M2 EEE Data Analytics

Topic Modeling New York Times Articles

Andrew Boomer, Nikita Marini, Jacob Pichelman, Luca Poll

February 26, 2021

Table of Contents

1 Introduction

Natural Language Processing (NLP) and related techniques like topic modelling allow to tame the hydra of information overload in times of digitization. More specifically, the statistical modelling of unstructured data allows the retrieval of patterns and valuable information from a variety of high frequency micro-level data.

This is especially beneficial when applied to easily accessible unstructured data like social media posts or news archives. While sentiment analysis can be of great use to model acceptance or dissent, topic modelling allows the classification of documents into categories, as well as getting a sense of the content of a corpus of documents without having to read them. This can be especially useful when a researcher wants to deal with a large corpus of documents like newspaper texts for instance. In topic modelling, not all the dimensions can be optimized quantitatively. Some are qualitative such as the relative interpretability of different token sets. Noting that these distinctions are dependent on the researcher and hence can suffer a bias, we show how small differences in the process of topic modelling can lead to great differences in the outcome.

More precisely, we will use historic news articles from 1981-2020 concerning world affairs and will evaluate their interpretability and presumed accuracy. This will be done by firstly presenting the scraping of the data and the data itself, followed by a description of the conducted preprocessing steps. Third, we will present the topic model employed and the results of these models.

2 New York Times Articles

The New York Times (NYT), as one of the most renowned newspapers in the world, has a long tradition of reporting in detail about world affairs through their numerous foreign correspondents. Besides covering most countries, the NYT also gives public access to their archives.

2.1 Scraping Articles

We scraped the New York Times online archive to gather a corpus of text (https://developers.nytimes.com/). The NY Times makes their archive publicly available through a URL based API. In order to query the API, an API key must be obtained through registering a free account. JSON files are returned from each URL request, and we parse these JSON files into data frames using the *fromJSON* function in the *jsonlite*

R library.

There are several different API's available through the NY Times developer website, we chose to query the full archive, which must be queried in monthly increments. The initial JSON file returned from the monthly query to the archive API is the metadata of each article in that time range. This metadata file contains information such as the publishing date, the section name, the word count, and the URL of the article itself. Using the article URL, we then request the source `HTML` code of each article in the full metadata file.

The full text of the article is found within the section tag named *articleBody* of each `HTML` file. We use the *html_nodes* function in the *rvest* library, and specify the xpath argument as *"//section[@name="articleBody"]* to get the location of the article text in each file. Finally, the *html_text* function is used to get the text string from this location in each `HTML` file.

We ran into some HTTP related errors while scraping this data from the NY Times API. These errors were related to a URL not being found or the API URL being incorrectly specified. Rather than breaking out of the loop when this arose, we implemented the *tryCatch* function in R to catch these errors and proceed to the next iteration if there was a bad URL path from the metadata file.

The error catching functionality and reading of the `HTML` files were combined into a function that was passed to the *sapply* function in R to avoid looping through the URL's.

2.2 Sample

The New York Times makes the entire range of their news desks and newspaper sections available for scraping. For our purposes, however, we identified articles belonging to the section *World* as the most useful. As shown in Figure 1, the number of articles published in the *World* section varies over time. With a total of 303,177 articles published in this section from 1981-2020, analyzing the full corpus would be too large a task for the memory and computing power we have at hand. We therefore take a representative random sample from the relevant articles.

To generate the sample, articles were randomly extracted from the whole dataset on a weekly basis, such that the weekly number of articles in the sample corresponds to 6% of the number of articles published during the respective week. The comparison of the full dataset and the random sample are shown in Figure 1. Overall, the number of articles in the sample used for the analysis amounts to 16,951.

The average article consists of 352 distinct words as shown in Figure 2a. When visualizing the terms in the corpus further, as through the word cloud of Figure 2b, we quickly see that the most frequent

