The Utility of Corpora Comparison for Generating Delete Lists

Geoffrey P. Morgan
July 10, 2017
CMU-ISR-17-109

Institute for Software Research
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Support for this work was provided by the Office of Naval Research (ONR) MURI N0014081186 as well as ONR Minerva – Dynamic Statistical Network Informatics N000141512797. Additional support was provided by the center for Computational Analysis of Social and Organizational Systems and the Institute for Software Research at Carnegie Mellon University. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Office of Naval Research, or the U.S. government.

The Center for Computational Analysis of Social and Organizational Systems
CASOS Technical Report

Keywords: Text Analysis, Corpus Comparison, Delete Lists, TF-IDF

Abstract

Delete Lists are lists of words that have been determined to have little useful meaning for textual analysis. One subset of words that are frequently deleted are stop-words. Stop-Words are textual tokens, such as “and”, “a”, or “the”, that provide structural or grammatical impact to a sentence but do not themselves have significant inherent meaning. Identifying stop-words is a routine process in most text-cleaning applications, but frequently is done via user-maintained word lists. I suggest that the corpora comparison technique I devised for word-score polarization can be used to identify low-value words while preserving the bulk of the text tokens. I will use both known and random draw corpora comparisons for this process. By “known” corpora, I mean corpora drawn from explicit data-sources, the emails of one company and the emails of another, for example. “Random-Draw” corpora are created by drawing document sets at random, and therefore this technique could be applied to any sufficiently large text corpus of interest. I use the ability to identify stop words as a proxy for performance in generating useful delete lists. Random-Draw and Known Corpora Comparison techniques outperform an iteration of TF-IDF (Term Frequency – Inverse Document Frequency), which performs quite poorly on this email data.

Table of Contents

1 Introduction

In textual analysis, word removal is a common cleaning process. Words are usually removed because they have little value to the analyst's goals. Stop words are one such set of low-value words. These stop words, such as "a", "and", "the", and "or", have significant structural and grammatical importance, but do not themselves have or convey intrinsic meaning. Often, words which are generally considered to have low-value are all placed in a single document, called a "delete-word list" or, more simply, a "delete list".

Word removal through delete lists significantly reduces the size and complexity of resulting analysis products, such as semantic networks and linguistic tree structures, making it easier to get useful results from these later analyses.

Delete lists vary from group to group, and these lists are maintained with significant analyst effort. The exact composition of a delete list may depend on the goals of the group, the text medium, and the analysis goals. There may not be one delete list that every analysis group could agree to.

Because it requires significant human effort to maintain delete lists, I offer an automated algorithmic technique for inferring which words are of low-value and can thus be discarded safely. I do this via corpora comparison – taking two sets of documents and identifying terms that do not usefully distinguish the two corpora. This technique should be applicable across various mediums, as long as the two corpora are comparable. I explore this approach with both "known" corpora, where there is a reason to distinguish between the documents, and "random draw" corpora, where document sets are drawn at random. I compare these approaches with a well-regarded token identification approach, TF-IDF (Term Frequency – Inverse Document Frequency) an approach that scores tokens as to their probable value within a set of documents (Salton & Buckley, 1988).

I would expect that the "known" case, where I have two sets of documents of interest would perhaps have better results than the random case. However, the random case is able to leverage many draw iterations, and with enough iterations; performance at the task of identifying stop words actually exceeds the "known" case. Both forms of corpora comparison out-perform TF-IDF at identifying stop words in email.

In the remainder of this chapter, I identify the algorithm used to identify these low-value words, explore the performance from the "known" case, and then identify the additions to the algorithm used to address the random-draw case and report performance on various settings of the variables used in random-draw corpora selection. Finally, I compare known corpora comparison, random-draw corpora comparison, and TF-IDF performance.

2 The Motivating Case: Comparing Two Corpora with Known Differences

This algorithm relies on the odds-ratio of the two corpora in comparison. A token is highly useful and/or interesting if the token is highly distinctive. Essentially, the quantity of interest would answer the question: "if I saw this token, would I immediately know which corpus this word is from?" I use the normalized odds ratio to compare the frequency of the word in each corpus.

