Inference of modular structures in dynamical systems and the application to Telegram data

by

Sebastian B. Mohr

A document submitted in partial fulfillment of the requirements for the degree of

Master of Science

at

University of Göttingen

Supervised by *Prof. Dr. Viola Priesemann*

Abstract

Social media platforms like Telegram have transformed the way information is consumed and shared between people. Amid the COVID-19 pandemic, unregulated information exchange on Telegram has led to an increased spread of false or misleading information, with unknown impact on the course of the pandemic. Analyzing the structural traits of Telegram channel networks could offer crucial insights into information dynamics. Yet, limited public data availability and the platform's closed nature present substantial challenges for investigation. This thesis presents a novel dataset of COVID-19-related Telegram channels, collected using a custom-built Telegram crawler, and a comprehensive analysis of its structural characteristics. The dataset contains over 128k channels and to our knowledge comprises the largest Telegram dataset to date. Structural analysis reveals a hierarchical organization within the network, with distinct modules and submodules. The hierarchical representation reveals language patterns, prevalent topics, and COVID-19 relatedness, providing the first step for a nuanced understanding of the underlying structure. The results significantly contribute to the understanding of the Telegram network's structural characteristics and provide a valuable resource for future investigations. The dataset and analysis serve as a foundation for further research, fostering a deeper understanding of the dynamic landscape of online communication, misinformation, and social interactions.

Contents

1 Introduction

In the digital age, social media has become a ubiquitous part of our lives. It has revolutionized the way we communicate and interact with each other. Not only has it changed the way we connect with other people, be they friends, family, or strangers, but it also allows us to communicate with people all over the world in real-time. Moreover, social media has changed the way we consume news and information in general. Instead of relying on traditional media outlets, such as newspapers, radio, or television, we increasingly access information through social media platforms. For instance, instead of waiting for the morning newspaper or the evening news broadcast, many people now get their news by scrolling through their social media feeds[1].

This shift in news consumption is often criticized as social media platforms are not held to the same standards as traditional media outlets. Social media platforms lack the quality control and fact-checking mechanisms of traditional media outlets. This might be due to the sheer amount of content being shared on these platforms and the providers' inability to keep up with the moderation of the content.

Among the variety of social media platforms, Telegram has emerged as a significant player reporting 700 million monthly active users[2]. It was first mainly known as an instant messenger but has evolved into a hybrid between a messenger and a social media platform. Telegram boasts unique policies, including encrypted messaging, group chat capabilities that can be similar to a public forum, and a sense of anonymity that some users find appealing. As Telegram proudly advertises it has never shared data with a government agency and will not do so in the future [3], it is a popular platform for privacy-conscious users. For instance, Telegram has been used by activists and journalists to communicate securely and anonymously [4–6].

However, this same lack of regulation and accountability has also attracted criminals and extremists, who use Telegram to organize and coordinate their activities but also to host dialog on extremist topics. Telegram is the major platform for US extremists [7, 8], for instance, right-wing US extremists used Telegram to organize the storming of the US Capitol in 2021 [9]. In Germany, the far right and corona protesters are also heavily relying on Telegram [10]. Further, Telegram is also used as a marketplace for restricted goods [11].

During the COVID-19 pandemic, a surge in active Telegram users [2] prompted a flood of pandemic-related discussions, from measures and vaccinations to symptoms and treatments.

[1] According to the Eurobarometer, a survey on public opinion in the European Union, 28% of people in Europe have not read any written press in 2022 [1]

[2] These 2022 numbers are an increase of 200% in comparison to pre-pandemic numbers in 2018 [2]

Consequently, a variety of often contradictory information was shared, ranging from conspiracy theories to scientific studies. Even though not all of this information is wrong per se, it is still often misinforming and/or misleading. This spread of misinformation posed challenges for public authorities, who had to counteract it while educating the public about the pandemic. The European Commission approved the Digital services act (DSA) in 2022 [12], which imposes legal responsibilities on online platforms and intermediaries, including measures to combat the spread of false information and harmful content. However, the DSA impact on the COVID-19 pandemic misinformation is arguably small, as the DSA was introduced after the pandemic had already been ongoing for several years. Further Telegram is just not big enough to fall under the scope of the DSA as it only applies to platforms with more than 45 million users in the EU [12, 13] and Telegram has self-reported 38.5 million users in the EU [2].

The lack of regulatory responsibility for social media be it in terms of quality control and fact-checking during the COVID-19 pandemic has resulted in a significant increase in both *COVID-19* cases and fatalities, a phenomenon aptly characterized as the "Infodemic" [14–17]. Understanding how misinformation is shared between individuals and how these individuals form communities to further share this misinformation is crucial to understand the impact of social media and misinformation on society. Not only to retrospectively understand the impact of misinformation on big or even global events such as the COVID-19 pandemic or elections, but also to give us the tools to combat the spread of misinformation in the future.

To enhance our understanding of misinformation and its impact on society, we must navigate the intricate landscape of social media networks. These platforms can be perceived as intertangled webs of users, communities, and the content they share, collectively shaping a dynamic network. Communities, as integral components within this network, play a crucial role in shaping the dissemination of information and influencing user interactions. They may arise organically, as users form groups based on shared interests, common goals, or linguistic commonalities. Communities may share similar characteristics, such as the type of content they share, the frequency of their interactions, or their size. These characteristics can be used to identify and classify communities within the network, providing valuable insights into the dynamics of information flow.

Within the specific context of Telegram, the formation of communities might be a multilevel process. At the foundational level, communities materialize as users join groups or channels. Expanding to a higher level, these groups and channels may further interconnect and form structures by sharing similar interests or just speaking the same language. This yields modular structures which describe discrete functional distinct components within the network on multiple levels.

This potential hierarchical structure of modular components within a social media platform is of particular interest, especially as it's plausible to hypothesize the existence of structured and/or organized groups dedicated to seeding misinformation. They maliciously share information with the intention of distributing it to a large audience. When thinking about spreading information with political or ideological motives, for instance, to manipulate public opinion is plausible. There is some evidence that such influencing campaigns are already being deployed in large-scale operations. For instance, the Russian Internet Research Agency allegedly created fake accounts on social media platforms to represent themselves as US citizens and spread information to influence the 2016 US presidential election [18].

Understanding the dynamics between these modular components in general and how they emerge and interact with each other is crucial to understanding the impact of social media and misinformation on society. Not only for countering the impact of misinformation during a crisis like the pandemic but also for safeguarding the integrity of information ecosystems in the future.

1.1 Notes on the structure of this thesis

This thesis is structured into three main sections. The initial section, which you are currently reading, serves as an introduction, presenting the overarching topic, and the central research question and gives some background information. The subsequent section focuses on the analysis of Telegram data, encompassing discussions of outcomes and a summary of key findings. The final section delves into the methods employed throughout this thesis. Placing the methods section towards the end of the thesis is a deliberate choice, as it may prove challenging for some readers without a background in statistical physics. Nevertheless, comprehending these methods is pivotal, as they underpin the results presented in the second section.

This document is designed to be read in a double-page format, mimicking the layout of a traditional book. For an optimal reading experience, consider viewing the pages in pairs, with the page numbers located on the outer edges.

1.2 Background

This section provides essential insights into the contextual background of this thesis, focusing on social media, particularly Telegram. Further, we lay the groundwork for the central theme of modular structures and their significance in community detection.

The background serves as a foundation for the subsequent exploration, designed to be accessible to a broad audience. For mathematical details please refer to the methods, see Chapter 4.

1.2.1 Telegram's Unique Landscape: Unregulated Environment, Privacy Emphasis, and Structural Distinctions

Telegram stands out from other social media platforms for several distinctive reasons. Unlike many mainstream platforms, Telegram operates in a mainly unregulated and unmoderated environment. It lacks fact-checking, censorship, and content moderation mechanisms that are seen on other social media platforms like Facebook, Reddit and Twitter. Further, Telegram is marketing itself as a privacy-focused platform, which is in stark contrast to the data collection practices of other social media platforms. Telegram stores data in multiple data centers around the world, making it difficult for governments to enforce law compliance as data is distributed over multiple jurisdictions. According to Telegram, they have never shared any data with any government agency [2].

The growth in Telegrams' popularity and its unique features have led to it becoming a focal point for discussions about misinformation and disinformation. The platform's unregulated and unmoderated environment, combined with its commitment to user privacy, has created an environment where the spread of misinformation can occur relatively unchecked. Researchers face challenges in obtaining data for analysis due to Telegram's emphasis on privacy. Unlike more accessible platforms, Telegram does not give researchers access to an API for data collection.

Telegram's structure is built around the core elements of groups, channels, and users, creating a versatile and user-friendly communication platform. Groups serve as hubs for interactive discussions, allowing multiple users to participate in real-time conversations. They can be public or private, fostering various levels of engagement. Channels, on the other hand, are ideal for broadcasting information to a wide audience. They function as one-way communication streams, making them perfect for news outlets, businesses, or individuals looking to share content with their followers. Depending on the channel's settings, users can interact with the content by liking, commenting, and sharing by forwarding a message to another user, group, or channel. Lastly, users are at the heart of the Telegram ecosystem, connecting with others through individual chats and forming relationships within groups and channels. The multi-faceted nature of Telegram's structure gives users the flexibility to choose how they want to engage with content and connect with others on the platform, making Telegram a dynamic and adaptable messaging service but also a versatile social media platform.

Telegram's unique design sets it apart from other social media platforms in several ways. Unlike many other networks, Telegram does not have a large-scale search engine[3], making it challenging to discover specific groups and channels of interest. Users must rely on alternative methods, such as sharing links within other channels or relying on word-of-mouth recommendations, to access niche or lesser-known content. This means that discovery on Telegram is often driven by the platforms' organic growth and community-driven sharing, rather than a centralized search algorithm. While this can make it more difficult to find content initially, it also cultivates a sense of community and encourages users to engage with content they discover, creating a different dynamic compared to platforms with robust search features.

While the research community has already created datasets for studying Telegram, such as the TGDataset [19] or the Pushift dataset [20] these datasets lack specificity or were not available at the start of this thesis. The TGDataset is the largest publicly available Telegram dataset, containing ≈120k channels, including messages. It contains a broad mix of content, for in-

[3] The existing search is very limited only showing little results and is not searching message content.

stance, it consists of 23.9% religious content (4725) and 8.7% COVID-19-related channels (1716). On the other hand, the Pushift dataset is small compared to the TGDataset, containing only 28k channels. These channels were collected before 2020, thus not including a majority of COVID-19-related channels.

While the existing datasets offer valuable resources for studying Telegram, they fall short in addressing the specific requirements of our further research. Our focus on understanding the dynamics of misinformation and disinformation related to COVID-19 within the Telegram ecosystem necessitates a more specialized dataset. The TGDataset, while extensive, lacks the granularity needed for our investigation, with a broad mix of content that extends beyond our research scope. Furthermore, the Pushift dataset, though useful for historical context, does not capture the surge in COVID-19-related channels that emerged after 2020. Given the dynamic nature of information dissemination during a global pandemic, it is crucial to have up-to-date and relevant data for a comprehensive analysis.

1.2.2 Modularity and Community Structures in Telegram

Given Telegrams' unregulated nature and the challenges associated with data access, studying the platform presents a set of intriguing research opportunities and complexities. In particular, we are interested in the structure of the network formed by Telegram users and their interactions. Of special interest are the communities that emerge organically, comprising individuals, channels, and groups. Within this context, we explore the concept of modular structures as it pertains to the segmentation of Telegram users, channels, or groups into discrete, functionally distinct components.

Modularity refers to the property or characteristic of a system, structure, or network being composed of distinct, self-contained, and interrelated components or modules. These modules are designed to perform specific functions or tasks while maintaining a degree of independence. Depending on the context and application, modularity can take various definitions mainly depending on the field of study. For instance, in biology, modularity refers to the ability of an organism to be divided into discrete functional units. In computer science, modularity refers to the ability of a system to be divided into independent modules. In network analysis, modularity refers to the ability of a network to be divided into distinct communities or modules. Most definitions of modularity share the common theme of dividing a system into discrete, functionally distinct modules.

In the literature on network analysis, modularity is predominantly recognized as an approach to community detection, where one maximizes a modularity score by maximization. It is a quite popular approach, as it is easy to implement and understand. However, modularity maximization suffers from a variety of serious conceptual and practical flaws, which have been documented extensively [21–24]. Its usage can be considered harmful for a variety of reasons, including overfitting, resolution limit, and sensitivity of the result to the size of the network. However the underlying idea of modularity, namely to divide a network into discrete and functionally distinct components, still gives valuable insights into the structure of the network.