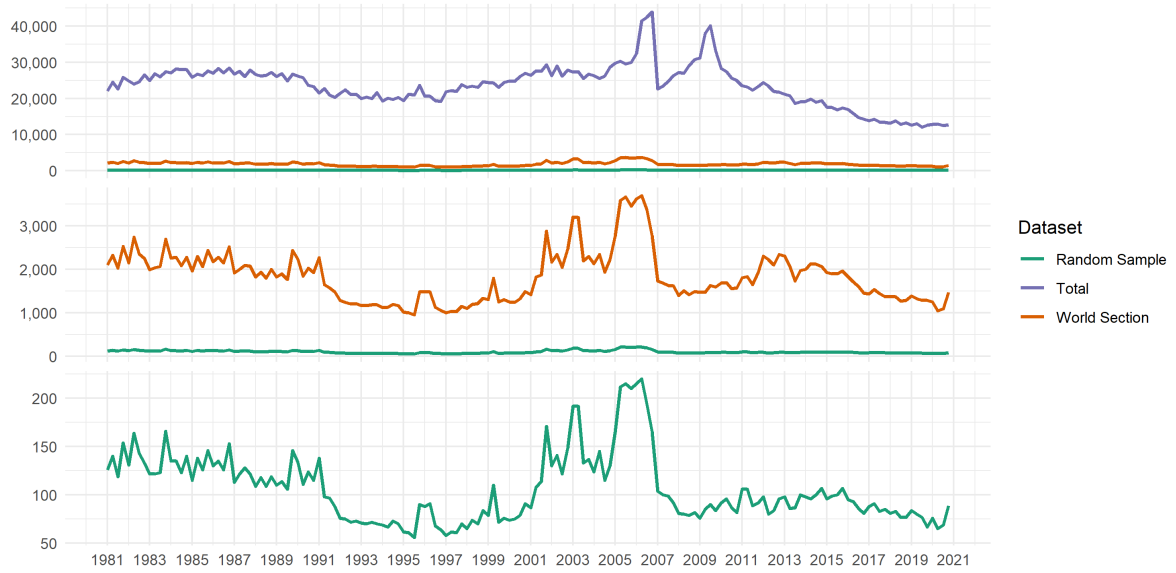
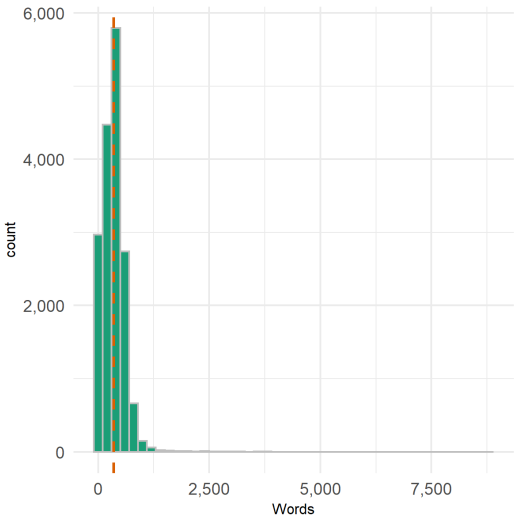
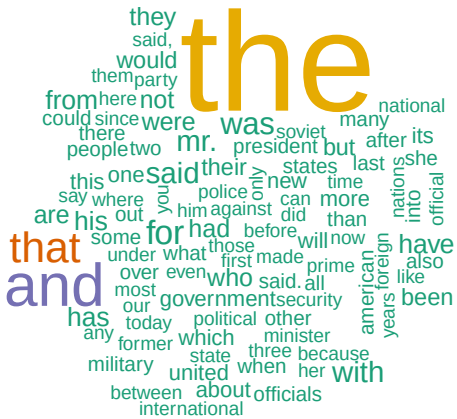


Figure 1: Published Articles in the NYT

terms across all documents are common words like "the", "and", "that" or "with". Although these terms appear at a high frequency, they carry little information. To account for this problem and enable meaningful information retrieval, text preprocessing has to be conducted.



(a) Distinct terms in article



(b) Most frequent terms in corpus

3 Pre-processing

In order to make the human-readable articles interpretable for machines, cleaning and so called "preprocessing" of the unstructured is performed. While the cleaning mostly involves removal of unwanted noise in the text[1], preprocessing involves more generalized data cleaning procedures. Since the different steps of preprocessing can have a significant effect on topic identification, we follow the approach presented by Martin and Johnson (2015) and define three models with a different extent of preprocessing (note that the listed order does not represent the order in which the steps are undertaken):

Model 1 (basic)	Model 2 (lemmatization)	Model 3 (only nouns & names)
Lowercase	Lowercase	Lowercase
Tokenization	Tokenization	Tokenization
Remove punctuation	Remove punctuation	Remove punctuation
Remove numbers	Remove numbers	Remove numbers
Remove Stopwords	Remove Stopwords	Remove Stopwords
Stem tokens	Stem tokens	Stem tokens
	Lemmatize tokens	Lemmatize tokens
	Account for bigrams	Account for bigrams
		Keep only nouns and names

Table 1: Models by their preprocessing steps

While the first model represents the most commonly undertaken steps, lemmatization and keeping only nouns for information retrieval are performed less often. In order to improve the accurancy and interpretability, we added the detection and labeling of frequent bigrams to models 2 and 3.

