

---

# INCORPORATING BLOGS IN POLLUX

---

TECHNICAL REPORT

**Tobias Holtdirk\***

GESIS – Leibniz Institute for the Social Sciences  
Unter Sachsenhausen 6-8  
50667 Köln  
tobias.holtdirk@rwth-aachen.de

**Nina Smirnova**

GESIS – Leibniz Institute for the Social Sciences  
Unter Sachsenhausen 6-8  
50667 Köln  
nina.smirnova@gesis.org

March 27, 2024

## ABSTRACT

This technical report describes the incorporation of political blogs into Pollux, the Specialised Information Service (FID) for Political Science in Germany. Considering the widespread use of political blogs in political science research, we decided to include them in the Pollux search system to enhance the available information infrastructure. We describe the crawling and analyzing of the blogs and the pipeline that integrates them into the Pollux system. To demonstrate the content of the incorporated blogs, we also provide a visualization of the topics covered by the blog posts during the first three months following integration.

**Keywords** political blogs · data crawling · political science · social media

## 1 Introduction

Political blogs are widely used in political science research (Akinnubi & Agarwal, 2023; Coleman & Wright, 2008; Peng et al., 2023; Wallsten, 2007, 2008; Wright, 2009). Coleman and Wright (2008) claim that communication through political blogs might enhance the responsibility and openness of governance overall. Peng et al. (2023) examined the process of opinion formation and evolution by analyzing a larger social network of political blogs. Wallsten (2008) investigated how political bloggers utilize their platforms, focusing on their predominant function: expressing opinions, mobilizing, seeking feedback, or disseminating information. Pettersson and Sakki (2020) studied the usage of political blogs by right-wing politicians for political communication and persuasion. Balakhonskaya et al. (2020) examined strategies for discrediting opponents in the Russian political blogosphere.

Considering the broad usage of political blogs in political science research, we decided to include political blogs in the search index of Pollux<sup>2</sup>. Pollux is the Specialised Information Service (FID) for Political Science, which provides literature and information infrastructure in the field of political science in Germany. The following technical report explains the crawling procedure and analysis of political blogs we included in our collection. Section 2 explains the generation of the list of RSS feeds used for incorporating blogs in Pollux. Section 3 describes findings from the collected RSS feeds that motivated the decision of how to implement them into Pollux. Section 4 provides a detailed description of the procedure that handles the integration of blogs in Pollux. Section 5 shows visualizations of the topics included in the incorporated blog posts.

## 2 Sourcing of RSS Feeds

In this section, we describe the generation of the RSS feeds list used to incorporate blogs in Pollux. A graphical overview of the process is provided in Figure 1.

---

\*Corresponding author

<sup>2</sup><https://www.pollux-fid.de/>

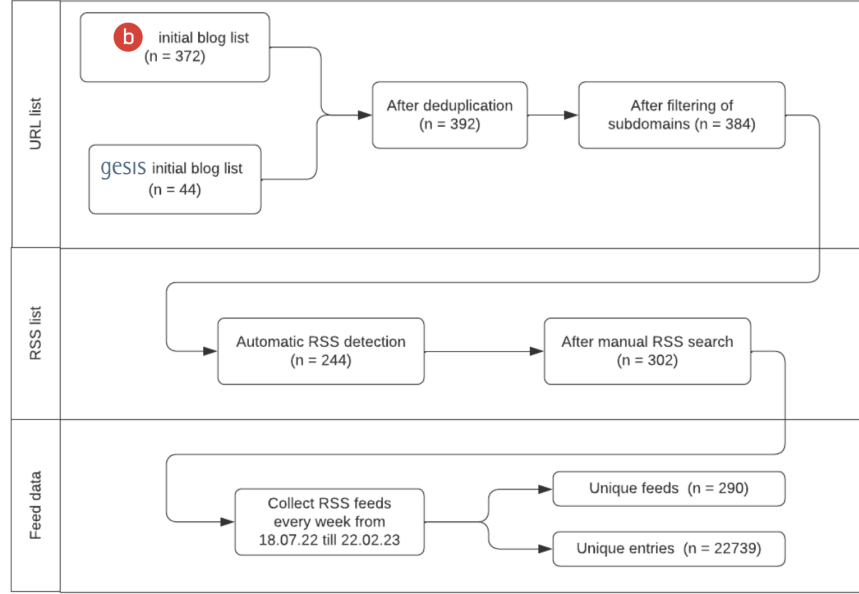


Figure 1: Pipeline for initial blog list generation.