Recognizing the limitations of modularity maximization, researchers have turned their attention to alternative methods like Stochastic blockmodels (SBMs) which similarly aim to divide a network into discrete and functionally distinct components. SBMs offer a probabilistic framework for modeling complex networks, and they have gained traction for their ability to address most of the shortcomings of the modularity-maximization approach. E.g. using prior knowledge about the network structure, SBMs can overcome the resolution limit and are less sensitive to the size of the network. Additionally, SBMs can handle networks with overlapping or hierarchical structures, making them a valuable and more generally applicable method for community detection [25].

Notably, statistical physics has emerged as a significant contributor to the subfield of modularity and community detection in network analysis. Offering an array of statistical methodologies originally proposed for analyzing complex systems. Among these contributions, *Tiago P. Peixoto* has played a pivotal role in advancing the realm of community detection within networks. His work combines methodologies from statistical physics and Bayesian inference, resulting in a profound impact on the field [26–35].

Nonetheless, in this work, we will use modularity in a broader sense, as a measure of how well a network can be divided into discrete and functionally distinct modules. As it was recently shown that modularity maximization and SBMs are equivalent under certain (but relatively limited) conditions [36, 37], thus this approach to modularity seems justified.

1.3 Research goals

The research goals of this thesis are twofold, first, we create an extensive but COVID-19-specific Telegram dataset, which includes channels, messages, users, and most importantly the interactions between them. This dataset and software created will be published and made available to the public to facilitate further research on Telegram and misinformation/disinformation during the COVID-19 pandemic. We enhance the typical approach of snowball crawling [19, 20], by adding a guidance mechanism, which allows us to target a specific topic and only collect data that is relevant to our research.

Further, we construct a network from this data and analyze its structure. We take a specific interest in the communities emerging from its structure organically, as they can be seen as communities of like-minded individuals and groups. We will use the framework of SBMs to infer these communities and perform model selection to determine the most likely generative model for the recorded data. This allows us to validate the existence of hierarchical structures within the network. We will then further use the most likely model to analyze the communities inferred within the network and their characteristics. This will allow us to draw conclusions about the meaning of these communities and their role within the network.

2 Results

As the goals for this thesis are twofold, so are the results. For one a software package was developed which allows us to extract data from Telegram and second we analyzed the resulting dataset. In the following we will give a brief overview of the key findings, please refer to the following sections for more details.

During the duration of this thesis, we developed and deployed a Telegram crawler software to collect, the largest currently available dataset on Telegram, encompassing over 128k distinct channels including all their messages and if available users. As further described in the methods Section 4.1 a guidance mechanism was proposed and implemented, which allows to systematically target specific topics in the crawling process. This allowed us to create a *COVID-19* targeted dataset, which is not only large but also mainly focused on a single topic.

We used this resulting dataset to find and analyze organically occurring modular structures within the network of Telegram channels. To this end, we used Bayesian model selection to evaluate a variety of different generative stochastic blockmodels on the network representation of the Telegram dataset. We found very strong evidence for the hierarchical and degree-corrected stochastic blockmodel mgb^{ch}.

We found the inferred modules of the hierarchical degree-corrected stochastic blockmodel mgb^{ch} to encode semantic meaning. The modules group languages together and are also able to distinguish between different topics. Notably, the model was able to partition the network into meaningful modules without any knowledge about the content of the channels.

Chapter contents

2.1 Collected dataset

The herein-presented Telegram dataset is the largest available as of today. As of writing this, we found a total of 433.12k channels of which we fully downloaded 128.15k. This includes 2.10B messages and 15.56M users of which 5.91M are members of a channel. Further, we recorded 1.82B text entities, 543.79M reactions to messages, and 3.69M polls with a total of 14.39M answers. The data was collected over the course of 19 weeks and totals 1.47 TB on disk. For a summary of all collected data see Table 2.1 and the corresponding structure in Fig. 4.3.

Metric	Count	Size
Messages	2.10B	1.22 TB
Entities	1.82B	202.00 GB
Reactions	543.79M	41.82 GB
Channels	433.12k	126.53 MB
Polls	3.69M	718.75 MB
Poll answers	14.39M	1.14 GB
Users	15.56M	1.02 GB
Channel Members	5.91M	406.73 MB
Total		1.47 TB

Table 2.1: **Summary of collected Telegram data.** The number of users, channels, messages, polls, reactions, and entities and the approximate size of the corresponding table in the database including indexing.

Most of the data collected are messages. These come in a variety of types, e.g. text, images, web pages, videos, stickers, documents, voice recordings, polls, and games. Because of storage, bandwidth, and potential copyright issues, we decided not to record any binary data, further, we do not record any interactive message types as these need server-side validation. Therefore we disregard videos, images, audio, and games. Nonetheless, we collect all message metadata, this includes potential text captions and timestamps. See Table 2.2 for a complete list of message types and their respective frequency. Further, we record entities within messages, these are mainly links to other channels, web pages, and mentions of users. These could be extracted post hoc but as Telegram automatically parses these we decided to record them directly (see Table S1.1).

The dataset was collected using snowball crawling with a self-created guidance mechanism and a heavily biased seed list (see Section 4.1). Thus, the collected data might not be representative of the full Telegram ecosystem. Nonetheless, the priority of our guidance mechanism decreased by 99% since the beginning of the collection, indicating that the crawler changed from a guided to a random walk (see Supplementary Fig. S1.6). However, we expect the content to be highly biased towards the keywords used in the guidance mechanism, i.e. the dataset should be highly targeted. This was the intention from the beginning to ensure a focused investigation into misinformation in the specific theme of the *COVID-19* pandemic. This same process allows other researchers to create their own targeted dataset, allowing further research on Telegram.

Type	Count	Fraction (%)
PHOTO	675,803,742	32.12710
TEXT	598,493,991	28.45187
WEB_PAGE	492,988,094	23.43621
VIDEO	211,153,063	10.03803
DOCUMENT	44,894,001	2.13422
STICKER	40,465,639	1.92370
AUDIO	14,557,606	0.69206
ANIMATION	13,809,547	0.65649
VOICE	5,730,157	0.27241
POLL	4,625,447	0.21989
VIDEO_NOTE	814,939	0.03874
DICE	167,547	0.00797
LOCATION	12,986	0.00062
CONTACT	9,616	0.00046
VENUE	4,242	0.00020
GAME	949	0.00005
Total	2,103,531,566	100.00000

Table 2.2: **The number of messages per type** and their respective fraction to the total number of messages.

The keyword list used for the guidance mechanism was translated into 44 different languages. Nonetheless, we find the majority of messages collected are in Russian, English, and German (see Supplementary Fig. S1.5). The same holds true for the majority languages of the channels i.e. the language with the most messages per channel. If one accounts for the number of active speakers of each language we find a slightly more diverse distribution (see Fig. 2.1). The diversity in language usage is skewed towards low values, indicating that the majority of channels are not multilingual but rather use a single language (see Supplementary Fig. S1.4). The language distribution and the normalized language distribution of the channels indicate that even though the keywords used in the guidance mechanism were included in 44 different languages, the collection preferentially targeted specific languages. Possible explanations are, that Telegram is just more used in these languages, the keywords are more frequently used in these languages or there might be overlap between similar keywords in different languages.

Initially, we found no correlation between the number of daily posted messages and the progress of the COVID-19 pandemic. We find no significant correlation between the OxCGRT [40] stringency index (a measure of the strictness of Nonpharmaceutical Interventions (NPIs)) nor the new cases (daily reported cases). Nonetheless, we find that the number of channels created increased during the first European COVID-19 vaccination campaign [41] and the initial phase of the Russian invasion of Ukraine [42] (see Fig. 2.2 and Supplementary Fig. S1.3). Further, the number of messages posted slightly decreases on weekends and during the night (see Supplementary Fig. S1.2). Indicating that the dataset mainly contains channels from european time zones.

Additionally to a messages' metadata and text content the dataset also includes reactions to messages and polls. These are distinct features of our dataset and are to our knowledge not

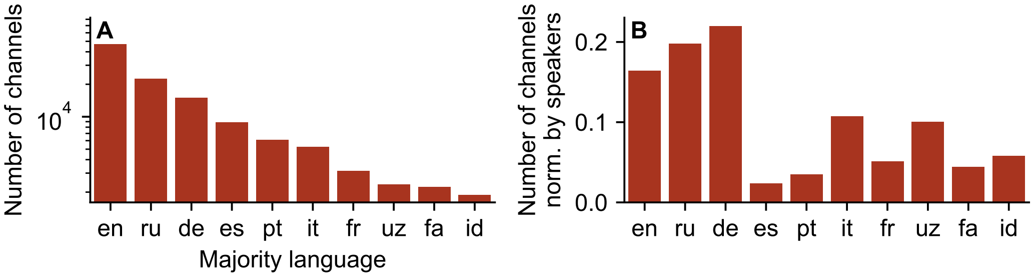


Figure 2.1: **Top 10 languages used in channels**. The majority of channels are in Russian, English and German, which is not surprising as these are the most commonly spoken languages (**A**). If one accounts for the number of active speakers of each language we find a slightly more diverse distribution (**B**). We identify a language as majority language if the number of messages per channel in that language is greater than the number of messages in all other languages. We only included messages where the language was detected with a confidence of at least 90%. Detection was done with FastText [38]. The number of active speakers per language is taken from Ethnologue [39].

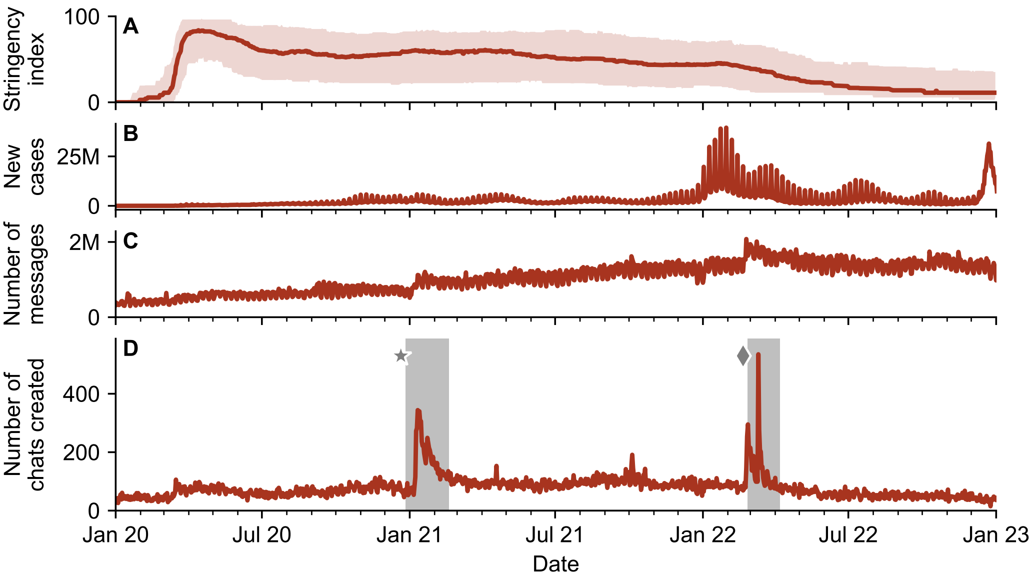


Figure 2.2: **Recorded messages and the relation to the *COVID-19* pandemic**. We find no significant correlation between the number of daily messages (**C**) and the number of new cases (**A**) nor the stringency index (**B**). Nonetheless we find that the number of channels created (**D**) increased during the first European COVID-19 vaccination campaign [41] (star) and the initial phase of the Russian invasion of Ukraine [42] (diamond). The stringency index is a measure of the strictness of NPIs and is estimated by the *Oxford COVID-19 Government Response Tracker* [40]. The new cases are aggregated by *Our World in Data* [43]. For the correlation analysis see Supplementary Fig. S1.3.

included in other available datasets. Reactions are a Telegram feature introduced at the start of 2022 and allow users to react to messages with emojis [44]. Reactions are used frequently as we found 9.75% of all messages posted to contain at least one reaction this includes messages posted before the introduction of the feature as users might react to these messages afterward. After the 1st January 2022, we found 45.6% of all messages to contain at least one reaction and 17.4% to contain at least ten reactions. We found a slight increase in the diversity of reaction usage during 2022 which might indicate that users are getting more familiar with the feature. The most common reaction is the thumbs up (39.4%) followed by the heart (11.8%) and the grinning face emoji (10.8%) (see Fig. 2.3). The feature was officially introduced on the 1st of January 2022 but we found an explosive growth in usage at the end of February 2022 this coincides with the initial phase of the Russian invasion of Ukraine [42]. Alternatively, this could also be explained by a delayed roll-out of the feature on different platforms.