3.1 General Preprocessing Steps

Common preprocessing steps include setting all characters to *lowercase* such that "Hello World", "hello world" and "hELLO wORLD" are equivalent. *Tokenization* represents the following step, during which the text strings are divided into tokens, which in most cases represent words. Tokens that represent *punctuation* or *numbers* are stripped, since they do not contain informative value. Furthermore, we identify tokens which represent *stop words*, which are words with a high frequency that carry little information.These stopwords are taken from the snowball stopword list provided by the stopwords package.

[1] like `HTML` code chunks or repetitive and uninformative text elements among others

This choice is rather conservative, since the snowball stopword list includes 174 words while the onix or the SMART stopword lists include 404 and 571 terms respectively. Once the stopwords are removed, the remaining tokens can be stemmed. *Stemming* refers to the process of replacing the words by their respective word stems. Hence 'dancing', 'dances' and 'danced' will all be transformed to 'dance' without any loss of information. For stemming, the snowball stemmer contained in the stemDocument function is used. *Lemmatization* on the other hand aims at replacing the words by their lemmas. Therefore, words like 'go', 'went', 'gone' will all be transformed to 'go'. For the lemmatization procedure the function lemmatize_strings from the textstem package is used.

3.2 N-grams

The presented general steps of text preprocessing consider tokens as individual units or words. Words do have a meaning when standing alone such as "west" and "bank", but once combined they change the meaning or form a name like "West Bank". Words that co-occur are called n-grams. Since topic modelling procedures to not consider the order of the tokens, we account for bigrams by manually labeling them. We retrieve all co-occurring tokens and subsequently filter the bigrams restrictively[2] for noisy co-occurrences such as "of the" or "and then". The 200 most frequent terms, which are shown in Figure 2 where the shade of the arrow indicates the frequency and direction, are concatenated together such that they will stay together throughout the course of preprocessing.

3.3 Part-Of-Speech Tagging

Martin and Johnson (2015) argue in their study that the subject of an article is most typically represented in the articles' nouns. In order to reduce the corpus to only nouns (and names), one first has to conduct so called 'Part-of-speech tagging' (POS-tagging). Through POS-tagging, the word class of every word in an article across the entire corpus is determined. This is done through the udpipe_annotate function from the udpipe package. Since the POS-tagger determines the word class of every word in the whole corpus (14,128,985 words), this procedure is computationally quite expensive. The words are assigned to one of 17 different classes, out of which subsequently only the nouns and pronouns can be retrieved. Figure 3 represents the respective amount of words that were assigned to 14 out of the 17 categories.

[2]By using the snowball, onix and SMART stopword lexicon among other selected noisy terms

Figure 2: 200 most frequent bigrams in articles

4 Initial Model Comparison

Before moving on to the Topic Modeling algorithms, examining the word clouds of the different preprocessing models can provide an idea of the type of information contained in each processed corpus. These word clouds for the preprocessing models 1, 2, and 3 are shown in Figures 4, 5, and 6 respectively. One thing that immediately jumps out is that two of the most heavily weighted tokens in Model's 1 and 2 are "say" and "will". The Model 1 cloud also contains other tokens similar to these, such as "much", "like", "back", and "told". These sorts of tokens are unlikely to contain any meaningful information, and we would therefore expect the presence of these tokens to weaken the interpretability of the topics generated through topic modeling.

Consistent with Martin and Johnson (2015), taken as a whole, the word cloud from Model 3 in Figure 6 seems to contain the most information. Given we chose a subset of articles under the NY Times section "World", some of the strongest tokens are "government", "president", "people" and "country". At the same time, it is also possible to imagine distinct topics within this world cloud, noting the slightly weaker tokens such as "police", "force", and "party". To not simply rely of the intuition, however, we will employ topic modelling techniques.

