Independent Research Project

Fauna Mahootian
Cornell University

Abstract

In order to carry out research, sociologists must often search through a corpus of thousands of documents to find ones relevant to their query. Sometimes a keyword search is satisfactory, but sometimes the factors that distinguish relevant documents from irrelevant are more complex than a keyword search can distinguish[1]. Our goal is to provide sociologists searching a corpus for a concept of interest with an automated search method to quickly narrow the corpus to a subset that contains a maximal number of relevant documents and minimal number of irrelevant documents according to the concept of interest. We use a combination of metrics to filter for various types of relevant documents, according to our typology. We want to suggest thresholds for each of these metrics that will generally create good quality subsets of corpuses. We create a function that scores the quality of the subset created by the input set of thresholds, and find its maximum using Bayesian optimization with Gaussian processes. We ascertain that 0.26-0.27 is a good threshold for total topic proportion, and that more tests need to be run to find good thresholds for our other metrics.

1. Introduction

In order to carry out research, sociologists must often search through a corpus of thousands of documents to find ones relevant to their query. Sometimes a keyword search is enough to come up with a satisfactory subset of documents. However, sometimes the factors that distinguish relevant documents from irrelevant are more complex than containing keywords[1]. Our goal is to provide sociologists with an automated search method to quickly narrow the corpus to a subset that contains a maximal number of relevant documents and minimal number of irrelevant documents based on the researchers concept of interest, using a combination of methods that target different types of documents.

Our method uses LDA topic modeling and identification of relevant topics to construct various proportional signifiers for each document that can indicate its relevance. We suggest general thresholds for each of these proportions that can be applied as filters to return a subset of documents that have at least one proportion that falls above the threshold.

In order to identify general thresholds for our recommendation, we optimized over a function that scores sets of thresholds on the precision and recall of the subset of documents they returned (based on a set of 500 documents that had been labeled as relevant and irrelevant).

1.1. LDA Topic Model

We wanted the topics and to be automatically defined and modeled, to reduce demand on users of our method. We chose to use an LDA topic model because it is one of the most widely used models[2]. Intervention is required on the part of the researcher to choose the number of topics (K) for the model to estimate. The optimal choice of K is unique to each corpus and is nontrivial to identify. We chose 50 for our corpus, and it resulted in several recognizable topics. To use our method, the researcher would first create an LDA topic model of their corpus, then would label each of the topics it produces as relevant, marginally relevant, or not relevant.

1.2. Metric

In order to filter for the following types of relevant documents, we employed several measurements as metrics for relevance. These measurements include max topic proportion, total topic proportion, "super" keywords, and vocabulary proportion. If the document surpasses our chosen threshold for any of the following metrics, it is considered "relevant" by our algorithm.

Maximum topic proportion is the largest proportion among the relevant topics in a document.

Total topic proportion is the sum of the proportions of relevant topics present in the document.

Vocabulary proportion is the proportion of words in the document that are also in a list of vocabulary words, generated by a method that orders all words in the corpus based on the words frequency in the relevant topics vs. in the corpus overall. The length of this list can be varied.

"Super"keyword list is a keyword list created by the researcher choosing words from the vocabulary list, words that would only be used to refer to the concept of interest.

1.3. Typology

We have defined a typology for documents according to how they relate to the research concept of interest. We have used these definitions to implement methods to identify documents of each type. The four types of relevant document are:

1. documents that are mostly about the concept of interest. To identify this type of document, we check if the total topic proportion or the maximum topic proportion surpass the suggested threshold.
2. documents that are partially about the topic of interest. To identify this type of document, we check if the vocabulary proportion surpasses the suggested threshold.
3. documents that briefly refer to the concept of interest. We identify this kind of document using the "super" keyword list. If a document contains any of those keywords, it is labeled relevant.
4. documents that refer to the concept of interest in uncommon terms. To identify this kind of document, we use the vocabulary proportion; the document is relevant if it contains a large enough proportion of words in the vocabulary list. Total topic proportion might also work here.

The two types of irrelevant document are:

5. documents completely about non-relevant concepts.
6. documents about related concepts, therefore containing related terms.

We employ a mix of three methods with the goal to ensure that the subset we generate contains a maximal number of relevant documents and excludes the least relevant documents.