We use lists of URLs with relevant political science blogs provided by SUUB and GESIS. The lists are parsed, tested for dead links, and then filtered to deduplicate and fix subdomain inconsistencies between the lists.

To get an RSS list from the URL list, we designed a Python script for identifying RSS feeds on web pages. It reads a list of URLs from a CSV file, sends a GET request to each URL, and uses a regular expression to find any RSS feed links in the HTML of the page. The regular expression is designed to match the href attribute of link elements with rel="alternate" and type="application/rss+xml", which is a common way to specify an RSS feed on a webpage. Finally, a dictionary containing the original URL, the response status code, the content type of the response, and the list of RSS feed links is written to a JSON file, which is used for the analysis of the RSS feeds, as well as the later integration into the Pollux database.

After the automatic RSS feed detection, the remaining feeds are checked manually. Like in the automatic checking, the HTML of the URL is used. We look for broader patterns like an "rss" string appearing in a link on the page. After that, we check the remaining URLs by visiting the website and searching for visual information like an RSS feed logo. The additional RSS feeds identified this way are added to the automatically detected ones, providing a JSON file as seen in Figure 2.

```

1  [
2    {
3      "url": "http://blog.yalebooks.com/category/current-affairs/political-science/",
4      "status_code": 200,
5      "content_type": "text/html; charset=UTF-8",
6      "rss_links": [
7        "https://yalebooks.yale.edu/category/current-affairs/political-science/feed/",
8        "https://yalebooks.yale.edu/feed/",
9        "https://yalebooks.yale.edu/comments/feed/"
10     ]
11   },
12   {
13     "url": "http://www.demokratie-goettingen.de/verzeichnis/blog",
14     "status_code": 200.0,
15     "content_type": "text/html; charset=UTF-8",
16     "rss_links": [
17       "https://www.demokratie-goettingen.de/verzeichnis/blog/feed"
18     ],
19     "error": null
20   },
21 ]

```

Figure 2: Two entries in JSON after retrieving RSS feeds from URLs.

To analyze the feeds later, we downloaded the RSS feeds each week for seven months, from July 2022 to February 2023. We designed a Python script for this purpose that was executed once per week. The script downloads and saves RSS feeds from the list generated earlier. The feed content and metadata, such as the timestamp, status code, and content type, were saved. If an error occurs during a request, the request is retried multiple times.

### 3 Analysis of Initial Blog Data

After generating the initial feed data, we analyzed the resulting 290 feeds and 22,739 entries to identify which metadata is available and assess the quality of the available metadata.

For an overview of the feed-level data we parsed from the RSS feed, see Table 1. At the feed level, only a little metadata is transmitted. However, the available metadata is included in almost all feeds and can, therefore, be used as information in Pollux. The fields mostly contained high-quality data, with only the *subtitle* field showing some inconsistencies, as some feeds had considerably longer entries than others. Therefore, we were able to use all the shown field data in the Pollux integration.

Table 1: **Feed Structure.** XML elements (fields) that were most often included in the RSS responses at the feed level.

Field	Inclusion	Example
title	100%	PoliSciZurich
subtitle	99%	A blog by political scientists in Zurich
blog url	100%	<a href="https://poliscizurich.wordpress.com">https://poliscizurich.wordpress.com</a>
rss url	100%	<a href="https://poliscizurich.wordpress.com/feed/">https://poliscizurich.wordpress.com/feed/</a>
last updated	92%	Wed, 17 Aug 2016 03:54:00 +0000
language	89%	en-US

For an overview of the entry-level data we parsed from the RSS feed, see Table 2. The entry level has more metadata available, sometimes even including the whole blog post in the content field. However, the quality and availability of data are worse. The *title* is always available and has mostly good quality, with some titles being overly long and some not including the actual title but rather a "not available" statement. Important metadata like *link* and *publication date* also had good quality. Other metadata was more spotty; *content* often includes HTML artifacts and had inconsistent content, with some entries having their entire blog post in the field and others just the first sentence. On the other hand, the *summary* field is available for all entries and has more consistent quality. Therefore, we use the summary rather than the content as the displayed "abstract" in Pollux. The *tags* and *comments* fields provide interesting metadata. With tags, the blog entries can be categorized into different subsets, and comments can be used to build some kind of popularity metric. However, comments have a very low inclusion rate at 33% and are therefore not reliable enough for a metric. The amount and kind of tags differ heavily for different entries, making them less valuable than an automatic topic generation based on the summary.

