Empirical Results I: Corpus Modification and its Effects on Topic Modeling

1 INTRODUCTION

The goal of this project is to explore the effects of corpus modifications on topic modeling. In previous sections, we built a state-of-the-art approach to topic modeling. Then, we defined several corpus properties that were both easy to measure and easy to modify. Finally, we defined metrics for topics that would allow us to measure the effects of changing these corpus properties.

During behavioural exploration, we were able to narrow down the set of corpus properties to explore. The three we chose are: the number of documents, the average document length, and the presence of stopwords. The first two have already been extensively studied in the related literature. [Tang et al.] shows the effects of changing either of these two properties indepently and at the same time. We limit our experiment to independently increasing or decreasing the three chosen properties.

Our expectation is that the results will follow those observed by Tang et al for modifications of the number of documents and the length of documents. Since to our knowledge there have been no experiments on the effects of inserting or removing stopwords, we rely on an understanding of the underlying phenomena of word-coccurrences between topics to predict the outcome in this portion of our experiment.

We also show those metrics that are the most useful in observing the intended behaviour. Often, it is hard to find labeled data with which to calculate perplexity. Though this is not a complete study of the best metrics for determining topic quality, it is enough to show that despite a lack of ground truth it can still be possible to reasonably assess topic quality.

2 SETUP

This experiment was written entirely in Python. Most of the code is contained in Jupyter notebooks with several utility Python files or objects existing outside of the notebook.

In the behavioural analysis portion we created a custom corpus reader object that knows more about the documents it reads than the original NLTK corpus reader. We will not go into detail on this object here since it was previously described.

A new object for the purpose of experiments is the MetricsModel object. This object models the key components of a generic topic model and can be queried for various topic-specific metrics such as the cosine distance between two topics or the top words from a topic. The entire list of available metrics has been explained in more detail in previous sections.

The code relies on several external libraries. The most substantial of these are the Natural Language Toolkit (NLTK) for tokenizing corpora and extracting stopwords and SciKit-Learn for the Latent Dirichlet Allocation (LDA) model and document translation into bag-of-words models.

Bokeh was used for plotting because of the ease with which it allows interactions. One of the challenges of performing multiple iterations of Latent Dirichlet Allocation is that it does not maintain consistent topic orderings. Asking which topic is which across multiple iterations is in some ways the wrong question to ask. A better question woud be to ask which topics most closely resemble other topics and at what point do they become impossible to distinguish? A metric such as cosine distance could be used to solve this task of pairing topics across iterations. However, we

are explicitly seeking to test the degradation of topics. Therefore, it is destined to become an impossible task. Instead of attempting to "match" topics, we concluded that the best method would be to leave the pairing to a human. Therefore, all of our plots were made so that hovering over any topic point will show the top three words from that topic. In this way, the general trends are still visible over all topics and individual topics that may be of interest can be tracked through human interaction before being further investigated.

