2-2017

ES-ESA: An Information Retrieval Prototype Using Explicit Semantic Analysis and Elasticsearch

Brian D. Sloan
The Graduate Center, City University of New York

ES-ESA: AN INFORMATION RETRIEVAL PROTOTYPE USING EXPLICIT SEMANTIC ANALYSIS AND ELASTICSEARCH

by

BRIAN SLOAN

A master's thesis submitted to the Graduate Faculty in Linguistics in partial fulfillment of the requirements for the degree of Master of Arts, The City University of New York

2017

ES-ESA:
An Information Retrieval Prototype Using Explicit Semantic Analysis and Elasticsearch

by

Brian Sloan

This manuscript has been read and accepted for the Graduate Faculty in Linguistics in satisfaction of the dissertation requirement for the degree of Master of Arts.

________________	________________________________
Date	Dr. William Sakas
	Thesis Advisor
________________	________________________________
Date	Dr. Gita Martohardjono
	Executive Officer

THE CITY UNIVERSITY OF NEW YORK

ABSTRACT

ES-ESA:

An Information Retrieval Prototype Using Explicit Semantic Analysis and Elasticsearch

by

Brian Sloan

Advisor: Dr. William Sakas

Many modern information retrieval systems work by using keyword search to locate documents in an inverted index by matching those documents based on terms in a user's query. While highly effective for many use-cases, one notable drawback to simple keyword-based searching is that the *contextual knowledge* surrounding the user's underlying information need may be lost, particularly if the user's query terms are ambiguous or have multiple meanings. Research in the field of semantic search aims to make progress towards resolving this. One methodology in particular, *explicit semantic analysis*, works by modeling a document not only as a set of the unique terms it contains but also as a set of *concepts* which describe it; these concepts are derived from some authoritative or curated source and assigned to each document in a collection. This paper presents a prototype information retrieval system called "ES-ESA" which borrows from the principles of explicit semantic analysis and implements them using the Elasticsearch framework. The ES-ESA system is qualitatively evaluated using a corpus of academic research abstracts.

TABLE OF CONTENTS

Introduction **1**

Background and Related Work **6**

ES-ESA **9**

Overview **9**

The Test Document Corpus **9**

System Components **10**

Document Indexing and Semantic Interpretation **12**

Search API **13**

The Test Corpus and Sample IR Tasks **14**

TF-IDF (Baseline) 15

Boosting 16

Query Expansion 19

Boosted Query Expansion 21

Evaluation and Further Work **22**

Works Cited **24**

LIST OF ILLUSTRATIONS AND TABLES

Fig. 1: ES-ESA architecture……………………………………………………………………………..11

Fig. 2: Search API endpoint and query parameter format……………………………………...14

Introduction

In computing, an Information Retrieval (IR) system facilitates the task of finding information relevant to satisfy a user's particular information need. Most modern IR systems, particularly in the area of web search, utilize keyword search to locate documents containing terms that match terms in the user's query in an inverted index. Most readers will likely recognize keyword search as a familiar paradigm for carrying out common IR tasks, and with good reason: it tends to work well in many contexts for a variety of common search and retrieval tasks. However, there are some scenarios where the weaknesses of keyword term matching for information retrieval become apparent.

Two commonly known (and reasonably well-understood) problems in text-based IR are 1.) how to handle so-called "stop words" in a query, and 2.) how to quantify the relevance of documents matching a multi-term query. Stop words are words that are typically ignored for the purpose of search and relevance ranking.[1] Examples of stop words in English include "the", "and", "in", "a", "an" and so on. One naïve (but still effective) method of dealing with the problem of stop words is to simply ignore them at query time and at indexing time. This method solves for the most common terms, such as the articles listed above, but it does not gracefully handle terms that appear commonly but do not fit easily into the same group of categorically irrelevant articles. Solving this problem – what is essentially the problem of search relevance – is more challenging.

A number of clever methods for dealing with the problem of search relevance have been proposed; some have proven very successful and have been widely adopted. Notably,

[1] Hans Peter Luhn has been credited with coining the concept of stop words, if not the exact phrase "stop word". See Luhn (1966) and Flood (1999).

one such method handles the presence of commonly occurring terms in a corpus by attempting to appropriately de-emphasize them relative to what are likely more important terms; this method uses what is known as the term frequency / inverse document frequency statistic, or TD-IDF for short (see Jurafsky and Martin 2009, p. 771). General-purpose search engines commonly handle queries using TF-IDF to assign a lower weight to documents matching on terms that occur with high frequency in the corpus. In this way, query terms are assigned a weight according to how this statistic determines their importance relative to the corpus.

Another potentially challenging problem with solely using term frequency statistics for relevance ranking are the related problems of *polysemy* and *synonymy* (Egozi, Markovitch, and Gabrilovich 2011, p. 2). Polysemy describes a word or phrase that, although spelled exactly the same way in a given language, can have multiple (and sometimes disparate) meanings in different semantic contexts. The problems arising from a polysemic search query in a simple keyword-based IR system are obvious: if, for example, the user of a search engine enters the query string "meat packing" intending to find information about the upscale district of Manhattan but instead returns a number of articles about the act of packing and distributing meat, the search has failed – no relevant results are returned with respect to the user's actual information need. Synonymy is similar in that it describes the problem that arises when a user's search query and documents relevant to that query use different terms to describe the same concept. The semantic context of both the user's query and the search space of documents does not translate to IR systems based solely on term frequency counting – because the user's query does not contain the exact same terms used in the relevant documents, the query does not return them.

