A Comparison of Statistical Filtering Methods for Automatic Term Extraction for Domain Analysis

Jason Tilley

Thesis submitted to the faculty of the Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
In
Computer Science

William B. Frakes, Chair
Gabriella Belli, Member
Gregory Kulczycki, Member

December 22, 2008
Falls Church, VA

Keywords: Domain Engineering, Term Extraction, Vocabulary Extraction

A Comparison of Statistical Filtering Methods for Automatic Term Extraction for Domain Analysis

Jason Tilley

ABSTRACT

Fourteen word frequency metrics were tested to evaluate their effectiveness in identifying vocabulary in a domain. Fifteen domain engineering projects were examined to measure how closely the vocabularies selected by the fourteen word frequency metrics were to the vocabularies produced by domain engineers. Six filtering mechanisms were also evaluated to measure their impact on selecting proper vocabulary terms. The results of the experiment show that stemming and stop word removal do improve overlap scores and that term frequency is a valuable contributor to overlap. Variations on term frequency are not always significant improvers of overlap.

Table of Contents

Table of Figures

Tables

1. Introduction

In today's business world, software customers continue to pressure software producers for ever-shortening product life-cycles and low cost goods. Software companies have been looking at software reuse as a means to shorten time-to-shelf of their products and the costs of those products. Studies have shown that reuse also offers many other benefits (Frakes & Kang, 2005). A key step in software reuse is domain analysis. In domain analysis, an engineer studies several related systems and their documentation to understand the commonalities and variables in each system in order to know what type of reuse mechanisms can be created.

Several domain analysis techniques have been introduced. One such methodology is the Domain Analysis and Reuse Environment (DARE) (Frakes, Prieto-Diaz, & Fox, 1998). This methodology uses a domain book to document different views of the domain. The book captures useful code fragments, architectural diagrams, feature tables, a facet table, and a vocabulary. The vocabulary is typically one of the first products created during the process, and is also one of the most important as its formation leads to the creation of subsequent models, like a facet table and a generic architecture.

While the vocabulary can be drawn from many sources, like subject matter experts and the code base, it is typically drawn from the domain documents. These documents are a subset of documents from the domain that describes systems that exemplify the domain, and discuss the theory of the domain. Currently the domain vocabulary is selected manually, with the domain engineers reading all of the documents and selecting vocabulary as they process the information. Current teaching of domain engineering suggests using tools that provide basic term frequency analysis as a starting point for selecting a vocabulary. However, not all words that appear frequently are important, and not all important words appear frequently. Domain engineers currently base their decisions on their understanding of the domain. This task is very time consuming and prone to error due to a single human's perspective on what is important in the study of the domain. The reuse field could greatly benefit from accurate automation of vocabulary generation. Current generation relies on term frequency analysis and a human interpretation of the term frequency metrics.

For many years now, computer scientists and linguists have been studying automation of computer understanding. In particular, information retrieval has been studied to improve corpus compression, storage, and retrieval. As part of that exercise, large vocabularies are extracted out of corpuses to create indexes for searching. To improve search results, some scientists are turning toward semantic understanding of documents, where the vocabulary is no longer just a set of all words in documents, but an understanding of the context of those terms. Some current schemes for attempting to understand the importance of terms in documents are statistically driven, using information retrieval measures such as term frequency and inverse document frequency. Many current measures in terminology extraction are more aligned with natural language processing and information extraction theories. These measures use linguistic properties of words as well as statistical properties. These linguistic properties are concepts like part of speech and phrase cohesion. This paper will measure the effectiveness of many current statistical test measures as well as derivatives of them. We will seek to find a highly accurate statistic for automatically generating a domain vocabulary from a small corpus of documents.

A formal model for our test follows in Figure 1.

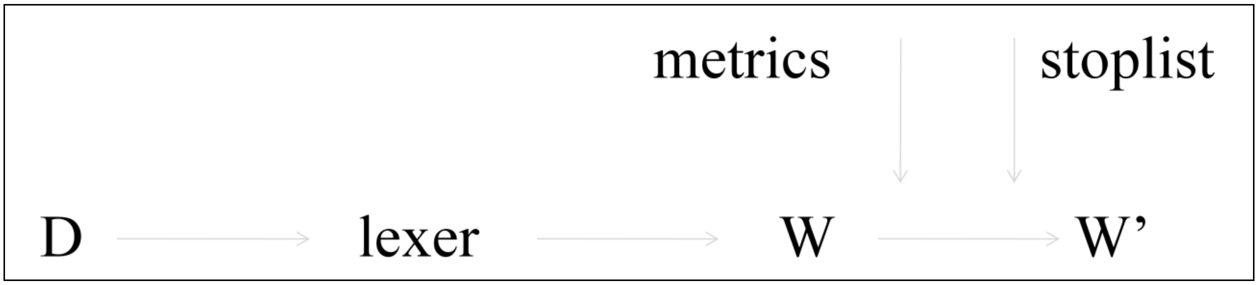
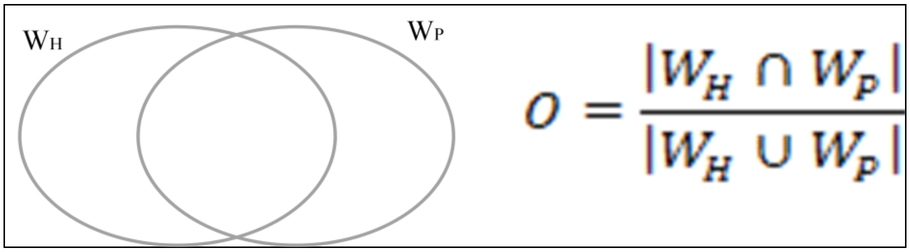


Figure 1: Formal model for Automated Vocabulary Extraction process

Our process begins with a small corpus of documents D. We ran D through a lexing process to extract the set of words in the corpus, W. The process then applied some combination of metrics and stop lists to W to produce W’. The size of W’ is limited to the size of the expert vocabulary, and in the case of ties, the terms are selected arbitrarily. This last set of words was considered our final automatically constructed vocabulary and was measured for overlap with the expert’s manual selections. We measured overlap as the intersection of automatically picked vocabulary, W_H, and expert vocabulary, W_P, over the union of the same two sets. In all of our equations, the size of the automatically generated vocabulary is limited to the size of the expert’s vocabulary. In the case of ties, the tied vocabulary terms are selected arbitrarily. A formal definition of WH is below.

$$W_H = \{t \in T_{Filtered} \mid \forall x \in T_{Filtered} - W_H, weight(t) \geq weight(x), where\ |W_H| = |W_P|$$

This definition holds in 12 of the 14 metrics. Two of the metrics use static allotments from each document in the corpus. The selection algorithm for those metrics is discussed in Section 3.3.2. In Figure 2, a Venn diagram is used to demonstrate overlap.



$$O = \frac{|W_H \cap W_P|}{|W_H \cup W_P|}$$

Figure 2: Equation for Overlap

As an example, suppose that a subject chose 12 words to be in the domain vocabulary. Suppose also that the term extraction algorithm selected 12 words from a corpus, and that four of them were in the subject’s vocabulary. The intersection of these two vocabularies would have a cardinality of four. The size of the union would be 20, because the automatically constructed vocabulary had eight unique terms. The overlap would be 4/20, or .2.

This experiment seeks to determine which of 14 word frequency metrics will best approximate, i.e. have the highest overlap with, the word sets done by humans. The paper also discusses whether the metrics impact the effectiveness of stemming and stop word removal. Furthermore, this paper seeks to determine whether stemming and stop word removal have a significant effect on the 14 word frequency metrics.

2. Problem Background

The concepts of the paper come from three separate fields. Section 2.1 briefly discusses the historical background and relevance of software reuse. Section 2.2 talks about parts of information retrieval which are relevant to this paper. Section 2.3 discusses past work in term extraction.

2.1 Software Reuse discussion

As the software engineering field matures, consumers are increasing their expectations of the software products they use and the companies that produce them. Software projects are becoming larger and more complex, yet customers want more reliable systems and expect the products to be delivered more quickly and inexpensively. These factors have caused software producers to focus on increasing productivity and improving software quality. One method to achieve these goals is software reuse (Frakes & Kang, 2005).

Software reuse is defined as the attempt to reuse existing software artifacts and software knowledge to create new software products. Domain engineering is a directly related area that aids in software reuse. Domain engineers understand that businesses rarely develop completely new products. Most companies build a suite of products that are highly related, and make relatively small changes during the lifecycle of the products they create. Domain engineers try to use tools, theories, and processes to help companies reuse their existing products to build new software. Domain engineering has two phases: domain analysis and domain implementation. Domain analysis is identifying and documenting the common and different aspects of multiple systems within one domain. Domain implementation is building a system based on domain analysis that incorporates reusable assets. These assets can be architectural constructs, code fragments, or even high level requirements. The main goal of domain engineering is identifying the common and varying ideas, concepts, and artifacts in different systems within the same domain or business line. By extracting this information, domain engineers can construct products that help businesses reuse their own software.

Software reuse is not a new concept; in fact it can be traced back to a proposal (McIlroy, 1968) for the software industry to focus more on component based software engineering. Some of the first domain engineering papers were written on programming families (Parnas, 1976) and on domains and domain analysts (Neighbors, 1980). Since the publication of these papers, many trends in reusable software have occurred. Some of these low level trends are software library creation, abstract data types, architectures, and software frameworks. Changes have also occurred at the higher level to increase reuse in software processes, and organizational changes to better aid reuse. There has also been more of an emphasis on designing to interfaces.

Several methods exist to aid domain engineers in their reuse work. These methods all revolve around finding differences and commonalities and creating a reusable product or products. Some of the methodologies are FORM, Koala, ODM, DSSA and Korba (Frakes & Kang, 2005).

Domain Analysis and Reuse Environment (DARE) is a domain engineering tool that places emphasis on repeatable processes and automation. It relies heavily on domain documents, subject matter experts, and a code base to extract similarities and differences between systems, so that generic architectures can be constructed for future related systems. DARE's creators feel that its emphasis on low-level abstractions enables the methodology to be automated more readily than other domain engineering techniques. The high level tasks in DARE, such as generic architecture design, can be guided by the low level results.

The domain analysis process starts with creating a well defined scope for the domain of interest (Frakes W. , 2000). Then source materials and experts are identified. Once these steps occur, the resources are analyzed and catalogued in the DARE domain book. The code base is analyzed using static and dynamic code analyzers similar to reverse engineering tools. This process can aid in creating the generic architecture. The domain's text documents are run through lexers, stop lists, and sometimes conflation algorithms to discover the vocabulary of the domain. At this point, the domain engineer groups the vocabulary terms into facet clusters. The facet clusters provide the basis for a system feature table, a generic feature table, and ultimately the generic architecture. The work done via text and code analysis should augment high level views of the system architecture provided by domain experts. This architecture should have enough rigidness to support the commonalities in all of the systems, but enough adaptability to allow for the variability between systems (Frakes, Prieto-Diaz, & Fox, 1998).

DARE has had 3 implementations over the years, the most recent being a DARE-COTS implementation that relies on COTS products to construct the domain book. The domain book is a Word document that contains sections for each of the main artifacts mentioned above, in addition to the source documents, a thesaurus, glossary, and other useful sections. DARE-COTS relies on several different tools for clustering, text analysis, code analysis, and other forms (Frakes, Prieto-Diaz, & Fox, 1997). Currently, a fourth implementation of DARE is being created with a web interface, and more tightly coupled analysis tools.

2.2 Information Retrieval

Information Retrieval is a field in computer science which seeks to improve the effectiveness and efficiency of searching for documents, searching for concepts in certain documents, and searching for metadata about documents. One aspect of information retrieval that relates this field to terminology extraction is indexing. Indexing is the process of selecting terms from a document to serve as index terms. Index terms are words that describe the document's content, and can help in retrieving documents when users query a system. This field still receives a lot of research effort because of the growing need for fast and effective systems. The quantities of data being produced in digital forms are staggering and the only way to allow users to access them in a reasonable amount of time is through effective information retrieval techniques. There are currently many models for information retrieval, although the main models are still the vector space model, probabilistic models, and Boolean models (Baeza-Yates & Ribiero-Neto, 1999).

The vector space model, developed in 1975, is still considered one of the most effective models. It views documents and queries as vectors of terms. Documents with vectors similar to query vectors score more highly and are selected over documents with low weights. Weights are scores given to a document by taking the dot product of the two vectors.   While some studies show that long documents are more likely to be relevant to queries, many vector based theories normalize the document vector to allow shorter documents to be weighted fairly. Normalization is the process of dividing some quantitative measure of a document by the size in words of that document. This has the effect of making all documents in a corpus equal in weight despite their size. This model clearly uses term frequency as a factor in a document's weight, but this is not enough for successful retrieval. Articles, prepositions, and other terms occur very often in literature, but clearly are not good indicators of a document's relevance. Terms are also weighted by their inverse document frequency, which is the number of different documents that they appear in across a corpus. Terms that are in fewer documents are given more weight. The theory behind this approach is that terms with low document frequencies are more specific to the documents that contain them. Overall the weight of a term increases the more it is found in one document and decreases the more it is found in other documents.

2.3 Terminology Extraction

While information retrieval saw many of its early theoretical successes in the 1970's, we find that terminology extraction's most notable research did not come until the 1990's (even though some substantial work happened even as early as the 1960's), particularly with Justeson and Katz's work about term frequency and noun phrase patterns (Justeson & Katz, 1993). Following this work we see much research into different measures regarding selecting technical terminology from a given domain, especially with regard to linguistic properties of the corpus.

Justeson and Katz laid the foundation by showing that term frequency was a major contributor to selecting terms that identified a domain. However, their studies also revealed that frequency was best coupled with some linguistic properties of the terms selected. They found that when they examined noun phrases, certain patterns in these noun phrases were more likely to contain appropriate terms than other patterns. The original paper suggests three noun phrase patterns cover most terms, while a follow on work by (Arppe, 1995) suggests the eight noun phrase patterns covered over 90% of the terms in his work.