2.1 Comparing Two Corpora

I will use the following notation. For each corpora, C, there is a set of tokens, T. Each token, t, appears a certain quantity of times and is noted as the quantity t_C. The sum of these quantities would be noted as T_C. In this case, there are two known corpora, A and G.

The equation, assuming t_A and t_G are both non-zero, ranges from -0.5 to 0.5. If the token appears only in A, the score is .5, if the token appears only in G, then the score is -0.5.

Equation 1. The Transformed Normalized Odds Ratio

$$odds(t) = \left(1 - \left(\frac{1}{\left(\frac{|t_A|}{|T_A|} \Big/ \frac{|t_G|}{|T_G|}\right)}\right)\right) - .5$$

I can also "pre-treat" the corpus by removing words that occur less than a certain number of times; this affects the ultimate distribution of the words. Removing words that occur less than 3 times is often recommended. Histograms of token scores across multiple thresholds are presented in Figure 1 - a normal distribution with a spike at both tails is expected. The effect of cleaning is evident and definitely valuable when moving from no filter to the "3 or more" words filter.

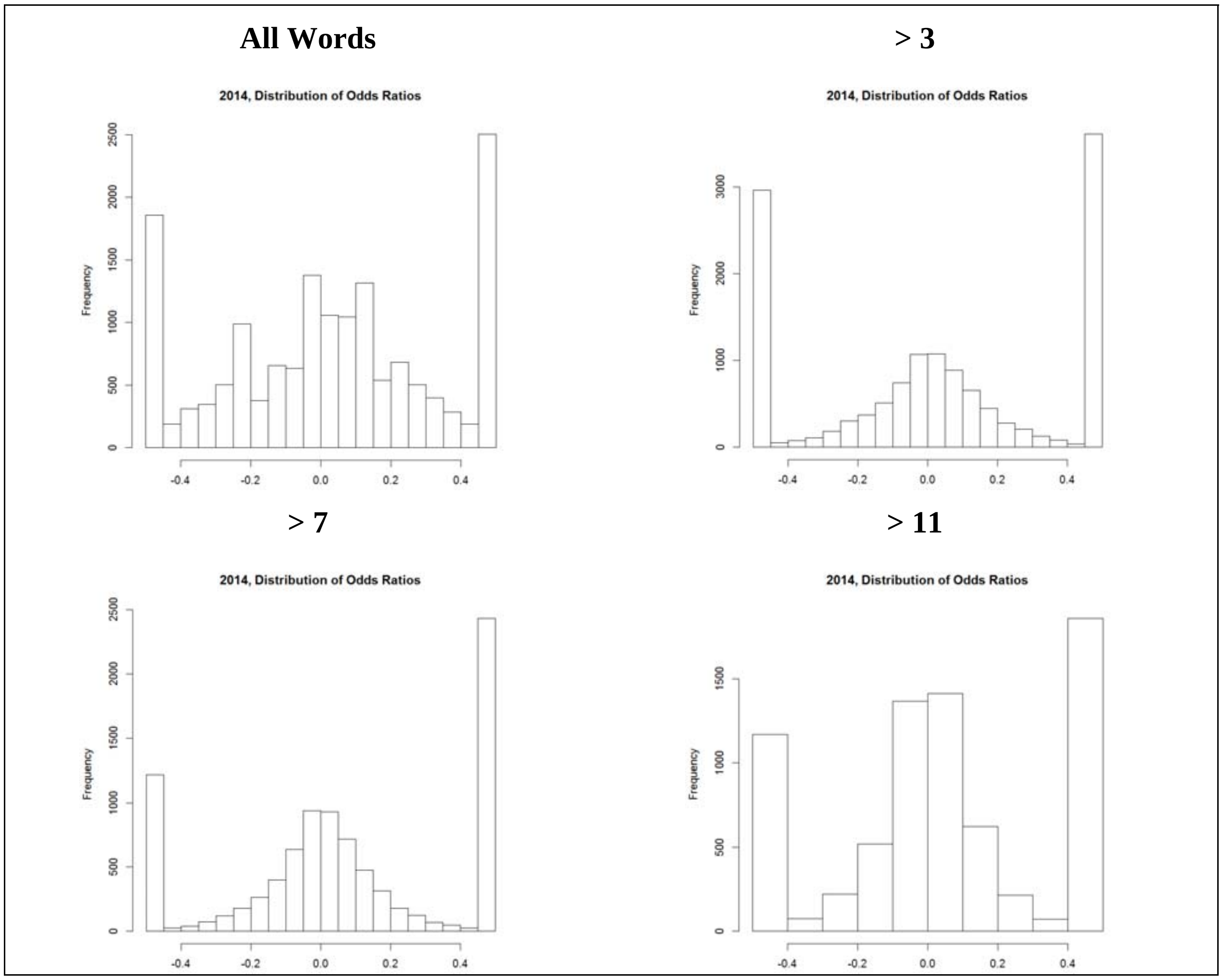

Figure 1. Distributions of token scores across multiple cleaning thresholds

The results in Figure 1 indicate the tested corpus' robustness to word removal. Without special knowledge of the corpora involved, I would expect to see a normal distribution with spikes at both ends. These spikes represent words that only one group or the other uses. Words in each spike usually represent the identities of products, trademarks, or objects related to the company's work. Many words will be used interchangeably across corpora, and this is represented by the bulge at 0.

2.2 "Known" Case Performance

In the "Known" case, I have two sets of emails at similar periods of time. The authors of these emails are from two companies, which are undergoing a horizontal merger. I compare the corpus of emails from Group A against the corpus of emails written by Group G from the same time-period.

If I remove terms based on their odds score, Figure 2 displays the ROC curve from known corpora comparison of Early 2013, other corpora results are similar. The X-Axis is the percentage of all terms removed, while the Y-Axis is percentage of known stop words removed. Random performance is the diagonal – if 30% of the terms are removed at random, 30% of the stop words should be removed as well. The best performance is at the top left of the graph.