A computer has a clear advantage over a human when it comes to quickly matching documents in a large text corpus that are relevant to a search query; similarly, sorting the results of the same search according to some statistical relevance technique such as TF-IDF is not highly problematic for a computer. However, the wealth of world knowledge and context that human experience brings to a search query, particularly an ambiguous or polysemic one, is not an easy problem for a computer to solve. The burgeoning field of semantic search deals with the problem of how to model and represent this world knowledge in a way that is computationally feasible for a computer to work with.

There are a number of semantic search methods proposed in the literature and too many to fully review here (Dong, Hussain, and Chang, 2008, provide a review of many of these methods). Notably, latent semantic analysis, or LSA, is one of these semantic search methodologies. LSA uses statistical methods to compute the similarity of groups of terms within a text corpus (see Landauer and Dumais, 1997). These conceptual groupings are used as the basis for ultimately determining contextual similarity of texts. LSA has been shown in some applications to correlate better than random with human judgments of the same comparisons (for one example, see Cohen, Blatter, and Patel, 2005).

Explicit semantic analysis, or ESA, differs from LSA in that the concepts that documents are modeled from are pre-defined instead of computed. In ESA, a human-curated knowledgebase is used, typically Wikipedia, which becomes the model for representing words as well as the documents they appear in. In this model, words are represented as vectors of concepts, with each concept being assigned a weight to indicate the “strength of association” of that particular word to that concept. According to Gabrilovich and Markovitch’s proposed ESA algorithm (2007, p. 1607), the weight assigned for each concept is the TF-IDF score of that word within the (typically Wikipedia) concept corpus. Texts are processed by a semantic interpretation module which translates the input text into a concept vector in the space of all Wikipedia concepts; short strings and entire documents can be represented simply by summing the scores per concept for all terms in the text. In this way, it is possible to model a document so that it is no longer strictly a term vector, but it also contains some level of semantic context encoded as a weighted list of concepts. The ESA algorithm is described more formally in the next section, but the intuition behind this algorithm is that given a pre-existing, human-curated corpus of concept data such as Wikipedia, and assuming the concepts themselves are orthogonal with respect to one another, it is possible to harness the semantic knowledge encoded in this corpus and use it to apply deep contextual understanding to texts. It can then be used in a number of IR applications, such as to quantify document similarity or to enable semantic search.

In this paper, I present a prototypical but fully functional IR system designed to allow software developers a facility for experimenting with and exploring the possibilities of ESA in semantic search applications. The prototype is based on the principles of explicit semantic analysis and is implemented as a simple HTTP web service. Low-level search and indexing functionality is provided by the Elasticsearch framework[2] and HTTP request handling is provided by the Flask framework.[3] I’ve called the prototype application itself Elasticsearch-Explicit Semantic Analysis, or ES-ESA for short. The purpose of this project is

[2] https://www.elastic.co/guide/en/elasticsearch/reference/2.4/index.html
[3] http://flask.pocoo.org/

not necessarily to show that ESA can be used as a fool-proof augmenter for any given IR task in any given corpus; rather, it is simply to provide a frame of reference for comparing how various ESA-derived techniques may (or may not) improve certain search tasks in a given corpus. The corpus of text used for demonstration purposes in this paper is a collection of just over 300,000 academic abstracts in the field of economics, described in more detail in the next section. The corpus of concepts used is Wikipedia.[4] Although these corpora are used in the demonstration queries and related discussion in this work, it is my hope that by making this project freely available and open-source, other researchers and software developers may build on it and experiment with different concept sources and text corpora.[5]

At a high level, ES-ESA provides a web-based API over an inverted index of documents that allows a user to submit a multi-term search query in the form of a simple HTTP GET request. The user's query is then executed by the service in one of four different ways, depending on the parameters of the user's request:

1. as a TF-IDF search using only the terms in the query (this is the "unenhanced" baseline search);
2. as an expanded search, with the original query expanded to include not only the terms of the query itself but also the ESA concepts derived from the query string;
3. as a boosted search, using query-time relevance boosting based on the ESA concepts (and their associated weights) that the query text has in common with any documents returned in the baseline TF-IDF result set; or

[4] Specifically, the full English-language Wikipedia dump was used, as available on WikiMedia's official site in October of 2016: https://dumps.wikimedia.org/enwiki/latest/
[5] The source code for ES-ESA is available on GitHub: https://github.com/bsloan/es-esa

4. as an expanded *and* boosted search, using both the ESA-derived boosting and query expansion techniques to augment the baseline TF-IDF keyword search results.

