Assignment 2

Name : Name : Smitesh Nitin Patil

Student ID: Student ID : 22223696

Question 1: Given a query that a user submits to an IR system and the top N documents that are returned as relevant by the system, devise an approach (high level algorithmic steps will suffice)

• To suggest query terms to add to the query. Typically, we wish to give a large range of suggestions to the users capturing potential intended query needs, i.e., high diversity of terms that may capture the intended query context/content. Discuss the efficiency of your approach. (10 marks)

Answer :

Problem Statement:

A search query that user inputs could have many different interpretations based on context of that query. This is evident when we try to frame ambiguous queries. An ornithologist who wants to study about cranes a family of birds may formulate a query for an information retrieval system correctly, but if the system is not designed to handle ambiguous documents with high diversity it may return results about mechanical machine crane.

Consequently, we want to develop a robust IR system that can handle such queries and return documents with high diversity for convenience of the user.

Query Expansion Terminology:

*Search result diversification (SRD)* is a technique that aims to diversify the top N documents retrieve by a user query to any IR system. It deals with re-ranking results such that many aspects of the queries are described in the top documents returned. Search result diversification technique gives us diverse results for a query, but we could instead expand the query itself and make a user query unambiguous to give us better result documents. *Document Query Expansion (DQE)* is a task of identifying a set of terms to add to a query that can be used filter a diverse set of documents.

Problem Definition:

Given any set of documents D and a query Q from the user U, the solution requires us to return a list of suggestions S which are highly diverse but related to the initial query. Suggestions S can be appended to the query Q that can later be used on IR system to retrieve documents.

Select, Link and Rank Algorithm:

This method was proposed in the paper cited, we have defined the steps in this section and suggested improvements later.

Select Phase :

Initially we retrieve terms in Document corpus D to the query Q, denoted as . From those documents we retrieve terms t based on a metric called Bo1, an informativeness measure that quantifies randomness.

Where, denotes frequency of term t in document collection . The top Bo1 words found in the paper for the word ‘jaguar’ where ‘Panthera’, ‘cars’, ‘racing’, ‘Jacksonville’, ‘cake’, ‘tyres. The Bo1 words are passed on as candidates of query expansion as

Link Phase:

In the link phase, we link terms in to Wikipedia articles for extra context. Since, we are working towards query expansion we are going to append Cand(Q,D) term to the query Q. Entity linking methods are used for entity description in Wikipedia . In the end we get relatedness score r(term, entity) which will later be used to rank terms for expansion. For example, ‘Panthera’ got linked to wiki pages like ‘Jaguar’, ‘Panthera’ and term cars with the query jaguar got linked to wiki articles like ‘Jaguar E-type’ and ‘Jaguar Cars’.

Ranking Phase: Our motive is to diversify our set of expansion terms. We first create a graph linking entities with their terms selected as candidates for query expansion. Then we apply *Vertex-reinforced random walk (VRRW)* , VRRW is an algorithm that randomly visits vertices in a graph in a continuously changing environment. When the random walk proceeds, the vertices with high visit counts get even more visits. Thus, the nodes around a highly visited vertex get low counts. After a certain number of iterations, we have a group of nodes with high visited counts that are diverse as the nodes arounds highly visited nodes are not in the sample

Improvement:

We suggest during the linking phase, the algorithm should take entities input from query logs along with the Wikipedia. Assuming that the query logs stores queries along with the user the relatedness of the previous user queries with the current one could be high. When the query along with the terms obtained from select phase could be used to retrieve entities in the query logs to suggest terms that could be later ranked in Ranking phase.

High Level Algorithm Proposed with the improvement:

In : User U, Query Q, Documents D

Output: Query with Set of Expansions E

Select Phase

1. Retrieve terms for query expansion Cand(Q,D) from the document corpus D using the metric Bo1

Link Phase

2. Link each term t in Cand(Q, D) to k Wikipedia articles

3. Link each term t in Cand(Q, D) to j query logs

4. Combine set of terms from query logs and Wikipedia entities.

5. 4. Let relatedness score be r(t, e) using the linking methods.

Rank Phase

5. Construct G(Q), graph of linked entities and neighbours

7. Perform VRRW on G(Q), entity scores initialized

8. Collect the top-scored entities based on VRRW scores as E

9. Construct E, a diversified term ranking using entity scores and term-entity relatedness

Efficiency:

Efficiency of the proposed algorithm would be lower than that of the Select, Link, Ranking algorithm as we have included query logs as a new paradigm for query expansion. Depending on the performance, we can change the value of the number of terms matched in Select phase, the number of Wikipedia articles selected per term, the number of query logs selected in linking phase.

2. Consider the following scenario: a company search engine is employed to allow people to search a large repository. All queries submitted to the system are recorded. A record that contains the id of the user and the terms in the query is stored. The order of the terms is not stored and neither is any timestamp. Each entry in this record is effectively an id and a set of terms. Any duplicate terms in a query are ignored.

The designers of the search engine, decide to use this information to develop an approach to make query term suggestions for users, i.e., at run time once a user an entered their query terms, the system will suggest potential extra terms to add to the query.

Given the data available, outline an approach that could be adopted to generate these suggested terms. A brief outline is sufficient that captures the main ideas in your approach. Identify the advantages and disadvantages of your approach (briefly). (10 marks)

Answer:

Data stored in any company’s repository would be domain specific. For example, a law firm would have a repository of law cases which would have high similarity among documents of the repository. Along with such a repository we have previous user queries in format Q(userid, set of terms T).

Approach for suggesting extra terms to add to the query on runtime:

A two-fold approach could be used to suggest extra terms for query.

Firstly, as we are storing previous queries along with the user id’s, we can use the term sets from the previous user queries . We can filter out the previous user queries by using metrics like cosine similarity or distance-measuring metrics like Euclidean distance If we have weighted matrices of the user queries.

Secondly, we can extract suggestion terms from the repository as the corpus is domain specific. Highly correlated query terms could be sorted out from the document corpus using Pearson correlation index or by using bayes rule for a probabilistic value. A term as a suggestion given some other term exist in query i.e., conditional probability.

Advantages:

1. Previous user queries are taking into account for suggesting terms, many times while using a search engine we tend to use previous searched terms for continuing our search i.e., we build on previous queries and understand the data returned to formulate a new query.
2. User preferences are considered by using user query logs.
3. Searching for suggestion terms in query logs or local repository is computational inexpensive compared to searching for suggestion terms on any remote web repositories

Disadvantages:

1. If the size of repository is large, the efficiency of search for suggestion terms for query expansion would be computationally expensive, in some cases you would have to stick to the query logs as it would take a long time to process suggestions terms from the repository.
2. The diversity of suggestion terms doesn’t improve using this method as we are constraint by the data available in the repository. Taking the example of a repository of a law firm, if the law firm specializes in dealing with financial sector clients it would be difficult to find suggestions for civil cases.