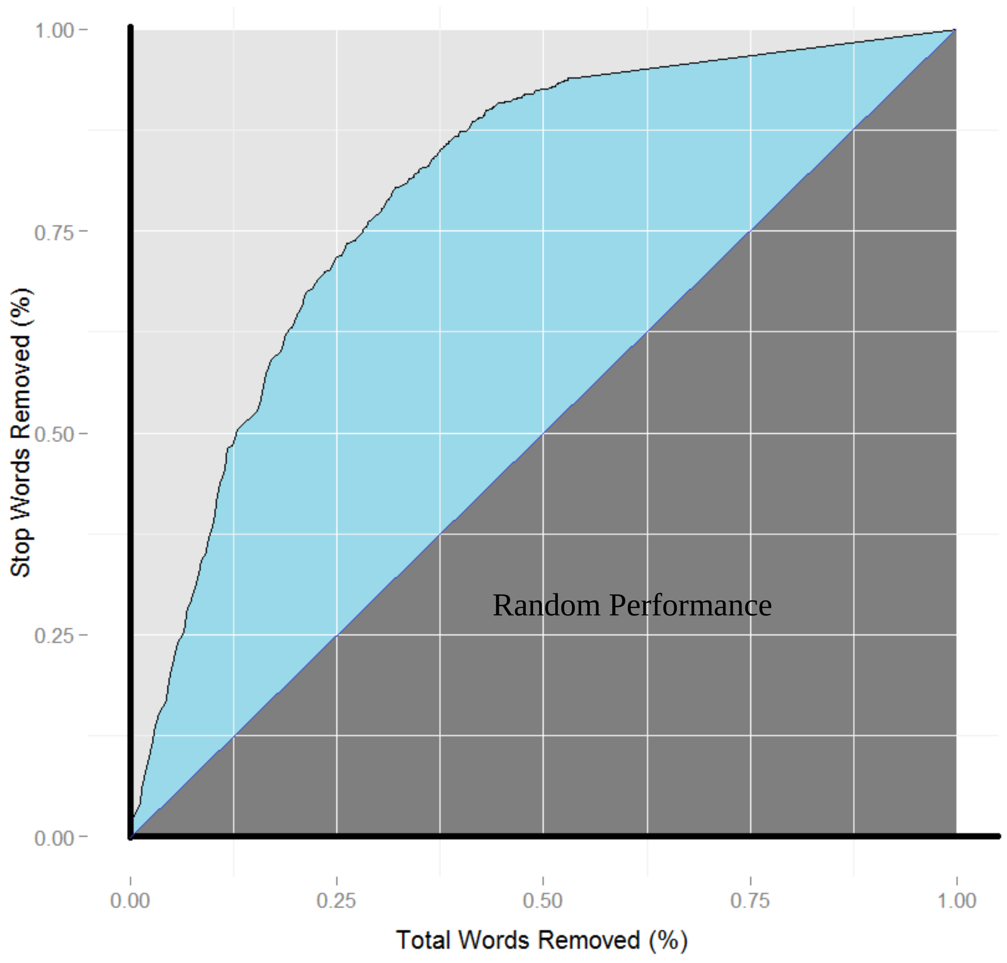

Figure 2. The Performance of the Known Case on Early 2013 data The area in blue indicates the relative performance gains compared to a random classifier.

There are two common aggregates of performance on a ROC Curve – the area under the curve, which indicates the overall fitness of the classifier, and the closest point to the ideal optimum: 0,1, which indicates the best performance achieved by this classifier across the entire curve. Another way to think of them is the area serves as a proxy for typical performance, while the closest point gives a measure of best performance. These quantities are highly correlated, but there may be trade-offs between them. For application purposes, there may be a sub-region of the curve that can be considered (it may not be feasible to retain 50% of the words, even if that's the best found performance). Both measures can be calculated for a sub-region.