Table 2: **Entry Structure.** XML elements (fields) that were most often included in the RSS responses at the entry level.

Field	Inclusion	Example
title	100%	The 2022 Midterms: In the Senate elections, [...]
id	100%	<a href="https://blogs.lse.ac.uk/usappblog/?p=47109">https://blogs.lse.ac.uk/usappblog/?p=47109</a>
link	100%	<a href="https://blogs.lse.ac.uk/usappblog/2022/11/">https://blogs.lse.ac.uk/usappblog/2022/11/</a> [...]
publication date	96%	2022-11-16 09:57:58
authors	84%	Blog
summary	100%	In this year&\#8217;s midterm elections, [...]
content	65%	a class="a2a_button_twitter" href="https: [...]
tags	65%	2022 Midterms', 'Elections and party politi [...]
comments	33%	<a href="https://blogs.lse.ac.uk/usappblog/2022/11/16">https://blogs.lse.ac.uk/usappblog/2022/11/16</a> [...]

Since we have high-quality data both at the feed and entry levels, we can use them for two distinct types of records in Pollux. As Pollux is designed as a database for academic research, we can use the preexisting structure of paper and journal records as a template for entry and feed records.

## 4 Incorporation into the Pollux Pipeline

This section provides a detailed description of the pipeline that handles the integration of blogs in Pollux.

### 4.1 Downloading RSS Feeds

The Python script, `rss_downloader.py`, is designed to download and store RSS feeds from a list of URLs. It does this through a series of functions that each handle a specific part of the process.

The `run` function is the main function that gets called to initiate the process. It reads a list of RSS feed URLs, shuffles the list to avoid querying the same domain consecutively, and then attempts to download each feed. If a feed is successfully downloaded, its metadata is stored. If a feed cannot be downloaded, its URL is added to a list of error URLs. The function then saves the metadata of all downloaded feeds to a JSON file and returns a `Result` object containing metrics about the number of feeds downloaded and the number of errors, as well as the error URLs.

Multiple helper functions are used in the process. The `extract_feed_urls` function reads in a JSON file containing blog information and extracts a list of unique RSS feed URLs. It expects the JSON file to contain a list of dictionaries, each with a key `rss_links` that maps to a list of URLs. The function returns the list of unique URLs.

The `load_rss` function handles the HTTP request that attempts to download an RSS feed from a given URL and save it to a specified directory. It first sends a GET request to the URL. If the response's content type is not XML, the function raises a `ValueError`. Otherwise, it saves the response's content to a file in the specified directory and returns a dictionary containing the URL, the timestamp of the download, the filename, the status code of the response, and the content type.

The `get` function handles the GET request used above. It sends a GET request to a given URL with specified headers and URL parameters. If the request is successful and the status code of the response is OK, the function returns the response. If the status code is a client error (400-499), the function raises a `ValueError`. If the status code is a server error (500-599), the function waits for a certain amount of time and then tries to send the request again. If the request fails three times, the function raises a `ValueError`.

### 4.2 Converting RSS Feeds

The Python script, `rss_converter.py`, is designed to parse and convert RSS feeds and their entries into the format used by Pollux entries. It does this through a series of functions that each handle a specific part of the process.

The `run` function is the main function that gets called to initiate the process. It reads in a dump of RSS feeds, parses the feeds and their entries, splits the feeds and entries into blog and comment data, converts the blog feeds and entries into a specific format, and returns the converted data along with some metrics and logs.

The `parse_rss_dump` function parses the RSS feeds and their entries. It adds additional information from the metadata to each feed and entry. The `split_comments` function separates the blog data from the comment data based on the URL of the feed or entry.