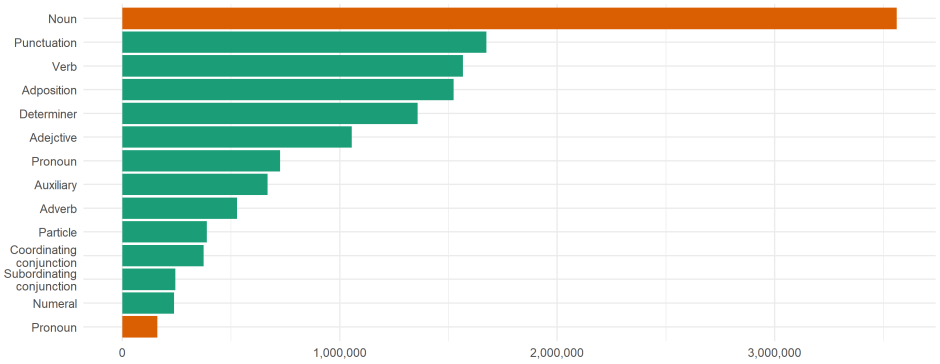


Figure 3: Words in corpus assigned to word classes

5 Topic Modeling

A topic model is a statistical model that allows discovering patterns that occur in a corpus of documents. These patterns of word co-occurrence are commonly conceived as *topics*. Topic modeling can foster informative insights and provide means of classification for largely unstructured data. In the context of this project, knowledge about the shares of each topic over time allows linking news coverage to key historical events and related paradigm shifts.

Figure 4: Word Cloud Model 1

Figure 5: Word Cloud Model 2

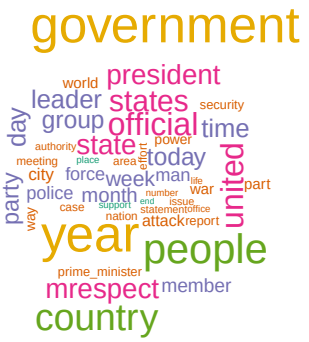


Figure 6: Word Cloud Model 3

5.1 Latent Dirichlet Allocation

We employ the Latent Dirichlet Allocation (LDA) developed by Blei, Ng, and Jordan (2003) as the algorithm to model topics from the corpus. With LDA, a document (or article in the context of this project) is taken as a collection of observable words where each word is assumed to be drawn from an underlying and unobserved (hence "latent") topic. LDA calculates the probabilistic distribution of a certain word over a set of topics (subsequently denoted as β), and the distribution of a given topic over

a set of documents (subsequently denoted as γ). Topics are then derived by calculating patterns of word co-occurrence across the corpus. Each topic in the model will be distributed across each article as a probability. Naturally the total probabilities of each topic appearing in an article will therefore sum to 1.

5.2 Number of Topics and Hyper Parameter Tuning

One of the key arguments that has to be passed to the LDA algorithm is the number of topics K that should be distilled from the corpus. The literature argues that there is no universally applicable decision rule for the choice of K, only that it has to be (qualitatively) evaluated on the performance of the model. Suominen and Toivanen (2016). To put this in layman terms, the current consensus is to follow a trial and error approach. We experimented with $K = \{3, 5, 10, 15, 25\}$ and found $K =$ to result in sensible topics.

Moreover, LDA has two additional parameters that specify the document-topic density (i.e. how widely a topic is distributed over an article) and topic-word density (i.e. how many words there are in a topic), which are denoted as α and β, respectively. Intuitively, a higher α leads to articles being made up of more topics and a higher β results in topics being composed by more words present in the corpus. We follow Griffiths and Steyvers (2004) and set $\alpha = 0.1$ and $\beta = \frac{50}{K}$.

6 Evaluation

As mentioned before, the search for the optimal specification of the topic model has to be conducted in large part qualitatively. As Carter et al. (2016) stress, the quality of a topic model is best measured by how well it accomplishes the task that it was built for. A necessary condition for doing so is the interpretability of a topic. We identify two main factors that affect the interpretability of retrieved topics and discuss them in the following sections.

6.1 Noise Removal

We briefly mentioned that some words have a high informational value (mostly nouns according to Martin and Johnson (2015)) while others, like stopwords, contribute little information. The latter can often be perceived as noise when trying to interpret a topic since words like 'show' or 'will' can occur in many different contexts and hence will help little to identify an underlying topic. Assuming that we specified a noisy model with otherwise optimal parameters, the topics might be correctly identified, but nevertheless difficult to interpret since the most important terms are likely to consist of many noisy terms. Hence,

through erasing specifically these noisy terms without losing considerable informational value, one needs to identify the optimal level of noise removal which is done through the preprocessing.