This remainder of this paper is structured as follows: the next section presents a brief background and review of published literature on ESA as it relates to semantic search; the following section presents the ES-ESA system as it is currently implemented and provides a brief qualitative analysis of the system's performance on a handful of test tasks, using a test corpus of academic abstracts; and the final section presents suggestions on the possible direction of future work on the ES-ESA system based on the qualitative analysis presented.

Background and Related Work

Gabrilovich and Markovitch initially introduced explicit semantic analysis as a novel technique for improving text categorization (2006). In 2007, the same authors used ESA to improve on the then state-of-the-art for automatic document similarity computation, showing that ESA could produce a 72% correlation with human judgments when determining similarity between texts and a 75% correlation with human judgments when determining similarity between words. In both of these articles, Gabrilovich and Markovitch used Wikipedia as the corpus of concepts, on the assumption that it is the closest approximation to the "breadth of knowledge available to humans" (Gabrilovich and Markovitch 2006, p. 1301), while at the same time being available in a format that is readily applicable to these computational tasks. Interestingly, ESA has also been shown to be effective on document similarity decisions using vectors consisting of document metadata (such as author or publication name) and notably *not* using Wikipedia or any external source of concept information other than this metadata (Martín, Schockaert, Cornelis, and Naessens, 2013).

Świeboda, Krasuski, and Janusz (2014) showed that iterative improvement on the results of ESA is possible using human feedback and machine learning, rather than leaving the results of the semantic interpreter as static. The authors demonstrate their iteratively-improving ESA using a corpus of scientific research articles and PubMed subject headings as the source of concepts. ESA (and various extensions of it) have been applied to question answering (Figueroa and Neumann, 2006) and sentiment analysis in microblog sites (Montejo-Ráez, Díaz-Galiano, Martínez-Santiago, and Ureña-López, 2014), as well as in other corpora. Interestingly, ESA has also been shown to have significant potential for multilingual IR applications. Potthast, Stein, and Anderka (2008) and Sorg and Cimiano (2012) use ESA methods to demonstrate multilingual search systems that are capable of returning relevance-ranked results in one language when the query itself is entered in another language.

Anderka and Stein (2009) presented an analysis of ESA in which they challenged the assertion that an organized, curated, and encyclopedic knowledge source such as Wikipedia (which was initially chosen for ESA under the assumption that the orthogonality of concepts is crucial) was not necessary; they suggested, rather, that similar results are achievable by using news articles as the source of concepts, calling into question some of the assumptions underlying ESA, particularly that the concepts must be orthogonal with respect to one another. Gottron, Anderka, and Stein (2011) provided further theoretical analysis of ESA, taking a more systematic approach to explaining *why* applications of ESA behave in the way they have been observed to behave.

Egozi, Gabrilovich and Markovitch (2011) describe an ESA-augmented IR implementation. The authors show that using ESA for query expansion can have a positive effect on search recall when the query terms do not exactly match the terms used in the corpus for a semantically-equivalent topic. However, they also demonstrate that such a system is susceptible to "query drift" – returning irrelevant documents because of "excessive text-based query expansion" (2011, p. 12). The authors go on to describe and evaluate methods for tuning the concepts assigned to the query terms in order to minimize excessive expansion by selecting only the most relevant "features" (e.g., concepts) of the query at query-time. They conclude by showing that this fused approach, in which the query's concepts are pre-calibrated to the corpus using a bag-of-words search before the ESA-based search is executed, provides the best results, significantly better than the BOW baseline search as well as the ESA-only approaches (p. 26-27).

In their research, Gabrilovich and Markovitch demonstrated some progress towards solving the problems of synonymy and polysemy by using a mixed approach to ESA-based information retrieval, combining TF-IDF and a form of ESA-based query expansion. The work of these authors is the main inspiration for the development of the ES-ESA project. It is not my intention here to propose an improved methodology for tuning ESA-based query expansion, but instead to provide the basis for an open-source system where the parameters to similar IR systems can be built on, expanded, tuned, and generally made more easily testable, in hopes that this provides an open-source basis for such systems to be improved upon and implemented in practice.

ES-ESA

Overview

In its current form, ES-ESA can be thought of as a development environment for testing and tuning ESA on different specialized corpora; this section presents an exercise of the system on one such corpus. The ES-ESA API is intended to demonstrate both the potential strengths and the potential shortcomings of using explicit semantic analysis to improve precision and recall in certain search tasks. This section will present a short description of the corpus of documents used for testing and example searches; an architectural overview of ES-ESA; and a series of brief, qualitative examples of searches executed in all of the query formats supported by ES-ESA. These sample queries and summary discussions are based on a corpus of test documents indexed using the ESA algorithm presented by Gabrilovich and Markovitch, using Elasticsearch for lower-level search and indexing functionality. The source of ESA concepts used as the basis for semantic interpretation in all of these examples is derived from a full dump of all English-language Wikipedia articles as of October 2016.