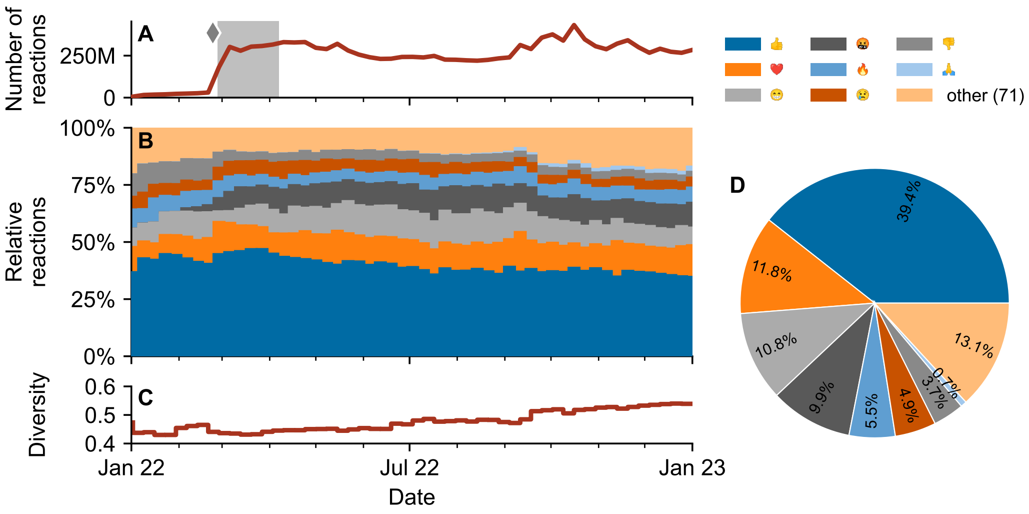


Figure 2.3: **Number of different reactions to messages**. The number of weekly reactions massively increased at the end of February 2022. This coincides with the start of the Russian invasion of Ukraine [42] (diamond). Afterwards the number of reactions stayed mostly constant (**A**). The relative reactions per week of 2022 stay mostly constant (**B**), but there is a small increase in diversity of reactions usage during 2022 (**C**). The most common reaction is the thumbs up followed by the heart and the grinning emoji (**D**), we observed a generally lower usage of ”negative” emojis such as the puking face, angry face, or shit emoji.

Telegram allows users to create polls with up to 10 options, depending on the poll settings one or multiple options can be chosen by a single user. The dataset includes 3.7M polls with a total of 14.4M distinct options. The distribution of votes per poll shows no apparent conformity to a visible distribution (see Supplementary Fig. S1.1). Out of all polls, we could collect the number of voters for 33.0% of polls, as the visibility of the number of votes can be disabled. Further out of all polls 5.6% allow multiple choice answers.

We also record the users that are part of a channel. Even though the visibility of users can be disabled by a channel administrator, this is mostly not the case for channels with the purpose of discussion. We encountered 17.6k channels where the public user list was enabled, which is 13.49% of all downloaded channels. The number of users per channel has a median of 13 (mean of 404) and it is correlated with the number of messages per channel in logarithmic space with a Pearson correlation of 0.33 (95% CI: 0.28 - 0.37)(see Fig. 2.4). This correlation is not surprising as it suggests that channels with more users tend to have more messages, which aligns with the expectation that discussions are typically more active in larger groups. The opposite case could also be possible, since a lot of channels are one-way communication channels, e.g. news channels, or ticker channels.

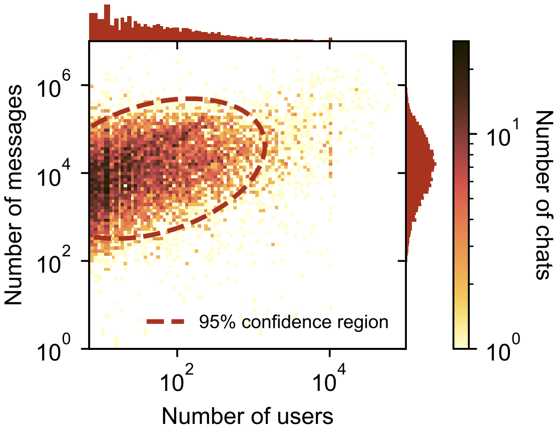


Figure 2.4: **Number of users and messages per channel**. The number of users per channel is correlated with the number of messages per channel with a pearson correlation of 0.33 (95% CI: 0.28 - 0.37). The number of users per channel is truncated at 6 users for improved visibility. The heatmap shows the number of channels with a specific number of users and messages. Dashed line indicate the 95% confidence region of a truncated multivariate normal distribution fitted to the data.

The dataset is collected by using the forwarded messages and their metadata, out of all messages 17% are forwards. We can leverage these forwards and create a network representation of the dataset which allows for further analysis of the underlying network structure of the Telegram ecosystem (see Section 4.1.5 for further details). The number of ingoing and outgoing forwards per channel i.e. the in and out-degree of the resulting network does not strictly[1] follow a power-law degree distribution (see Fig. 2.5). This might be an unexpected result for a social networks [45] but on the other hand power-law degree distributions and therefore scale free networks are generally considered rare [46–48].

Because of organizational overhead and the sheer size of the dataset, it is not yet publicly available. Additionally, it is not clear to which extent the data can be published without violating the privacy of users and still conforming to European data protection laws. However, we plan to publish as much of the dataset as possible in the near future. Nonetheless, the network representation of this dataset without including the content of messages is available.

[1] I was told by a colleague and neuroscientist, that the out-degree is power-law like with an exponential cutoff.

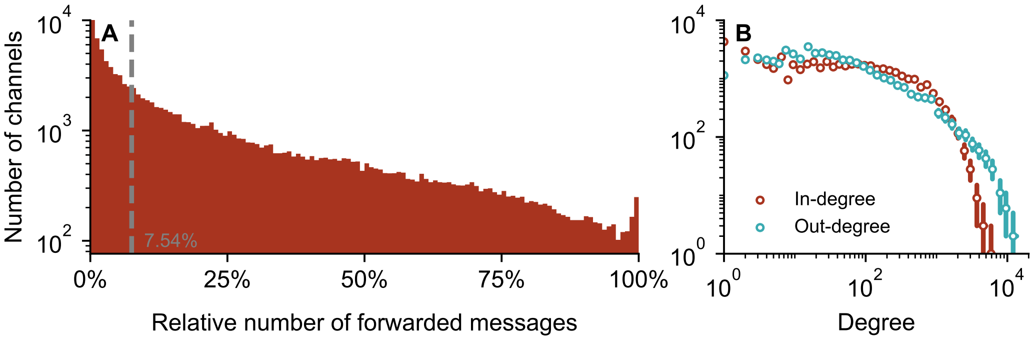


Figure 2.5: **Summary of forwarded messages and the resulting network structure**. The distribution of the number of forwards relative to the total number of messages in the chat (**A**) shows the majority of messages in a given channel are not forwards (median 7.54%). The in- and out-degree distributions of the resulting network do not follow a power-law degree distribution (**B**). Whiskers indicate the 95% Confidence interval (CI) obtained by bootstrapping.

For instance, this can be used to reproduce most of the results of the following section. The network data is published on *GRO.data*[49].

2.2 Model selection

We compared 8 different Stochastic blockmodel (SBM) variants where we varied the inclusion of different model features. The features we included are:

- **Directed edges:** - As edges in the graph correspond to messages being forwarded from one channel to another. This is arguably a directed process.
- **Hierarchical structure:** - Hierarchical structure is included as we expect the Telegram network to be organized in a hierarchical fashion (communities of communities).
- **Degree correction:** - Degree correction is included as the standard SBM assumes high homogeneity in the degree distribution of the nodes, which is often not a reasonable assumption for real-world graphs.

For a full description of all model variants please refer to Section 4.5. We compare these variants using Bayesian model selection as described in Section 4.3.

All proposed models can be defined on a multigraph, i.e. a graph with multiple edges between the same pair of nodes, and on a simple graph, i.e. a graph with at most one edge between the same pair of nodes. We opted to only compare multigraph models as it is common for Telegram channels to forward messages to the same other channel multiple times [2].

[2] Further, multigraph and simple graph models are difficult to compare as the observed graph data is different.

The most likely model out of the tested models is the hierarchical model with degree correction mgb^{ch}. It has a Minimum description length (MDL) of 34 509 kbit (95%CI: 34 502 kbit to 34 518 kbit). Compared to all other models, it is at least $K = \ln(897997) \approx 9 \cdot 10^{389994}$ more likely. This result is significant as normally a factor of more than 20 is considered decisive evidence that the data is more strongly supported by one model over the other [50, 51]. For an overview of the model comparison i.e. the posterior odds ratios between the models see Fig. 2.6 and Table 2.3.

Abbreviation	$ln(K)$	95% CI		$\approx K$	features
mgb^{ch}	0	-13031	13045	1E+0	hierarchical, deg. corrected
mgb^{h}	897997	790727	1019571	9E+3509	hierarchical
mgb^{dch}	1650389	1637745	1662641	6E+6450	hierarchical, deg. corrected, directed
mgb^{c}	1965230	1956391	1972519	2E+7681	deg. corrected
mgb^{dh}	3359806	3274056	3516317	2E+13132	hierarchical, directed
mgb^{dc}	3551768	3542819	3559135	4E+13882	deg. corrected, directed
mgb	3727290	3718436	3734538	5E+14568	
mgb^{d}	7029211	7020369	7036482	5E+27474	directed

Table 2.3: **Model comparison of all models compared to the best model (mgb^{ch}) as baseline.** The log posterior odds ratio $\ln(K)$ is show and an approximation for the Bayes factor $K \approx e^{ln(K)}$ is given. Further the 95% CI for the log posterior odds ratio is given and the features of the model are listed.

In earlier runs using a smaller subset of the data we also included a SBM variant which allowed for overlapping communities mgb^{o}. However, this model did not converge with the full dataset as the overlap models are more complex than other models. Nonetheless, in earlier runs we found it to be less likely than most other models. Therefore it is not included in the final model comparison results.

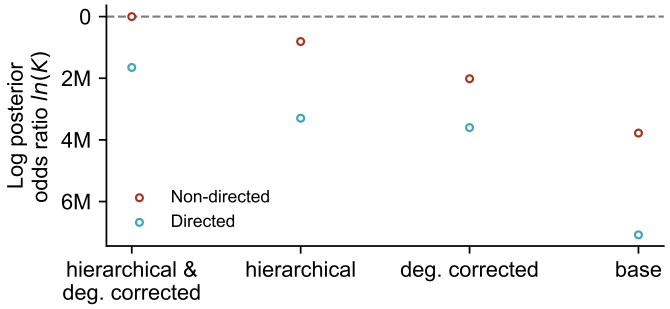


Figure 2.6: **Model comparison of all models compared to the best model (mgb^{ch}) as baseline.** We compare the all models to the most likely one via their log posterior odds ratio $\ln(K)$. We choose to keep the value logarithmic and as the differences are large. On average the non-directed models variants perform better. Red denotes the non-directed and blue the directed model variants. Whiskers denote the 95% CI of the log posterior odds ratio.

We find the directed and non-directed variants show consistent results between different model variants. For instance, the most likely directed model is also the hierarchical and degree corrected variant (mgb^{dch}). The same internal ranking for the non-directed and directed model variants is observed. Unexpectedly the non directed models perform better on average (see Fig. 2.6). This is surprising as the edges in the graph correspond to messages being forwarded from one channel to another. This is arguably a directed process and thus we would expect the directed models to perform better. However, the ability of users to join the original channel through forwarded messages introduces a bidirectional aspect to the communication flow. This bidirectional aspect challenges the assumption that the communication process is strictly directed and thus might be a reason why the non-directed model variants perform better on average.

As the log posterior odds ratios between the different model variants are very decisive we choose to only further analyze the best model for our collected data i.e. the hierarchical and degree corrected model (mgb^{ch}).

2.3 Structural analysis

The hierarchical and degree corrected model mgb^{ch} explains the collected data best. During sampling of the equilibrated model, we found 4875 modules (95%CI: 4874 modules to 4876 modules) as the most likely number of modules. As it is a hierarchical model, modules might include other modules and thus form a hierarchy. We found that ten hierarchies i.e. layers describe the data best. After the tenth layer, all channels are assigned to the same singular module. The number of modules is mostly stable within each layer, as the number does not change significantly during sampling (see Fig. 2.7). This does not necessarily indicate that channels are assigned to the same module in each sample. For instance, two channels might swap their module assignment during sampling.

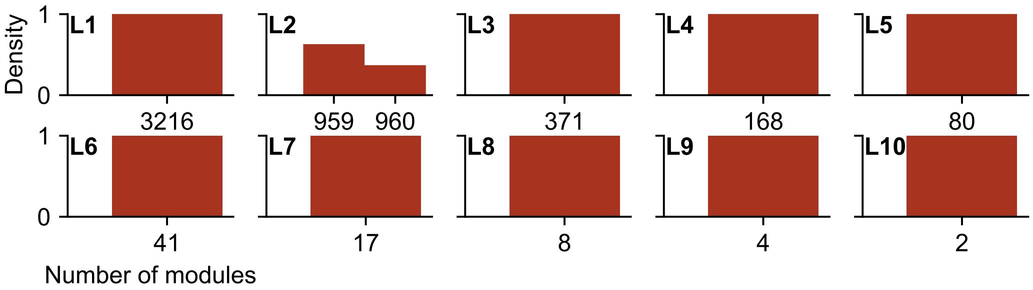


Figure 2.7: **Marginal posterior distribution of the number of modules per layer.** The number of modules is mostly stable across all layers (**L1** - **L10**) during sampling. Only the second layer shows a ambiguous number of modules where the model can't decide between 959 and 960 modules.