In the following we will compare the three proposed preprocessing models and will pay special attention to the noise removal. Figure 7 shows the most characterizing terms for each of the 15 retrieved topics when performing LDA on the Model 1 corpus. Overall, we get a good impression about the possible latent topics behind the retrieved topics. When considering the respective terms closer, however, we see that words with relatively little information like 'said', 'will', 'like', 'new', 'can', 'now' or 'live' appear rather often among the characterizing terms. Besides not contributing much information, these terms appear across different topics and hence reduce the exclusivity of the topics which introduces distortion. Besides these frequent terms we observe for topic 18 for instance 'iraq/iraqi' and 'iran/iranian' or in topic 3 'isra/israel' as well as 'serb/serbian' and 'bosnia/bosnian' for topic 2 (among others) which from an informational standpoint refer to the same context while being classified as two different words. This results in inefficiencies which are accounted for by lemmatization.

The most characterizing terms for the lemmatized model (Model 2), can be found in Figure 8. We can quickly spot that many of the singular/plural combinations mentioned before disappeared. Overall the topics look quite informative although we can still spot 'say' and 'will' across different topics and must hence conclude that we were not entirely successful in removing the noise.

Considering the terms presented in Figure 9, we see that since we only kept names and nouns, we could drastically reduce the noise level which results in a increased interpretability and a higher degree of exclusivity of the topics. We observe, however, that the term 'government' appears rather often in the individual topics (8/25). This might be explained by the fact that a government usually has a vast area of responsibility and hence will appear as a relevant part of different topics.

6.2 Coherence

The second dimension over which the LDA must be evaluated is that of coherence. Such evaluation is most often carried out at a qualitative level; in particular, a coherent model is defined as one in which the words that define each topic "make sense together". While assessing this is a matter of subjectivity, certain tests do exist: Chang et al. (2009) for instance suggest a "word intrusion test", whereby an external reader is presented with the set of words defining a certain topic, plus one that has been randomly added. If the reader is able to identify the intruder, the topic is deemed coherent. Additionally, we can think of a topic being coherent if its characterizing words point in a unique direction, i.e. there is no ambiguity in the terms that define the topic.

Figure 7: Characterizing terms Model 1 for K=25

These measures of coherence are likely to be correlated with the number of topics K that we choose for our modelling. For instance, increasing the number of topics will most likely lead to better results in terms of the "uniqueness" of said topics, as the characterizing terms will become more and more specific. However, it must be noted that increasing the number of topics has its costs: it must be remembered that, like all models, LDA should deliver a simplified version of the vast amount of underlying information; one that is rather simple to make sense of, especially to someone who does not have a deep knowledge of every single item in the original data. Therefore, a balance between coherence and interpretability must be chosen by the researchers (Yau et al., 2014).