The Test Document Corpus

The collection of documents to serve as the test index for the queries described here consists of an arbitrary collection of slightly over 300,000 academic abstracts downloaded from publically available collections from the *Research Papers in Economics* web site, or RePEc. All documents are available from the RePEc metadata API and were downloaded from this source.[6] RePEc represents "a collaborative effort of hundreds of volunteers in 89 countries to enhance the dissemination of research in economics and related sciences"

[6] The RePEc XML repository is available from this endpoint: http://oai.repec.org/?verb=Identify

(http://repec.org/). RePEc was chosen as the source of articles for a number of reasons: first, all of these documents are academic abstracts, not full text articles. The intuition here is that although the length of each document is relatively short, it is also densely packed with information and relevant keywords. Both of these traits should provide a good starting point for testing ESA. Second, the subject matter of the documents is diverse, despite falling under the general domain of economics. Finally, the data was freely and publicly available.

These RePEc abstracts were selected because they provide a sufficiently representative body of documents for the purposes of this paper; that is, they can approximate the target(s) of varied information needs within a domain. This corpus is not intended to simulate a full web-scale search such as what is provided by Google, Bing, or other major search engines. The documents were downloaded in XML format via the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) endpoint.[7] It should be noted that only an arbitrary subset of abstracts from RePEc was indexed and is discussed here for the purposes of analyzing ES-ESA; the articles were not deliberately selected and are not meant to be a representative sample of all articles in RePEc – their selection should be considered random, with no regard to title, author, or subject matter.

System Components

ES-ESA consists of several components as illustrated in the architecture diagram in Fig. 1. The three main components of the system are the *Search API*, which provides an HTTP web service for querying the documents index; the *Document Indexer*, which indexes the individual input documents of the corpus and semantically interprets them by adding

[7] More information about the OAI-PMH protocol is available here: http://www.openarchives.org/OAI/openarchivesprotocol.html.

Wikipedia concepts and weights; and the *Wikipedia Indexer* (this can also be thought of more generically as the "concept" indexer) which creates the inverted index of concepts that both the Search API and Document Indexer depend on for semantic interpretation. The Document Indexer and the Search API rely on the same two Elasticsearch indexes: one containing the full text of all English-language Wikipedia articles (or, thought of more abstractly, any other corpus of source concepts), and the other containing the corpus of documents to be searched by end-users of the system. The Wikipedia Indexer takes as its input the XML format made available on Wikipedia's public archive (https://dumps.wikimedia.org/enwiki/latest/) and uses this to generate the index.

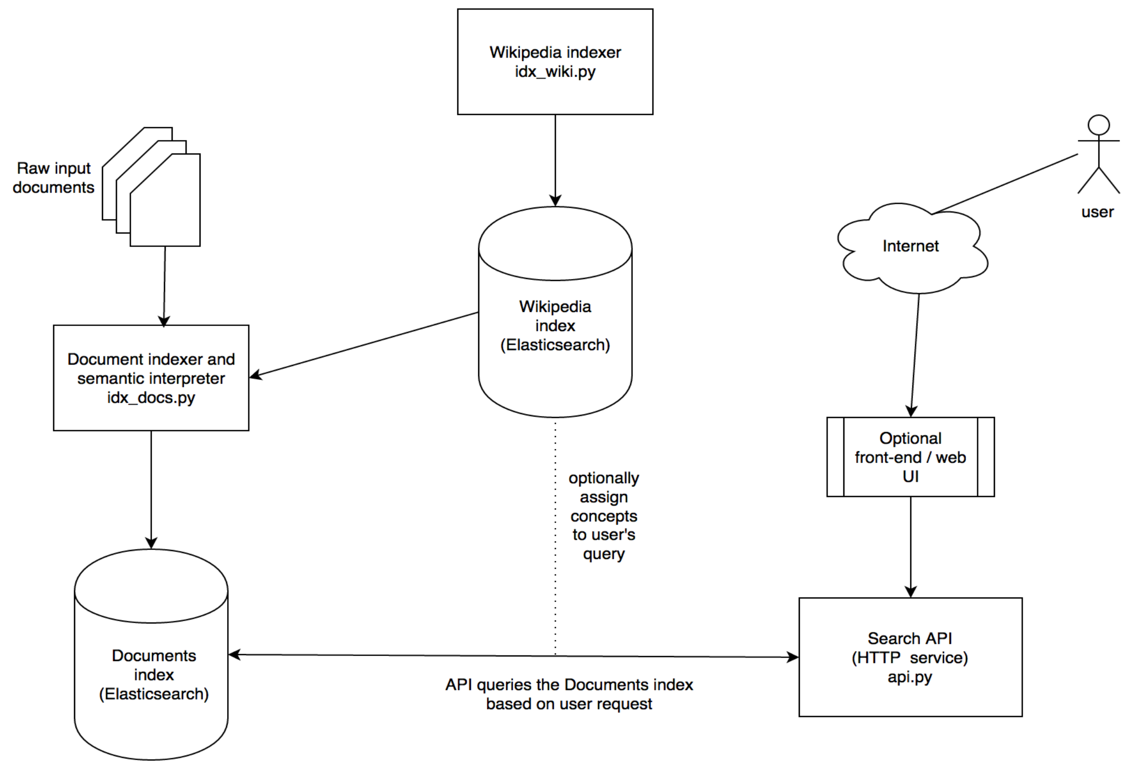
Wikipedia indexer
idx_wiki.py
Raw input documents
Wikipedia index (Elasticsearch)
user
Internet
Document indexer and semantic interpreter
idx_docs.py
Optional front-end / web UI
optionally assign concepts to user's query
Documents index (Elasticsearch)
Search API (HTTP service)
api.py
API queries the Documents index based on user request

Figure 1. ES-ESA architecture.[8]

[8] The "Optional front-end / web UI" is not implemented at this time and is not described in this paper. This component is not necessary to demonstrate the system's functionality.