Out of all 128k channels we found 762 (0.6%) channels with ambiguous module assignments. The probability for this node to belong to more than one module is higher than 20%. The model is not able to assign these nodes to a single module with confidence. The reasons for this can be manifold. For instance, the node might statistically belong to multiple modules or the model might not be able to distinguish between the modules. This might be due to the fact that the model is not able to capture the underlying structure of the data or that the data is inherently ambiguous. Only a small percentage of channels are assigned to multiple modules, thus the model confidently assigns most nodes to a single module on the first layer.

To visualize the channels we use their module assignment in each level as a prior for a Scalable Force-Directed Placement (sfdp) layout [52]. This allows the computation of a 2D embedding for each node in the graph based on the inferred communities. Compared to a simple spring layout, the sfdp layout, especially using the inferred modules as a prior, allows for better separation of the different modules. The resulting layout is shown in Fig. 2.9. Here, we show the channel module assignment of each layer in a separate panel. For following graphics using this layout we opted to not show any edges (forwarded messages) as they mostly overlap and reduce the readability of the figure.

Even though the sfdp layout gives an overview of the inferred modules, without any coloring it is not clear how the modules relate to each other. An alternative to the sfdp layout is a radial layout [53]. Here the channels are arranged in a circle with lines or bundles of lines as connections between the channels. Layers are circles with decreasing radius here the lines are bundled together if channels are assigned to the same module. This allows us to highlight the relationship between modules or channels across different layers. For an example of this layout with coloring corresponding to the module assignment see Fig. 2.8. This comes with the trade-off of a reduced readability of the modules itself but one can see how modules are grouped across layers.

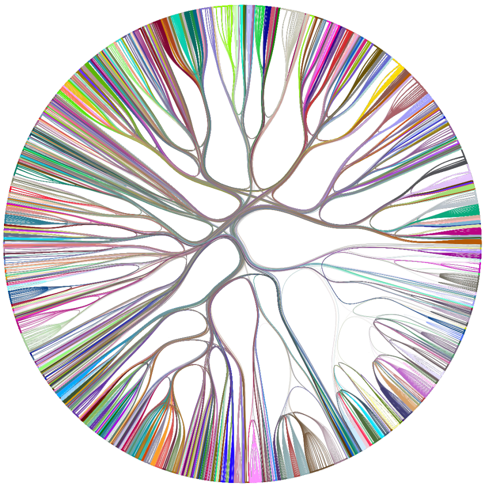


Figure 2.8: **Example of a radial layout of the inferred model.** The modules are colored according to the module assignment on the second layer. For a comparison using the sfdp layout see Fig. 2.9 (**L2**).

Of course, it is not clear if the inferred modules are meaningful. For instance, the model might not be able to capture the underlying structure of the data or the data might be inherently ambiguous. Basically, we could be just fitting noise. In the following we investigate the modules in more detail to answer the question of whether the inferred modules are meaningful or not. We do this by labeling the modules and investigating said labels.

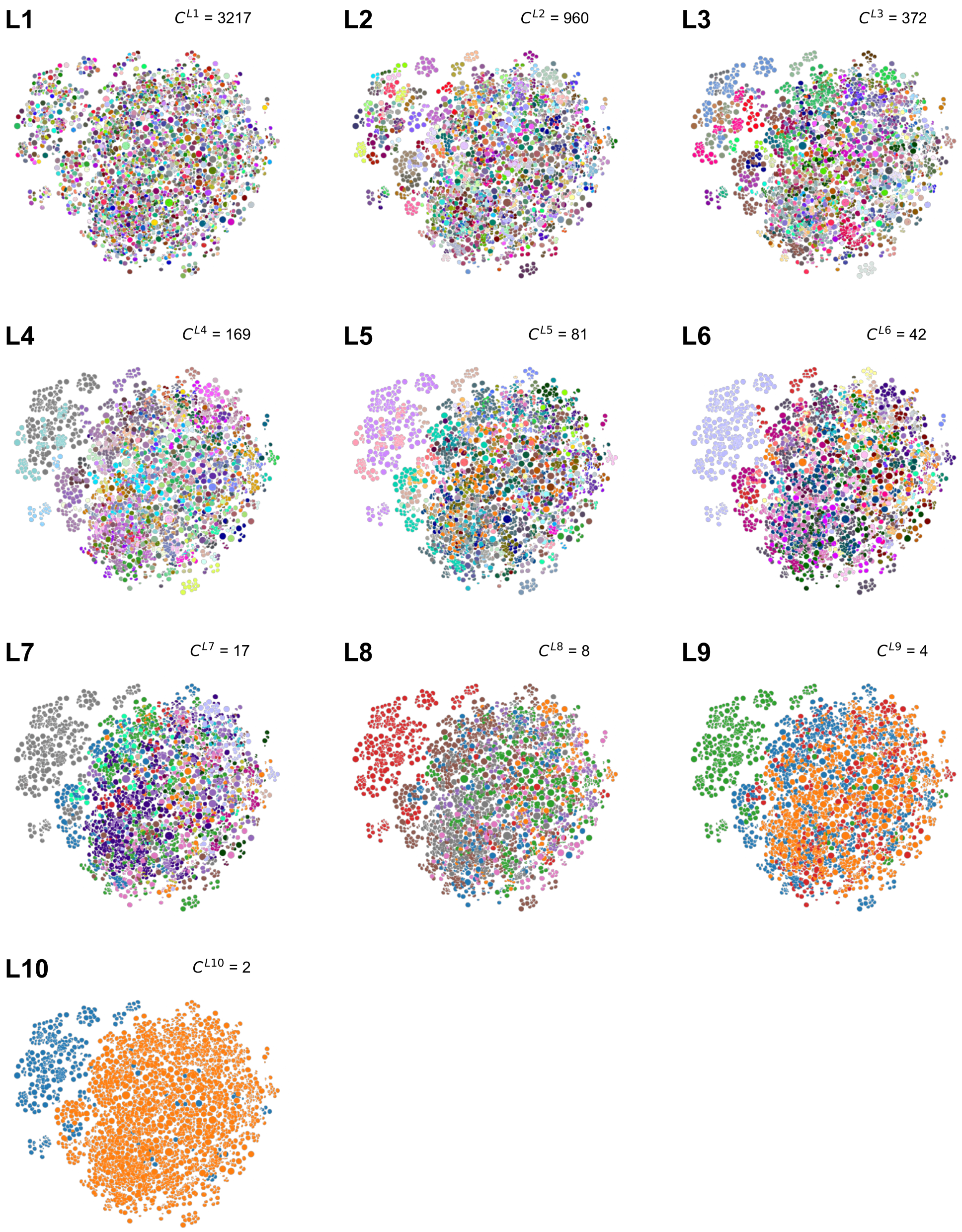


Figure 2.9: **Overview of the inferred modules in each of the nine layers.** Each point represents a channel and is colored according to its module assignment on the corresponding layer (**L1-L10**). Colors are reused across layers and do not indicate a cross-layer relationship. For the first three layers, colors might also be reused for module assignment, as picking more than 200 perceptually distinct colors for quantitative data is difficult if not impossible [54]. Colors might be reused for the different modules in these lower layers. The number of modules shown in each layer C^{L*} is indicated in the top right corner of each panel.

2.4 Meaningfulness of the inferred modules

The partitioning of the channels by the model is done without any prior knowledge of the content, purely based on the structure of the graph of channels and their forwarded messages. It is therefore not clear if the inferred module assignments encode a humanly interpretable and meaningful properties of the channels.

To enhance the interpretability of the partitioned modules, we explore various labeling strategies to discern patterns and relationships within the inferred modules. These strategies aim to unveil underlying patterns and relationships within the inferred modules. Labeling modules can be done in a variety of ways e.g. by the topic of the channel, by the language used in a channel, or by the number of messages sent in the channel. The absence of a singular correct method makes the labeling process inherently subjective. Because of simplicity, we opted for the following approaches to identify labels for the inferred modules and investigate their meaning:

- **Language:** We used the language of the channel as a label. We used the *fasttext-langdetect* package [38] to detect the language of all messages. For each channel, we then assigned the language with the highest probability over all messages in the channel.
- **Topic:** We used the most frequently used words in each channel and used the models hierarchical structure to extract topics on different layers. This is very similar to tf-idf approaches [55]. This method was chosen as it is straightforward to implement and gives a good overview of the content of each channel and module.
- **COVID-19 relevance:** We investigate the priority values of each channel as computed earlier for the guidance mechanism in the data acquisition process. This gives us a value for the COVID-19 relatedness of each module.

We observed that all the examined labels are organized into distinct modules. Labels associated with language and COVID-19 exhibit a clear separation between modules, while the topic labels are more difficult to interpret as they are not as clearly separated as the other labels and interpretation is more subjective. Nevertheless, we generally found strong evidence that the inferred modules are meaningful and that the model can capture the underlying structure of the data without any prior knowledge of the content.

2.4.1 Language

People tend to communicate more with people that speak the same language. Forwarding a message into a channel with a different majority language is less likely as the recipients might not understand the content of the message. This should lead to more edges intra-language than inter-language. Overall, it is thus expected that different languages are grouped into different inferred modules.

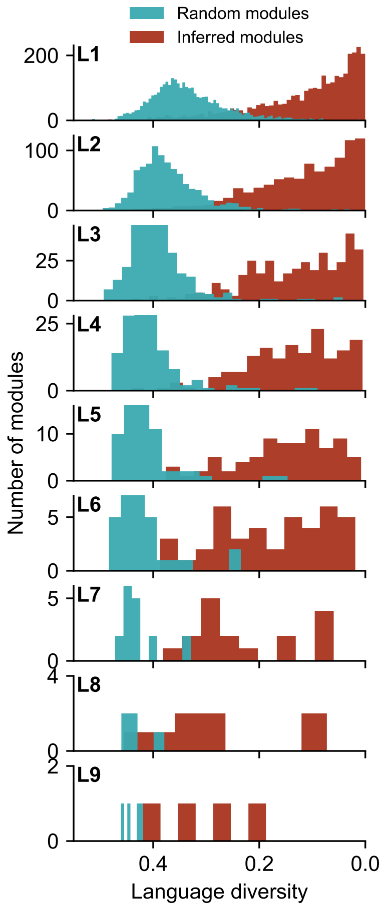


Figure 2.10: **Language diversity in the inferred modules per layer.** The diversity in the language usage per inferred module (red) is lower than what would be expected for a random assignment of channels to modules (blue).

Out of all the used languages within a module, one or a small number of languages is favoured, which is shown by the general low diversity in language usage per module (see Fig. 2.10). The diversity in language usage within a module is consistently lower than what would be expected under a random assignment of channels to modules. This pattern holds across all layers, although the difference is less pronounced in upper layers where the number of modules is smaller than the total number of identified languages.

The upper layer modules seem to consolidate similar languages, with less frequently used languages being assimilated into the higher layer (elevated) modules. Notably, the merging process in the lower layers appears to prioritize languages that share similarities or are geographically proximate, e.g. this is observed for Ukrainian and Russian or Portuguese and Spanish (see Fig. 2.11).

Without any prior knowledge of the content and languages used in the channels, just by using the structure of the graph of channels and their forwarded messages, the model separates different languages into different modules and groups the same or similar languages into the same module. This indicates that the model can capture the language structure of the data.

The computed language diversity is limited by the language detection algorithm. The used language detection algorithm/-model *fasttext-langdetect* [38] has an unweighted accuracy of 76.8% on the *WiLi* [56] dataset and (only) includes 139 languages. While this algorithm performs reasonably well, it is important to acknowledge that the accuracy rate introduces a level of uncertainty in the language assignments. Misclassifications occur and consequently, the observed language patterns and the inferred modules should be interpreted with a degree of caution. Future iterations of this analysis could benefit from the incorporation of more sophisticated language detection models or techniques to improve accuracy and mitigate potential misclassifications.