Many of the works that followed continued to study the effects of proper noun phrase selection, while a few others introduced new concepts. A paper dealing with better understanding of phrase selection relied heavily on linguistic properties and manipulation of the phrases (Church & Dagan, 1994). Their paper showed that grouping terms by noun phrase heads brought out terms that were otherwise unlikely candidates. While their work was based on technical term translation, it has direct relevance to terminology discovery. Some other papers detailing research in noun phrase detection are (Voutilainen, 1993)and (Bourigault, 1993). These two researchers also released tools that provide noun phrase extractions, called LEXTER and NPtool, respectively. There are many other tools that provide the same functionality with varying results.

One of the problems with syntactic methods for terminology discovery is that more terms are selected as candidates than should be. Further filtering must be done by statistical methods. Studies show that 50-80% of terms run through statistical filters are not candidates to be selected as vocabulary terms (Velardi, Missikoff, & Basili, 2001). Many of the statistical studies also revolve around noun phrases. Mutual Information (Fano, 1961) and the Dice Factor (Smadja, McKeown, & Hatzivassiloglou, 1996) are two statistical measures designed to eliminate poor candidates. These statistical values, referred to as association measures, try to determine the strength of the relationship between words in a phrase. Cohen shows that infrequent words can be extracted as terms, if they are in contained infrequent n-grams (Cohen, 1995). Another statistical approach that has been studied was morphemic frequency. By looking at morphemes that occur frequently we can extract less frequent words based on the frequency of their morpheme in the domain. (Heid, 1998) (Witschel, 2005)

A few more statistical measures that have provided benefits to terminology extraction are cohesion, consensus, and relevance. Domain relevance is a measure that scores a term's probability that it will be found in a domain document versus its probability that it will be found in a document in another domain. Terms that are only likely to be found within the domain of study are scored highly and usually are good candidates. Domain consensus measures the distribution of a term among the documents within one domain. Well distributed terms can be good candidates because they demonstrate universal recognition as domain terms among many domain authors. Cohesion is a term metric associated with phrases. It measures the likelihood that terms within a noun phrase will be found next to each other. Terms paired with other frequent terms may be eliminated if their cohesion is low. (Sclano & Velardi, 2007) (Velardi, Missikoff, & Basili, 2001)

The theories that have been studied generally have come from related, but slightly different fields and therefore have had slight changes to the filters. Keyword extraction tries to select a very small subset of words that embody the content of the document, so that people searching for it can find it. This field adds the measure of structural relevance. Words that are close to the beginning of the document or that are part of the title of a subsection are more likely to be keyword choices and are scored higher. Ontology builders seek to extract the terms from a domain and relate them to each other. Low level ontologies also allow for instances of concepts to be recognized, so proper nouns are not necessarily filtered as they are in thesaurus and glossary building. Glossaries and thesauruses mainly rely on linguistic filtering because of the need for context when trying to relate phrases. Most, if not all, of the theories use multiword noun phrases because single words tend to be too generic and polysemous.

3. Methods

3.1 Hypothesis

The purpose of this paper is to evaluate various automatic vocabulary extraction methods. Each methodology gave weights to terms within a corpus, and those terms were evaluated against a manually selected vocabulary. We compared **o = f(K,C,S)**, an overlap score for each methodology K. The variable C represents the conflation variable which had two possible values: stemmed with Porter's stemming algorithm, or not stemmed. The variable S represents 3 possible stop word list options: no stop list, short stop list, or long stop list. We believed that the different metrics would produce a significant difference for the dependant variable, overlap. Furthermore, we believed that stemming and stop word removal would significantly affect overlap. We also believed that different stop word lists would affect overlap.

3.2Test set preparation

3.2.1 Test set selected

Over the years, several Virginia Tech graduate school classroom exercises have been conducted in which students used the DARE methodology to analyze a domain of their choice. The results of these exercises were complete, or partially complete DARE domain books. Each subject put bounds on their domain and then selected an arbitrary number of documents that discussed that domain. A minimum of three documents were selected. These documents were typically research papers and web pages, but also included system documentation. With the help of those documents, subjects selected terms that were pertinent to the domain. The instructions suggested that subjects use tools that could automatically provide statistics on the terms in their corpus, but were not bound to choosing highly frequent terms. The vocabulary was then used to create other artifacts in the domain book. A subset of these domain books, and their corresponding vocabulary selections, were used for our test bed. We selected fifteen domain books to analyze. The vocabularies, as determined by the students, were considered our expert determined vocabularies, and the source documents that each subject's project references were the domain corpuses used to automatically generate a vocabulary.

Of the 15 subjects, seven of them chose conflation algorithms as their domain. Four of them chose domains related to application or programming metrics. The remaining four domains were in personal information management, military medical systems, encryption, and sentence alignment. Figures 3-6 display the frequency of subjects' chosen corpus sizes, token counts, unique token count, and the size of their selected vocabularies.

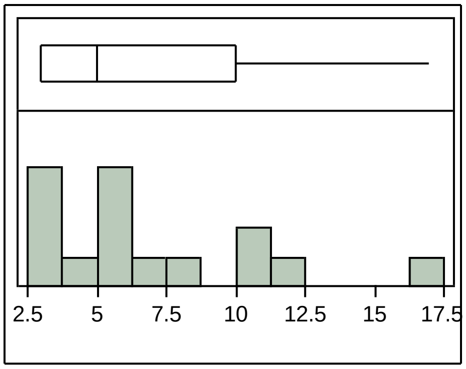


Figure 3

Frequency of Test Corpus Sizes (Documents)

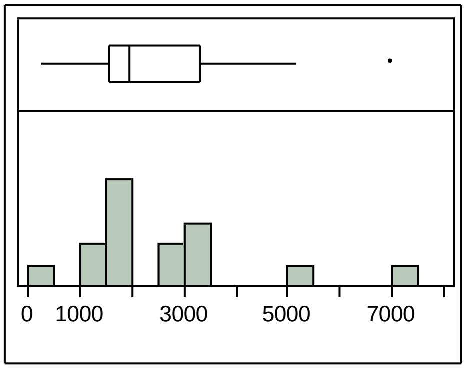


Figure 4

Frequency of Test Unique Term Counts

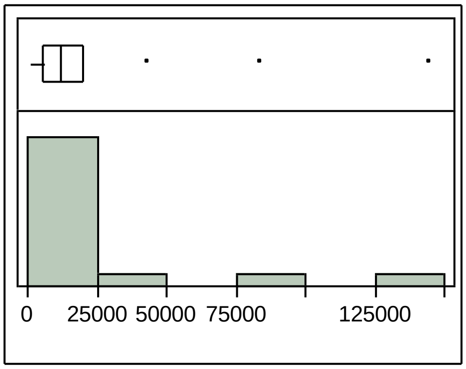


Figure 5

Frequency of Test Total Token Count

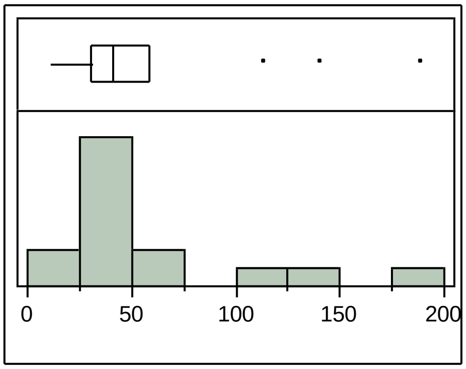


Figure 6

Frequency Test Vocabulary Sizes

3.2.2 Overlap

We used a series of information retrieval related metrics to predict the vocabularies contained in each domain, and then compared the results to the "expert" vocabulary. We used a measure that we call overlap. We define overlap as the number of terms in the intersection of the expert selected vocabulary and generated vocabulary, divided by the number of terms in the union of the same two vocabularies. Overlap is used to measure and compare our independent variables, namely the information retrieval statistics used to statistically filter the vocabularies. Overlap is a measure that is directly related to coefficient of similarity, a metric used to compare the automatic and manual indexing results (Salton & McGill, 1983). In our experiments, we limit the size of the automatically selected vocabulary to the size of the test vocabulary. In the case of ties, the selected terms are arbitrary.

3.2.3 Comparison

After processing each vocabulary through various automatic term extraction algorithms, we compared the metrics' overlap measures against each other using notched box plots (Crawley, 2007). Notched box plots are a variation on normal box plots that provide a simple visual demonstration of the experiments' statistical significance. The box plots also tell us the median overlap of each metric, as well as provide indicators of variance. The box plots will also show outliers. It is through these visual displays of statistics that we can judge whether one metric was significantly better or worse than others. If the notches do not overlap, statistical significant differences are present.

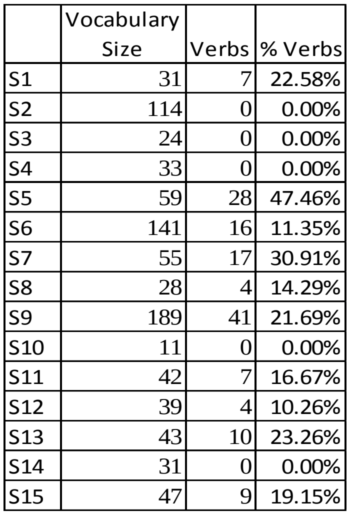
3.2.4 Preparation

We prepared our corpuses first with tokenization. The frequency with which each word appears in a document was counted and recorded. The recorded frequencies are the number of occurrences of a word in an individual document, not the corpus. This recorded frequency list was the baseline for an individual document. We created several variations of this baseline. The first two variations were created through stop word removal. Stop words are terms that occur in language so often that they are generally never associated with a domain, like articles, prepositions, and exclamations. We used two stop word lists to create two variations, one being a short stop list (Alex, 2008) and the other being large (Lektek International). With the baseline and the two stop word variations, we also created another variation by using a stemmer. Stemming is the process of grouping related terms into one central term, often by removing suffixes and prefixes (Frakes W. B., 1992). The Porter Stemmer has been chosen because of its popularity, and the convenience of having many implementations available. Stemming was most likely to reduce the number of terms by bringing similar terms into the same grouping, such as plural and singular nouns, and past and present tense verbs.

3.2.5 Vocabulary creation

These steps left us with large vocabularies that we trimmed down using statistical metrics. We used Java to write several algorithms that use corpus and document statistics to filter the corpus's vocabulary to a few terms that best exemplify the vocabulary of the domain. Although most work in this field has strictly used noun phrases for term candidates, our research used any single word, not just nouns, in a corpus as a possible candidate. Two reasons exist for this decision. Our gold standard vocabularies incorporate a substantial number of verbs (14.51%), and some also incorporate adverbs and adjectives. The percentage of verb selection is shown in Table 1.

Table 1 Percentage of Verbs in Expert Vocabularies



	Vocabulary Size	Verbs	% Verbs
S1	31	7	22.58%
S2	114	0	0.00%
S3	24	0	0.00%
S4	33	0	0.00%
S5	59	28	47.46%
S6	141	16	11.35%
S7	55	17	30.91%
S8	28	4	14.29%
S9	189	41	21.69%
S10	11	0	0.00%
S11	42	7	16.67%
S12	39	4	10.26%
S13	43	10	23.26%
S14	31	0	0.00%
S15	47	9	19.15%

Domain engineering is also different from many of the other fields requiring vocabularies in that the vocabulary is a precursor to future artifacts, particularly a generic architecture. Software projects often use verbs to model methods and transitions, and to automatically create a generic architecture including these methods, the vocabulary must have verb candidates.

3.3 Metrics

For each metric, we weighted each term in a subject's corpus and assigned it a value, w. The term weights were sorted and the top V terms were selected (unless noted otherwise), where V is the size of the expert selected vocabulary for the domain corresponding to the corpus. Table 2 provides an abbreviation for metric names that will be used in any following tables. The table also provides a short description of the weighting rationale of each term. Figure 7 provides a taxonomical view of the metrics in the experiment.