Document Indexing and Semantic Interpretation

The Wikipedia Indexer uses an implementation of the ESA algorithm as described by Gabrilovich and Markovitch (2007, p. 1607) for assigning concepts to documents. Gabrilovich and Markovitch's (2007) algorithm follows:

> Let T = { w_i } be input text
>
> Let < v_i > be its TF-IDF vector where v_i is the weight of word w_i
>
> Let < k_j > be an inverted index entry for w_i where k_j quantifies the strength of association of word w_i with Wikipedia concept c_j , { c_j in $c_1, \ldots, c_N$ } where N is the total number of Wikipedia concepts
>
> Then, the semantic interpretation vector V for text T is a vector of length N, in which the weight of each concept c_j is defined as: $\sum_{w_i \in T} v_i \cdot k_j$

ES-ESA uses Elasticsearch as the search and indexing backend, which in turn uses the Lucene framework to provide methods for creating and interacting with inverted indexes (http://lucene.apache.org/). These frameworks provide APIs readily capable of executing the methods described in Gabrilovich and Markovitch's ESA concept-generating algorithm, since by default, Elasticsearch and Lucene represent indexed documents using the TF-IDF scheme. Within the ES-ESA prototype, the intuition behind creating this index of Wikipedia articles is that later, any input text, when it is tokenized and executed as a term query against the Wikipedia index, is essentially transformed into a TF-IDF vector corresponding to *v* in the algorithm above. Running the input text as a query in Elasticsearch produces search results ranked by their cosine similarity. In other words, the search hits to this query are an ordered list of each concept (Wikipedia article), which can thus be said to correspond to *V* above: the input text's "interpretation vector" as defined by Gabrilovich and Markovitch (2007).

It follows that the first step in setting up a working implementation of ES-ESA is to create an inverted index of concepts from Wikipedia containing each word *w* above and the “strength of association” metric *k*; the creation of this index implicitly results in an inverted index representing *w* and *k.* Elasticsearch, via Lucene, internally represents each indexed document and query string as a TF-IDF term vector. The similarity between the query string vector and each document vector is computed at the time of the query using cosine similarity.[9] [10]

Search API

The search API provides a simple client/server interface that provides a relatively straightforward request format for formulating test queries over the corpus of indexed documents. As outlined in this section’s overview, queries can be formulated as TF-IDF, ESA-expanded, ESA-boosted, or some combination of TF-IDF and expanded and/or boosted using ESA concepts. Clients of this API are able to define the type of query using URL query parameters, as described in Fig. 2 below:

[9] See the official Elasticsearch documentation on relevance scoring at https://www.elastic.co/guide/en/elasticsearch/guide/current/scoring-theory.html.

[10] The ES-ESA system uses Elasticsearch’s English-language analyzer when creating the Wikipedia index. The English analyzer provides stemming of terms and drops common English stop-words. For more details on the processing carried out by the English analyzer, see https://www.elastic.co/guide/en/elasticsearch/reference/2.4/analysis-lang-analyzer.html#english-analyzer.

Endpoint	Description of search executed
`/search?q=<query>`	"baseline" TF-IDF search using terms from the query string
`/search?q=<query>&expand`	TF-IDF search is run, expanded to include tagged concepts using ESA
`/search?q=<query>&boost`	TF-IDF search is run, with relevance scores updated using ESA concept scores
`/search?q=<query>&boost&expand`	TF-IDF search is run using ESA query expansion and the boosting algorithm

Figure 2. Search API endpoint and query parameter format.

It should be noted that all of the baseline TF-IDF queries are processed by the same English-language analyzer that is used when indexing Wikipedia and the documents themselves. This means the terms are stemmed the same way and the same English stop words are dropped as when the Wikipedia index is created. Also, adding an additional parameter `&debug` to any of the above URLs will execute the query in "debug" mode, exposing the exact relevance scoring formula used by the framework to rank the results.

The Test Corpus and Sample IR Tasks

A simple methodology was followed to test the ES-ESA system. A number of articles within the corpus were selected at random, and various queries were formulated with the intention of mimicking what an end-user might enter into the same system intending to find the same abstract. Several of these test queries are described and qualitatively evaluated here. A more in-depth empirical evaluation of the system is needed, but these examples serve as a starting point, suggesting some of the advantages and disadvantages each query method presents in this particular corpus and as a starting point for suggesting solutions or further analysis of problems.