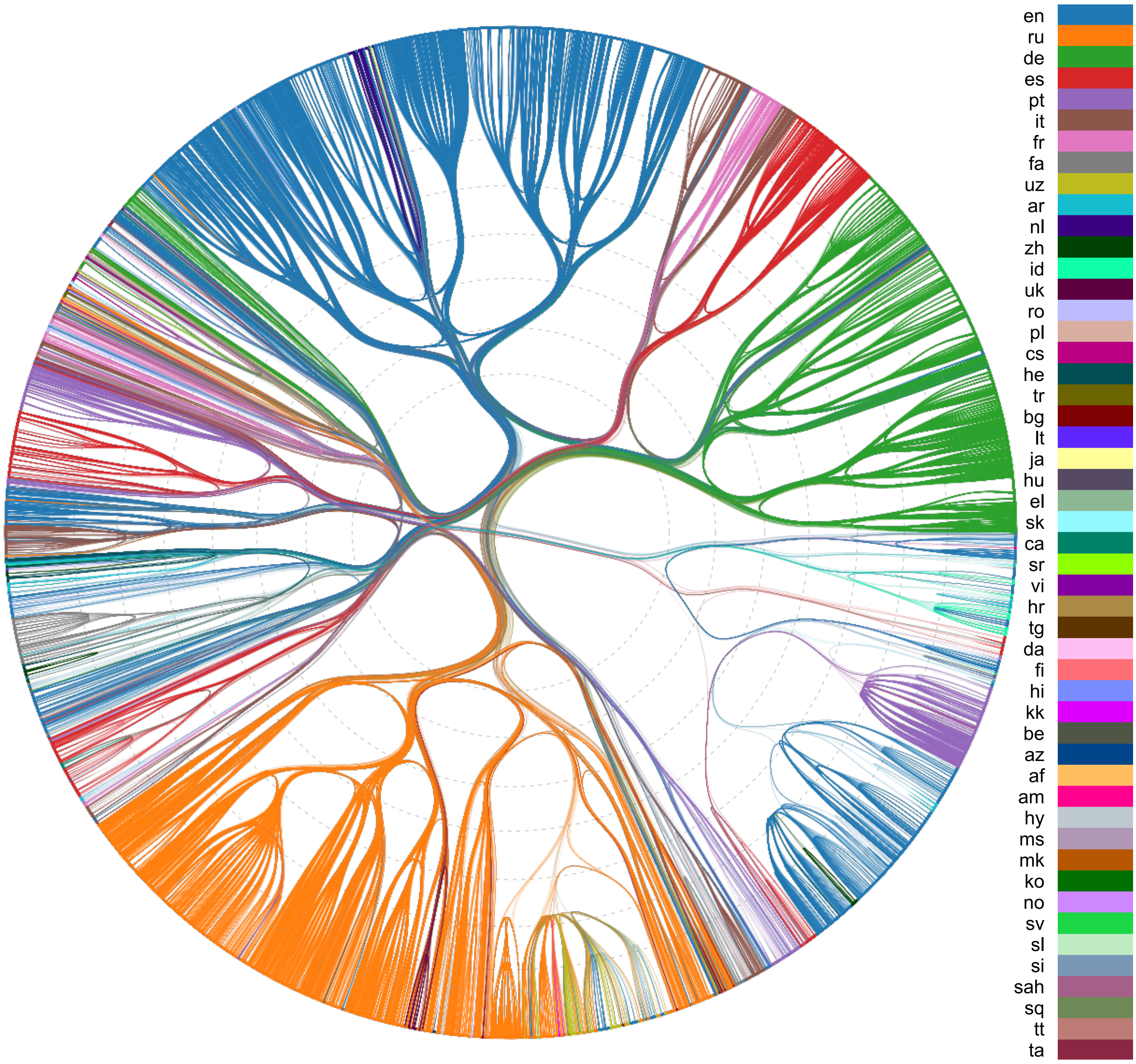


Figure 2.11: **Majority languages of the modules across hierarchies.** Each channel is represented by a dot on the outer ring. The channel is colored according to the majority language of the first layer module it is assigned to. Lines show 75k subsamples of the edges in the hierarchical model. The inner rings, where lines are bundled shows the different layers of the hierarchical model. The most used languages stay separated across most layers (see e.g. orange, green, blue). Smaller languages get assimilated into larger ones, for instance, the Ukrainian language (red-purple) gets merged into the Russian language (orange) in layers 2-4 (see at the bottom).

2.4.2 COVID-19 relatedness

Similar to the used languages we expect the model to separate COVID-19-related channels from non-COVID-19-related channels. This is due to the assumption that COVID-19-related channels are more likely to interact with each other and these channels share a similar audience. For instance, messages from a channel dedicated to COVID-19 news are more likely to be forwarded to channels discussing vaccines, treatments, or prevention strategies. This pattern of interconnectivity should lead to the model effectively distinguishing between different topics and separating them into different modules.

During the crawling process of the dataset, we already computed a COVID-19 relatedness score for each channel based on the number of COVID-19-related words used in the channel that connect to this channel via a forwarded message (see Section 4.1). We use the same metric but on the channels content itself to compute a COVID-19 relatedness score for each channel. We expect that this score is a good initial indicator for the COVID-19 relatedness of a channel and allows us to label modules accordingly.

We found that the model is on average able to group channels into modules based on their COVID-19 relatedness. The median COVID-19 relatedness per module shows a larger variance than what would be expected under a random assignment of channels to modules. This is most predominant in the lower layers up to the sixth layer (see Fig. 2.12). In the upper layers, this difference is less pronounced which indicates, that the upper layers might not prioritize the segregation of COVID-19-related information to the same extent as the lower layers (see also Supplementary Fig. S2.12). The nuanced layer-specific behavior further suggests that COVID-19 relatedness is structurally less important to the network of channels as compared to the language of the channel which showed a more consistent pattern across all layers.

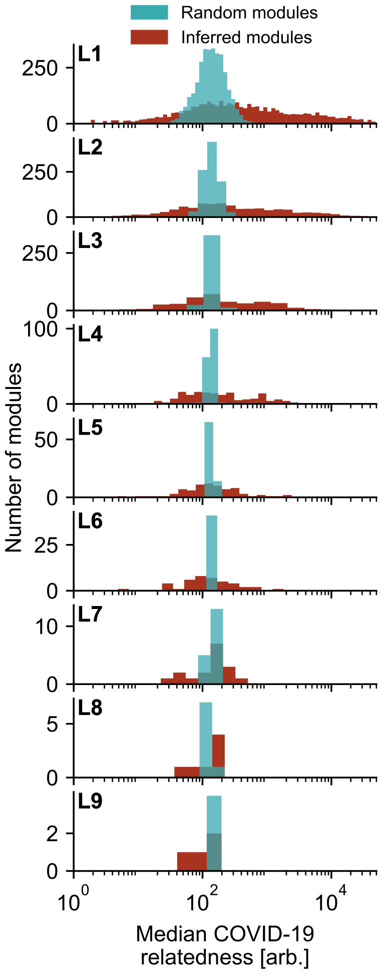


Figure 2.12: **Median COVID-19 relatedness in the inferred module per layer.** The variance of the median COVID-19 relatedness per layer is higher (red) than what would be expected under a random assignment of channels to modules (blue). This especially holds for the lower layers **L1-L6**.

As the relatedness is a continuous value, we do not expect a clear classification of COVID-19-related and non-COVID-19-related channels. Instead, we expect a variety of modules with different COVID-19-relatedness values. Indeed, the inferred modules seem to group channels with similar COVID-19 relatedness into the same module (see Fig. 2.13). While some modules exhibit a strong decoding of COVID-19 related channels, others group modules without COVID-19 related channels. This suggests a potential suppression or exclusion of COVID-19-related information in specific channels.

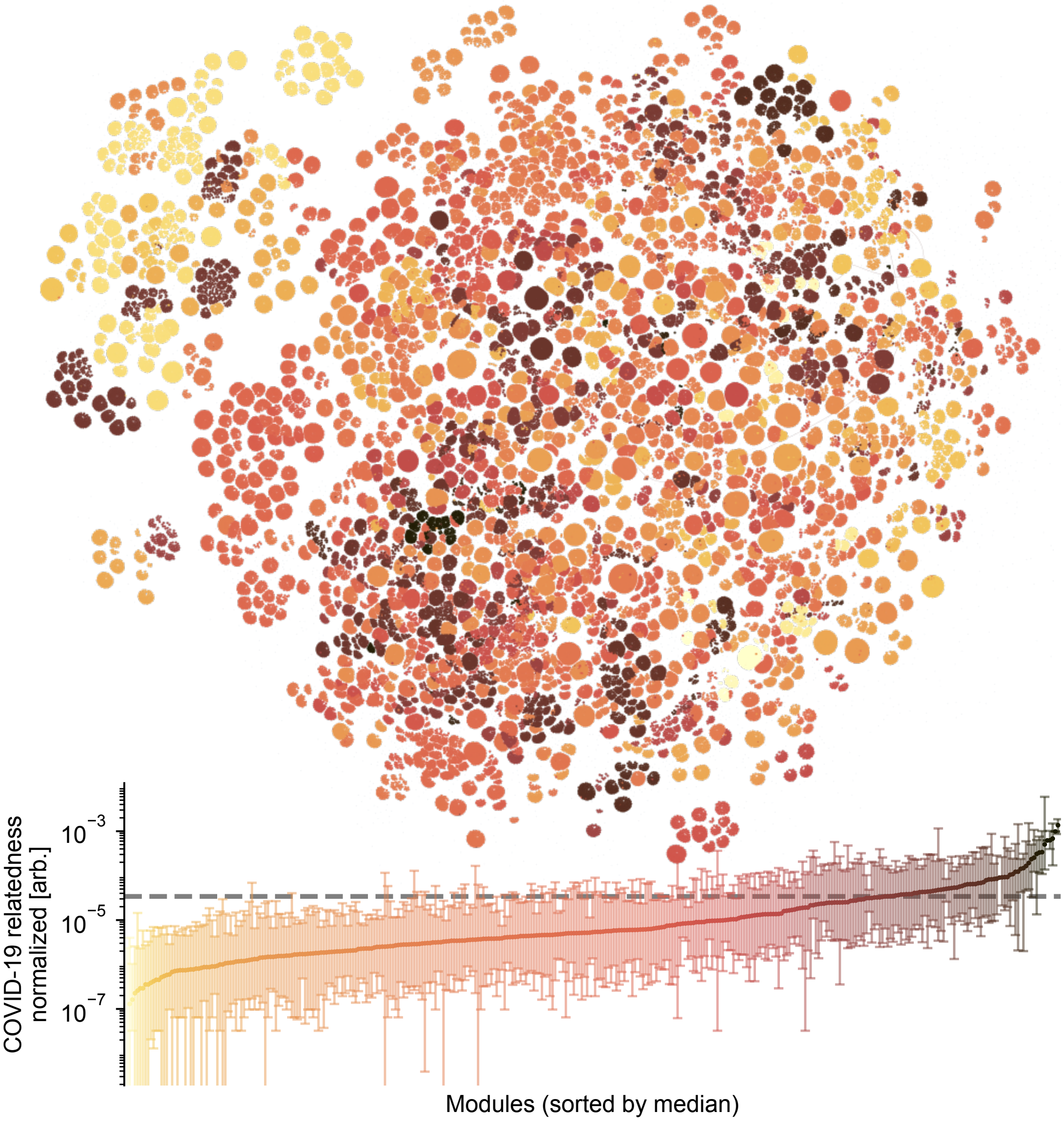


Figure 2.13: **COVID-19-relatedness of the inferred modules on the third layer.** The COVID-19-relatedness of the channels is computed as the median guidance priority of all channels in the module. A small number of modules hold the most (black) and least COVID-19-related channels (yellow). Compared to the baseline, assuming a random assignment of channels to modules (gray), the model is able to group channels with similar COVID-19 relatedness into the same modules. Here represented is the third layer where this separation is most predominant. Whiskers indicate the 68% CI of the median.

2.4.3 Topic labels

To gain insights into the topics discussed within a channel and, subsequently, within the inferred modules, we computed module-specific word frequency distributions. These distributions serve as indicators of the topics within a module and allow further validation of the assumption of topic separation by the model.

We incorporated language information from the earlier analysis and removed stopwords accordingly to improve the accuracy of the topic extraction. This step enhances the topic representation by eliminating common words such as “the”, “a”, and “and” that do not contribute significantly to the topic.

We collected channels in a variety of sizes, some with a few hundred messages and others with several million messages. These channels might be grouped together into the same module by the model. To keep the word frequency distribution comparable between channels of different sizes, we weight the word frequency by the number of messages in each channel. This is very similar to established tf-idf approaches [55] where the goal is to compare the importance of a word in a document to a corpus of documents.

Since the model includes hierarchical partitions, we extended this weighting strategy to the word frequency distribution of modules within a module. E.g. the word frequency distribution of a module in layer 5 is the weighted average of the word frequency distribution of the submodules in layer 4. This hierarchical weighting mechanism ensures that the word frequency distribution remains representative and comparable across modules in different layers (see Fig. 2.14).