What tokens are identified? In Figure 3, I show tokens that are removed at various cleaning levels in the known case for Early 2013.

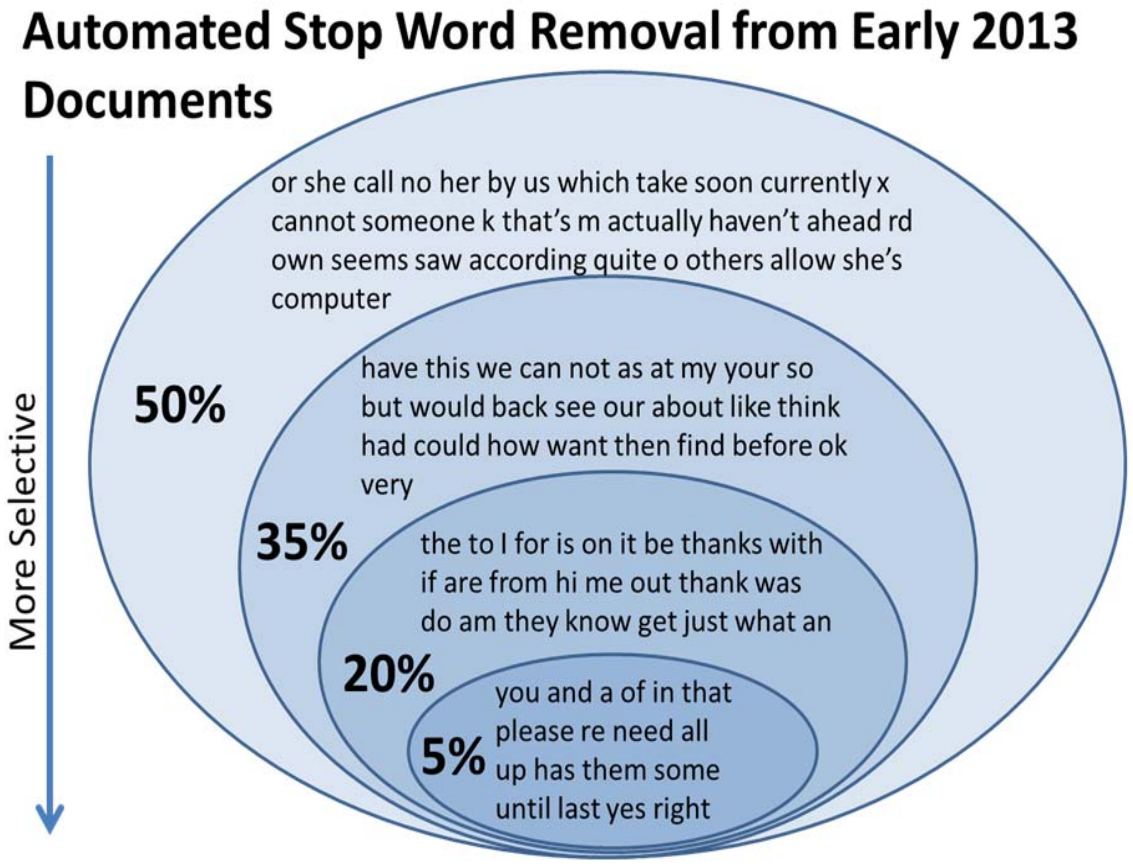
Automated Stop Word Removal from Early 2013 Documents
More Selective
50%
or she call no her by us which take soon currently x cannot someone k that's m actually haven't ahead rd own seems saw according quite o others allow she's computer
35%
have this we can not as at my your so but would back see our about like think had could how want then find before ok very
20%
the to I for is on it be thanks with if are from hi me out thank was do am they know get just what an
5%
you and a of in that please re need all up has them some until last yes right