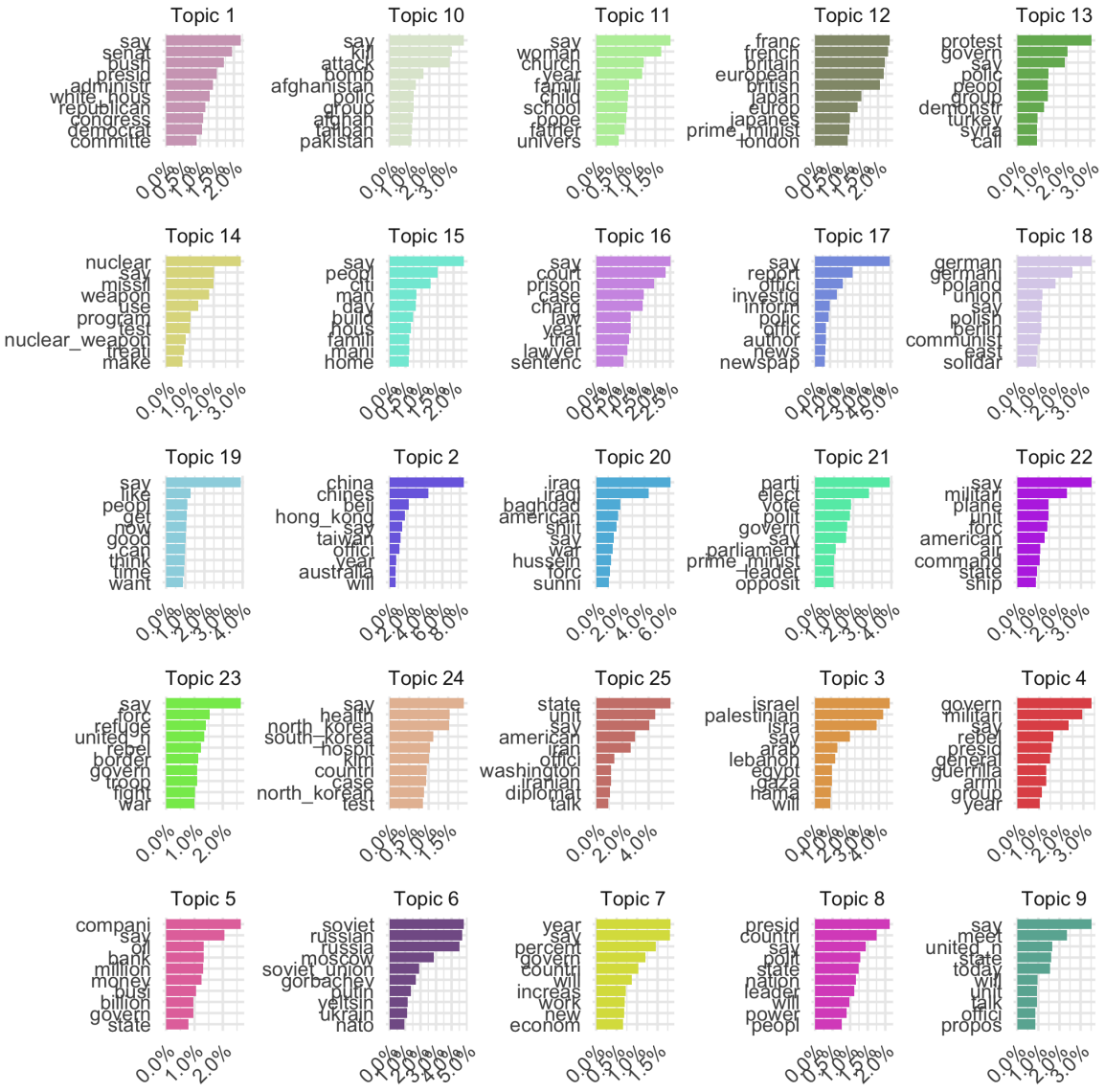


Figure 8: Characterizing terms Model 2 for K=25

In our running example, we focus our attention on three different specifications of Model 3, for $K = 15, 25,$ and 50. Figure 10 reports the case for $K = 15$. We can clearly see that the characterizing words for each topic still tend to be relatively vague, thereby contradicting our "uniqueness" measure. For instance, Topic 12 is characterized by words such as 'government', 'people', 'state', 'country', suggesting that we are dealing with the political domain, but we cannot infer anything about the spatial and temporal context of these terms. Additionally, Topic 5 seems to also violate the "intrusion" measure, as it is hard to see what role the word 'israel' plays in relation to the remaining ones.

Figure 9 displays the top 10 words characterizing our 25 topics. It can be seen that noticeable improve-

Figure 9: Characterizing terms Model 3 for K=25

ments have been made compared to the previous case with respect to how much information is conveyed by these words. For instance, Topic 16 can be immediately be associated to the Israeli-Palestinian conflict, and similarly Topic 23 is very closely associated to the Iraq War. However, we can still see that improvements can be made by increasing the number of topics, as Topic 7 seems to be grouping different conflicts under the same label (the War in Afghanistan with the Yugoslav Wars).