Within the scope of this thesis, it is not feasible to explore and present all computed topics and labels. Therefore, we have chosen to focus on the branch of modules primarily containing German channels (refer to Fig. 2.11, highlighted in green) for further exploration. The topics identified within this German subset reflect a diverse range of discussions, including geopolitical events, and public health concerns. Some topics are more clearly defined, such as the discussion of the COVID-19 pandemic or the Ukraine-Russia war, while others are more ambiguous(see Fig. S2.11). The identified topics are distinct from each other, thus indicating once more that the model is indeed able to separate different topics into different modules.

It is essential to acknowledge certain limitations in our approach. Firstly, the approach ignores word order during topic extraction. This limitation might affect the nuanced understanding of context and meaning, particularly in languages where word order plays a crucial role [57]. Additionally, while our analysis includes stopwords removal for precision, we did not employ lemmatization [58]. Lemmatization could enhance the identification of core word forms, potentially refining the accuracy of topic extraction and making the results more interpretable but it is nuanced and might not be applicable to all languages.

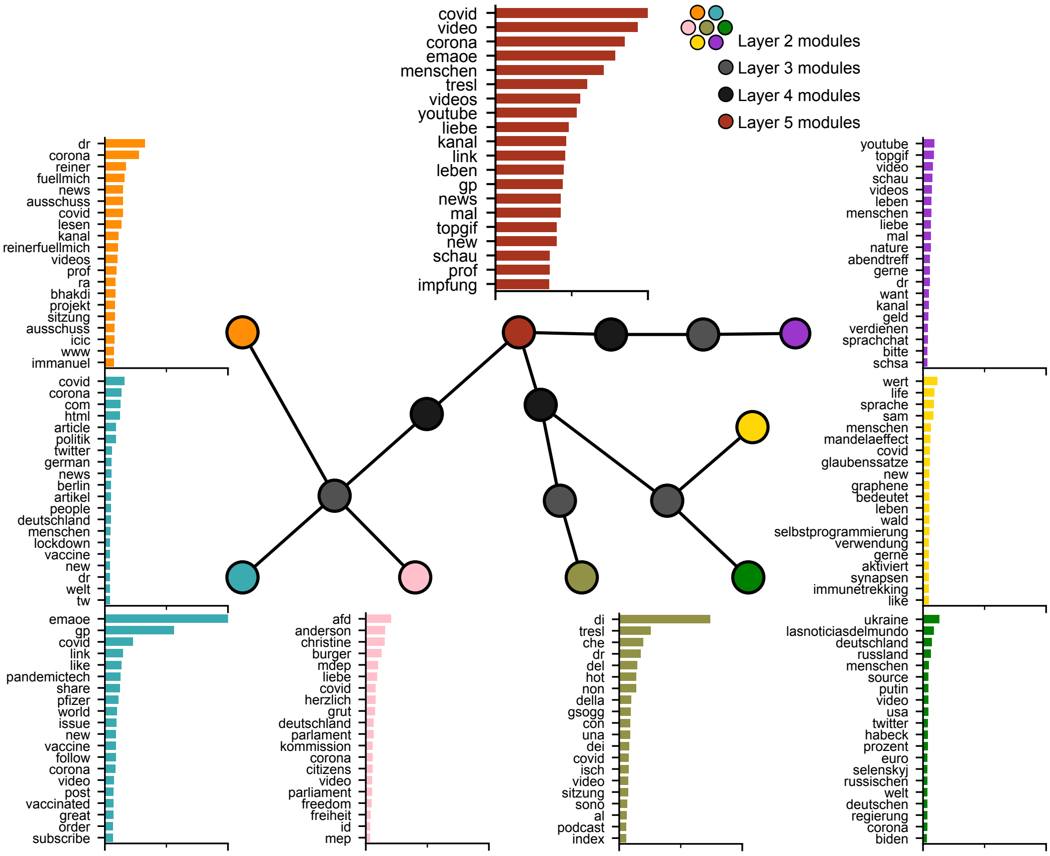


Figure 2.14: **Hierarchical topic extraction using inferred modules.** The hierarchical structure of the inferred modules allows the extraction of topics on the different layers. The topics on the first layer (smaller colored bar plots) get combined to form broader topics on the upper layers based on the module structure (colored to gray to red dots). An example is shown from a module in layer 5 (red) and its corresponding submodules (gray to colored). We picked this branch from the german speaking modules see Fig. S2.11 in green.

3 Conclusion

In conclusion, this thesis has conducted a comprehensive examination of Telegram data, revealing insights into its structural characteristics and setting the stage for future investigations. The presented dataset and analysis aim to be a valuable resource for researchers in diverse disciplines, fostering a deeper understanding of the dynamic landscape of online communication, misinformation, and social interactions. The Telegram dataset itself provides a wealth of information for exploring communication dynamics, user interactions, and network structures on social media platforms. As we wrap up this thesis, let us contemplate the findings, potential avenues for future research, and broader implications.

3.1 Summary

The collected dataset is both unique and extensive, providing a rich source of information for analysis. Its global scope enables the observation of prominent geopolitical events reflected in the communication patterns within the data. It's crucial to note that data collection is an ongoing process, ensuring the dataset remains dynamic and relevant for continuous examination.

In the structural analysis, the framework of Stochastic Block Models has proven highly effective for handling large-scale real-world data. Utilizing this framework, very strong evidence for a hierarchical structure within the Telegram network was found. The hierarchical organization allows for the separation and grouping of modules within different layers of the model, providing a nuanced understanding of the underlying structure. Importantly, the hierarchical representation not only reveals the macro-level organization of the network but also offers insights into the micro-level relationships among channels.

The inferred modules encode language patterns, prevalent topics, and COVID-19 relatedness. Along with the capability to further analyze these modules, the analysis serves as a valuable baseline for in-depth investigations. The framework of Stochastic Block Models not only shows the potential for detailed examinations of social networks' structural intricacies and dynamics but also opens avenues for more targeted studies, such as the exploration of evolving trends, identification of influential channels, and understanding the flow of information within specific topic clusters. In essence, the framework not only captures the static structural features but also provides a dynamic lens to delve into the evolving nature of the dataset, paving the way for a more profound and nuanced understanding of its complexity.

3.2 Outlook

While this thesis offers a comprehensive analysis of the Telegram data and its structural characteristics, numerous open questions and potential avenues for future research remain. The ongoing data collection presents opportunities for further exploration across various research domains. In the following we discuss some of the potential applications of the herein presented dataset and analysis.

One promising avenue for future research involves incorporating prior knowledge into the partitioning process. In this thesis, we have chosen an uninformative prior, assuming equal probabilities for all possible partitions. However, the integration of domain-specific information or external data sources as informative priors could significantly enhance the accuracy and the relevance of the structural analysis, which allows to tailor the partitioning process to the specific research question.

The disregarded models in the model comparison, while not included in the final analysis, may still prove useful depending on the research question at hand. While our study focused on selecting the most likely model for the Telegram network, the exclusion of certain models does not diminish their potential relevance in different contexts or for specific research objectives. Further the potential to create or refine more models that better suit the Telegram network remains.

The emergence of Large Language Models (LLMs) in recent years raises concerns about the presence of bots that can effectively mimic human behavior. A critical question it is possible to reliably identify these bots within our dataset. Investigating methods to discern between genuine human interactions and those orchestrated by LLMs could provide valuable insights into the evolving landscape of online communication and influence mechanisms.

While research has explored the connection between misinformation spread and the impact on the spread of COVID-19, there are ample opportunities to further validate and deepen this understanding using the herein presented dataset. Investigating the dynamics of misinformation within Telegram channels and its potential influence on public perceptions and responses to the pandemic could provide valuable insights into the role of online communication in the context of public health crises.

In the domain of social science, the identified modules may offer insights into the dynamics of online communities, social interactions, and the formation of digital subcultures. Understanding how individuals engage with one another, form connections, and contribute to the construction of online spaces is essential for comprehending the evolving landscape of contemporary social interactions.

Given the recent nature of the collected data, there is a unique opportunity to conduct analyses on contemporary geopolitical events. Specifically, the ongoing data collection allows for the exploration of how the Telegram platform is utilized to discuss and disseminate information related to current global events. For instance the currently ongoing Russian-Ukrain war. Analyzing how information related to the conflict is disseminated, received, and discussed within different channels could contribute to a nuanced understanding of online discourse during geopolitical events.

3.3 Ethical Considerations

In the exploration and analysis of large-scale datasets, particularly those involving user-generated content and interactions, ethical considerations and the protection of user privacy are of utmost importance. The Telegram dataset, being a substantial collection of communication data, raises several ethical concerns that need consideration prior to the dataset publication.

The dataset includes user information such as usernames, first and last names, which raises concerns about individual privacy. Even though the names can be chosen freely by the user, they may still contain sensitive information. For example, a user might opt to use their real name as their username for easier identification by friends and family. While the dataset may be pseudonymized, replacing problematic information with unique identifiers, the risk of re-identification remains, especially if combined with the other available information. As the data is still publicly available through the Telegram platform, it is possible to reconstruct the original dataset or parts of it even if the dataset is pseudonymized. For instance, utilizing message content, timestamps, and channel details could potentially lead to the identification of users, undermining the intended privacy protections.

While the dataset does not directly include health information, inferences about users' health statuses can be drawn from the content of their messages. For instance, a user mentioning that they have tested positive for COVID-19 allows for the inference of their health status. Handling and analyzing such information requires careful consideration to prevent potential stigmatization or misuse.

A comparable issue emerges in relation to potential criminal activities or unconstitutional behavior documented within the dataset. We have not actively searched for such content, but given the size of the dataset and the nature of Telegram as a platform, it is likely that some of the collected data contains information about criminal activities. This raises important ethical considerations for future research. When delving deeper into the message content, it becomes crucial to establish clear guidelines on how to address potential criminal activities or other illegal actions recorded. Questions arise regarding the ethical responsibility of researchers, including whether law enforcement agencies should be informed and whether such reporting falls within our responsibility. Defining an approach to handling such cases is essential for ensuring that future research is conducted in a responsible manner.

4 Methods

In the following we will formalize the problem of inferring modular structures in graphs and describe the collection process of the herein presented Telegram dataset. We introduce the Stochastic blockmodel (SBM) as a foundational framework for community detection. However, we recognizing its limitations and therefore also explore refined variants to better suit real-world complexities. Central to this approach is the usage of Markov chain Monte Carlo (MCMC) methods, enabling us to navigate intricate graph structures, even when analytical solutions are intractable. Further we explore model selection in this context using the Minimum description length (MDL) criterion.

Central to our analysis is also the creation of the used dataset itself. We define data collection process via snowball crawling and extend it by introducing a guidance mechanism, which allows to target specific topics in the crawling process. We will also discuss the preprocessing steps necessary to prepare the data for the analysis.

Our overarching objective is to unravel the modular structure embedded within the Telegram network, where nodes denote distinct channels and edges forwarded messages between these channels. Notably, this network may possess multiple edges between the same pair of nodes and the edges are directed. We will thus extend the standard SBM to account for these complexities. We will also discuss the implications of the multigraph structure and the directed edges on the inference process. Further we will define a hierarchical extension of the SBM to account for the nested structure of the Telegram network and an extension to account for degree heterogeneity.

Infoboxes

In the following you will find small infoboxes like this one, which will contain additional information about the topic at hand. These are not necessary to understand the main text and can be skipped on first reading. However, they might contain useful information for the interested reader. I opted for this approach instead of moving content into footnotes or the appendix, as I find them more readable and less distracting.

First we will start with the data collection process and the preprocessing steps necessary to prepare the data for the analysis, this is followed by a formalization of the problem of inferring modular structures in graphs.

♦

Chapter contents

4.1 Telegram data acquisition

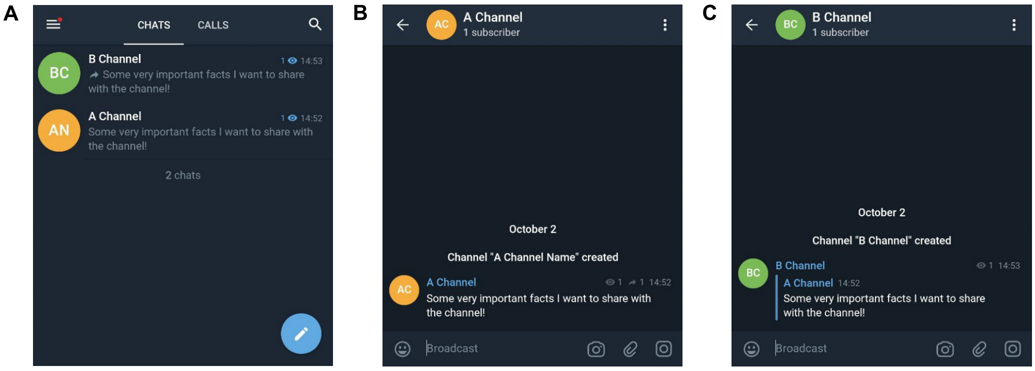
Telegram does not provide direct a way to download its data nor is there a program for researchers to access the data. Therefore we had to develop a custom solution to collect the data. We here describe the data acquisition process, the software used and the data itself.