Table 2 Metric Summary and Abbreviations

Abbreviation	Metric title	Rationale
TF	Corpus Term Frequency	Rewards high term count, large documents have advantage
LTF	Logged Term Frequency	Minimize effect of unusually high terms, similar to normalization
USN	Document Term Frequency	Rewards words that appear a lot in one document
ED	Evenly Distributed	All documents contribute same number of terms
BD	Favor Big Documents	Reward for large documents
NTF	Normalized Term Frequency	Rewards high term count, but negates large document skewing
DR	Document Relativized	Less reward for large documents, penalizes verbose documents
CD	corpus relativized	Less reward for large documents
DRDA	Document Relativized - Document Average Frequency	Less reward for large documents
CRDA	Corpus Relativized - Document Average Frequency	Less reward for large documents
TFIDF	Term Frequency and Inverse Document Frequency	Reward terms that are in few documents, but that appear frequently
LTFIDF	Term Frequency and Logged Inverse Document Frequenc	Flattens distribution of document frequency, making outliers less powerful
DC	Distribution consensus	Rewards terms that occur in same frequency in multiple documents
BC	Binary Consensus	Rewards consensus, rewards minimum frequency of one

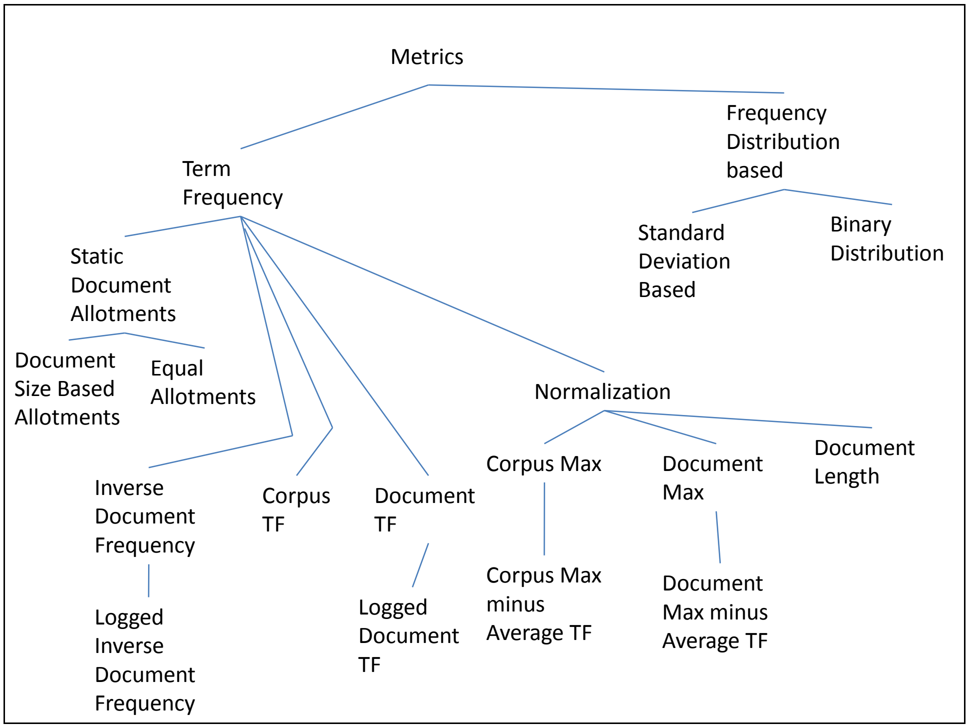
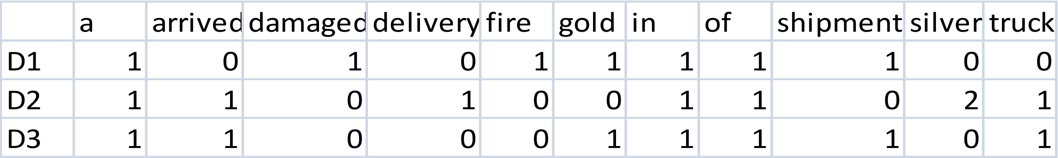

Metrics
Frequency Distribution based
Term Frequency
Standard Deviation Based
Binary Distribution
Static Document Allotments
Document Size Based Allotments
Equal Allotments
Normalization
Corpus Max
Document Max
Document Length
Inverse Document Frequency
Corpus TF
Document TF
Corpus Max minus Average TF
Document Max minus Average TF
Logged Document TF
Logged Inverse Document Frequency

Figure 7 Metric Hierarchical Ordering

We used an example from an information retrieval text (Grossman & Frieder, 1998) that contains three documents and a query. We used the query terms in the original example as our vocabulary terms. The example can be used by readers to help understand the scoring system for the metrics. The example results assumed no stop words have been removed, and no stemming has been performed. In each metrics subsection, the following corpus's terms are ranked according to the metric being described. The table following the corpus description provides a summary of the term occurrences within each document. The term occurrences are summarized in Table 3.

Vocabulary: gold, silver, truck
Document 1: Shipment of gold damaged in a fire
Document 2: Delivery of silver arrived in a silver truck
Document 3: Shipment of gold arrived in a truck

Table 3 Example Corpus's Term Occurrences



	a	arrivec	damagec	delivery	fire	gold	in	of	shipment	silver	truck
D1	1	0	1	0	1	1	1	1	1	0	0
D2	1	1	0	1	0	0	1	1	0	2	1
D3	1	1	0	0	0	1	1	1	1	0	1

Table 7 provides a set of definitions for the variables that are used in the metric definitions.

Table 4 Formula Variable Descriptions

N	Number of documents in a corpus
T_c	Number of terms in a corpus
T_j	Number of terms in document j
$\|V\|$	Number of terms in expert selected vocabulary
tf_{ij}	Number of occurrences of term i in document j
tf_{ic}	Number of occurrences of term i in a corpus
P_j	Number of occurrences of most frequent term in document j
P_c	Number of occurrences of most frequent term in corpus
n_i	Number of documents in which term i appears
w_i	Weight of term i in a corpus
K_j	Number of terms to be selected from document j
V	Size of an expert's vocabulary

3.3.1Term Frequency Based

Most existing indexing algorithms and term extraction algorithms base their results on some calculation involving term frequency. It stands to reason that words that are used often are likely candidates to be terms used in the vocabulary of the domain. This assumption is justified by (Justeson & Katz, 1993), although they further their results with restricting candidates to particular patterns. The original paper on the importance of term frequency in selecting index terms for information retrieval dates back to (Luhn, 1958). Our study neither includes phrases, nor does it exclude non-nouns, so this value was a good baseline with which to compare other values. Justeson and Katz eliminated verbs based on their research of technical dictionaries in multiple domains that showed 97% of terms were noun phrases. Our experts have selected 15% of their terms to be verbs.

3.3.1.1 Corpus Term Frequency

The first metric is a purely term frequency metric, calculated over the entire corpus. This metric favored words that appeared often, but it consequently favored large documents.

$$\mathrm{w_i} = \sum_{j=1}^{N} \mathrm{tf_{ij}}$$

This equation states, sum up the term frequencies, tf, from each document j, in the corpus of size N. The sum will be $\mathrm{w_i}$, the weight of term i. We refer to this methodology as **corpus term frequency**.

In our running example, the term ‘a’ appears once in every document and would receive a score of 3, while ‘silver’ appears twice, but only in one document, and would receive a score of 2. Table 4 presents all of the scores for the terms in our example and bolds the terms that would be selected as vocabulary terms.

Table 5 Corpus Term Frequency Example Results

a	3
in	3
of	3
arrived	2
gold	2
shipment	2
silver	2
truck	2
damaged	1
delivery	1
fire	1

3.3.1.2 Logged Term Frequency

Logarithms are powerful modifiers of data. They have the ability to reduce the range of values in a set. We used logarithms to reduce the range of terms in any given document. This dampens the data, making the distribution of frequencies smaller.

$$\mathrm{w_i} = \sum_{j=1}^{N} \ln(\mathrm{tf_{ij}} + 1)$$

A different example is necessary for this metric. Suppose two terms exist, term A and term B. If term A appears in two documents twice and one document fourteen times, it has a term frequency of sixteen. If term B appears in all documents five times, its term frequency is eighteen. In corpus term frequency term A outscores term B, but when we apply the natural log to all term counts prior to summation, term B will have a higher score. Term B scores 4.83, while term A scores 4.02. The metric is called **logged term frequency**.

3.3.1.3 Document Term Frequency

A unique metric that we measured was maximum term frequency with a document. Instead of summing all the term frequencies together, we just selected the words that appeared most within their respective document. This weight was normalized, so as not to penalize words in short documents. This possibly provided new terms to our vocabulary by finding terms that appear often in one document, but not in any others. It favored unevenly distributed word frequencies. This methodology is called **document term frequency**.

$$\mathrm{w_i} = \max_{1 \leq j \leq N} tf_{ij}$$

In our example, even though the term ‘a’ appears three times in the corpus, it only appears at most once in any document, and therefore scores a one. The term silver appears twice in a document and scores a two. Most terms have a maximum frequency of one, but only 3 vocabulary words exist, so only 3 vocabulary words will be automatically selected. Words that have a tie score are arbitrarily selected. Table 5 presents the example scores for document term frequency.

Table 6 Document Term Frequency Example Scores

silver	2
a	1
arrived	1
damaged	1
delivery	1
fire	1
gold	1
in	1
of	1
shipment	1
truck	1

3.3.2 Static Document Allotments

The following two metrics use a different approach to term selection than the rest of the metrics. The weights given to each term are specific to individual documents within the corpus and are not summed, nor used in a maximization formula. Each document is allotted some number of terms, K_j, to add to the set of terms to be returned to the user. The value of K_j is different in each algorithm. The scoring internal to each document is a normalized term frequency score as discussed in section 3.3.3.1. The algorithm is as follows:

```
$W_{AG}$ = {}
FOR( J=1;J<=N;J++){//FOR EACH DOCUMENT
    RANKTERMSBYWEIGHT();//SORT THE TERMS BY THEIR WEIGHT, IN DESCENDING ORDER
    FOR(I=1;I<$T_J$;I++){//FOR EACH TERM IN DOCUMENT J
            IF($T_{IJ}$ EXISTS IN $W_{AG}$){
                    CONTINUE;
            }
            ELSE{
                    $W_{AG} = W_{AG} + T_{IJ}$
                    IF($|W_{AG}|>K_J$)
                            BREAK;
            }
    }
}
```

3.3.2.1 Equal proportions

Another alternative to this corpus term frequency is to take a fixed proportion of terms from each document. This method alleviated favoring large documents, but still holds the premise that the terms occurring often were the best terms to select.

$$K_j = floor\frac{V}{N}$$

V is the size of the manually selected vocabulary. We refer to this methodology as **evenly distributed**.

In our example we have three vocabulary words and three documents, so each document will contribute one word to the automatically selected vocabulary. Again, all ties are arbitrarily decided, so the only certain word to be added to the vocabulary is silver from document two. Some word with a score of one will also enter from document one and document three.

3.3.2.2 Length based proportions

It may be possible that large documents reflect a greater depth of a field than shorter documents, and as such their terms should be given priority over terms from smaller documents. This concept is supported in a term normalization paper presented in 1995. (Singhal, Salton, Mitra, & Buckley, 1995) In our version of this metric, each document in the corpus is allotted a quota of terms to provide to the domain vocabulary based on the percentage of terms the document has in the term size of the corpus. This metric is the **big document** metric.

$$K_j = floor\frac{T_c}{T_j}$$

Our running example is not very fit to demonstrate how this metric works, because each document basically contributes the same number of terms to the corpus. In an example where the subject chose six vocabulary terms out of three documents that contained fifty, one hundred, and one hundred and fifty terms the documents would contribute one, two, and three terms, respectively.

3.3.3 Normalization Based

The concept of term normalization has been a standard metric for information retrieval since the late 1970's (Noreault, McGill, & Koll, 1980). Normalization occurs by dividing the frequency of a term in a document by the total number of terms in a document. By normalizing each document, we removed the effect of size, and each term frequency is now a percentage of another characteristic of the document, such as the document's term count.

3.3.3.1 Document Terms Counts

The most common way of normalizing a document is by dividing a term's frequency by the number of terms in a document.

$$\mathrm{w_i} = \sum_{j=1}^{N} \mathrm{tf_{ij}} \,/\mathrm{T_j}$$

$\mathrm{T_j}$ represents the total term count in document j. We refer to this metric as **normalized term frequency**.

In our example, the term silver would be evaluated with a score of .25 (0/7+2/8+0/7), and gold would get a score of .28 (1/7 + 0/8 +1/7)

3.3.3.2 Document Maximum Frequency

This paper also explored other metrics that have a similar effect to the standard normalization metric, such as a relative score. Instead of dividing the term frequency by the number of words in a document, we divided by the most frequent term in a document, and then summed up the results afterward. This gave the most frequent word a score of one for the document for which it is the most frequent term, and added on any scores it received by occurring in other documents. This had a similar effect to normalization because no score contributed to a term from any single document will be greater than one, but the scores resulting from each document will be different than the scores after standard normalization. The weight of term contributions was a ratio of the term frequency and the most common term, rather than the frequency and the document size.

$$\mathrm{w_i} = \sum_{j=1}^{N} \mathrm{tf_{ij}} \,/\mathrm{P_j}$$

In the example, the term 'a' would receive a score of .5 (1/2) from document B, because the maximum term frequency in document B is two (silver occurs twice). In document A it would receive a score of 1 (1/1), because it is the most frequent term (along with many other terms that occur once in that document.)

3.3.3.2.1 Document Maximum Frequency and Term Average Frequency

This metric used normalization based on the most frequent word in the document also, but subtracted the average frequency that term ‘i’ appears across as documents in the corpus as well. It is referred to as **document relativized minus average TF**.

$$w_i = \left(\sum_{j=1}^{N} tf_{ij} / P_j \right) - \frac{\sum_{j=1}^{N} tf_{ij}}{N}$$

Using the same document relativized score for ‘a’ above, we subtract 1 from each score, because ‘a’ occurs three times total over three documents for an average frequency of one. So its total score would be -.5 ((1-1)+(.5-1)+(1-1)).

3.3.3.3 Corpus Maximum Frequency

The paper also explored the above maximum frequency normalization technique by using the most frequent term in the corpus. Because the corpus’s most common term is fixed, we have a constant. Our results should be similar to the results of term frequency, if not exactly the same.

$$w_i = \sum_{j=1}^{N} tf_{ij} / P_c$$

This metric will be called **corpus relativized**. Table 6 shows the results for the terms in the example corpus. Notice that they are proportional to the score of corpus term frequency.

Table 7 Corpus Relativized Example Scores

a	1.5
of	1.5
in	1.5
arrived	1
gold	1
shipment	1
silver	1
truck	1
damaged	0.5
delivery	0.5
fire	0.5

3.3.3.3.1 Corpus Maximum Frequency and Term Average Frequency

The above metric was run again, but with the normalization being based on the most frequent term in the corpus. This metric is **corpus relativized minus average TF**.

$$w_i = (\sum_{j=1}^{N} \frac{tf_{ij}}{P_c}) - \frac{\sum_{j=1}^{N} tf_{ij}}{N}$$

The example scores would use the same value from the summation as in the table above, but would need to factor in the average term frequency. The term ‘a’ had a score of 1.5, but we would now subtract 1 (3/3), the average frequency of ‘a’ in the documents. The new score for ‘a’ is .5.

3.3.4 Inverse Document Frequency

Inverse document frequency measures favor words that appear in very few documents. It is used quite often in indexing, because the documents in the corpus being indexed are generally quite varied, so appearing in few documents makes a term a good identifier for those documents. Coupled with term frequency, inverse document frequency helps to select words that occur often, but only in few documents.