Lastly, we report the case of $K = 50$ in Figure 11. Noticeable improvements can be observed if we return to the aforementioned example about the Afghan and Yugoslav Wars: while in the previous case they were grouped together, they are now assigned to Topics 27 and 22, respectively, and are rather easy to

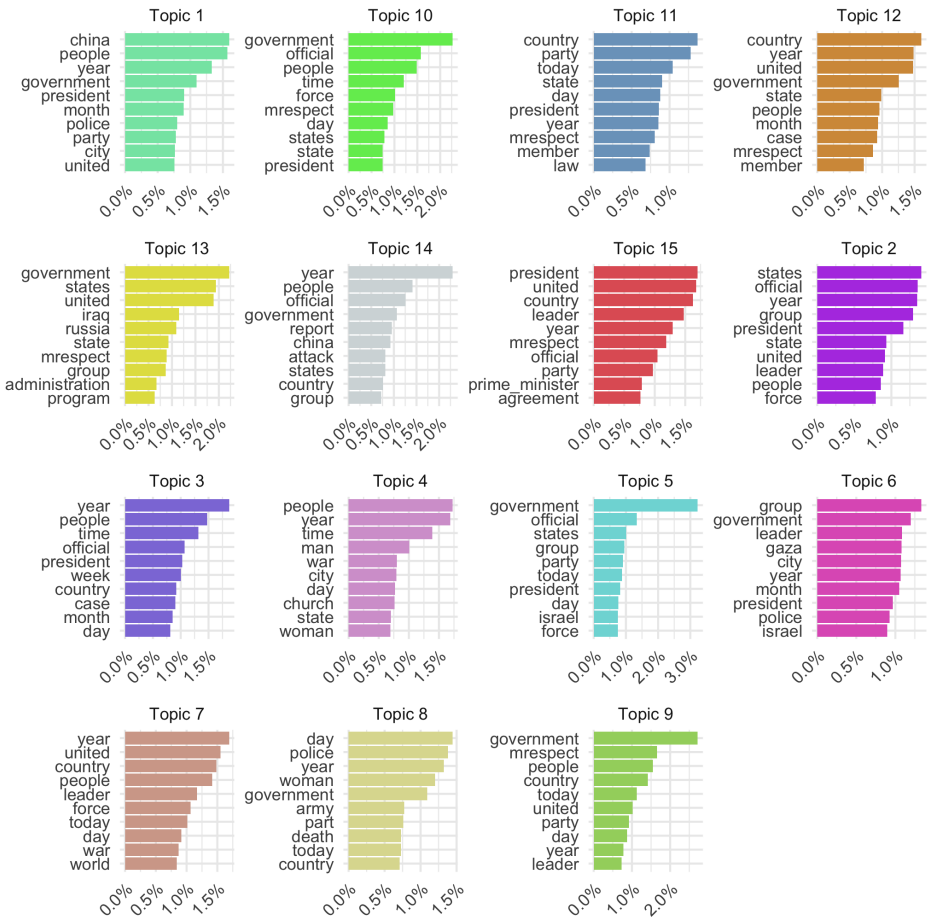


Figure 10: Characterizing terms for 15 topics obtained from Model 3 corpus

label, thanks to characterizing words such as 'afghanistan', 'taliban', 'al_quaeda' on one hand, and 'serb', 'bosnia', 'kosovo', and 'war' on the other. Given these results, we deem the model with $K = 50$ to be superior to the previous specifications, as it offers more informative categorizations, while maintaining a high level of interpretability that does not require in-depth knowledge of the various subjects contained in the news.

Figure 11: Characterizing terms for 50 topics obtained from Model 3 corpus

6.3 Model and reality

Figure 12 offers an interesting insight over the goodness of our categorization. The figure reports how the share of the 25 computed topics varies over the timespan considered; looking at some of the most evident trends and comparing them to the events that took place in those periods can give us an indication of how accurate our interpretation of the data is. We highlight some of these:

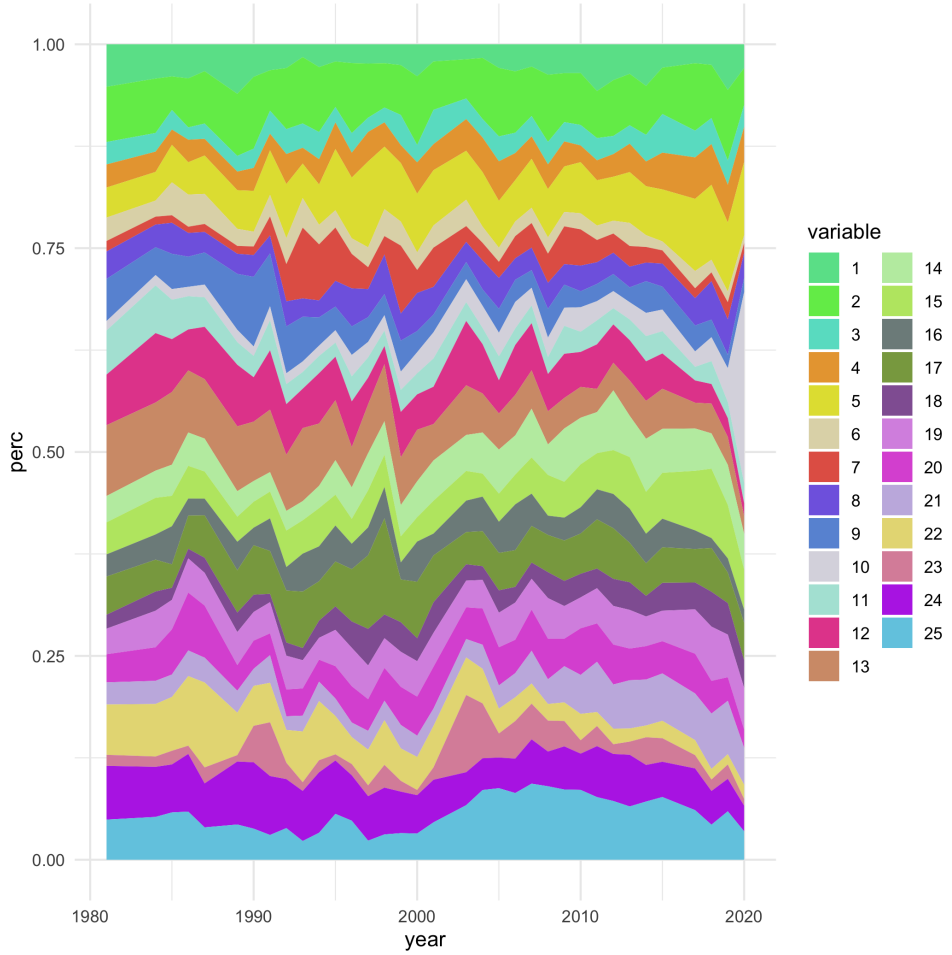


Figure 12: Evolution of the 25 topics' shares over time

- Topic 10, characterized by words such as 'health', 'hospital', 'disease', 'virus', can be seen to take up a major share in the very end of our timespan, in correspondence with the spread of the Covid-19 pendemic (year 2000);
- Topic 23, characterized by words such as 'iraq', 'hussein', 'war', shows two peaks, one around 1990 and one between 2000 and 2005, in correspondence with the First and Second Gulf wars;
- Topic 12, denoted by words such as 'force', 'troops', 'army', 'guerrilla', 'commander' shows a very sharp drop after 2015, which coincides withe the peace treaty signed between the Colombian rebel groups and the government, ending a civil conflict that we can see has been steadily present before then.

References

Blei, D. M., A. Y. Ng, and M. I. Jordan (2003). "Latent dirichlet allocation". In: *the Journal of machine Learning research* 3, pp. 993–1022.

Chang, J., S. Gerrish, C. Wang, J. Boyd-graber, and D. Blei (2009). "Reading Tea Leaves: How Humans Interpret Topic Models". In: *Advances in Neural Information Processing Systems*. Ed. by Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta. Vol. 22. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2009/file/f92586a25bb3145facd64ab20fd554ff-Paper.pdf.

Griffiths, T. L. and M. Steyvers (2004). "Finding scientific topics". In: *Proceedings of the National academy of Sciences* 101.suppl 1, pp. 5228–5235.

Martin, F. and M. Johnson (Dec. 2015). "More Efficient Topic Modelling Through a Noun Only Approach". In: *Proceedings of the Australasian Language Technology Association Workshop 2015*. Parramatta, Australia, pp. 111–115. URL: https://www.aclweb.org/anthology/U15-1013.

Suominen, A. and H. Toivanen (2016). "Map of science with topic modeling: Comparison of unsupervised learning and human-assigned subject classification". In: *Journal of the Association for Information Science and Technology* 67.10, pp. 2464–2476.

Yau, C.-K., A. Porter, N. Newman, and A. Suominen (2014). "Clustering scientific documents with topic modeling". In: *Scientometrics* 100.3, pp. 767–786. ISSN: 1588-2861. DOI: 10.1007/s11192-014-1321-8. URL: https://doi.org/10.1007/s11192-014-1321-8.