Telegram allows developers to create their own clients, e.g. if you are creating an operation system for a smart fridge you are able to create a Telegram client for it. This is possible as all official Telegram apps are open source project[1]. We can use this to our advantage and create our own client which allows us to interact with the Telegram Application programming interface (API) and use it to download data. We here use the Pyrogram [59] API framework

[1] https://github.com/tdlib/td

which is a python wrapper around Telegrams MTProto encryption protocol and allows for asynchronous interaction with the Telegram API.

To systematically download content from Telegram we use the fact that messages can be forwarded from one channel to another (see Fig. 4.1). The metadata of a such a forwarded message contains its origin. Therefore we can use this to discover new channels and download their content. We start with a single channel and download all messages from it. We then look for forwarded messages in these messages and download the channels from which these messages were forwarded. We then repeat this process for the newly discovered channels. This allows us to discover new channels in a recursive way. This process is called snowball crawling (see Algorithm 4.1).



B Channel 1 14:53
Some very important facts I want to share with the channel!

A Channel 1 14:52
Some very important facts I want to share with the channel!

2 chats

A Channel
1 subscriber

October 2

Channel "A Channel Name" created

A Channel 1 1 14:52
Some very important facts I want to share with the channel!

Broadcast

B Channel
1 subscriber

October 2

Channel "B Channel" created

B Channel 1 14:53
A Channel 14:52
Some very important facts I want to share with the channel!

Broadcast

Figure 4.1: Example of message forwarding between two Telegram channels **A** shows previews of the two channels “A Channel” and “B Channel”. **B** shows the messages of the “A Channel”, here a single message was posted. In **C**, the message of “B Channel” are shown, notably a message from “A Channel” was forwarded and is show. Note the metadata of the message which shows the origin of the message as “A Channel”.

The Telegram API is not completely public, depending on the request it is necessary to provide Telegram account credentials. Further, there are some limits[2] on the number of messages which can be viewed/downloaded per day and the number of channels which one can be part of at the same time. To overcome these limitations we used multiple Telegram accounts and distributed the crawling process. This allowed us to partially sidestep these limitations and download more data in a shorter time.

Multiple accounts and distributed crawling

For the crawler, we used 198 Telegram accounts. Each account was created by using telephone numbers i.e. by sim card. Luckily because of business contracts by the Max Planck Institute for Dynamics and Self-Organization we were able to get them relatively cheap. On the other hand the effort activating each Telegram account and

[2] https://limits.tginfo.me/en

```
Data: Initial channel $C_{\text{initial}}$
Result: List of discovered Telegram channels
// Initialize a queue with the initial channel
$q \leftarrow \{C_{\text{initial}}\}$
while $q$ is not empty do
    // Pop item from the queue
    $C_{\text{current}} \leftarrow \text{Dequeue}(q)$
    // Get all messages from the current channel
    $m \leftarrow \text{GetMessages}(C_{\text{current}})$
    // Get forwarded messages
    $m_f \leftarrow \text{GetForwardedMessages}(m)$
    // Discover new channels
    $C_{\text{new}} \leftarrow \text{DiscoverChannels}(m_f)$
    // Add newly discovered channels to the queue
    $\text{Enqueue}(q, C_{\text{new}})$
end
```

Algorithm 4.1: **Snowball crawling for Telegram channels.** Details on distribution of the crawling process are omitted for clarity.

getting an API authorization key was quite high, it took us about 3 days to activate all accounts with a four people team.

We then distributed the crawling process over 3 machines, one running the database and two running the crawler workers itself. Luckily Telegram does not limit the number of accounts which can access its service from one IP-Address. Otherwise, this project might be way more complicated.

4.1.1 Guidance

Given the sheer volume of public channels, it's essential to focus our efforts on those channels most relevant to our analysis. For instance, channels solely dedicated to reporting cryptocurrency prices should not be downloaded or only considered once more meaningful channels have been exhausted. To this end, we developed a guidance mechanism to prioritize the crawling of channels with a higher likelihood of containing pertinent data.

As the content of messages is not available before downloading the channel, we predict the relevance/content of a channel based on already collected channels which have links to it i.e. channels which contain forwarded messages from it.

We rank each new found channel by a priority score and place them in a download queue. The score is predicted by the number of keywords in the channels description, the channels name and the number of keywords in the messages of the channel where the new channel is found. For instance if we are currently crawling a channel called "CovidTruths" and we find a new channel called "CovidFacts", we compute a priority score for the new channel

”CovidFacts” by counting the number of keywords in its metadata and in the messages of ”CovidTruths. We than download the channel with the highest priority score first.

Channels which we predict to be more relevant to our analysis are thus downloaded with a higher priority. The snowball crawling approach is guided towards a topic hence the name guidance mechanism. This process is illustrated in a simplified fashion in Fig. 4.2.

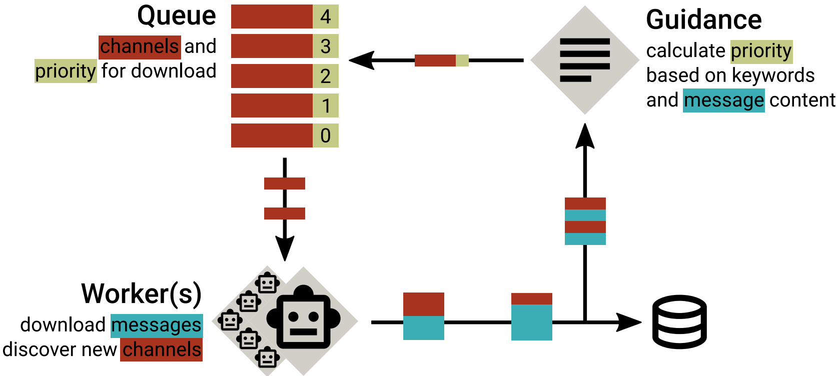


Figure 4.2: **Simplified overview of the crawling process including the guidance mechanism.** A number of workers download messages of channels in parallel. Once a channel is fully downloaded all content is written to a database and further used for the guidance mechanism. The guidance mechanism then calculates or updates the priority score for each newly found channel. The channel with the highest priority is then downloaded next. This process is repeated indefinitely.

To formalize this, let us consider we find a new channel c_{new} within the messages M of the channel c_{current}. We then calculate the priority of c_{new} by the following

$$P_{c_{\text{new}}} = \left(1 + ck(N_{c_{\text{new}}}) + ck(D_{c_{\text{new}}})\right) \cdot \sum_{m \in M} ck(m) \tag{4.1}$$

where ck is the keyword count function, $N_{c_{\text{new}}}$ is the name of the new channel and $D_{c_{\text{new}}}$ is its description and m is a message in the set of messages M of the current channel. This process is additive in the sense that the priority of a channel is the sum of the priorities of all channels that lead to it.

The keyword count function ck is counting the the number of times a specific string occurs in a given text. We also transform all text to lower case, i.e. if the keyword is ”covid” and the text contains ”Covid19” it is counted as a match. This is done to avoid at least some stemming and lemmatization, which can be computationally expensive in an online setting. Further, stemming and lemmatization is not always possible as we are working with multilingual data and not all languages have a good stemming or lemmatization libraries.

All of this allows to predict the relevancy of a chat without having to download it, which is useful as the download of a chat can take a long time, especially if it is a large chat with many messages and by the limitations imposed by Telegram. This allows us to prioritize

```
Data: Initial channel $C_{\text{initial}}$, keyword list $K$
Result: List of discovered Telegram channels
// Initialize a priority queue with the initial channel
$q_p \leftarrow \{(C_{\text{initial}}, 0)\}$
while $q$ is not empty do
    // Pop item from the queue with the highest priority
    $C_{\text{current}}, p \leftarrow$ Dequeue($q_p$)
    // Get all messages from the current channel
    $m \leftarrow$ GetMessages($C_{\text{current}}$)
    // Get forwarded messages
    $m_f \leftarrow$ GetForwardedMessages($m$)
    // Discover new channels
    $C_{\text{new}} \leftarrow$ DiscoverChannels($m_f$)
    // Calculate priority of new channels, see (4.1)
    $p_{\text{new}} \leftarrow$ CalculatePriority($C_{\text{new}}$, $m$, $K$)
    // Add newly discovered channels to the queue
    Enqueue($q_p$, ($C_{\text{new}}, p_{\text{new}}$))
end
```

Algorithm 4.2: **Snowball crawling for Telegram channels with guidance mechanism.** Details on distribution of the crawling process are omitted for clarity.

downloading channels which are more likely to be relevant to our analysis and hence create a more focused dataset as we do not download every chat we find. For a pseudo code implementation see Algorithm 4.2.

As we mainly wanted to focus on the diffusion of (mis-)information and specifically in the context of the COVID-19 pandemic, we created a keyword list based on COVID-19 pandemic related words. The main list contains 87 different keywords in English language, which are further translated into 44 different languages. The main list can be found in listing 4.1. We selected the words in this list by picking 10 channels which are relevant to our analysis preliminary to the crawling process.Than we use a bag-of-words approach to find the most commonly used words in these channels. This initial list is then further refined by hand and translated into other languages using google translate. The translation can be found in the supplementary information S3. We combine all languages into one keyword list and use this list to calculate the priority values with the ck function.

This guidance mechanism introduces a potential bias, especially in the early stages of the data collection process. The bias arises from the way channels are prioritized for crawling based on the number of keywords in their descriptions, names, and the messages of the current channel. While the intention is to focus on channels that are more likely to be relevant to the analysis, this method inadvertently favors channels that heavily use the keywords. For our analysis this is not necessarily a problem and in the limit this bias should disappear as the number of channels with a high keyword count are limited. However, this should be kept in mind when interpreting the results of the analysis.

```
keywords = ['covid','corona','virus','pandemic','lockdown',⌋
↪  'health','mask','distancing','outbreak','symptom',⌋
↪  'quarantine','influenza','vaccine','vaccination', 'pandemic',
↪  'ventilator', 'isolation', 'immunity','hospital','icu',⌋
↪  'intensive care unit','treatment','virologist','clinic',⌋
↪  'homeopathy','sick','pharma','polymerase chain reaction',
↪  'pcr','pertussis','emergency', 'injection','doctors',⌋
↪  'cellular','remote work','frontline','covid-19','sars',⌋
↪  'sanitation','pathogen','propaganda','disease','epidemic',⌋
↪  'diarrhea','adjuvants','respiratory','hygiene','protein',⌋
↪  'medicine','new cases','positive','scientist','contagious',⌋
↪  'mandate','variant','infect','deaths','ill','cough',⌋
↪  'measure','viral','mrna','prevent','healthcare','contract',⌋
↪  'shutdown','smallpox','booster','antibody','dose','evidence',⌋
↪  'misinformation','isolation','observed','mandatory',⌋
↪  'allergic','allergy','immune','shortage','syndrome','drug',⌋
↪  'chinese','test','restriction','spread','vax','experimental']
```

Listing 4.1: Base list of English keywords used in guidance mechanism to calculate priority values. For the list of keywords in other languages, see supplementary information S3.

Nonetheless, the guidance mechanism remains a valuable approach for data collection across diverse research contexts. Its foremost advantage lies in its efficiency, enabling the prioritization of channels with a higher likelihood of containing pertinent data. This strategic approach allows us to concentrate our data collection efforts on channels more likely to yield valuable information, such as those engaged in discussions about the COVID-19 pandemic. It was precisely this efficiency that motivated the development of the guidance mechanism.

4.1.2 Seed list

We initially started the crawling based on a single chat named ”Corona_Fakten” but later on switched to a more diverse seed list. The seed list is a list of channels which are used to start the snowball crawling. The download queue is initialized with these channels and the crawling process starts using them. The seed list is created by picking manually picking a language diverse set of 100 COVID-19 related channels from the TGStats website[3]. The seed list can be found in listing S.45.

4.1.3 Software

The software package is created with the FAIR Principles in mind, emphasizing data Findability, Accessibility, Interoperability, and Reusability [60]. The source code is readily available on GitHub[4]. The solution is containerized using Docker, allowing for straight forward deployment and use in various computing environments.

[3] https://tgstat.com/

[4] https://github.com/Priesemann-Group/telegram_crawler

We opted to use the Pyrogram [59] API framework which is a python wrapper around Telegrams MTProto encryption protocol and allows for asynchronous interaction with the Telegram API. Further we use SQLalchemy [61] as an Object Relational Mapper (ORM) to interact with the database which allows for efficient and high-performing database access.

Furthermore, we have also developed an auxiliary monitoring website to track and analyze the performance and usage of our software. This website provides real-time insights into the system's behavior, making it a valuable tool for maintaining and optimizing the crawlers operation.

All data is stored in a relational database. This allows to query the data in a structured manner, allowing to filter, search and perfom other analysis in the future. We use a MariaDB [62] database as it is open source and because of previous positive experiences with it. Here we deploy two database replicas to increase throughput.