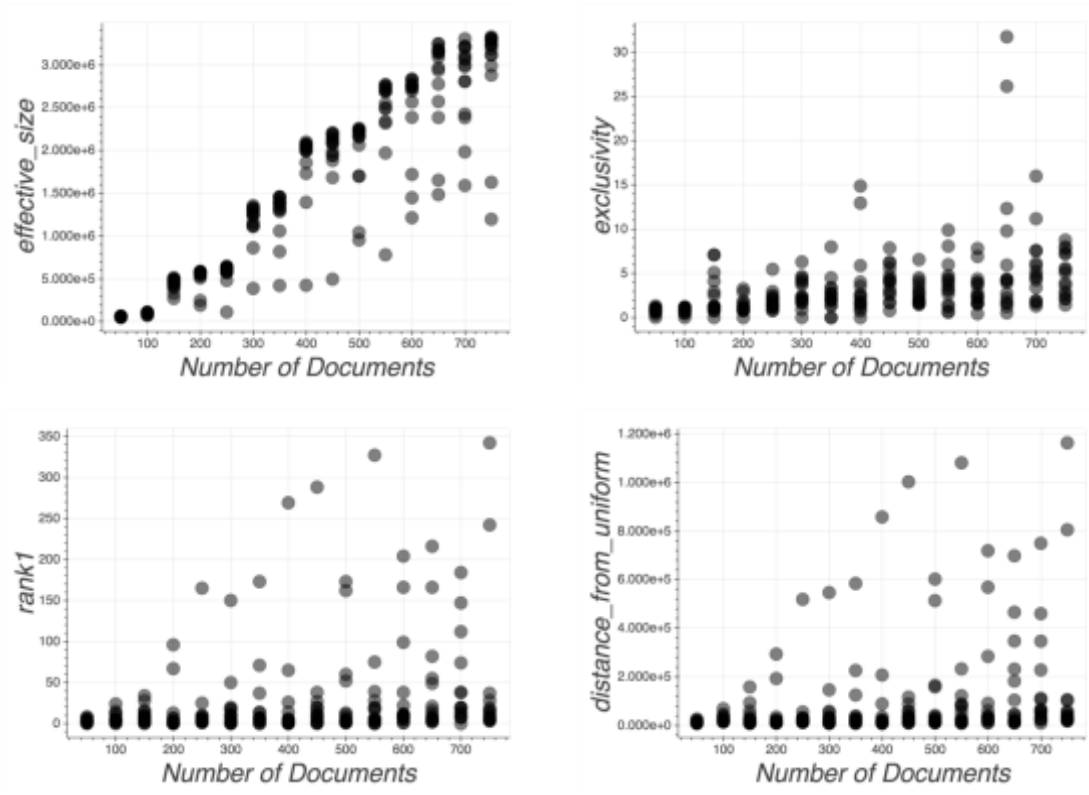
Unfortunately this hoverability is hard to translate into PDF format so if you would like to experience the wonders of Bokeh's HoverTool, please visit the repository at https://www.github.com/cassiecorey/snr-topic-modeling/.

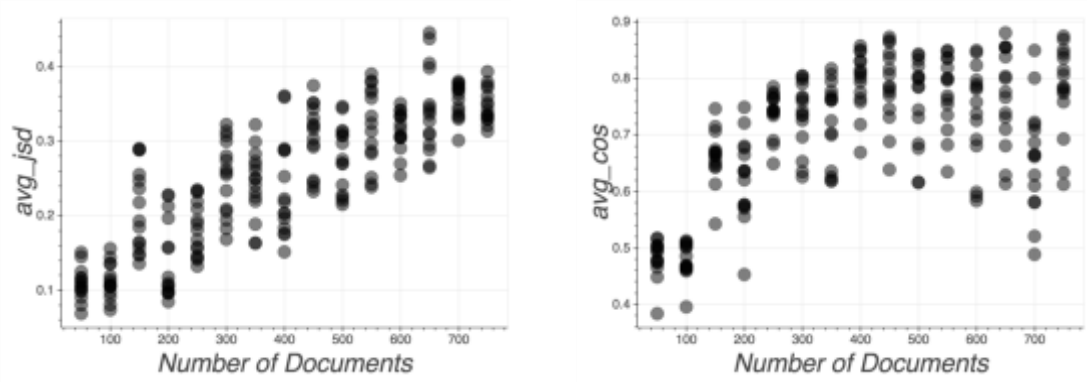
3 NUMBER OF DOCUMENTS

The algorithm for modifying the number of documents is relatively straightforward. NLTK corpus reader objects construct corpora from a given root directory and list of file ids to read from that directory. Therefore, we modified the number of documents in the test corpora by randomly sampling from the list of file ids before building the corpus with the sampled ids. The smallest number of documents tested was 50 and the largest was roughly equivalent to the original corpus size (in this case: 764). In between these two values we tested corpus sizes in 50-document increments.

3.1 Results

Experiments were run on each plaintext corpus from NLTK as well as the three we created specifically for this project: ABC Science, ABC Rural, and Wine Reviews. Results are shown for the ABC Science corpus. The other corpora all showed similar trends.