3.3.4.1 TF IDF

This metric multiplies the term frequency by the number of documents in the corpus and divides by the number of documents that the term appears in. We refer to this as **TFIDF** for the remainder of this document.

$$w_i = \sum_{j=1}^{N} tf_{ij} * N/n_i$$

The term ‘a’ scored highly in the corpus term frequency metric, but in this one its score will not improve much. The number of documents in the corpus is three and the number of documents that ‘a’ appears in is also three, so the expression N/n_i ends up being one. This term scores a three, contrasted to the term ‘silver’ which only appears twice in the corpus, but in only one document. The term silver scores a six (0 + 2*3/1 + 0).

3.3.4.2 Logged IDF

This metric is similar to TD IDF, except that we are weighting term frequency more highly. The logarithm function decreases the range of IDF values. Henceforth, we refer to this as **logged TFIDF**.

$$w_i = \sum_{j=1}^{N} (tf_{ij}) * \ln(\frac{N}{n_i})$$

Our running examples don’t provide a wide range of inverse document values, so we construct another example with two terms, and one thousand documents. Suppose term A appears 20 times, but in only 1 document. In TFIDF, term A will score 20,000 (20*1000/1). Term B appears 40 times but in 10 different documents. It will score 4000 (40*1000/10). In TFIDF, Term A outscores Term B dramatically. When we apply natural logs to the IDF component however the IDF factor will decrease in importance. Term A will score

(20*ln(1000))=138, but Term B will now score (40*ln(100))=172. Clearly logging IDF will play a factor in the results.

3.3.5 Distribution Based

Previous term extraction work that looks at noun phrases has shown that consensus can be an important factor in selecting a suitable vocabulary. (Sclano & Velardi, 2007) This metric will reward terms that have consensus. Consensus is a reflection of a term's popularity by many authors. Terms that display an even probability distribution across the documents of the domain have consensus.

3.3.5.1 Standard Deviation Based

The paper measured consensus based on frequency distribution. Terms that appeared the same number of times in each document scored highly. This metric normalized the terms prior to evaluating the variance of the term frequencies. We will refer to this metric as **distribution consensus**.

$$w_i = \sqrt{\frac{1}{N}\sum_{j=1}^{N}(tf_{ij} - (\frac{1}{N}\sum_{j=1}^{N} tf_{ij}))^2}$$

An example for this metric will be discussed in the next section to provide a contrast to the metric below. One interesting note about this metric is that it is the only metric considered that values lower scores more highly than high scores.

3.3.5.2 Binary Distribution Based

We also measured consensus based on a term's binary distribution. For every document a term appears in, the term gets a score of one. The more documents a term appears in the higher its score was. This metric is a contrast to terms favored by inverse document frequency. We refer to this metric as **binary consensus** for the remainder of the paper.

$$\mathrm{w_i} = \sum_{j=0}^{N} \mathrm{bin_{ij}}, \qquad \text{where } \mathrm{bin_{ij}} = \begin{cases} 0, \text{if } \mathrm{tf_{ij}} = 0 \\ 1, \text{if } \mathrm{tf_{ij}} > 0 \end{cases}$$

Suppose that 3 documents exist with 3 different terms. Suppose term A appears once in document A and document B. Suppose term B appears once in document A, once in document B, and 9 times in document C. Term A will score .58 in distribution consensus and a 2 in binary consensus. Term B scores better in binary consensus, with a value of 3, but has a higher distribution consensus, 4.6 (which is worse than .58 in this metric.)

4. Results

In this chapter we discuss the resulting data and how it supports or contradicts our hypothesis that pre-filtering methods and statistical metrics impact term extraction to different degrees. Section 4.1 will show the impact of pre-filtering methods, as metrics are held constant. Section 4.2 will discuss differences in each metric as filtering is fixed. Within sections 4.1 and 4.2, each subsection will be titled with the fixed metric or filtering method, respectively. Section 4.3 will discuss general anomalies in the data.

4.1 Pre-filtering impact

It is very clear that stemming and stop word removal had positive impacts on overlap. This result is not surprising as these methods are consistently used in most information management fields, like retrieval and term extraction. While it is clear the results are better on the whole, we found that there are cases when improvement is not guaranteed, particularly with stemming. Figure 8 provides the results of the baseline run. The results displayed are overlap scores received when no stemming or stop word removal was done.

Stemming is almost imperative when it comes to retrieval, when speed is of concern. In our experiments it improved vocabulary extraction. There are a few problems that did occur however. Many of the preselected vocabularies did indeed contain morphological variants of each other, and stemming eliminated any chance of having two words with the same stem from being selected as terms. One example would be the preselected vocabulary of subject 5. Stem, Stemming, and Stemmer were all vocabulary terms for the domain of conflation. In every filtering metrics' output, besides Distribution consensus and Logged TF IDF, the word "stem" was chosen. However stemmer and stemming were consistently omitted. The number of morphological variants in the corpuses is very low, so we don't see tremendous loss of variants being excluded, which is why we see an overall gain in overlap when stemming.

As expected the stop word lists both successfully increased overlap on the average, but we can find some cases where stop word list removal can hurt effectiveness. An obvious example is medicine, where vitamin types have short names like 'A' and 'B'. Many stop word removal lists remove single letter words from texts. Most stop word lists also remove articles and prepositions. The first stop word list we used, a short stop list, removed many of these and improved performance. It did however leave many words off of its list that the big stop list included. For instance, "end" and "ending" are both on the big list, but not on the small list. The vocabulary created by subject 15 included both "end" and "ending" in its vocabulary. The short stop list performed better than the big stop list in this case because it allowed these words to be selected.

Both stop lists took away many candidates that scored highly in the base line run, like "the" and "of". The big stop list found more words than the short stop list, and therefore received better overlap results. For instance, subject 12 received a 26% overlap score versus a 20% overlap score when using the document term frequency metric. Some words that appear in the short stem vocabulary that did not appear in the big stem vocabulary were "I", "our", "you" and "each". In this case, removing these words was very effective.

In most cases, a combination of both stemming and stop word removal was more effective than either filtering method alone. The big stop word removal list was more effective overall than the short stop word list, even in conjunction with stemming. The results of using a big stop list and stemming were the highest scoring results in our tests.

4.1.1 Term Frequency

Figure 8 displays the overlap scores for our filtering methods when the corpus term frequency metric was applied. As discussed above, the baseline performs poorly because words common in everyday language are not omitted, like ‘the’ and ‘or’. In fact, ‘the’ was the top word in almost every test. It also performs poorly because words that have several variants are counted individually. The baseline could not overcome the problems of word variants and common terms in almost all runs, and therefore the distribution had small variability.

The test shows that the big stop word removal and the stemming filterer had very similar improvements on the baseline, and the big stop word removal and stemming in the same test had significant improvement. The base run had the smallest distribution because the same set of stop words caused noise in every run, and unstemmed words were not consolidated into a single variant.

The outliers in the figure belong to two subjects that remain outliers in almost every test. The reasons behind these outliers will be discussed later.

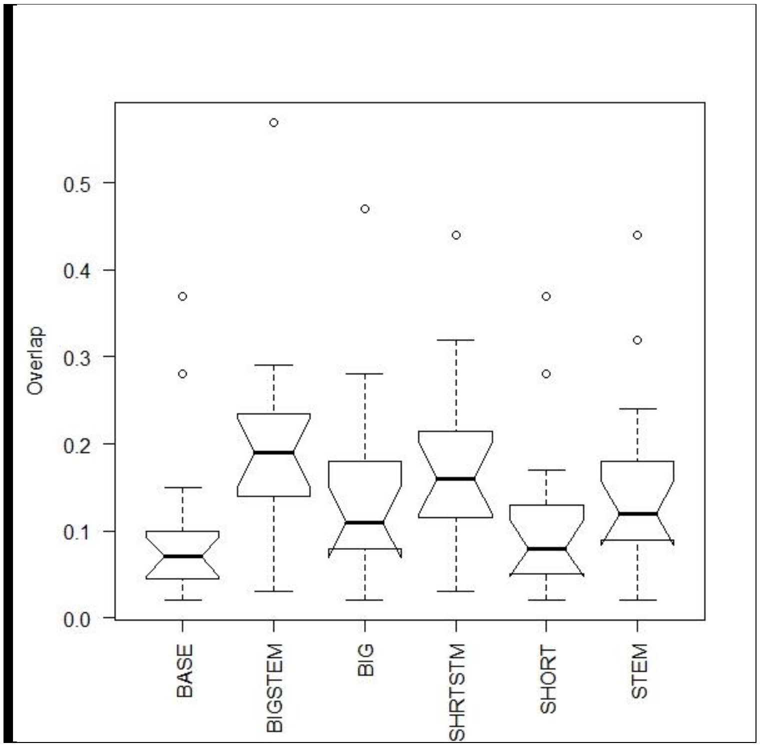


Figure 8 Prefiltering Overlap Results after TF

4.1.2 Logged Term Frequency

Figure 9 displays the overlap scores for our filtering methods when the logged term frequency metric was applied. Although the means and distributions generated from LTF are different than the TF metric, the scores are almost the same relative to one another. The metrics that were calculated after the big stop list was applied resulted in a slightly wider distribution. This phenomenon occurs in several of the metrics, although certainly not all of them. This may be because of the sensitivity of overlap when errors occur. The occurrence of common words in the automatically created vocabulary instantly and almost without fail introduces a large constant population of erroneous terms like 'of' and 'the'. This effectively shrinks the population of possibly correct terms. In other words, without stop word removal, the automatically created vocabularies all have a portion of words that are the same, even though they are all incorrect. If the stop words are removed there is a larger degree of variation in the terms that are selected, creating an opportunity for variance in the overlap scores.

Stemming and stop word removal had similar impact on the baseline. Stemming combined with stop word removal significantly improved overlap over the baseline. The baseline had a smaller distribution because stopwords accounted for a large portion of the error in each test, and multiple word variants were not stemmed, lessening their chance for vocabulary selection.

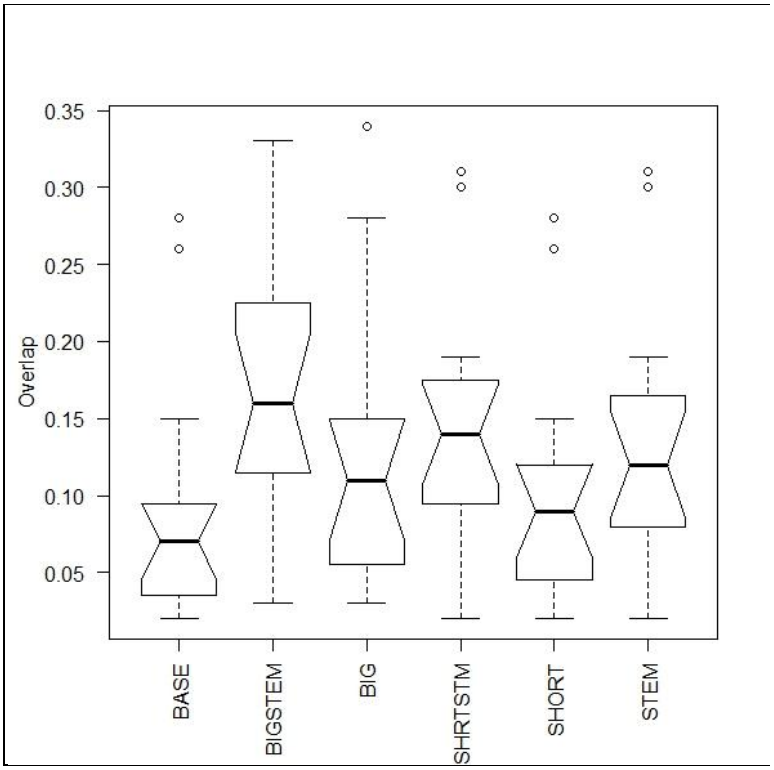


Figure 9 Prefiltering Overlap Results after LTF

4.1.3 Un-summed Term Frequency

Figure 10 displays the overlap scores for our filtering methods when the document term frequency metric was applied. The results for the document term frequency are similar again to

TF and LTF relative to one another. Stemming and stemming combined with stopword removal had significant impact on the baseline, but stop word removal alone did not in this experiment.

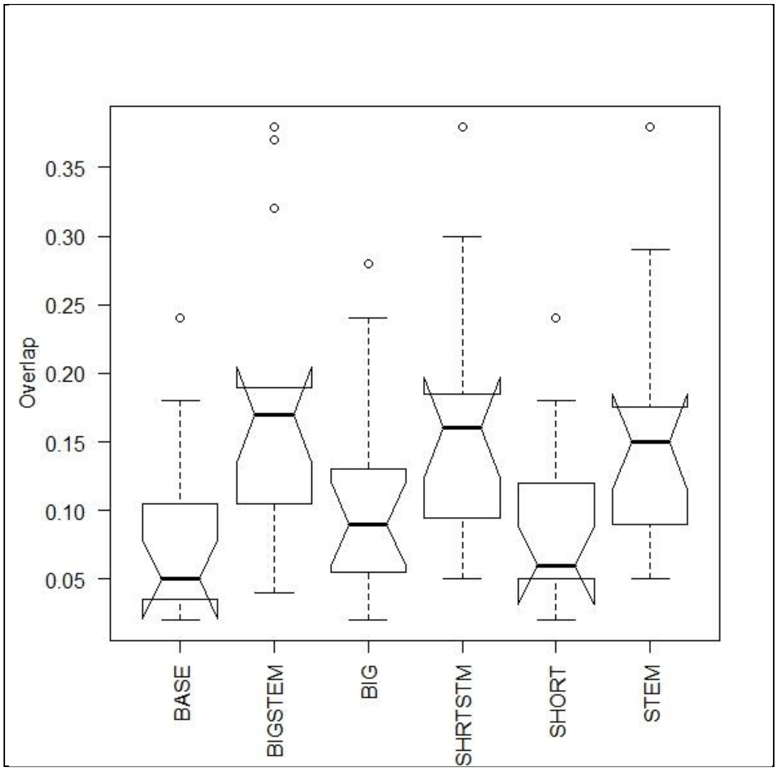


Figure 10 Prefiltering Overlap Results after USN

4.1.4 Evenly Distributed

Figure 11 displays the overlap scores for our filtering methods when the evenly distributed metric was applied. Most of the results for evenly distributed align with previous comments. One interesting observation is that the big stop list did not improve the baseline as much when ED was applied as in other metrics. This phenomenon also occurs in the BD metric. Both metrics allow certain proportions of the vocabulary to come from each document. It is not understood why this occurs. Stemming and stemming combined with stopword removal had significant impact on the baseline, but stop word removal alone did not in this experiment.