Figure 3. Stop Words removed at various thresholds in the Early 2013 corpus.

3 The General Case: Identifying Low-Value Words through stochastic draws

I don't always have two corpora to work with, and I would like to identify low-value words even when I only have a single set of documents (a corpus) to work with. I will do this by taking the set of documents, creating random subsets as comparison corpora and identifying low-value words based on those random draws.

I use multiple (even many) random corpora based on the original data because I expect words without the ability to distinguish corpora to do so relatively consistently over time. There may be the occasional outlier where "the" is not identified as a low-value word, but on draw after draw, it will be. Evidence accretes over each draw. This gradual accretion of evidence is what allows us to harness the Law of Large Numbers.

3.1 Algorithmic Extensions

Essentially, the algorithm is as follows:

1. Identify an appropriate corpus, ***C***.
2. Gather evidence via random draws
    a. Select documents at random without replacement (each comparison is guaranteed to be non-overlapping sets) to form two corpora, R1 and R2. The number of documents in each random draw is another parameter, ***D***. Larger corpora are less likely to be noisy (in terms of odds ratios), and more likely to include many terms.
    b. Calculate the transformed odds ratio (Equation 1) for R1 and R2, for each unique token
    c. Sort the token set based on absolute value of the odds ratio and take the ***A***% lowest tokens. The value of ***A*** indicates how much evidence the tool gathers per random draw. For each token in the ***A***% lowest, note it was marked lowest by iterating its "found lowest" count by 1
    d. Merge found tokens with the master set of all tokens ever found
3. Generate the ROC Curve by starting from the highest realized "found lowest" count and iterating down until all tokens are removed.

3.2 Evaluating Performance across Parameters

I evaluate this technique across multiple settings of ***C***, ***D***, and ***A*** to provide a more complete evaluation of the approach. Please see Table 1 for the virtual experiment. Theoretically, very low values and very high values of ***A*** should produce sub-optimal performance closer to random – I use many settings of ***A*** in order to characterize the inverse-U shaped curve indicated by theory. I do multiple runs of each parameter setting to evaluate noise.

Table 1. Virtual Experiment for testing the Random Draw Corpora Case

Factor	**# of Values**	**Values**
Corpus (***C***)	3	Early 2013, Late 2013, 2014
Document Draw Size (***D***)	4	1000, 5000, 10000, 20000
Accretion Rate (***A***)	8	1, 5, 10, 20, 40, 60, 80, 99
Constants	**Setting**	
Filter Value	3	
Number of Draws	1000	
Outcomes		
Best Performance	The distance of the performance point closest to 0,1	
Average Performance	Area under the curve	
	Total Combinations	96
	Repetitions	10
	Total Runs	960

Our results indicate that, in general, accretion rate has the expected behavior, with area under curve describing a U-Shaped Curve, while optimal performance describes an inverse-U-Shaped curve – see Figure 3. Very small corpora (1000 documents) may benefit from very high accretion values, because there is relatively little information to be gained from each random draw. Larger corpora are usually better, although it appears that small corpora sometimes find a higher optimal performance than medium size corpora. The best performance is found when the accretion rate is about 40%.