The `convert_feed` and `convert_entry` functions convert a feed or an entry into the Pollux entry format. They call several helper functions to get specific pieces of information from the feed or entry. Ten different helper functions in the format `get_<field>` parse the blog information. For example, the `get_languages` function converts the BCP 47 language code found in the RSS to an ISO 639-3 language code. For detailed information on the functions, see the accompanying source code.

## 5 Topics of incorporated Blog Posts

After the integration of the pipeline into Pollux (see Appendix A and Czolkoß-Hettwer and Pfeifenberger (2023) for the website layout), we analyze the incorporated blogs. Integration started in July 2023, and we analyzed blog entries until October 2023. We analyze the topics of the blog entries by fitting a topic model on the `summary` field of the entries. Before fitting the model, we translated all summaries into English to make the topic model consistent, as we want to focus on the areas the blogs cover instead of language differences. We additionally use the `publication_date` field to visualize topics over time.

**BERTopic.** BERTopic (Grootendorst, 2022) is a topic modeling technique that combines BERT (Bidirectional Encoder Representations from Transformers) with classical topic modeling methods. It leverages the power of BERT embeddings to represent documents and then applies topic modeling algorithms to discover latent topics within the

document collection. BERTopic provides several methods for visualizing the discovered topics and the documents associated with them. We visualize document-level relationships and topics over time.

**Document visualization.** Using the fitted topic model, we visualize the documents in the context of the discovered topics, see Figure 3. We plot the documents in a two-dimensional space, representing each as a point. The position of the documents is determined by their topic distribution, such that documents with similar topic distributions will be clustered together. This visualization helps understand the relationships between documents based on their topic assignments. We can see that US-centric topics like the Supreme Court, monetary policy, and AI advances dominate the visualization’s upper half. The middle includes topics such as the coronavirus and climate change, which matter globally. The bottom half includes topics concerning the European Union, German newspapers, and the war in Ukraine, which are more relevant in Europe.

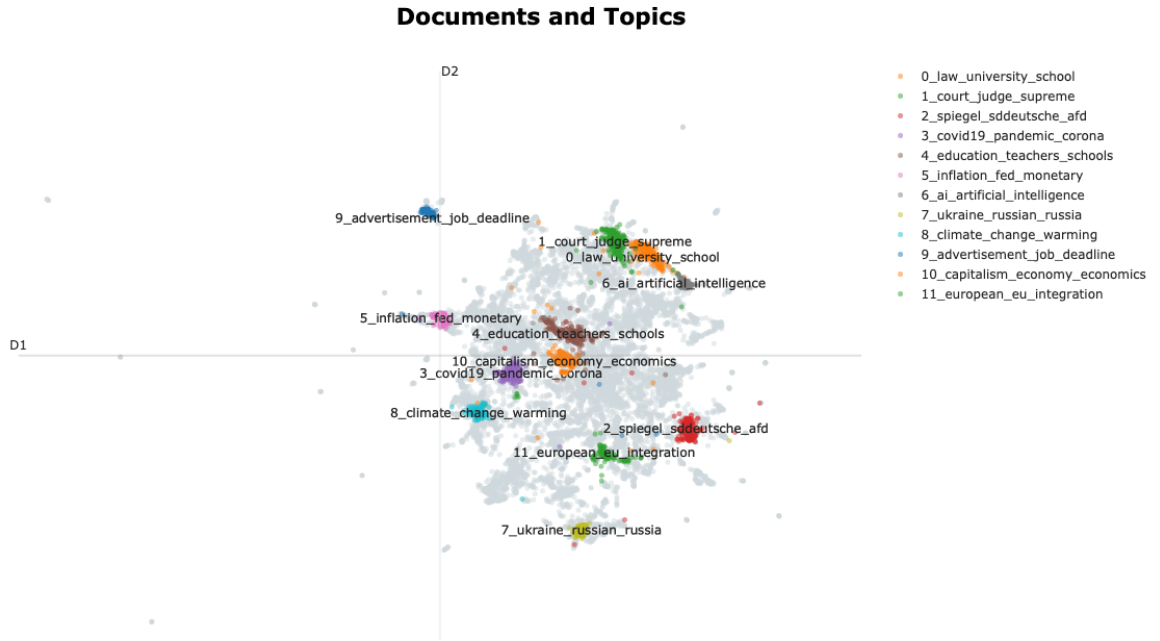


Figure 3: BERTopic visualization of blog entries incorporated in Pollux. The scatterplot shows embeddings for each blog entry reduced to a 2-dimensional space. The summaries of the entries were used to create the embeddings. (Interactive plot at [https://tobihol.github.io/pollux-rss-blogs/content\\_analysis/topic\\_viz\\_2d.html](https://tobihol.github.io/pollux-rss-blogs/content_analysis/topic_viz_2d.html))

