Original Research Paper

Review on Information Retrieval for Desktop Search Engine

Jahanvi. P¹, Jammula Nikhil¹, Arindam Chowdhury¹, S. A. Karthik¹

¹ Dayananda Sagar Academy of Technology and Management, Bangalore, Karnataka, India.

Abstract: Search is an important aspect of information management often taken for granted. Domain specific repositories are growing in both size and numbers calling for efficient search and retrieval of documents. This paper explores the possible techniques and necessary system components for a search engine charting several iterative optimizations over the last few years. This paper focuses on NLP models while retaining basic principles from other methods that assist in information search.

Keywords: Information Retrieval, Query Understanding, Feature Extraction, Entity Recognition, Similarity Measures.
1. Introduction
The Data generation has increased exponentially with time as computing has grown ubiquitous. The internet is a great facilitator when it comes to knowledge discovery and dissemination. This property of the internet stems from extremely efficient data cataloguing, storage, and retrieval from large scale real-time distributed systems. Initially, rudimentary information retrieval systems constituted a simple query to access a particular document from a local database. Over time, these databases have grown to span across networks and storage mediums, and these gave rise to search algorithms designed for user convenience. Accessing documents from a database through a query mentioning document ID or title proved to be difficult when one wanted to retrieve large numbers of documents. It was also an issue to actually give correct values for the document details such as title, author, keywords, domain, and more.

Later this led to the rise of Information Retrieval based on ontology where documents were classified with respect to some entities they contain. It used inverted indexing of the classes they belong to and the keywords they contain [1].

However, information retrieval improved to the levels where users need not necessarily know the keywords to get desired documents. It was possible due to efforts to understand the intent behind the user’s query and the concept the document entails.

2. Research Methodology
In this paper, the basic components that are imperative to a modern-day search engine are: Query Understanding, Feature Extraction, Similarity Measure and Scoring/Ranking, Information Retrieval.

2.1. Feature Extraction – Entity
Entity Recognition seeks to locate and classify the entity mentions of unstructured text into predefined categories.

<table>
<thead>
<tr>
<th>QUERY</th>
<th>NON-INSTANCE</th>
<th>TYPE</th>
<th>INSTANCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Watch Harry Potter</td>
<td>watch</td>
<td>verb</td>
<td>harry potter</td>
</tr>
<tr>
<td>most dangerous python in the world</td>
<td>dangerous</td>
<td>adjective</td>
<td>python</td>
</tr>
<tr>
<td>population of china</td>
<td>population</td>
<td>attribute</td>
<td>china</td>
</tr>
</tbody>
</table>

Table 1. Example of Instance Terms in Queries Which Are Mapped to ProBase [2]

Extracting entities is a preprocessing technique either based on linguistic rules or n-grams word or phrase extraction [2], [3]. Algorithms based on linguistic rules to tag parts of speech are popular for feature extraction. Their rules are manually written and updated with regards to standards of English Grammar.

Entities and their associative categories, either manually tagged [2], [4] or classified by an algorithm, become important features that influence retrieval models. The possibility of including the synonyms, antonyms, acronyms, and more with help of word embeddings enrich the extracted features to cover the meaning behind the context and help the model infer meaning behind the query.

Many approaches to word-category disambiguation have been developed that give the most optimal results in context understanding. Ontology-based entity classification [4], [5] is the most primitive yet classical attempt to entity disambiguation where the documents or their paragraphs can be mapped to a particular category to enrich the meaning behind the group of entities. It is still the preferred technique for information retrieval in closed domain as there are clearly defined boundaries and categories to map the context to their categories. Knowledge Bases provide more varieties of categories to enrich extracted features in a non-conventional manner. Knowledge Base such as Wikipedia [5] is a store for
structured data where the entities and their relationships can be mined. The facts are represented as attribute: fact pairs that gives flexibility in storing various types of relationships. This opens the door the inference in the highest levels of abstraction with regards to navigational queries. On the opposite end of this stream is ProBase which is a Knowledge base of isA relationships between entities that can be mined to build a concept graph in levels of abstractions [6] or a semantic network [7]. Mapping extracted entities of the text to ProBase results in a probabilistic model that classifies the text according to the probability based on Bayes theorem. Bayes theorem describes the probability of a term belonging to the particular concept given the prior information about its isA relationship occurrences.