TF-IDF (Baseline)

The TF-IDF baseline query is executed by Lucene via Elasticsearch, using Lucene's implementation of the algorithm. In short, TF-IDF provides a numeric score encapsulating the "importance" of a particular term relative to a corpus of documents; a document is scored highest for a term if that term appears frequently in that particular document but does not appear frequently throughout the corpus. TF-IDF was used as the baseline for ES-ESA because of its use in many modern text-based search and retrieval systems that use a vector space model for documents (Jurafsky and Martin, 771). As such, it provides a basic point of comparison against the boosting and expansion algorithms based on ESA.

Within the RePEc corpus under test, there are many examples of TF-IDF working well to produce results directly relevant to a query. One such example is a query for the terms *car ownership in Turkey*:

```
/search?q=car%20ownership%20in%20turkey&debug
```

The top five hits in this corpus are at least marginally relevant to the context of the query (studies of car ownership internationally), with the top result being directly relevant (car ownership in Turkey):

1. Title: "Modeling Car Ownership in Turkey Using Fuzzy Regression"
2. Title: "Demographic Determinants of Car Ownership in Japan"
3. Title: "A Stated Preference Study for a Car Ownership Model in the Context of Developing Countries"
4. Title: "Changes in level of household car ownership: the role of life events and spatial context"
5. Title: "Is demand for polluting goods manageable? an econometric study of car ownership and use in Mexico"

On the other hand, one simple query that illustrates the shortcomings of using only TF-IDF term matching within the test RePEc corpus is *minnow schooling*. Here is the URL in the ES-ESA API:

```
/search?q=minnow%20schooling&debug
```

Presumably, the information need underlying such a query would for articles related to the schooling behavior of fish. In this particular corpus, the term *minnow* does not occur at all, so the TF-IDF search returns matches on the term *schooling*. Predictably, this returns a number of irrelevant articles, even as the five most highly ranked:

1. Title: "Fertility shock and schooling"
2. Title: “Schooling and Prenatal Death”
3. Title: “Schooling, Parents, and Country”
4. Title: “Estimating Returns to Schooling When Schooling is Misreported”
5. Title: “Gender Parity and Schooling Choices”

Presumably, if any article in the corpus contained the word *minnow*, or both *minnow* and *schooling*, the results would be much better. But with the term *minnow* absent in the corpus, and with the IR system lacking any contextual knowledge of what the query is actually about, the system fares poorly for this use case.

Boosting

One way that ESA can be applied to improve search relevance is through the application of a query-time boost based on concepts that the query and documents in the TF-IDF search result set have in common. A naïve boosting algorithm on the basis of common concepts is presented below:

```
- Let $V$ = the semantic interpretation vector for the input text (query).

    (This is a collection of concept / weight pairs <$c$, $w$> where $c$ is the concept and $w$
    represents the strength of association of the query text with the concept according
    to the ESA algorithm described previously.)

- Let $R$ be the set of results returned by a TF-IDF term query over the documents index.

- For each document $d$ in $R$:

    - Let $C$ be the semantic interpretation vector of $d$
    - Let $t$ be the TF-IDF score of $d$

    - Let $I$ be the collection of weighted concepts <$c$, $w$> present in both $V$ and $C$

        (This collection $I$ essentially amounts to the concepts that the query and the
        document $d$ have in common.)

    - Let the boosted score for $d$ equal $t$ plus the sum of each weight $w$ in $I$
```

This algorithm is implemented in ES-ESA in the Groovy programming language and is applied as an Elasticsearch script scoring function; the Groovy implementation is available here: https://github.com/bsloan/es-esa/blob/master/script_score.groovy. The intuition behind this boosting algorithm is that if the query and a search hit using the TF-IDF relevance scoring are both tagged with the same semantic concept, then that search hit should receive a relevance boost proportional to both the strength of association between the same concept in the query text and the strength of association of that concept to the document text.

Many test queries were qualitatively evaluated using this boosting technique, and in most of them, the boosting did not make a noticeable difference in relevance quality (positively or negatively). However, there were a few key examples of where this boost did bring more relevant results to the top of the result set whereas the baseline search scored the same articles as less relevant. One such example is the query for the terms *untouchable caste job diversity*.

Query:

```
/search?q=untouchable%20caste%20job%20diversity&debug
```

Run as a TF-IDF baseline search, this returns the following as the top three most relevant search hits:

1. Title: "The Impact of Team Diversity, Task Interdependence, Team Conflict and Team Cooperation on Job Performance: Using Real Estate Brokers as Examples"
2. Title: "Double jeopardy? Caste, affirmative action, and stigma"
3. Title: "Caste at Birth? Redefining Disparity in India"

Contrast these results with the top three search hits for the same query, only with the ESA-based boosting algorithm applied to the result set.