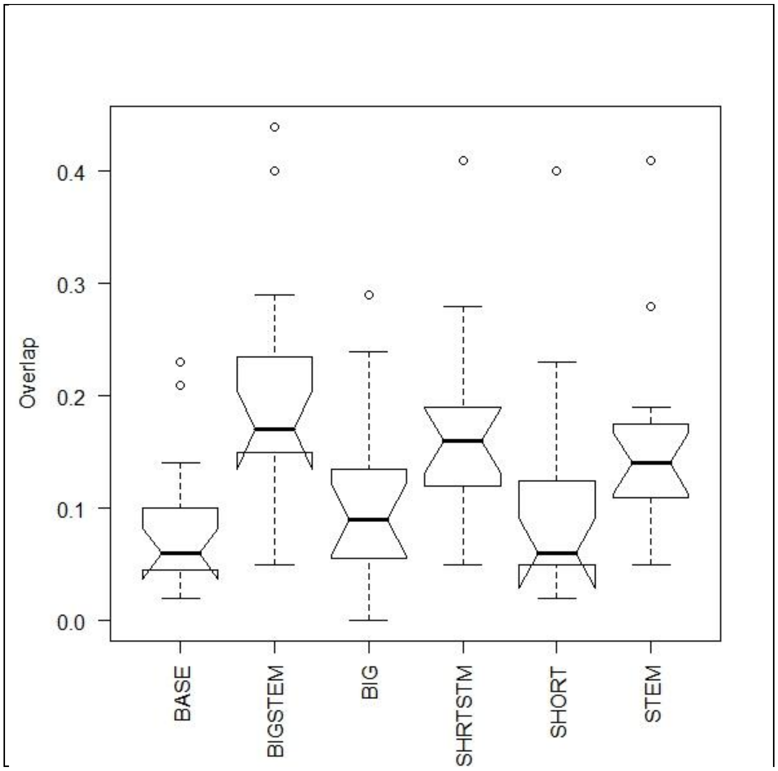


Figure 11 Prefiltering Overlap Results after ED

4.1.5 Favor Big Documents

Figure 12 displays the overlap scores for our filtering methods when the favor big documents metric was applied. Overall the results are similar with respect to each other. This metric is the only metric used that did result in any outliers. The outlier's discussion below suggests the outliers are generally high because the subjects used similar term frequency favoring metrics to create their vocabularies. If true, then it is possible that a portioning scheme such as BD may be different enough from the other term frequency schemes to produce drastically different results. This does not explain why ED has outliers though. The reason for this occurrence is not understood at this time.

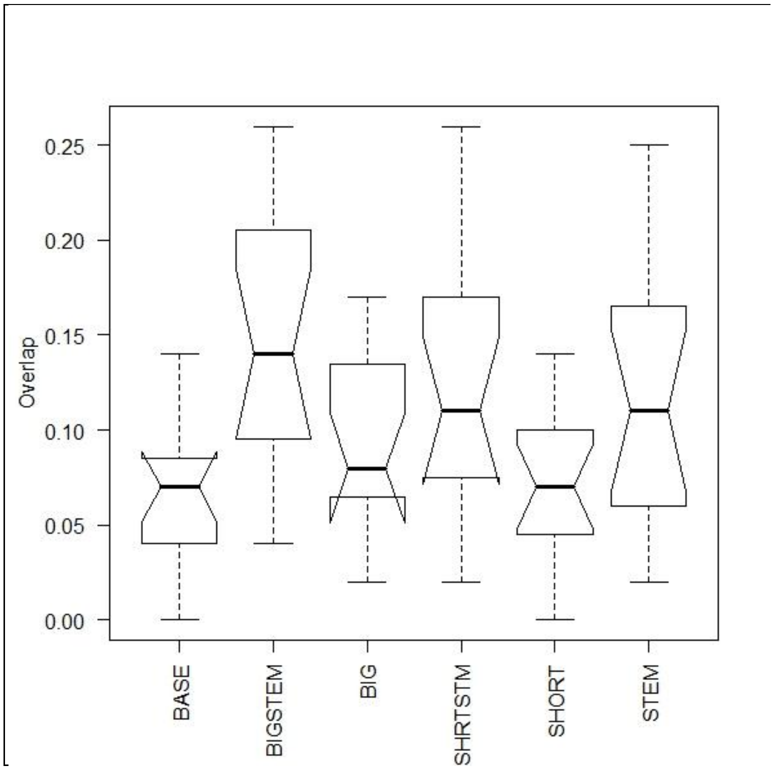


Figure 12 Prefiltering Overlap Results after BD

4.1.6 Normalized Term Frequency

Figure 13 displays the overlap scores for our filtering methods when the normalized term frequency metric was applied. Stemming and stemming combined with stopword removal impacted overlap significantly over the baseline. The short stop list did not impact overlap at all over the baseline. The distribution of the baseline is small because of the consistent error produced by stopwords.

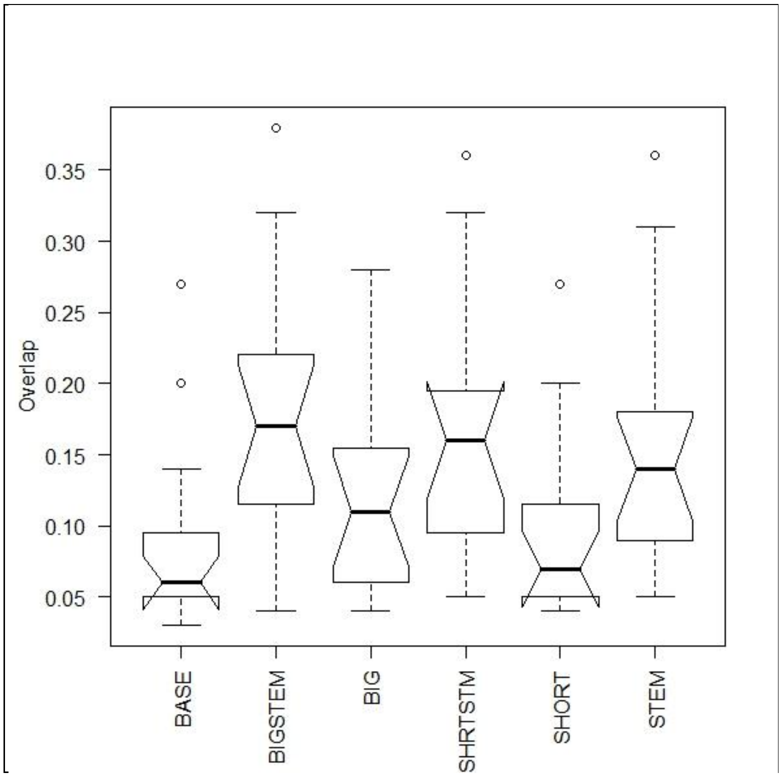


Figure 13 Prefiltering Overlap Results after NTF

4.1.7 Document Relativized

Figure 14 displays the overlap scores for our filtering methods when the document relativized metric was applied. The results are similar to other metrics in this paper. The stemming and stemming combined with stopword removal filters provide impact on the baseline, while stopword removal alone was not statistically significantly improved.

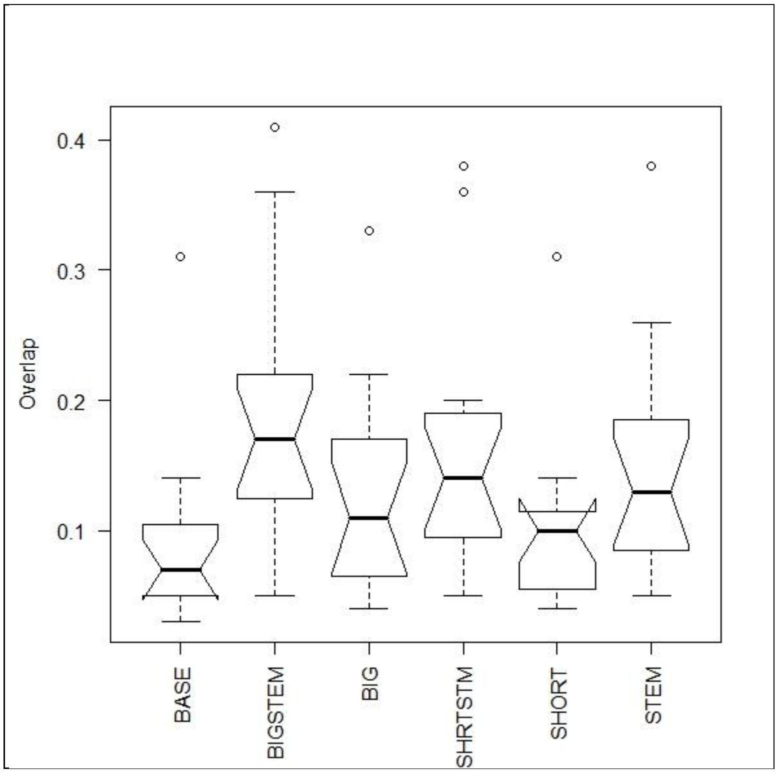


Figure 14 Prefiltering Overlap Results after DR

4.1.8 Document Relativized minus Term Frequency Average

Figure 15 displays the overlap scores for our filtering methods when the document relativized minus term frequency average metric was applied. The impact of stemming and stopword removal is not evident when this metric is applied. This occurs because the metric produces very overlap values no matter which pre-filtering method is used. The distribution is small in all cases because vocabulary selection was poor in all cases.

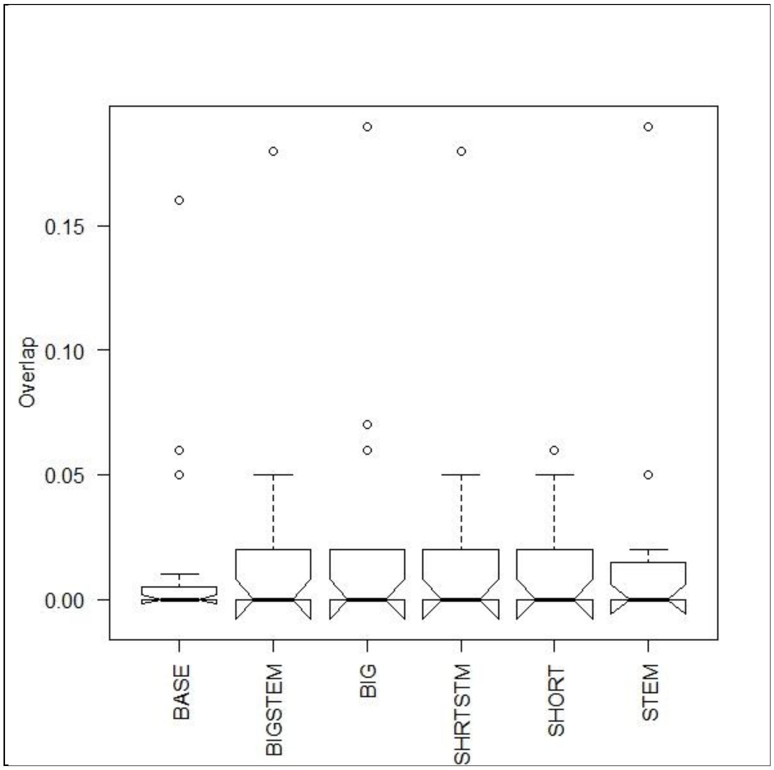


Figure 15 Prefiltering Overlap Results after DRDA

4.1.9 Corpus Relativized

Figure 16 displays the overlap scores for our filtering methods when the corpus relativized metric was applied. The effects of stemming and stopword removal on the CR metric are relatively similar to the effect that they have on other metrics. Stopword removal combined with stemming was significantly better than the baseline, and stopword removal and stemming alone were only moderately better.

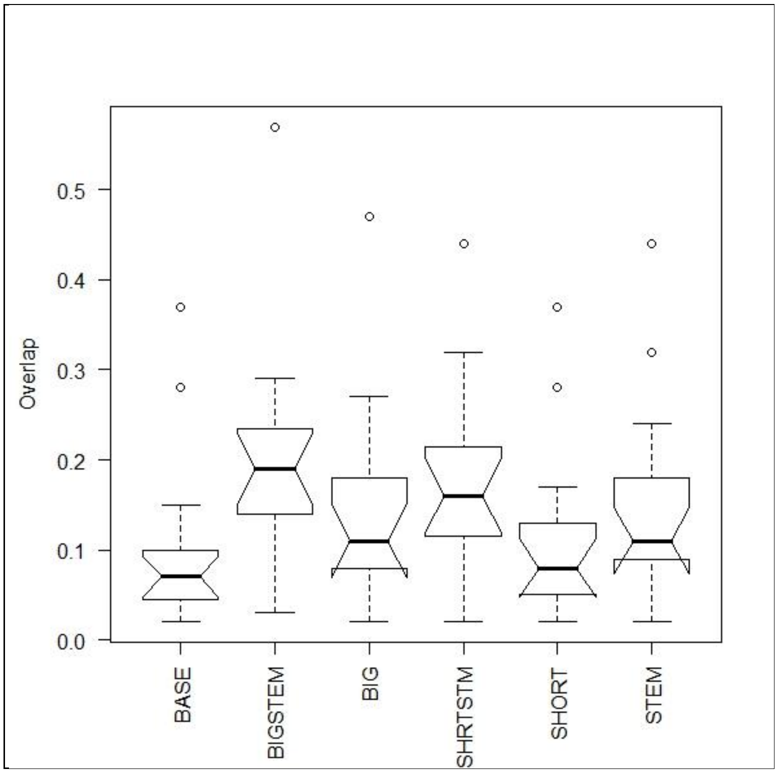


Figure 16 Prefiltering Overlap Results after CR

4.1.10 Corpus Relativized minus Term Frequency Average

Figure 17 displays the overlap scores for our filtering methods when the corpus relativized minus term frequency average metric was applied. The scale of the figure below is a little misleading. The results of this metric are very similar to DRDA which shows very little difference between the filtering methods. This figure is on a scale one tenth of the other scales, so the differences are more evident. Overall, the effects of stemming and stopword removal affect this metric very little because the metric is very poor. The combination of stopword removal and stemming is significantly better, but still not a good performer.