Effective size increases linearly with the number of documents. Exclusivity shows only a slight linear increase. The variance for both of these metrics also increases with the number of documents. For the Rank1 and Distance from Uniform metrics, the full range of possible scores increases with the number of documents. However, the mean stays relatively small.

3.2 Discussion

It is not surprising that the plots for Kullback-Leibler (KLD) and Jensen-Shannon divergence (JSD) between topics are similar because KLD is simply the asymmetric form of JSD. Neither is it surprising that the effective size of topics increases linearly with the number of documents. By adding more documents, we are inherently increasing the vocabulary size as well. Exclusivity of topics was largely unaffected by changes in the number of underlying documents.

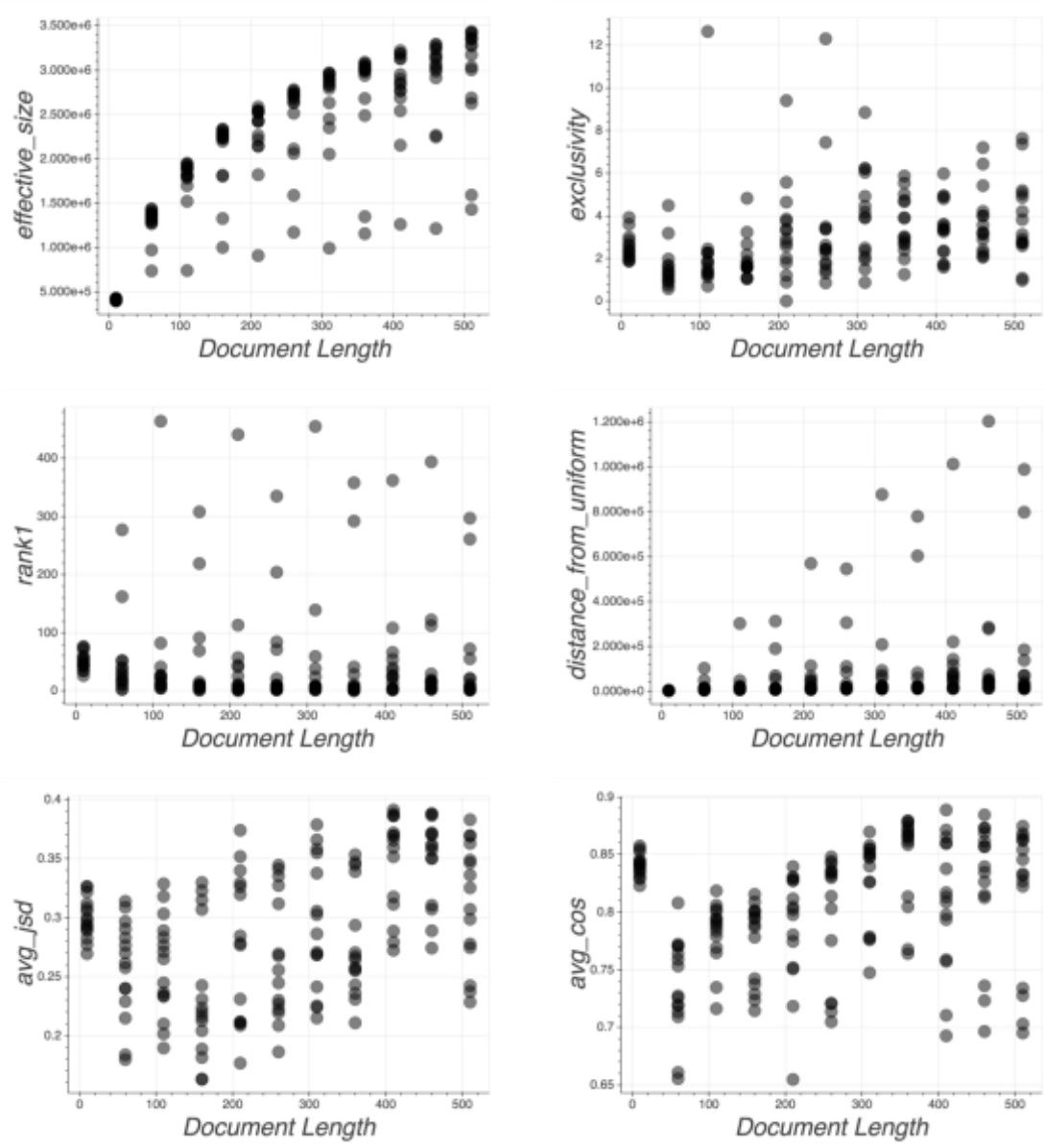
One limitation of this experiment is that we did not exceed the number of documents present in the underlying corpus. We felt this would lead to some odd behaviour if we were essentially repeating documents. Generating synthetic documents ventures into the realm of other more complicated research questions that we were not intending to address with this work.