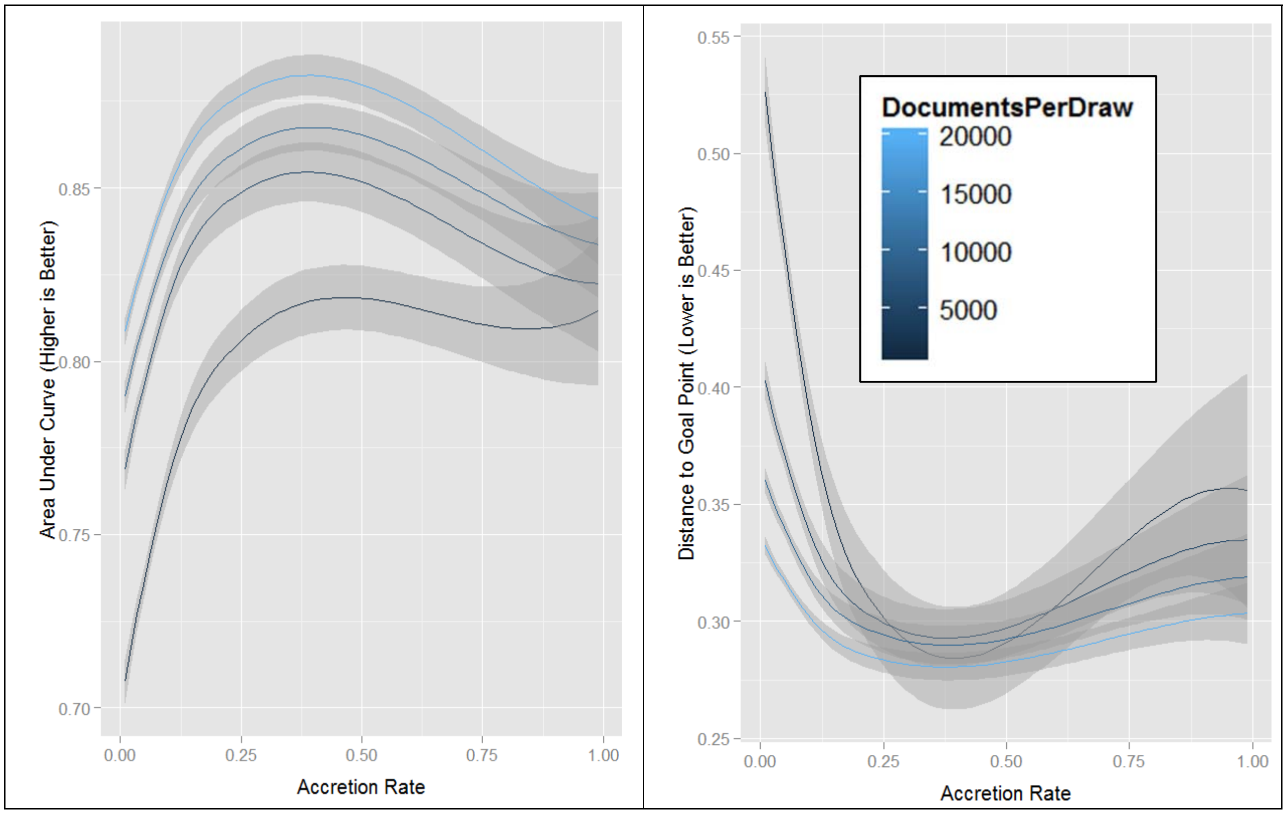

Figure 4. Performance based on Accretion Rate (*A*) – each line indicates a different value of *D*. I see the U-Shapes expected from theory.

Figure 4 shows our results for different corpora, ***C***. There is not a statistically significant difference between performance on these three corpora.