Query:

```
/search?q=untouchable%20caste%20job%20diversity&boost&debug
```

Results:

1. Title: "Is caste destiny? Occupational diversification among Dalits in rural India"
2. Title: "Social Identity and Educational Attainment: The Role of Caste and Religion in Explaining Differences between Children in India"
3. Title: "On Backwardness and Fair Access to Higher Education in India: Some Results from NSS 55th Round Surveys 1999-2000"

The first result of the boosted query set, "Is caste destiny? Occupational diversification among Dalits in rural India" is a relevant hit; at least, it is arguably the most relevant document in the RePEc test corpus. This document ranks seventh in the TF-IDF baseline results, suggesting that the boost had a significant positive impact on relevance in this query. The relevance of the results ranked second and third respectively in both sets of hits is subjective, although it is interesting to note that the hits ranked second and third in the boosted result set do not appear at all in the top ten results of the TF-IDF baseline result set, perhaps being offset by other, less relevant results.

Query Expansion

Another ESA-derived method for augmenting TF-IDF comes in the form of query expansion – that is, incorporating the concepts tagged on the text of the query into the query itself. This presents a number of challenges, not the least of which is to define the best way to appropriately assign weight to each concept and to prune the set of concepts used in the query to provide the most precise results. This requires removing terms that too aggressively expand the query causing irrelevant, "noisy" results to be returned, while at the same time expanding the query to the degree that the concepts included in the expanded query do, in fact, cause the result set to include relevant documents that would otherwise be missed by the same query without expansion.[11] What follows is a brief, qualitative evaluation of several examples of ESA-based query expansion on the RePEc test corpus.

It was challenging to find examples of query expansion that demonstrated higher precision than the baseline TF-IDF search; in the test document corpus, the best results of query expansion perform with similar results to TF-IDF. Searches for terms such as *Indonesia aquaculture* and *regional minimum wage variation* produce very similar top document results, as do a number of other test queries. It is difficult for a non-expert experimenter to determine which of the top hits are actually more relevant to the sample queries in these and similar examples.

[11] Gabrilovich and Markovitch (2011) experimented with a number of machine learning techniques for doing exactly this. In ES-ESA, these techniques are not yet implemented, but in future work it would be an interesting and important improvement to the system to incorporate some of these methods

It is easier, in fact, to find negative examples, where the "noise" introduced by the query expansion has a detrimental effect. Consider the top search hit for the terms *competition analysis tools* when executed using baseline TF-IDF:

Query:

```
/search?q=competition%20analysis%20tools&debug
```

1. Title: "Empirical Tools and Competition Analysis: Past Progress and Current Problems"

Compare this result, which seems highly relevant, to the top search hit returned from an expanded query:

```
/search?q=competition%20analysis%20tools&expand&debug
```

1. Title: "Approaching Retargetable Static, Dynamic, and Hybrid Executable-Code Analysis"

The TF-IDF result is arguably much better in this example. The reason for the poor performance of the expanded query is that a match for the terms *analysis* and *tools* introduce Wikipedia concept #28811 ("Static program analysis"), which in turn brings in search hits related to the analysis of computer code – probably not the hypothetical user's intention in a query related to competition analysis. The "Static program analysis" concept was assigned to the query text correctly, using the Wikipedia body of concepts and the ESA algorithm, but with respect to the information need underlying the query, this is "semantically" incorrect.

It is worth noting that Gabrilovich and Markovitch (2011) identified this problem and offered a number of solutions using machine learning techniques to improve the quality of the concepts assigned to the query. In particular, one method that was shown to be effective in improving precision in their system was to calibrate or "tune" the queries to the corpus text. They presented this solution in what they called the MORAG system (2011, p. 24). An interesting avenue for future development on ES-ESA would be to extend it to support MORAG-tuned queries (for more information on MORAG, see Gabrilovich and Markovitch 2011, pp. 24-35).

Boosted Query Expansion

In light of the results of query expansion and boosting described in the preceding two sections, it is worthwhile now to revisit the term query *minnow schooling* initially presented in the TF-IDF section. This query performed poorly when run as a TF-IDF query, at least in part because the term *minnow* is not present in any document in the test corpus. Although this fact does suggest that perhaps there are few documents in the corpus that are directly relevant to the query, it should be possible to return documents "more relevant" to the semantic context of the query, perhaps related to an animal behavioral pattern to avoid predators, rather than about "schooling" in the educational sense, which is what all of the articles returned by the TF-IDF query were about. Consider the top two hits when running the same query, *minnow schooling,* but this time expanding the query to include a search for its ESA concepts:

Query:

```
/search?q=minnow%20schooling&boost&expand&debug
```

Results:

1. Title: "Benefits of kin shoaling in a cichlid fish: familiar and related juveniles show better growth"
2. Title: "The effect of temporally variable environmental stimuli and group size on emergence behavior"

The top two results shown above are new, and ranked at the top of the results list (subsequent hits in the results list are the same as under the TF-IDF section). What is interesting to note about these new top results is that, because boosting is introduced along with query expansion, two contextually relevant articles are introduced into the result set via the concept #548423 ("Common minnow") and then boosted by the ESA-derived boosting algorithm, which raises them to the top. This seems to suggest that some combination of query expansion and ESA-based boosting has potential to improve precision *and* recall. If properly calibrated to the corpus as shown by Gabrilovich and Markovitch (2011), it is possible the results could be further improved.