Upon hovering on the outlier topics, the words are “says new researchers.” Therefore, it seems the only topics that were significantly negatively affected by increasing the number of documents were the most meaningless to begin with.

4 DOCUMENT LENGTH

Modifying document length was slightly more challenging than changing the number of documents but still relatively straightforward. The NLTK corpus reader object allows extracting a list of words/tokens from each file in the corpus. We chose random subsamples of various sizes from these lists to change the underlying document length in the corpus.

4.1 Results



Effective size increases non-linearly with the document length. We see again that Rank1 and Distance from Uniform remain relatively unchanged apart from a few outlier topics.

4.2 Discussion

The way effective size increases compared to the way it increased with changes in document size is more gradual due to the fact that we are changing every document in the corpus rather than adding documents of random size.

5 STOPWORD PRESENCE

Modifying the presence of stopwords in corpora was more challenging than changing the number of documents or even the number of words in documents. The algorithm is as follows:

1. For S in the range of stopword presences to test:
2. For each FILE in the corpus:

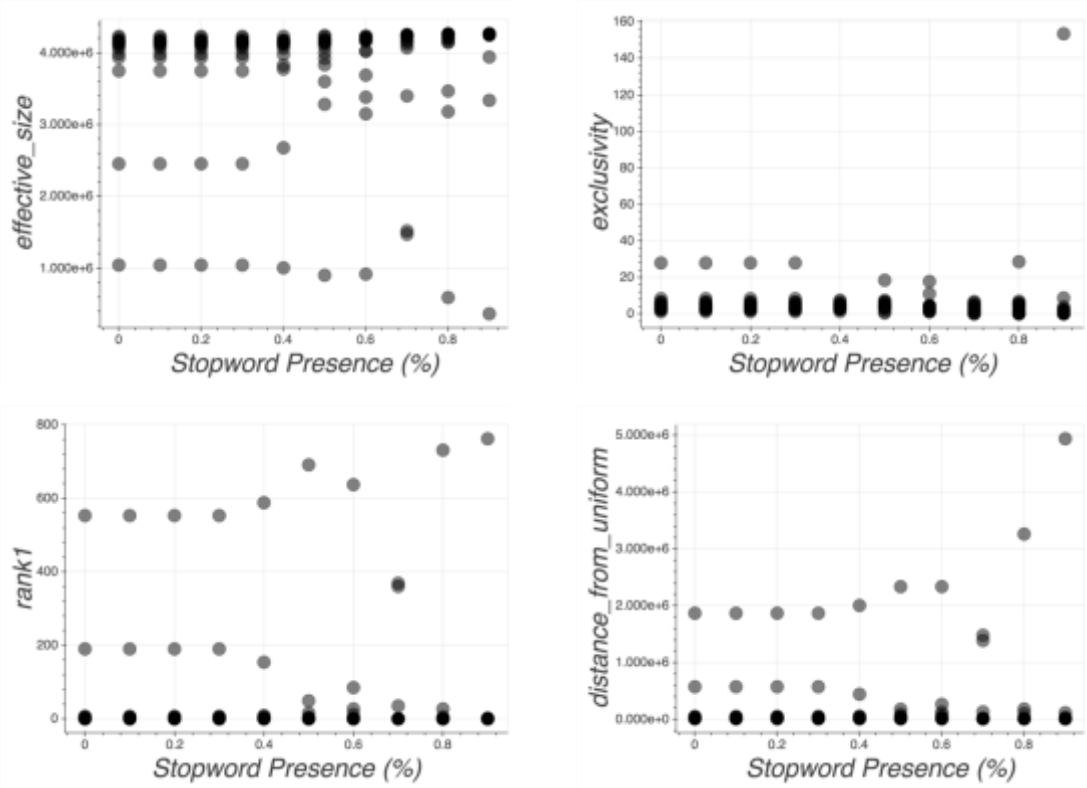
3. P ← the percentage of stopwords in the file
4. If P < S:
5. N ← the number of stopwords to add
6. Randomly select N from STOPLIST∩FILE
7. Else:
8. While the percentage of stopwords in the file is > S:
9. W ← Randomly selected stopword from STOPLIST∩FILE
10. Remove that word from FILE