**Over time visualization.** We use the publication date of the blog entries to show the trend of topics over time; see Figure 4. The dates are grouped into two-month intervals to make the plot more interpretable. The number of blogs included for dates before the incorporation into Pollux differs from blog to blog, as the number of records per RSS request is individual for each feed, and the frequency of new blogs per feed varies widely. For the visualization, we pick topics that should vary in relevance over time. The blogs cover important political events like the coronavirus pandemic or the war in Ukraine. We see the expected temporal development, where COVID is the dominating topic throughout 2020 and 2021, with events like the Russian invasion of Ukraine spiking at the start of 2022 and the AI-related blogs increasing around the time of the release of ChatGPT in November 2022.

## Acknowledgements

This work was funded by Deutsche Forschungsgemeinschaft (DFG) under grant number MA 3964/7-2, the POLLUX project. We thank Marie-Saphira Flug for supporting and reviewing the process of integrating the blog pipeline into Pollux. We thank Philipp Mayr for motivating and supporting the write-up of this technical report.

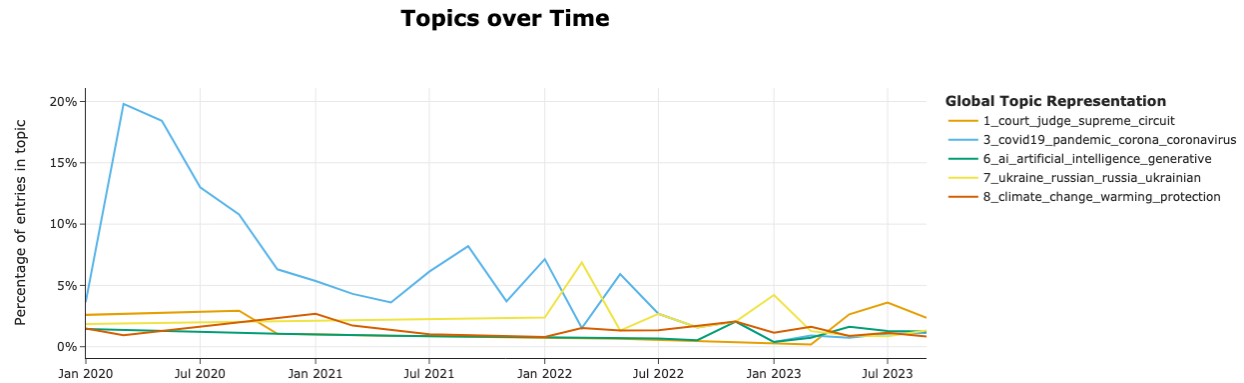


Figure 4: BERTopic visualization of blog entry topics over time. The topics are based on the summaries of the entries. (Interactive plot at [https://tobihol.github.io/pollux-rss-blogs/content\\_analysis/topics\\_over\\_time.html](https://tobihol.github.io/pollux-rss-blogs/content_analysis/topics_over_time.html))