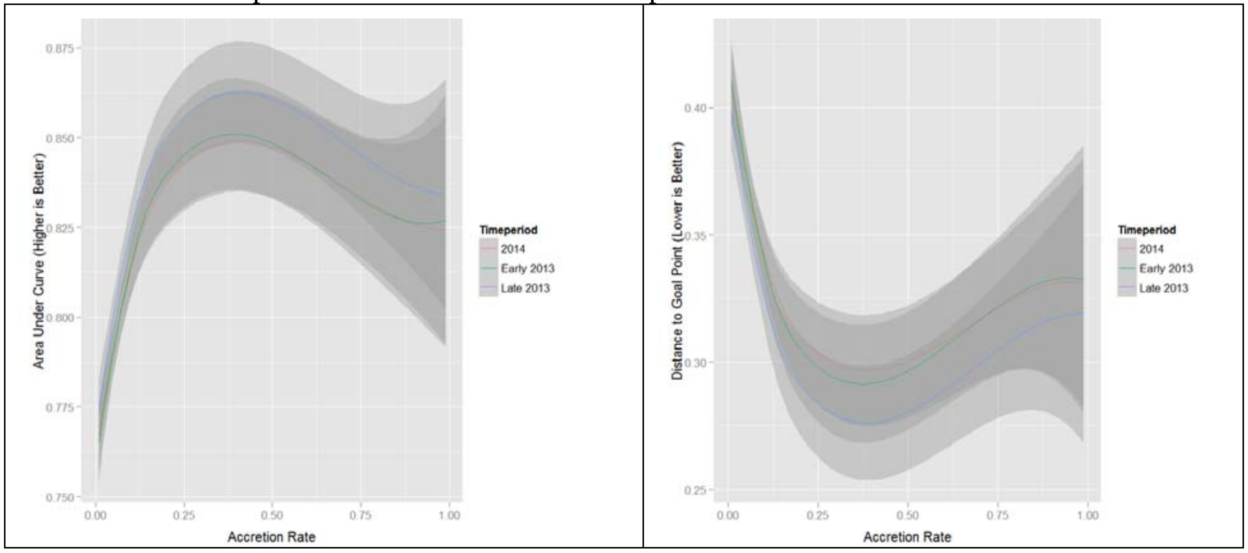

Figure 5. Corpora do not show statistically significant differences.

3.3 Comparison to TF-IDF

Another technique for identifying stop-words is the Term Frequency - Inverse Document Frequency (TF-IDF) algorithm. TFIDF uses the entire document corpus and characterizes the value of terms within the corpus. TF-IDF, as the name suggests, suggests that terms are important if they occur frequently, but are less important if they appear in every document, reasoning that words such as “the”, and “and” are common in documents and should be discounted. Both F and D are score vectors representing, respectively, the number of times a term appeared across all documents, and the number of documents the term appeared in. C represents the set of documents, so |C| represents the total number of documents in the corpus.

Equation 2. The TF-IDF term score is a function of the total count of the term’s appearance in the corpus, F_t, and the log of total documents in corpus, |C|, divided by the number of documents the term appeared in, D_t.

$$TFIDF_t = F_t * log \frac{|C|}{D_t}$$

I can then generate a ROC curve based on the TF-IDF scores, with terms that have the highest scores being retained the longest. I can then compare these ROC Curves, which is done in Table 2.

Table 2. Performance of different techniques

Classification	Best Distance from Goal (Less is Better)	Area Under Curve (More is Better)
Random Corpora Comparison	0.27	0.87
Known Corpora Comparison	0.365	0.76
Random	0.707 (Typical)	0.5 (Typical)
TF-IDF	0.99	0.135

TF-IDF, as defined here, performs very poorly on this task despite having access to entire document corpus, the area under the curve is .135 (below random), and the best distance is at 1 (when it essentially removes all documents), also worse than expected from random chance.

4 Discussion and Conclusion

In this paper, I introduced a well-known problem, the need to identify low-value words in a set of text documents, called a corpus. To address this problem, I created a new technique that generates an odds ratio of a term appearing in one of two corpora. This can be used to directly compare corpora to identify words that are low-value and suggest words for deletion, or can be used in conjunction with stochastic draws to suggest words to remove within a single corpus.

What is a low-value word is dependent on the text corpus and analysis needs, but stop words are usually low-value, and so I used the ability to identify stop-words as a way of comparing methodologies. I compared the known case, the stochastic technique, TF-IDF, as well as a null random classifier to contextualize that performance.

The stochastic technique performs best in our case, with TF-IDF performing the worst. I think this is because email data is messy, with irregularity of word choice and informal language being common along with widely varying data lengths. There may be versions of TF-IDF that perform much better on this data. This technique was primarily developed to help distinguish between group corpora, and thus the performance in identifying delete words is reassuring but not required to still find utility in the technique.

References

Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information processing & management*, *24*(5), 513-523.