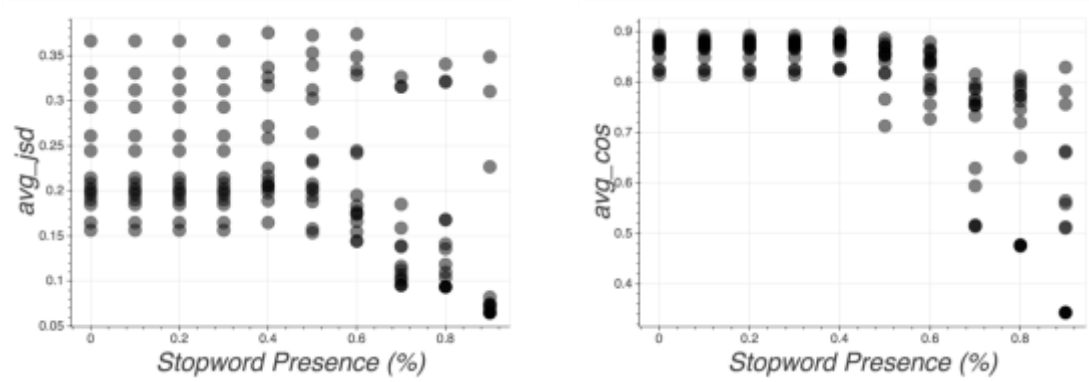
The proportion of stopwords in the text is calculated by taking the sum of the number of appearances of each word from the intersection of the stoplist and the text vocabularies and. The stoplist comes from NLTK's list of English stopwords. The number of stopwords N to add for a desired presence percentage p is calculated as follows:

$$N = \frac{text.count\_stopwords - text.length \cdot p}{p - 1}$$

This equation requires that $p \neq 1$. Therefore, we only test up to 0.9 for stopword presence.

5.1 Results





There are several outlier topics in the plots for Rank1, Distance from Uniform, and Effective Size that change drastically as the presence of stopwords it altered. This corpus in particular had a pre-existing stopword presence of about 35%.

5.2 Discussion

Modifying the presence of stopwords was perhaps the most interesting of the experiments we performed. As shown in the graphs, there was some odd behaviour observed that was not entirely expected by our hypothesis.

Our stoplist was extremely limited and general. In future work, it would be interesting to make a better stoplist that is more fine-tuned to the corpus being tested. This could be done using the top words from the outlier topics seen in the above plots. Doing this would remove the outliers from the plots and probably improve topic exclusivity since most topics often share a few undetected stopwords in their most-likely words.

REFERENCES

[1] Tang, J., Meng, Z., Nguyen, X., Mei, Q., & Zhang, M. (2014, January). Understanding the limiting factors of topic modeling via posterior contraction analysis. In *International Conference on Machine Learning* (pp. 190-198).