1.4. Labeled Dataset

Our test corpus contained about 26,000 documents, curated by a domain expert,[1] who also created a labeled subset containing 500 documents. Each document was labeled with its typology(1-6) and relevance (0, 1). Three of

[1]Annotation performed by Alicia Eads, U. Toronto

these documents were excluded because one of them was empty, and two were missing from the dataset, so the labeled dataset used contained 497 documents.

2. Methods

In order to suggest general thresholds for each of these values – max topic proportion, total topic proportion, vocab list proportion, and top m vocab words – we needed to first create a scoring function that, when given thresholds and a labeled dataset, creates a subset by filtering documents according to the thresholds, and then outputs a score indicating the quality of the subset. Then, we needed to optimize over that scoring function to identify values for each threshold that will work generally for filtering corpora into quality subsets.

2.1. Scoring Function

In the implementation of the scoring function, the filtering process included a document in the subset if any of its proportions pass the respective threshold. The score we used for the quality of the subset was the F1 score, which is a comparison of precision (how many documents in the subset were relevant?) to recall (how many documents are in the subset vs. how many are not). F1 score returns the weighted average of precision and recall, which is useful in situations where the costs of false positives and false negatives are dissimilar. This is true in our case, as a little pollution in the subset is less costly than leaving out relevant documents. Our goal is for the subset to contain all relevant documents in the corpus and no irrelevant documents, and the F1 score tells us how close we are to achieving that.

2.2. Optimization

The goal of optimizing the scoring function is to find the set of inputs (thresholds) that have the highest F1 score. The method we opted for was Bayesian optimization using Gaussian Processes. This method finds a distribution of functions that fit the data, in order to find the functions minimum value[3]. We had the score function output (1 - F1), so that the minimum value is the highest F1 score. This algorithm works by first assuming the function output, i.e. threshold scores, follow a multivariate Gaussian distribution. It picks random inputs within the given ranges, and uses the function value at that point to update what it knows about the shape of the function. It uses priors to choose the next set of inputs to check, and continues iterating until it reaches the number of iterations set at call time.

3. Results

The following tables show thresholds that the optimization settled upon, and the corresponding F1 scores.

Tables 1, 2, and 3 show the comparative performances of the three vocabulary list generating methods: relative entropy (Table 1), tf-idf (Table 2), and log-tf (Table 3), at 100 iterations. All performed relatively similarly, i.e. produced similar F1 scores.

Relative Entropy	MTP	TTP	VocP	m words	F1
Run 1	1.000	0.270	0.287	1	0.790
Run 2	0.711	0.270	1.000	130	0.790
Run 3	1.000	0.272	0.235	200	0.790
Run 4	1.000	0.261	0.593	1	0.787
Run 5	0.884	0.259	0.427	85	0.787

Table 1: Thresholds and F1 score suggested by the optimization at 100 iterations each run, with the vocab list created through the method relative entropy. The range for m, the number of words in the vocabulary list, was 1 - 200, and the range for the other three thresholds was 0.0 - 1.0.

tf-idf	MTP	TTP	VocP	m words	F1
Run1	1.000	0.270	1.000	65	0.790
Run 2	0.351	0.315	0.494	162	0.785
Run 3	0.364	0.268	0.200	187	0.790
Run 4	0.971	0.234	0.781	4	0.786
Run 5	0.364	0.167	0.382	14	0.777

Table 2: Thresholds and F1 score suggested by the optimization at 100 iterations each run, with the vocab list created through the method tf-idf. The range for m, the number of words in the vocabulary list, was 1 - 200, and the range for the other three thresholds was 0.0 - 1.0.

Vocabulary list size, optimized and fixed In Table 4, look at the column containing the vocabulary proportion threshold (VocP). We changed various aspects of the optimization to try to get vocabulary proportion threshold

Log-tf	MTP	TTP	VocP	m words	F1
Run 1	0.267	0.271	0.801	173	0.790
Run 2	0.487	0.262	0.547	1	0.787
Run 3	0.512	0.315	0.959	168	0.785
Run 4	1.000	0.317	0.286	75	0.785
Run 5	0.530	0.235	0.116	1	0.786

Table 3: Thresholds and F1 score suggested by the optimization at 100 iterations each run, with the vocab list created through the method log-tf. The range for m, the number of words in the vocabulary list, was 1 - 200, and the range for the other three thresholds was 0.0 - 1.0.

to converge, but it never did. Aspects we tinkered with include number of iterations, range of vocab list size, fixed vocab list size, and inclusion of max topic proportion. We used relative entropy to generate the vocabulary list. View Table 1 for the comparative impact on vocab proportion threshold of 100 iterations, and a 1-200 range of vocabulary list size.