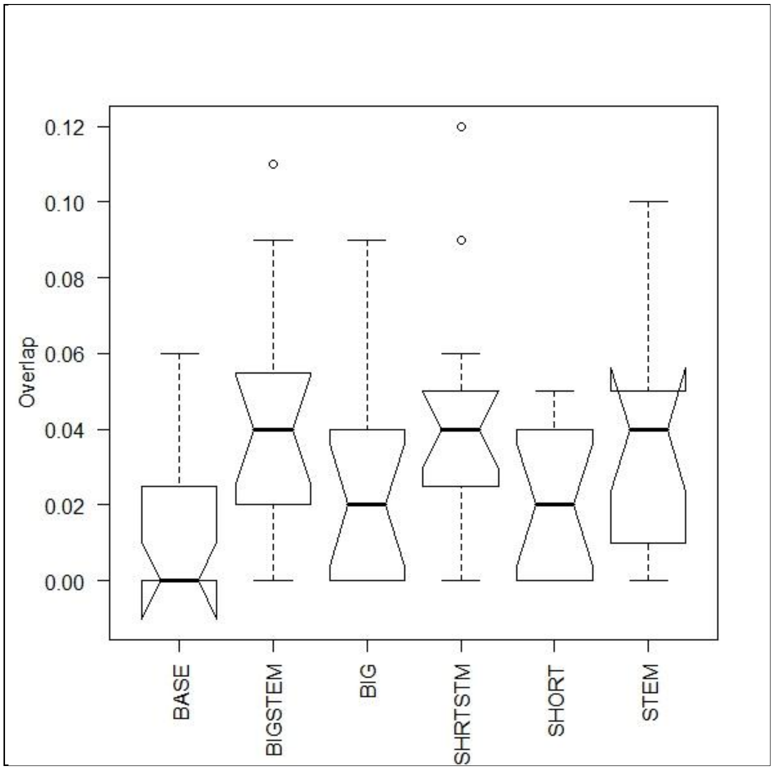


Figure 17 Prefiltering Overlap Results after CRDA

4.1.11 Term Frequency – Inverse Document Frequency

Figure 18 displays the overlap scores for our filtering methods when the inverse document frequency metric was applied. Generally speaking, the stemming filter has increased the mean overlap more than that of the two stopword removal filters. This difference, albeit a small one, is even less evident with this metric. The stemming makes variants similar to one another. This increases the likelihood that a term appears in other documents, decreasing the value of IDF.

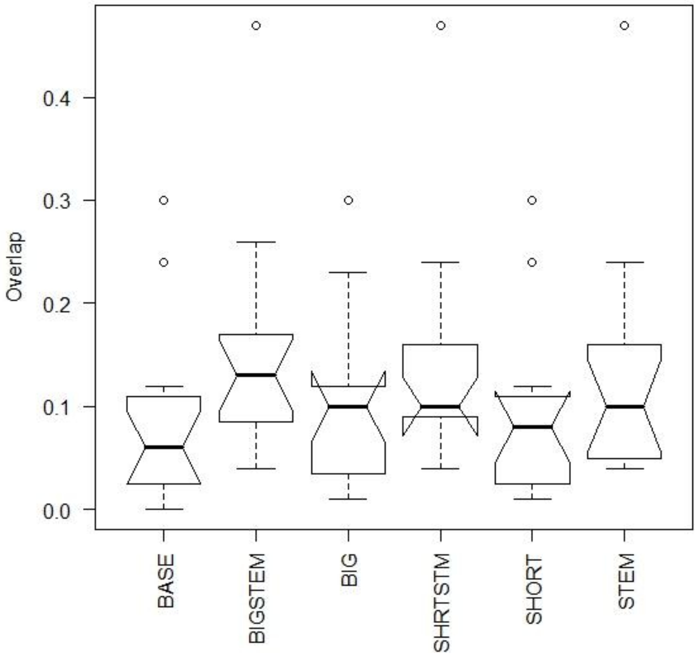


Figure 18 Prefiltering Overlap Results after TFIDF

4.1.12 Term Frequency – Logged Inverse Document Frequency

Figure 19 displays the overlap scores for our filtering methods when the logged inverse document frequency metric was applied. The impact of stemming and stop word removal is not nearly as evident when this metric is applied. This occurs because the metric is very poor and overwhelms the effect of stemming and stop word removal. None of the results are statistically significantly better or worse than any other result.

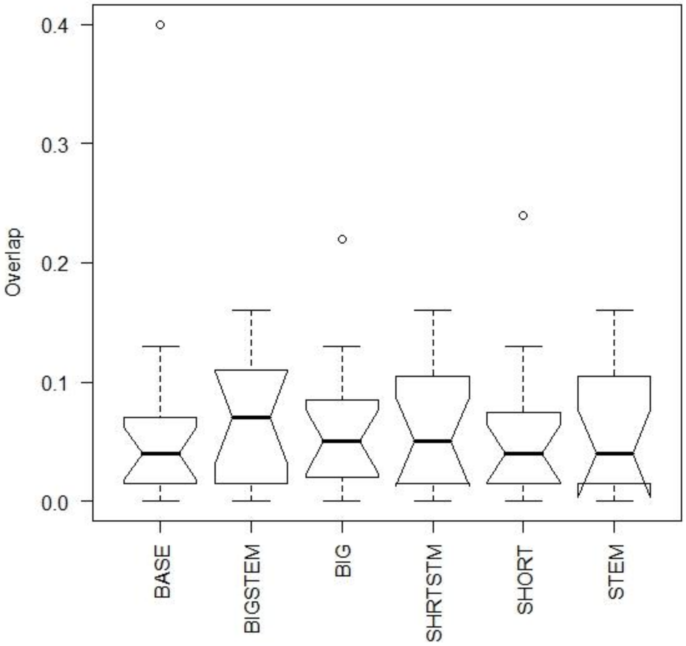


Figure 19 Prefiltering Overlap Results after LTFIDF

4.1.13 Distribution Consensus

Figure 20 displays the overlap scores for our filtering methods when the distribution consensus metric was applied. The impact of stemming and stopword removal is not evident when this metric is applied. This occurs because the metric is very poor and overwhelms the effect of stemming and stopword removal. The distribution is very low because the results are consistently zero.

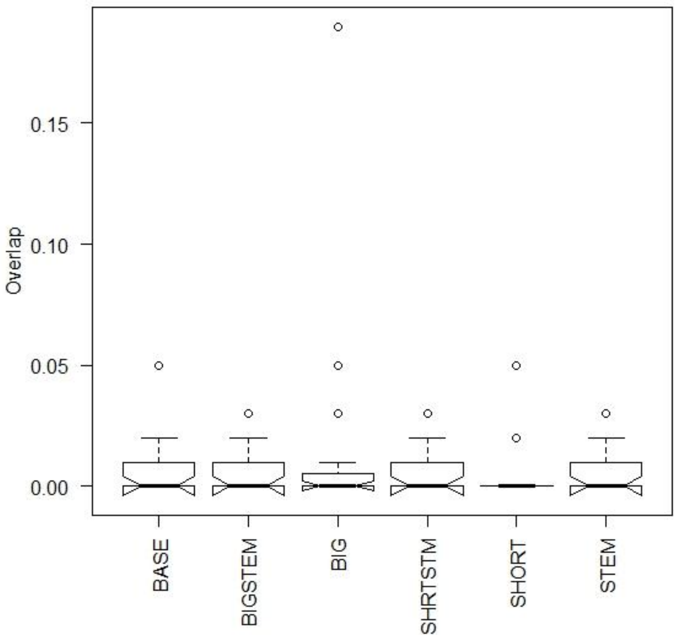


Figure 20 Prefiltering Overlap Results after DC

4.1.14 Binary Consensus

Figure 21 displays the overlap scores for our filtering methods when the binary consensus metric was applied. The TFIDF metric can be contrasted with the results in this table. We see very little impact of stopword removal when binary consensus is used, but large increases in the mean overlap score when stemming is applied to the non-stemming counterpart. This metric rewards terms that are in multiple documents, while TFIDF penalizes words for the same reason. While the results are not significant statistically speaking, this result does look promising. Stemming decreases the number of unique types in the corpus and increases the probability that a term will be in multiple documents. The distributions in each experiment are rather high in each filtering case.

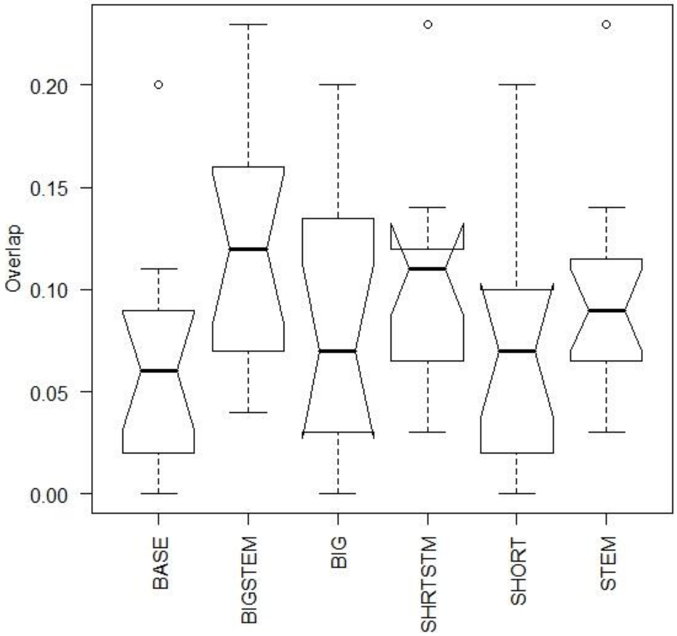


Figure 21 Prefiltering Overlap Results after BC

4.2 Filtering Metrics

While it is interesting to see how pre-filtering affected our results, another goal of this paper is to discuss the effects of filtering metrics based on term frequency statistics. This section of the paper will discuss the results of each metric when trying to achieve high overlap.

4.2.1 Baseline results

Figure 22 displays the overlap scores for our statistical metrics when no pre-filtering is performed. It is clear from the table that DRDA, CRDA, and DC are poor performers compared to the rest of the field. A primary component of the DRDA and CRDA metrics is subtracting average term frequency from a normalized frequency. In essence this penalizes terms that occur frequently within the corpus. This clearly has adverse effects on the results and should not be considered as a means to extracting terms. The DC metric's poor performance occurs when a term is found in one document one time, and is not found in other documents. This provides a low standard deviation, especially in a large corpus, and rewards infrequent terms. Terms that appear in all documents but with very different frequencies are penalized greatly.

The LTFIDF metrics was also another poor performer, although it received higher overlap scores than the three mentioned above. This result is slightly puzzling and may need further investigation. In term weighting methods for information retrieval the logged version of IDF is used more frequently than the non-logged version, yet here we see poorer results. A possible explanation for this is the information retrieval typically deals with very large corpora where logging document numbers is more drastic. Corpus term frequency and TFIDF did relatively well though, so lessening the impact of IDF should make the results look more like TF, and clearly it does not. An explanation for this could also be that because the corpora were so small that logging most corpus sizes produced small decimal values which greatly dampened term frequency. This dampening effect caused the term scores to be more closely lumped, which may in turn produce more errors.

The table also displays many outliers. Almost all of the outliers on this table and on all tables to follow belong to two subjects. The two subjects will be examined in section 4.3. The metrics DRDA and DC also produced outliers that did not belong to the two consistent outliers. This happens because small changes in the vocabulary selected can produce overlap outliers. The mean and distributions of these metrics are so low that subjects with only 2 or 3 correctly extracted terms scored well above the mean.

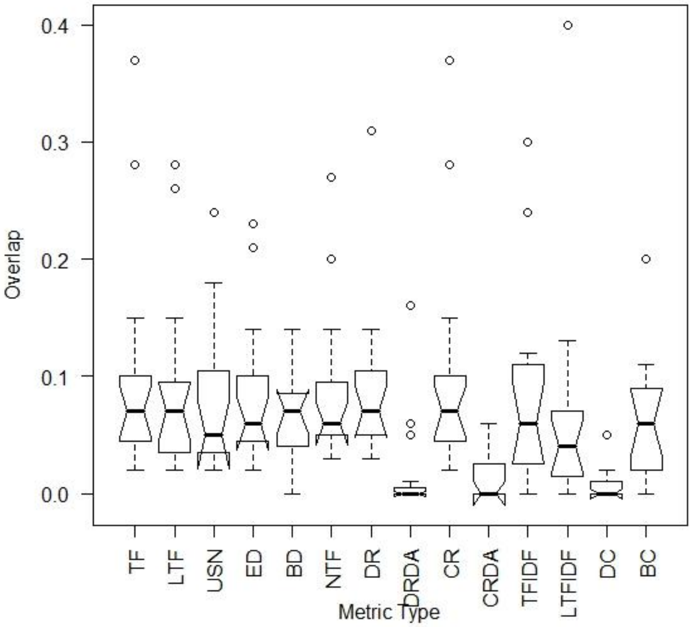


Figure 22 Metric Overlap Results for the Baseline

4.2.2 Stemming results

Figure 23 displays the overlap scores for our statistical metrics when stemming is applied to the tokens. The stemming results produced very similar results to the baseline. Although the means have risen, the same scores performed relatively as well as each other. The score for

LTFIDF has actually become more like the other three poor performers. One metric that performed more consistently than its baseline version is BC. This happens because terms that appear as different variants within different documents many receive low binary distribution scores when not associated with each other. When the terms are stemmed, they essentially become the same word, and their occurrence across the corpus becomes apparent.

