>
<td>0.67</td>
<td>0.75</td>
<td>0.67</td>
</tr>
<tr>
<td>Recall</td>
<td>0.7</td>
<td>0.7</td>
<td>0.5</td>
<td>0.6</td>
<td>0.54</td>
</tr>
<tr>
<td>F-measure</td>
<td>0.64</td>
<td>0.63</td>
<td>0.57</td>
<td>0.66</td>
<td>0.6</td>
</tr>
</tbody>
</table>

For non-synthetic ontologies, we calculate Precision, Recall and F-measure for all the retrieved individuals as depicted in Table 14. As we can see in Table 14.a, 14.b and 14.c, RET results in the best Precision, Recall and F-measure. Both BET using SI and RET using DI have very close results. For instance, in Bibliography domain, RET results in Precision=1, Recall=0.98 and F-measure=0.97 for both SI and DI. Our experimental results show that both precision and recall measures significantly improve by using RET. Based on the query keywords and structure of ontologies, BET or NET may be more useful in RET calculation. WN does not help much for query expansion in federated architecture.

7 CONCLUSION
In this paper, we have outlined a novel approach for query expansion in a federated architecture. This is achieved in two phases. In phase I, we rank ontologies using an efficient algorithm that finds the common subset of entities (CSE) on the graph of ontologies. CSE Algorithm uses name and structural similarity to determine commonality of entities in a pair of ontologies. Then, we propose EBD as a distance measurement between ontologies which identifies similarity of each pair of ontologies. Furthermore, we illustrate how to use the similarity between pairs of ontologies to rank them by naive and bisecting k-medoid clustering algorithm.

In phase II, first we determine central entity and BET in each individual ontology. Second, we use an ontology alignment algorithm to determine NET based on the similar entities between each pair of ontologies. Third, all expansion terms are weighted using the weight of semantic path, betweenness and the structure of classes in ontologies. Fourth, we find the robust expansion terms using heuristics. Ontology ranks are used to find dynamic threshold for robust query expansion. This helps to choose the k-top related terms across the federated architecture. Robust expanded terms are considered as entities in ontologies and we retrieve all the relevant individuals along with related web documents for them. Finally, we discussed the results of a series of query expansion experiments. Future efforts will focus on comparison between the text-based query expansion methods and ontology-driven query expansion. We will analyze effectiveness of text-based vs. ontology-driven query expansion.
Fig. 9. Different Domain Ontologies and F-Measure

REFERENCES


[8] Blackbook, a graph analytic platform for semantic web data, {http://ebiquity.umbc.edu/blogger/2009/10/03/blackbook-a-graph-analytic-platform-for-semantic-web-data/}

[9] Information Interpretation and Integration Conference, {http://www.atlmlco.com/projects/ontology/isscon.html}