Comparatively, the threshold for total topic proportion varies very little, hovering around 0.26-0.27. The threshold for max topic proportion fluctuated more at 100 iterations, but became more stable at 500 iterations, around 0.5.

4. Discussion

The results indicate that a value of around 0.26-0.27 is an appropriate threshold for the total topic proportion of a given document, when applying thresholds to filter a corpus to create a relevant subset. In table 4, almost all of the suggested thresholds for total topic proportion fall in this range.

The threshold for max topic proportion fluctuated more at 100 iterations, but became more stable at 500 iterations, around 0.5.

The threshold for vocabulary proportion fluctuated a lot at both 100 and 500 iterations. We tried fixing the number of words in the vocabulary list (rather than including it in the set of thresholds to optimize), but the vocabulary proportion threshold still did not converge. We excluded vocabulary proportion and optimized over max topic and total topic proportion, and found that the optimization reached the same F1 score. Although it is indicative, this isnt conclusive evidence to deem vocabulary proportion superfluous.

The three different vocabulary methods performed about the same, relative entropy seemed to perform the best, garnering the highest F1 scores,

m range: 10-200	MTP	TTP	VocP	m words	F1
Run 1	0.858	0.256	0.027	10	0.793
Run 2	0.575	0.268	0.539	10	0.790
m range: 10-200, no MTP		TTP	VocP	m words	F1
Run 1		0.269	0.235	200	0.790
Run 2		0.271	0.383	10	0.790
Run 3		0.267	0.463	200	0.790
Run 4		0.269	0.962	189	0.790
m range: 10-500	MTP	TTP	VocP	m words	F1
Run 1	0.508	0.270	0.556	10	0.790
Run 2	0.508	0.268	0.926	474	0.790
m range: 10-500, no MTP		TTP	VocP	m words	F1
Run 1		0.269	0.672	256	0.790
Run 2		0.268	0.118	54	0.790
Run 3		0.271	1.000	500	0.790
m = 200	MTP	TTP	VocP	m words	F1
Run 1	0.571	0.271	0.645	(200)	0.790
Run2	0.489	0.272	1.000	(200)	0.790
m = 200, no MTP		TTP	VocP	m words	F1
Run 1		0.271	0.883	(200)	0.790
Run 2		0.272	1.000	(200)	0.790

Table 4: Effects on vocabulary proportion threshold of range of vocab list size, fixed vocab list size, inclusion of max topic proportion. All treatments were run with 500 iterations.

but more repetitions are needed to say this conclusively.

It could be useful to run the optimization at a higher number of iterations (e.g. 1000). The highest number of iterations we ran it with was 500, and we did not get convergence on thresholds other than total topic proportion. Also, the optimization settled most frequently on the F1 score 0.7909, but on one run it found a higher one (0.7930). Perhaps running the optimization with more iterations could locate higher F1 scores with more frequency.

In order to determine if any of the thresholds are superfluous to constructing a high quality subset, we would need to create subsets by exclude each threshold, and compare the sets of documents incorrectly identified as relevant or irrelevant – the false positives and false negatives.

In general, it would be informative to examine the false positives and false negatives to identify ways to improve our implementation. Currently, the different sets of thresholds suggested by the optimization at 500 iterations produce the same F1 score (the 0.790 value in Table 4) the majority of the time. We have compared the sets of false positives and false negatives produced by each of the different threshold sets that produce this score, and the false positives are congruent for all sets, as well as the false negatives. There are 26 documents incorrectly identified as relevant, and 32 incorrectly identified as irrelevant.

We may want to change the balance of precision and recall by changing how each is weighted to calculate the F1 score. In the situation were designing for, its more costly to leave out a few relevant documents than it is to include a few irrelevant documents. Some good might come of experimenting with weighing recall more heavily than precision.

[1] J. Grimmer, B. M. Stewart, Text as data: The promise and pitfalls of automatic content analysis methods for political texts, Political Analysis 21 (2013) 267–297.

[2] D. M. Blei, A. Y. Ng, M. I. Jordan, Latent dirichlet allocation, Journal of Machine Learning Research 3 (2003) 993–1022.

[3] P. I. Frazier, A tutorial on bayesian optimization, arXiv preprint arXiv:1807.02811 (2018).