Our methodology section mentions that TF and CR would receive similar if not identical scores. The table for stemming shows that the distributions are different. This implies that the words chosen by both metrics are not the same. Careful analysis of the scoring of the words shows that the metrics are indeed scoring words similar to each other. The difference between the extracted vocabularies comes from a sorting difference. The algorithms sort the terms by metric score alone and ties are not sorted in any particular order. If multiple terms exist with the same metric score when deciding on the last terms to enter the extracted vocabulary, some scores with equal weights may be omitted because of our hard cap of V vocabulary words. In some cases the correct terms were high in the sorted list and made the automated vocabulary, and in other cases they were on the low end of the sort and did not.

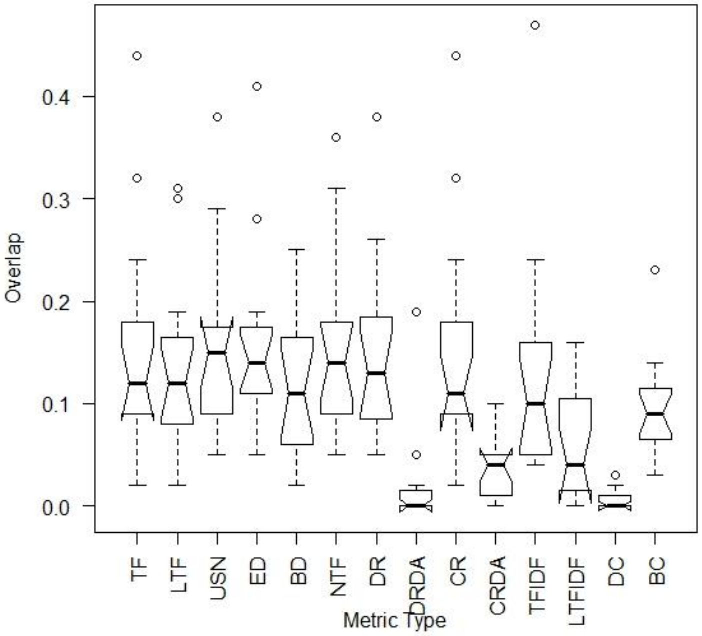


Figure 23 Metric Overlap Results after Stemming

4.2.3 Short stop list results

Figure 24 displays the overlap scores for our statistical metrics when no stemming is applied to the tokens but a small set of stopwords are removed. The table reemphasizes that DRDA, CRDA, LTFIDF, and DC are poor performers. Document relativization shows some promise in this table, but not significant enough to claim it is better than the other metrics. The outliers in the table also belong to the same outliers in other tables.

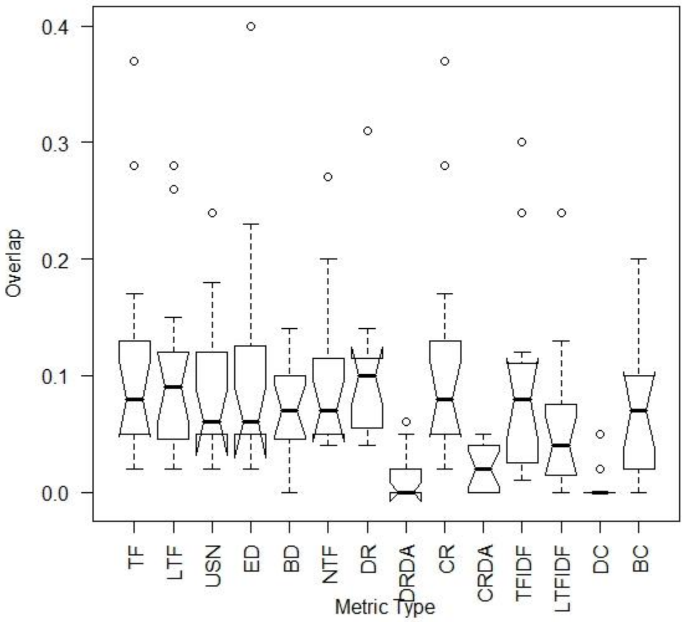


Figure 24 Metric Overlap Results after applying Short Stop List

4.2.4 Big stop list results

Figure 25 displays the overlap scores for our statistical metrics when no stemming is applied to the tokens but a large set of stopwords are removed. The same poor performers and outliers occur in this test as well. Other outliers that occur are the result of chance in poorly performing metrics.

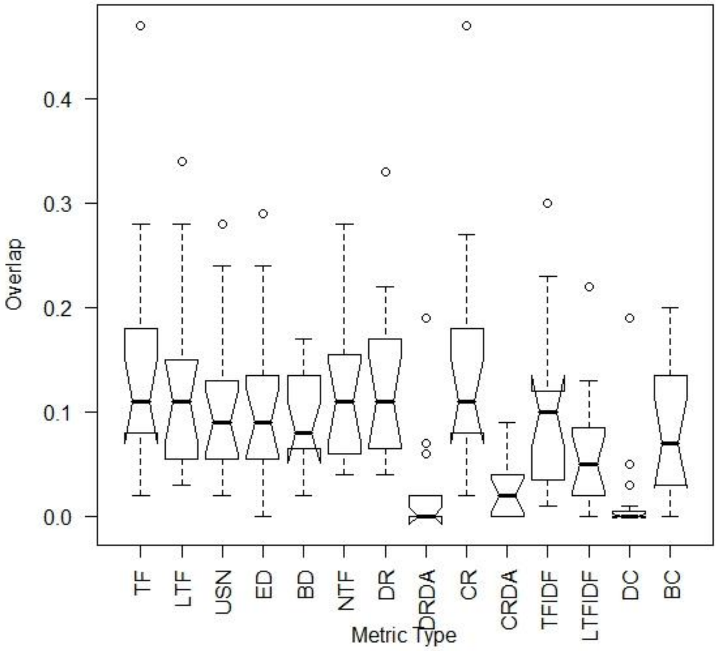


Figure 25 Metric Overlap Results after applying Big Stop List

4.2.5 Short stop list and stemming results

Figure 26 displays the overlap scores for our statistical metrics when stemming is applied to the tokens and a small set of stopwords are removed. The mean overlap for the metrics has been improved with the combination of stemming and stop-word removal, except for the poor performing metrics. Although the metrics BD and TFIDF are not significantly worse, their means have not improved as much as the other metrics. While TFIDF is a better metric for information retrieval than TF, the fact that its scores do not improve over pure TF in these tests may indicate that IDF is a value specifically useful for indexing. The value of IDF is intended to set words out as special to a subset of documents within a general corpus of many domains. Our corpora are specific to one domain, so we are not interested in isolating a subset of documents, or extracting words that belong to just one document.

The metric ED outperforms BD in this experiment, as well as in four of the other five experiments. Many subjects chose papers that represent different algorithms pertaining to their domain of choice. Conflation was a popular domain by the subjects. Many subjects chose to look at Porter’s algorithm, the Paice/Husk algorithm, and the Lovins algorithm. Porter’s algorithm uses a unique metric called measure that the other algorithms do not use. Six out of seven of the expert vocabularies chose this word as being important even though it only appears in one of the documents of the corpus. While the ED metric did not pull out measure in many cases, it did pull out ‘m’, which is a variable name associated with measure in the Porter papers. This indicates that placing some form of distinct emphasis on each paper is warranted, although the degree of importance is not known.

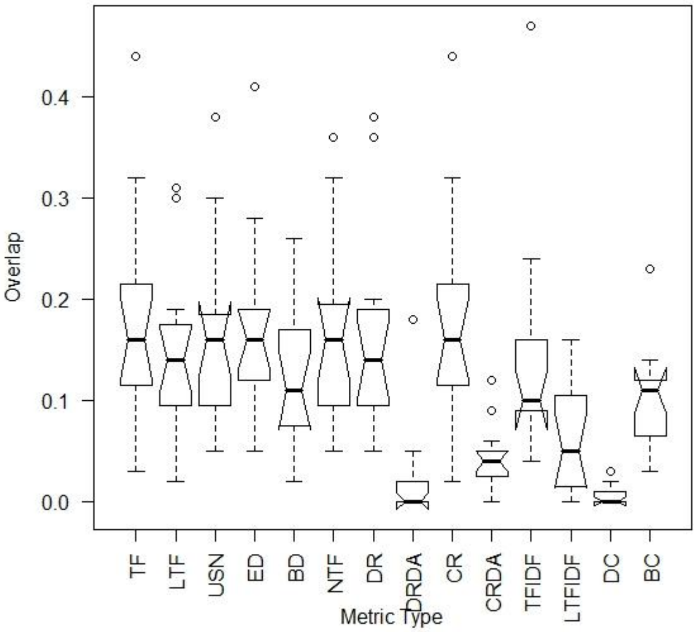


Figure 26 Metric Overlap Results after applying Short Stop List and Stemming

4.2.6 Big stop list and stemming results

Figure 27 displays the overlap scores for our statistical metrics when stemming is applied to the tokens and a large set of stopwords are removed. This filtering method provides the highest means for most metrics, approaching scores of 20 percent. The evidence still points to DRDA, CRDA, LTFIDF, and DC being poor metrics. We still see the same outliers occur.

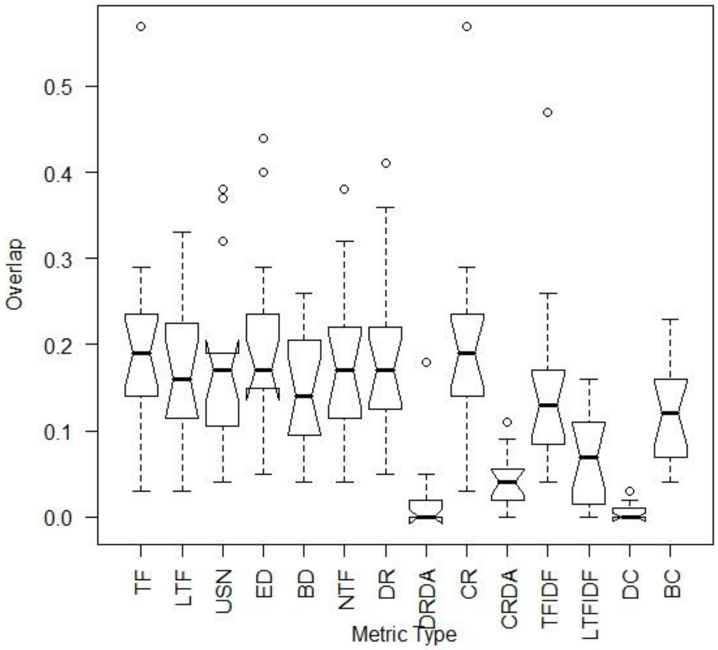


Figure 27 Metric Overlap Results after applying Big Stop List and Stemming

4.3 Consistent outliers

The first consistent outlier was subject 12. The only comment about the expert vocabulary in the domain book is that stemming was not used. The vocabulary terms in the domain book were presented with scores, leading us to believe that some type of statistic was computed to derive a vocabulary. Tokens that would typically be removed by humans, such as “1” and either “used” or “using” were included in the final vocabulary. This indicates that some metric was applied to the corpus, and few, if any, human alterations occurred. If the statistic that was used was term frequency related, that would account for the high overlap scores. Any differences in vocabularies could be accounted for by definitions of allowable tokens, pre-processing of data, and variation of the metric used.

Another possible reason that this subject’s overlap score was consistently high could be that the corpus used was heavily text oriented and did not include source code, pictures, or graphs. Source code and diagrams in other subjects created noise in other subjects’ results. Short variable names from source code and abbreviations from diagrams made their way into automatically constructed vocabularies, and these terms did not belong in the expert vocabulary.

The second consistent outlier was subject 9. This subject studied application analysis. It is less obvious why the results were better for this subject than many of the others, but it is most likely due to similar reasons as the other outlier. The subject stated that the domain vocabulary was created in less than one hour. This is unusual as many other subjects indicated it took longer than one hour, even though they had significantly less words in their domain vocabulary. This indicates that some quick approach was used. Another note about the domain vocabulary is that it stood out from the other vocabularies in that it used multiple tenses and capitalization in each term. It incorporated present and past tense words, singular and plural words, and upper and lower case words. We can infer that the vocabulary was quickly extracted out of the corpora, most likely in an automated fashion. If quickness of vocabulary creation and inconsistent tense are indications of automatically constructing a vocabulary, then this would put this subject into similar circumstances as the previous outlier. Unfortunately this paper cannot judge the validity of their vocabularies, so we cannot state whether the high metric scores for these subjects are the result of poor vocabularies or hints that these metrics can perform well in certain circumstances.

5. Conclusions

Section 5.1 discusses the implications of our findings with respect to the domain analysis community. Future work in this area is discussed in section 5.2.

5.1 Domain Analysis Relevance

Domain Analysis has a different set of needs than the typical term extraction user and unfortunately cannot directly use some current methods without term loss. This is mainly true because current methods extract only nouns which are vital to domain analysis. While the metrics in this study do show that some algorithms are better than others for extracting terms, a success domain engineer will need to use more than these statistics. They will still need to rely on subject matter expertise, intuition, and some guidance from these statistics.