Word Embedding is an alternative approach to entity disambiguation which uses a vector space model to retrieve semantically similar terms to the entities [1], [6]. It is quite simple and provides an opportunity to involve synonyms and acronyms of the extracted entities of the text. VerbNet [7] and WordNet [1, 6] are framework that provide reliable similar semantic representations to the entities and the possible categories they belong to.

**Figure 1. Universal Concept Graph of Agrawal et.al [7]**

### 2.2. Query Understanding

Query Understanding seem to have many approaches and perspectives which rather makes one be dispirited to get to the bottom of it.

Understanding the intent behind a user’s query regardless of the technique is query understanding. Larger part of Query Understanding seems to be POS tagging and segmentation [8]. This works for all ontology based, graph based and vector space-based models of Information retrieval [8]. If one wants to avoid the issue of not retrieving any document due to absence of a key word of query in the document it is better to introduce some word embeddings as part of query expansion[2] or sub-query generation[7] that later hit these queries against the search engine and gets the results.

It is also imperative to carefully choose a representation model for the knowledge in the documents. One can choose Index [8], Concept Graphs [7], Semantic Networks [9] or Vector Space Models.

Indexing is the method creating a list of keywords for every document. It is a basic File Structures method of Information Retrieval.

Inverted Indexing is the method of Indexing that evaluates whether a keyword is present in the document or not by creating a list of documents containing that particular keyword. This is the primitive method to provide search functionality [4, 8].

Later properties of graphs seem to be a suitable knowledge representation for the documents. The nodes contain a term or concept while the edges represent the relationship between the nodes in the graph. If the nodes and the edges have semantic relationship then the graph becomes a semantic network [3, 9]. If the terms in the nodes have a lexical relationship then the graph becomes a concept graph [7].

However, it is still not possible to complete the process of understanding user intent if one doesn’t consider the synonyms or alternatives for a concept. Kuzi et.al [2] specifically says to enrich the entity values of a query with alternative synonyms to help in retrieval.
2.3. Similarity Measure and Scoring/Ranking

One of the simplest formulae for scoring is mentioned in [8]. Since it is based on ontology, list of keywords is associated to the class and are compared to the terms in the user’s query. It uses Bag of Words approach to classify the query according to user’s Intent.

Agarwal et.al [7] presents that search engine performs better in low dimensions hence it is necessary to deal with only few concepts of user’s query at a time. User’s query traverses the concept graph built and then sub queries are hit against the search engine. The results retrieved are aggregated and relevance scores are computed. The highly relevant docs are retained and retrieved a result to the user’s query.

Gelbukh et.al [10] to compute max- common sub graph, to compare the similarity between graphs i.e. the document and the query. Vector mapping between the two graphs ensure to help score the max-common sub graph. Scoring of max-common sub graph is similar to vector scoring.

\[
P_{qd} = \frac{P(R_{qd}^>) + P(R_{qd}^<)}{2}
\]

(1)

Gelbukh et.al [10] also deals with ambiguous equal ranking by using the above equation where P is the precision,

\[
R_{qd}^> \text{ is the set of documents scored higher } d
\]

\[
R_{qd}^< \text{ is the set of documents scored higher or equally as } d
\]

2.4. Information Retrieval

In Agrawal’s and et.al paper [7] the query and document representations need not be the same. The documents are processed to make a universal concept graph and the combination of the core concepts and directly fed to the search engine.

Buttler et.al [5] presents a pseudo – feedback relevance feedback that helps to enrich the search functionality. A set o learned latent topics that are relevant to the query are provided as suggestion and if user chooses them, the query is augmented. Even Kuzi et.al [2] and Ganguly et.al [6] shows its results by using pseudo – feedback relevance model.

Techniques of Inference accommodate uncertainty. It is possible to delve deeper into the relationships with help of proposition logic while traversing the semantic network and retrieve the documents pertaining to the particular node [3].