Evaluation and Further Work

ES-ESA, in its current implementation, provides a starting point for researchers and developers to experiment with and compare various ESA-derived semantic search methodologies. The previous examples illustrate the potential, through qualitative evaluation of sample search queries on an arbitrary corpus, for advances as well as some of the drawbacks of these methods. The ES-ESA codebase provides a first stepping stone for developers to expand on and experiment with some of these topics on their own, while delegating the implementation of the underlying algorithms themselves to commonly used, stable, and open-source software frameworks.

Qualitative analysis of these methods on arbitrary data with arbitrarily chosen queries is only the very first, preliminary step – it is a superficial analysis, and is not sufficient for proving the superiority of any one search method over another, or advancing the state-of-the-art for semantic search. In future work on ES-ESA, it would be useful to do an empirical assessment of relevance quality using the same search systems discussed in this work, perhaps utilizing a measurement also used in industry for evaluating search relevance, such as Discounted Cumulative Gain (see Jarvelin and Kekalainen, 2002). In addition to doing empirical evaluation, the system could be evaluated more extensively in an end-user study. Ideally, this evaluation would solicit feedback from subject-matter specialists using ES-ESA to query a corpus of documents in their particular domain of expertise, and then provide feedback on which search method returned the most relevant results.

In combination, these two evaluation techniques would be highly informative as to the effectiveness and further development of ES-ESA in particular and semantic search methodologies in general. It is my hope that ES-ESA system may show promise and utility when released in the open-source community.

Works Cited

Anderka, M., & Stein, B. (2009). The ESA Retrieval Model Revisited. *Proceedings of the 32nd International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pp. 670-671, 2009.

Cohen, T., Blatter, B., & Patel, V. (2005). Exploring Dangerous Neighborhoods: Latent Semantic Analysis and Computing Beyond the Bounds of the Familiar. *AMIA Annual Symposium Proceedings*, 2005, pp. 151-155.

Dong, H., Hussain, F.K., & Chang, E. (2008). A Survey in Semantic Search Technologies. *2nd IEEE International Conference on Digital Ecosystems and Technologies,* pp. 403-408.

Egozi, O., Markovitch, S., & Gabrilovich, E. (2011). Concept-Based Information Retrieval Using Explicit Semantic Analysis. *ACM Transactions on Information Systems*, 29 (2), 1.

Figueroa, A., & Neumann, G. (2016). Context-aware Semantic Classification of Search Queries for Browsing Community Question–Answering Archives. *Knowledge-Based Systems*, 96, pp. 1-13.

Flood, B. (1999). Historical note: The Start of a Stop List at Biological Abstracts*. Journal of the American Society for Information Science,* 50 (12), 1066.

Gabrilovich, E., & Markovitch, S. (2006). Overcoming the Brittleness Bottleneck Using Wikipedia: Enhancing Text Categorization with Encyclopedic Knowledge. *Twenty First AAAI Conference on Artificial Intelligence.* Retrieved from http://www.cs.technion.ac.il/~gabr/papers/wiki-aaai06.pdf.

Gabrilovich, E., & Markovitch, S. (2007). Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis. *Proceedings of the 20th International*

Joint Conference on Artificial Intelligence, 2007, pp. 1606-1611. Retrieved from https://www.aaai.org/Papers/IJCAI/2007/IJCAI07-259.pdf.

Gottron, T., Anderka, M., Stein, B. (2011). Insights Into Explicit Semantic Analysis. *Proceedings of the 20th ACM International Conference on Information and Knowledge Management (CIKM)*, pp. 1961-1964, 2011.

Hurtado Martín, G., Schockaert, S., Cornelis, C., & Naessens, H. (2013). Using Semi-structured Data for Assessing Research Paper Similarity. *Information Sciences,* 221, pp. 245-261.

Järvelin, K., & Kekäläinen, J. (2002). Cumulated Gain-Based Evaluation of IR Techniques. *ACM Transactions on Information Systems* 20 (4), pp. 422-446.

Landauer, T. & Dumais, S. (1997). A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge. *Psychological Review*, 104 (2), pp. 211-240.

Luhn, H.P. (1966). Keyword-in-Context Index for Technical Literature (KWIC Index). In D. G. Hays (Ed.), *Readings in Automatic Language Processing* (pp. 159-67). New York, NY: American Elsevier Publishing Company.

Montejo-Ráez, A., Díaz-Galiano, M., Martínez-Santiago, F., & Ureña-López, L. (2014). Crowd Explicit Sentiment Analysis. *Knowledge-Based Systems* 69, pp. 134-139.

Potthast, M., Stein, B., & Anderka, M. (2008). A Wikipedia-based Multilingual Retrieval Model. *Proceedings of the 30th European Conference on IR Research (ECIR)*, 2008, pp. 522-530.

Sorg, P., & Cimiano, P. (2012). Exploiting Wikipedia for Cross-lingual and Multilingual Information Retrieval. *Data & Knowledge Engineering* 74, pp. 26-45.

Świeboda, W., Krasuski, A., Nguyen, H. S., & Janusz, A. (2014). Interactive Method for Semantic Document Indexing Based on Explicit Semantic Analysis. *Fundamenta Informaticae*, 132(3), 423-438.