The results of our experiment clearly demonstrated that filtering methods play an important role when extracting terms from a corpus. Stemming words into related groups and removing common, generic terms like prepositions will provide for a more accurate vocabulary. Experiments that used these methods performed better than experiments without them. In addition, a combination of the two even worked better.

While the overlap scores were not very high, there were some clear differences between different metrics. Metrics that penalized high term frequency, like CRDA and DRDA, both scored very poorly. This result confirms previous works that suggests term frequency is one of the most important factors in term extraction. Distribution consensus also scored poorly because it rewarded terms that were consistently absent. This suggests that consensus should be based on minimum frequencies.

The overlap scores for any metric and filtering combination rarely approached 30 percent, leading us to initially caution domain analysts from completely relying on automation for vocabulary selection. Although the automation cannot be relied on for good final results, we believe there are certain metric and filtering techniques that are adequate starting places for analysts. In most cases, the combination of stopword removal and stemming provided the best filter for high overlap scores. These two methods should be employed in all cases, although care must be given when choosing a stopword list. There are words in some lists that may be appropriate for certain domains and the list should be scanned ahead of time if possible. When the big stop list was applied and stemming was applied, term frequency had the highest median overlap. This was not statistically significant. Many of the metrics related to term frequency like normalized term frequency, document relativized term frequency, and unsummed term frequency performed well in several experiments. It is clear that term frequency is a positive indicator of domain relevance, and therefore should be used as a starting place for building a vocabulary.

5.2 Future work

There are many areas of this experiment that can be improved upon. Section 5.2.1 will discuss the samples. Sections 5.2.2 will discuss corpus preparation. Section 5.2.3 will discuss other possible metrics.

5.2.1 The Sample Set

The students who created these projects did work hard indeed on their class projects, but are not necessarily subject matter experts in the field they chose to study. They were also not expert domain analysts who may have trained how to select a proper vocabulary. The vocabularies that they created for even one domain, such as conflation, were quite varied over several subjects' work. This is also a result of human perspective. Using multiple experts to select one vocabulary would be beneficial as it would show consensus. Even with groups selected vocabularies, multiple groups would likely created multiple vocabularies. It may be beneficial to create an index of similarity.

We believe that several subjects were not really aware of what it meant to select a proper vocabulary. The vocabularies selected for several subjects came from their facet tables rather than the actual vocabulary. This was done because their selected vocabulary contained every unique term in their corpus, excluding stopwords. Subjects who completed their work more recently seemed to do a better job in this area, but this fact should reinforce the need to pick projects that were completed well.

Proper projects and vocabularies that have the consensus of multiple subjects would provide a better sample to work with. In addition, the corpuses used to describe the domain should also be subject to group consensus. The spectrum of documents used for each corpus was very wide and it would be interesting to see if certain document types led to a better vocabulary. Some examples of document types were research papers, system design documents, web pages, wiki pages, and help files.

5.2.2 Document preparation

When going through the words that were selected out of each automatic extraction run, we found many words that could be considered noise. Code fragments and pictures should be taken out of the text before tokenization. Comments imbedded in the code may be useful phrases to include in the analysis, but the cost of the variable names and their often high frequencies tend to lower overlap scores.

One of the subjects mentioned in his domain book that he did not do frequency analysis on his scanned image file because of the high error rates of optical scanning recognition software. While this paper cannot confirm high error rates, it may be prudent to avoid a possible noise source such as this. This paper did use the scanned images, and the subject's corpus was prone to low overlap scores.

Domain analysis is concerned with objects and methods that are vital to a system functioning. These objects and methods are almost always types, or classes of objects and not the objects themselves. It is for this reason that we think that entity filtering may have a positive impact on overlap, much like stopword removal. Entity filtering is the process of removing proper nouns from a document, such as names and places.

The metrics we chose also relied on randomness to break ties. In the real world, systems would choose terms based on their weights being greater than some threshold. Since we did not have a known threshold for our metrics, we used a set limit of terms to select. In case of ties, we chose the terms randomly. In the future, we could revisit the tests and remove the arbitrary decisions by alphabetizing the tied scores.

5.2.3 Other metrics

It may be interesting to revisit some of the metrics that performed poorly. For instance, the distribution consensus metric performed poorly because it rewarded consensus. The distribution was low even if the term frequency was also low. Combining the minimum frequency aspect of binary consensus with distribution consensus may have a positive impact on overlap.

The metrics that subtract average term frequency performed poorly. These metrics were not found in literature pertaining to entity extraction. However, there was an alternate version of average term frequency that may be useful. We used an average term frequency that measured the average number of times one term appeared across all documents. Some entity extraction papers talk about the average term frequency within a document. This is called verbosity. Papers that are long because they repeat the same information a lot are called verbose (Singhal, Salton, Mitra, & Buckley, 1995). It has been suggested that these documents be penalized. Future work could be done to integrate verbosity.

This paper also selected a certain number of terms based on the size of the subject’s vocabulary. This is obviously not possible when performing domain analysis, as the vocabulary is not given ahead of time. Therefore some work should be done to quantify typical vocabulary sizes, or establish minimum thresholds to metrics. Not only would this be more realistic to the intended use of extraction, but it would also aid in avoiding tie-breaking situations that are broken by qualitative means.

Bibliography

Arppe, A. (1995). Term Extraction from Unrestricted Text. Helsinki.

Baeza-Yates, R., & Ribiero-Neto, B. (1999). Modern Information Retrieval. Addison-Wesley.

Bourigault, D. (1993). An endogenous corpus-based method for structural noun phrase disambiguation. Proceedings of the 6th Conference of the European Chapter of the Association for Computational Linguistics, (pp. 81-86). Utrecht, the Nederlands.

Bourigault, D. (1995). LEXTER, a Terminology Extraction Software for Knowledge Acquisition from Texts. In the Proceedings of the 9th Knowledge Acquisition for Knowledge Based System Workshop. Banff, Canada.

Bourigault, D., & Jacquemin, C. (1999). Term extraction + term clustering: an integrated platform for computer-aided terminology. European Chapter Meeting of the ACL (pp. 15-22). Bergen, Norway: ACM.

Church, K., & Dagan, I. (1994). Termight: Identifying and Translating Technical Terminology. Fourth Conference on Applied Language Processing (pp. 34-40). Stuttgart, Germany: Association for Computational Linguistics.

Cohen, J. D. (1995). Highlights: Language- and Domain-Independent Automatic Indexing Terms for Abstracting. Journal of the American Society for Information , 162-174.

Collier, N., Nobata, C., & Tsujii, J. (2001). Automatic acquisition and classification of terminology using a tagged corpus in the molecular biology domain. Terminology , 239-257.

Crawley, M. J. (2007). The R Book. West Sussex, England: Wiley.

Daille, B. (1994). Study and implementation of combined techniques for Automatic Extraction of Terminology. The Balancing Act: Combining Symbolic and Statistical Approaches to Language , 29-36.

Evans, D. A., & Zhai, C. (1996). Noun-Phrase Analysis in Unrestricted Text for Information Retrieval. Proceedings of the 34th annual meeting on Association for Computational Linguistics (pp. 17-24). Santa Cruz, CA: ACM.

Fano, R. M. (1961). Transmission of Information: A Statistical Theory of Communications. MIT Press.

Frakes, W. (2000). A Method for Bounding Domains. IASTED International Conference Software Engineering and Applications. Las Vegas, NV.

Frakes, W. B. (1992). Stemming Algorithms. In W. B. Frakes, & R. (. Baeza-Yates, Information Retrieval: Data Structures and Algorithms (pp. 131-160). Englewood Cliffs, NJ: Prentice Hall.

Frakes, W. B., & Kang, K. (2005). Software Reuse Research: Status and Future. IEEE Transactions on Software Engineering , 529-536.

Frakes, W., Prieto-Diaz, R., & Fox, C. (1998). DARE: Domain Analysis and Reuse Environment. Annals of Software Engineering , 125-141.

Frakes, W., Prieto-Diaz, R., & Fox, C. (1997). DARE-COTS: A Domain Analysis Support Tool. XVII International Conference of the Chilean Computer Society (pp. 73-77). Valparaiso, Chile: IEEE Computer Society Press.

Grossman, D., & Frieder, O. (1998). Information Retrieval: Algorithms and Heuristics. Kluwer Academic Press.

Harris, C., & Frakes, W. (2006). Domain Engineering: An Empirical Study.

Heid, U. (1998). A linguistic bootstrapping approach to the extraction of term candidates. Terminology , 161-181.

Hersh, W. R. (1996). Empirical, Automated Vocabulary Discovery Using Large Text Corpora and Advanced Natural Language Processing Tools. AMIA Annual Fall Symposium, (pp. 159-163). Portland.

Jacquemin, C., & Bourigault, D. (2005). Term Extraction and Automatic Indexing. In R. Mitkov, The Oxford Handbook of Computational Linguistics (pp. 599-615). Oxford: Oxford University Press.

Justeson, J., & Katz, S. (1993). Technical Terminology: Some Linguistic Properties and an Algorithm for Identification in Text. In Natural Language Engineering (pp. 9-27). Almadem : IBM Research Division.

K, A. (2008, August 22). Retrieved September 2008, from https://projects.commandprompt.com/public/replicator/browser/trunk/src/backend/snowball/stopwords/english.stop?rev=1889

Kang, K., Cohen, S., Hess, J., Novak, W., & Peterson, A. (1990). Feature-Oriented Domain Analysis (FODA) Feasibility Study. Pittsburgh.

Kang, K., Kim, S., Lee, J., Kim, K., & Shin, E. (1998). FORM: A Feature Oriented Reuse Method with Domain-Specific Reference Architectures. Annals of Software Engineering .

Kantrowitz, M. (1999). Patent No. PCT/US1999/025686. US.

Klavans, J. L., & Muresan, S. (2000). DEFINDER: Rule-Based Methods for the Extraction of Medical Terminology and their Associated Definitions from On-line Text. Proceedings of the AMIA Annual Symposium. Los Angeles.

L., K., Y., P., T., F., Y., D., Y., D., & T., C. (2004). Glossary extraction and utilization in the information search and delivery system for IBM Technical Support. IBM Systems Journal , 546-563.

Luhn, H. P. (1958). The Automatic Creation of Literature Abstracts. IBM Journal of Research and Development , 157-165.

McIlroy, M. D. (1968). Mass produced software components. Proceedings of NATO Software Engineering Conference, (pp. 138-155). Garmisch, Germany.

Medelyan, O. (2006). Thesaurus based automatic keyphrase indexing. Proceedings of the 6th ACM/IEEE-CS joint conference on Digital Libraries (pp. 296-297). ACM Press.

Muresan, S., Popper, S. D., T., D. P., & Klavans, J. L. (2003). Building a terminological database from heterogeneous definitional sources. Proceedings of the 2003 annual national conference on Digital government research (pp. 1-4). Boston: Digital Government Society of North America.

Na, S.-H. N., Kang, I.-S., & Lee, J.-H. (2008). Improving Term Frequency Normalization for Multi-topical Documents and Application to Language Modeling Approaches. In Advances in Information Retrieval (pp. 382-393). Berlin: SpringerLink.

Neighbors, J. (1980). Software construction using components. Irvine: University of California.

Noreault, T., McGill, M., & Koll, M. (1980). A performance evaluation of similarity measures, document term weighting schemes and representations in a Boolean environment. Proceedings of the 3rd annual ACM conference on Research and development in information retrieval (pp. 57-76). Cambridge, England: Butterworth and Co.

Park, Y., Byrd, R. J., & Boguraev, B. K. (2002). Automatic glossary extraction: beyond terminology identification. International Conference On Computational Linguistics (pp. 1-7). Taipei, Taiwan: ACM.

Parnas, D. L. (1976, March). On the design and development of program families. IEEE Transactions on Software Engineering , pp. 1-9.

Prieto-Diaz, R. (2003). A faceted approach to building ontologies. IEEE International Conference on Information Reuse and Integration, (pp. 458-465).

Rydberg-Cox, J. (June 2002). Keyword Extraction from Ancient Greek Literary Texts. Literary and Linguistic Computing , 231-244.

Salton, G., & McGill, M. J. (1983). Introduction to Modern Information Retrieval. New York: McGraw Hill.

Sclano, F., & Velardi, P. (2007). TermExtractor: a Web Application to Learn the Shared Terminology of Emergent Web Communities. In R. J. Gonçalves, J. P. Müller, K. Mertins, & M. Zelm, Enterprise Interoperability II (pp. 287-290). London: Springer.

Singhal, A., Salton, G., Mitra, M., & Buckley, C. (1995). Document Length Normalization. Ithaca, NY: Cornell University.

Smadja, F., McKeown, K., & Hatzivassiloglou, V. (1996). Translating collocations for bilingual lexicons: a statistical approach. Computational Linguistics , 1-38.

Velardi, P., Missikoff, M., & Basili, R. (2001). Identification of relevant terms to support the construction of domain ontologies. Proceedings of the workshop on Human Language Technology and Knowledge Management. Toulouse, France: Association for Computational Linguistics.

Voutilainen, A. (1993). NPtool. A detector of English noun phrases. Proceedings of the Workshop on Very Large Corpora, (pp. 48-57). Columbus.

Wermter, J., & Hahn, U. (2005). Finding New terminology in Very large Corpora. International Conference On Knowledge Capture (pp. 137-144). New York: ACM.

Witschel, H. (2005). Term Extraction and Automatic Indexing: Comparative and Qualitative Comparison of Methods. In Proceedings of Terminology and Knowledge Engineering.

Witten, I. H., Paynter, G. W., Frank, E., Gutwin, C., & Nevill-Manning, C. G. (1999). KEA: Practical automatci keyphrase extraction. Proceedings of Digital Libraries 99 (pp. 254-256). ACM Press.