Seok et.al [11] uses a CRF model for NER task as CRF’s represent a probabilistic framework for labelling and segmenting sequential data.

\[
P(y|x) = \frac{1}{Z(x)} \prod_{i=1}^{n} \exp \sum_{j} \lambda_j f_j(y_{i-1}, y_i, x, i)
\]

(2)

Input: is a sequence of words x1 to xn; tags y1 to yn are allocated.

\[
f_j(y_{i-1}, y_i, x, i)
\]

(3)

Equation 3 is a feature function that takes input: sentence x, position i, label y of current word and label of previous word.

\[
Z(x) = \sum_y \exp( \sum_{i=1}^{n} \sum_j \lambda_j f_j(y_{i-1}, y_i, x, i) )
\]

(4)

Equation 4 is a normalization vector that ‘sums’ the scores.
3. Result Comparision
Agrawal et.al [7] provides a CGSimilarity algorithm to construct the concept subgraphs against the use of random walks over the concept graph to end with a semantically similar concept. While both techniques formed queries of same concept phrases, CGSimilarity is much preferred.

![Figure 2. Comparative Performance between CGSimilarity and Random](image)

Seok et.al [11] compares between vector space model as baseline with and without embeddings and proves that embedding is a feature that has impact.

**Table 2. Comparison between Types of Embeddings in Seok’s et.al [11]**

<table>
<thead>
<tr>
<th></th>
<th>Test A</th>
<th>Test B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>84.12%</td>
<td>77.06%</td>
</tr>
<tr>
<td>Baseline + GloVe</td>
<td>85.18%</td>
<td>79.48%</td>
</tr>
<tr>
<td>Baseline + Word2Vec</td>
<td>85.89%</td>
<td>80.72%</td>
</tr>
<tr>
<td>Baseline + CCA</td>
<td><strong>85.96%</strong></td>
<td>80.68%</td>
</tr>
</tbody>
</table>

Prasath et.al [8] shows there is a drop in precision score when 10 topics are considered and also proves in its experiment that entity recognition and resolution performs better than pattern matching in feature extraction and is shown in Figure 3.

![Figure 3. Query Classification Topic Score](image)

4. Conclusion
This paper concludes that knowledge representation of a document and query has a high impact on user’s query intent understanding. Building index is efficient when one deals with a collection of
documents where they are classified based on ontology i.e. the classes are pre-defined. If one wants to deal with information of open domain, representing data in terms of entities and concepts is recommended. Query Understanding can be tackled if one considers lexical and semantic relationships between the terms [9]. However, it is rather exhaustive to maintain a universal concept graph or semantic network for it does not give space for document to be edited or deleted. This is an issue that remains unexplored due to assuming that all documents are static.

Vector space models are not recommended for they do not consider the lexical and semantic relationships between the vectors in a text which puts the model at a disadvantage in understanding intent behind the query.

Table 3. Comparison of Techniques and Their Impact Retrieval

<table>
<thead>
<tr>
<th>Paper</th>
<th>Chapter</th>
<th>Technique</th>
<th>Baseline Measure/Score</th>
<th>Technique Measure/Score</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>[8]</td>
<td>Information Retrieval</td>
<td>Ontology based Topic classification of the recognized entities</td>
<td>Keywords: 0.46</td>
<td>Topic indexed based retrieval: 0.55</td>
<td>14.145% improvement</td>
</tr>
<tr>
<td>[7]</td>
<td>Similarity Measure</td>
<td>Concept Graph Based query generation</td>
<td>Random: 0.1</td>
<td>CGSimilarity: 0.8</td>
<td>Using CGSimilarity is much efficient in doc retrieval</td>
</tr>
<tr>
<td>[9]</td>
<td>Feature Extraction – Concept Clusters</td>
<td>Semantic Network</td>
<td>IJCAI111 – Bayesian Analysis: 0.84</td>
<td>Random Walk: 0.87</td>
<td>Seems that Random walk covers much more of User’s Intent Under this Technique</td>
</tr>
<tr>
<td>[2]</td>
<td>Query Understanding</td>
<td>GOV2 Model of Word Embeddings</td>
<td>RI score of RM3: 0.392</td>
<td>RI score after integration in RM3: 0.432</td>
<td>Including Synonyms is exhaustive</td>
</tr>
<tr>
<td>[6]</td>
<td>Feature Extraction-Word Embedding</td>
<td>Generalized Language Model</td>
<td>Recall score of LDA: 0.58</td>
<td>Recall score of GLM: 0.62</td>
<td>Vector word embedding is exhaustive</td>
</tr>
</tbody>
</table>

References