## References

- Akinnubi, A., & Agarwal, N. (2023, April 5). *Deliberative democracy, perspective from indo-pacific blogosphere: A survey*. <https://doi.org/10.31219/osf.io/8nj7y>
- Balakhonskaya, L. V., Strelchenko, V. I., Balakhonsky, V. V., Sadretdinova, T. A., & Beresneva, I. V. (2020). Communicative strategy of discrediting opponents in the russian political blogosphere. *2020 IEEE Communication Strategies in Digital Society Seminar (ComSDS)*, 27–33. <https://doi.org/10.1109/ComSDS49898.2020.9101281>
- Coleman, S., & Wright, S. (2008). Political blogs and representative democracy. *Information Polity*, 13(1), 1–6. <https://doi.org/10.3233/IP-2008-0140>
- Czolkoss-Hettwer, M., & Pfeifenberger, R. (2023, July 25). *Politikwissenschaftliche Blogs sichtbar machen*. Deutsche Vereinigung für Politikwissenschaft. Retrieved March 20, 2024, from <https://www.dvpw.de/blog/politikwissenschaftliche-blogs-sichtbar-machen-ein-beitrag-von-michael-czolkoss-hettwer-und-regina-pfeifenberger>
- Grootendorst, M. (2022, March 11). *BERTopic: Neural topic modeling with a class-based TF-IDF procedure*. arXiv: 2203.05794 [cs]. <https://doi.org/10.48550/arXiv.2203.05794>
- Peng, Y., Zhao, Y., & Hu, J. (2023). On the role of community structure in evolution of opinion formation: A new bounded confidence opinion dynamics. *Information Sciences*, 621, 672–690. <https://doi.org/10.1016/j.ins.2022.11.101>
- Pettersson, K., & Sakki, I. (2020). Analysing multimodal communication and persuasion in populist radical right political blogs. In M. A. Demasi, S. Burke, & C. Tileagă (Eds.), *Political communication* (pp. 175–203). Springer International Publishing. [https://doi.org/10.1007/978-3-030-60223-9\\_7](https://doi.org/10.1007/978-3-030-60223-9_7)
- Wallsten, K. (2007). Agenda setting and the blogosphere: An analysis of the relationship between mainstream media and political blogs. *Review of Policy Research*, 24(6), 567–587. <https://doi.org/10.1111/j.1541-1338.2007.00300.x>
- Wallsten, K. (2008). Political blogs: Transmission belts, soapboxes, mobilizers, or conversation starters? *Journal of Information Technology & Politics*, 4(3), 19–40. <https://doi.org/10.1080/19331680801915033>
- Wright, S. (2009). Political blogs, representation and the public sphere (B. Guner, Ed.). *Aslib Proceedings*, 61(2), 155–169. <https://doi.org/10.1108/00012530910946901>

## A Representation of Blogs on the Pollux Website

**Search** | Advanced Search Search tips ?


klimakrise Q

Alert from search 🔔

57 results Sort by: Relevance Year

Blog Entries ✕

Blog Entry ⓘ February 13, 2023 #1

 **Wie umgehen mit der Klimakrise?**

Blog: [Politikwissenschaft – Kohlhammer Blog](#)

[Frank Wagner](#)


**Abstract ▼**

Access Blog Entry

Export

---

Blog Entry ⓘ September 22, 2023 #2

 **Ziviler Ungehorsam in der Klimakrise**

Blog: [Verfassungsblog](#)

[Maxim Bönnemann](#)

**Abstract ▼**

Access Blog Entry

Export

**FORMAT**

electronic	57
print	

**TYPE**

Blog Entries	57
Articles	
Blogs	
Book Chapters	
Books	
Datasets	
Journals	
Newspaper	
Open Access	

**LANGUAGE**

German	30
English	3
Danish	
Norwegian	

**TIME RANGE**

1924

→

1944

>

1945

0

Figure 5: View of the search interface on Pollux when searching for "klimakrise".

Search | [Advanced Search](#)[Search tips](#) Blog Entry 

February 13, 2023

## Wie umgehen mit der Klimakrise?

Blog: [Politikwissenschaft – Kohlhammer Blog](#)

Frank Wagner

[Access Blog Entry](#)

### Abstract

Emil Kowalskis Essay vertritt die Auffassung, dass neben unseren Überlebensbedingungen in der Klimafrage ein weiteres hohes Gut auf dem Spiel steht: die liberale Demokratie. Zu oft werde übersehen, dass ein breit verteilter Wohlstand deren Voraussetzung ist. Vor diesem Hintergrund plädiert Kowalski eindrücklich für einen technologie-offenen Umgang mit dem Klimawandel. Denn mit unserem Wohlstand schützen wir zugleich unsere Freiheit. Lesen Sie erste Eindrücke in unserem Interview mit dem Autor. [Weiterlesen →](#)

### LANGUAGES

German

 [Export](#) [Report Issue](#)

Figure 6: View of a blog entry on Pollux.