Anecdots from the development

Of course during development we run into a number of problems, here I want to shortly take some time to discuss some of them.

We initially did not expect to collect as much data, therefore we never fought about limitations of integer as an identifier field. Indeed this lead to some serious problems as the messages table grew in size, the number of available identifiers in the 32 bit integer "id" column was exhausted. This lead to multiple days of migrating the database to a 64 bit integer "id" column. Luckily we were able to do this without any data loss but we lost a few days of data collection.

Initially we also did not use multiple machines but we always developed it with distributed computing in mind. As a single worker in the crawler (representing a Telegram account) does not take much Central Processing Unit (CPU) resources we always though running on a single machine was enough. However, we quickly realized that the crawler is very Input/Output (IO) intensive and that the CPU is not the bottleneck but rather the IO to the database. This lead to the decision to distribute the crawler over multiple machines and to use a single database server. This lead to a significant increase in performance and allowed us to collect more data in a shorter time.

4.1.4 Data description

We gathered a comprehensive dataset that encompassed a most of elements within the Telegram platform. Specifically, our data collection efforts included all available and non binary data, i.e. text messages, users, channels, polls and all related metadata. We opted to disregard binary data, such as images and videos, due to their large size and possible trademark/ownership issues. The database is structured as shown in figure 4.3.

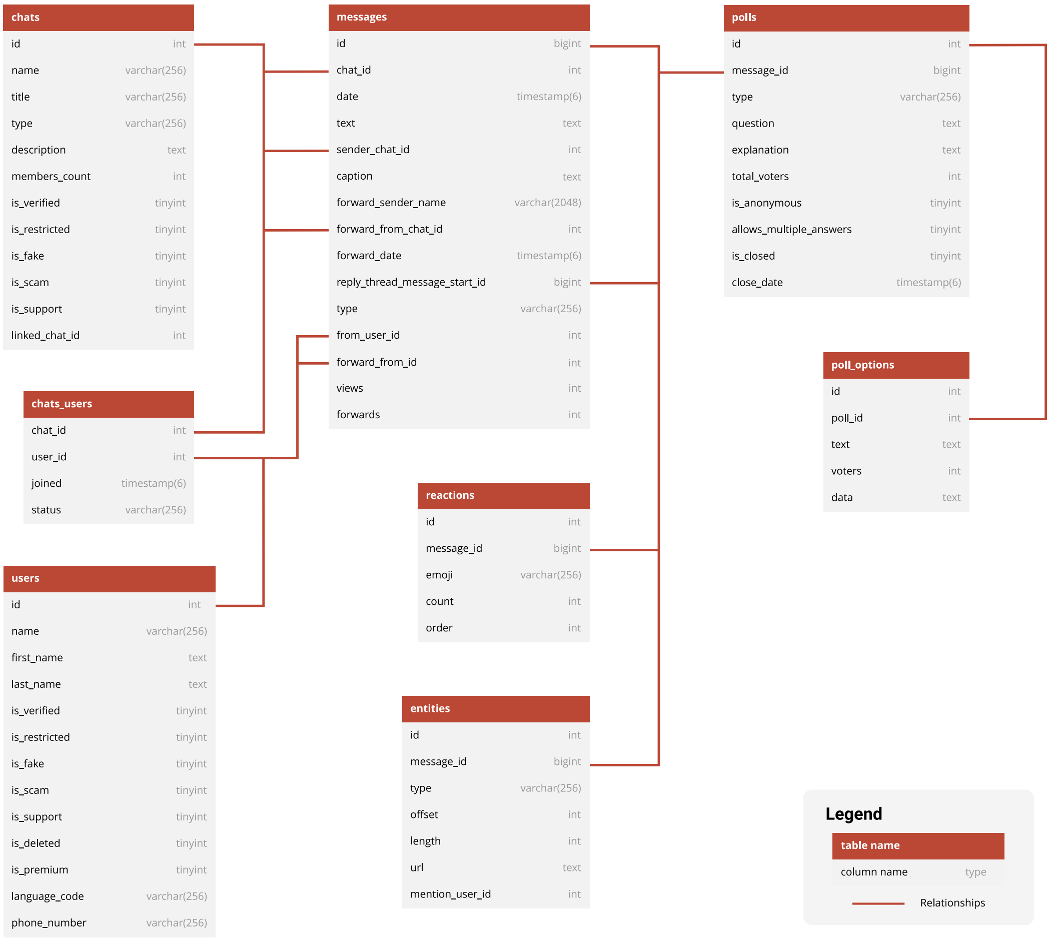


Figure 4.3: Database structure of the sql database storing the Telegram dataset. Boxes denote tables and red lines denote relations between tables.

Additionally to a messages' text content we record its metadata, including the number of views, reactions (i.e. likes, dislikes, etc.) and entities (i.e. mentions, hashtags, links). We also record the users of a channel if possible as it can be disabled by the channels admin and some general metadata of the channel i.e. its description and some tags which are set by Telegram. If the message is a poll we also record it in a structured manner with its possible options and votes for each option if available. We also record if a message was forwarded and from which channel and by whom.

4.1.5 Preprocessing

As we are not too interested in the content of the messages for this analysis. Rather in the structure by forwards, we disregard the text for now and create a directed multigraph to represent the interactions within the Telegram dataset. This graph is constructed based on forwards of messages from one channel to another. Each node in the graph represents a channel, and directed edges represent the forwarded messages. We only include fully downloaded channel, i.e. we know all outgoing edges of a channel. The graph is created using a single SQL join query (see Listing 4.2). The results of the query corresponds to the edge list of the multigraph. The graph is then stored in a the *gt* (graph-tool) binary format for the further analysis.

```sql
SELECT
    m.id as message_id,
    m.chat_id as to_chat_id,
    m.forward_from_chat_id as from_chat_id,
    q.status as to_status,
    q2.status as from_status
FROM messages as m
    JOIN queue as q ON m.chat_id = q.chat_id
    JOIN queue as q2 ON m.forward_from_chat_id = q2.chat_id
WHERE m.forward_from_chat_id is not null;
```

Listing 4.2: SQL query to create the edge list from the database.

4.2 General framework for stochastic blockmodels

With the graph data collected and preprocessed we can now start to think about how to infer the modular structure of the Telegram network. In this section we will introduce the general framework for SBMs and derive the entropy of the ensemble for the standard SBM. This will allow us to formulate the problem of inferring the modular structure as an optimization problem.

Let us consider an observed graph G which is composed of N nodes and E edges. In our case the nodes are the Telegram channels and the edges are forwarded messages between channels. For now we simplify the graph to only contain undirected edges, we set multiple edges between the same nodes to a single edge, and we disregard self loops. The resulting simplified graph is called a simple graph.

SBM are a class of generative models used in statistical graph analysis to describe and understand the structure of complex networks. These models are based on the assumption that nodes within a graph can be partitioned into latent groups or communities and the edges between nodes are stochastically generated based on the community membership of the nodes. A number of different variants of this model exist, which cater to various aspects and complexities of real-world graphs.

The standard SBM formulation assumes that all the nodes belong to the same community if they are statistically indistinguishable. This means that the nodes with same expected degree are part of the same community [63]. Assuming high homogeneity in the degree distribution of the nodes is often not a reasonable assumption for real-world graphs. To circumvent this problem, the degree corrected SBM [64] is used. One might also assume nodes can belong to multiple communities at the same time. Models including this assumption are called overlapping SBM [65] or mixed membership SBM [66]. By considering the temporal information of the graph, this allows to capture the changing block memberships and evolving patterns of connections over different time intervals [67–70].

What is a node's degree?

The term ”degree” in the context of graphs is used to denote the number of edges incident to a node. In other words, the degree of a node represents its connectivity or the number of connections it has with other nodes.

Even though there exists variety of SBMs, they all share the same general framework. Let us consider a simple graph for now i.e. the standard SBM, first. In this work, our goal is to find the optimal community membership c_i for each node i. In other words, we want to maximize the likelihood of observing the graph given a particular partition $\{c_i\} \in [1, C]$ of the nodes into communities. General speaking, one can directly derive the log-likelihood $\mathcal{L}$ by considering the probability to observe a particular graph realization P.

$$\mathcal{L} = \ln P \tag{4.2}$$

Let's assume all the graph ensembles are realized with the same probability $P = 1/\Omega$, where Ω is the number of possible graphs in the ensemble [5]. This can be interpreted as a microcanonical ensemble with an entropy $S = \ln \Omega$. The entropy for this ensemble can just be derived from the log-likelihood:

$$\begin{aligned}\mathcal{L} &= \ln P \\ &= \ln \frac{1}{\Omega} \\ &= -\ln \Omega \\ \mathcal{L} &= -S \end{aligned} \tag{4.3}$$

Expressing the log-likelihood in relation to the entropy of the ensemble, allows us to later on simplify and generalize notation. If we can compute the entropy of the ensemble for a given model, we can directly derive the log-likelihood. Thus this is applicable to any graph model where the entropy can be computed.

I have a random question, ”what is entropy?”

Entropy can be interpreted as the amount of disorder or randomness of a system.

In the context of our graph analysis, it represents the number of different ways the graph can be arranged while still satisfying certain constraints.

[5] Assuming the same probability, gives a uniform and noninformative prior for the partitioning. Other prior distributions are also possible, but we will not consider them in the following.

Think of it this way: if you have a puzzle with many pieces, the entropy would be low if all the pieces fit together in only one specific way. But if the pieces can be arranged in many different combinations and still form a complete puzzle, then the entropy would be high.

In our case, the entropy is related to the number of possible SBM ensembles that follow the same characteristics as the observed graph. The higher the entropy, the more different ways the partitions can be organized while still matching the observed graph edge and node distributions. On the other hand, a lower entropy means there are fewer ways to arrange the partitions while satisfying the constraints of the observed graph.

As a result of (4.3), instead of maximize the likelihood, we can equivalently minimize the entropy of the ensemble. This is the central idea of the stochastic blockmodel framework. We want to find the partition $\{\hat{c_i}\}$ that minimizes the entropy of the ensemble. Therefore, we can formulate the problem as:

$$\{\hat{c_i}\} = \underset{\{c_i\}}{\arg\min}\, S(\{c_i\}) \quad \text{or} \tag{4.4}$$

$$\{\hat{c_i}\} = \underset{\{c_i\}}{\arg\max}\, \mathcal{L}(\{c_i\}) \tag{4.5}$$

Finding the optimal partition $\{\hat{c_i}\}$ is generally not tractable, i.e. the exact enumeration to test all possible partitions is not feasible for most networks as naive testing via enumeration scales with $\mathrm{O}\binom{N}{C}$, where N is the number of nodes and C the number of communities. Instead one must rely on approximating methods which are able to sample partitions with a probability given as a function of the entropy S (see section 4.4).

The entropy consistently decreases as the number of communities C increases. This makes using entropy alone ineffective for identifying the optimal number of communities, especially when the true number is unknown. This can be seen as an example of overfitting, where the ”best” case puts each node in its own community. The most complex description minimizes the entropy, but it is not the most useful result. To address this, we introduce a penalty term through model selection, specifically the minimum description length (MDL) criterion (see section 4.3), which helps us determine the most suitable model, including the number of communities.

Before we delve into extending the general framework with model selection considerations, let’s first explore the entropy of the ensemble in the context of simple graphs. This will help us in deriving the entropy for more complex graphs later on and also give us a better understanding of the entropy itself.

Entropy of the standard stochastic blockmodel

The standard stochastic blockmodel [63] assumes an observed simple graph $\mathcal{G}$, i.e. there is only one edge between two nodes and no self-loop. Further, it assumes that the nodes in the graph can be partitioned into C distinct communities $\{c_i\} \in [1, C]$. This section considers the undirected case, the directed case is derived later. We also assume that the degrees of the nodes are homogeneous within each community, which especially for real-world graphs is often not the case. We will address this issue later on by introducing the degree corrected stochastic blockmodel.

As we have seen earlier, if we are able to derive the entropy of the ensemble, we can use our general framework to infer the community structure. To derive the entropy of a model one first needs to compute the total number of graphs in the ensemble. We define the number of edges between two communities x and y as e_{xy} and n_x, n_y as the number of nodes in the respective communities. For any given pair of communities x and y we can compute the number of possible graphs using the binomial coefficient. The total number of possible graphs realizations Ω is then given by the product of all possible graphs between all pairs of communities. Here we only consider half of the matrix, as the other half is symmetric. This is given by

$$\Omega_{xy} = \binom{n_x n_y}{e_{xy}} \tag{4.6}$$

$$\Omega = \prod_{x \geq y} \Omega_{xy} \tag{4.7}$$

We can now obtain the entropy of the standard stochastic blockmodel as $S_{ssb} = \ln \Omega$. Assuming the values of n_x are large enough that Stirling's approximation $\ln \binom{N}{r} \cong NH(\frac{r}{N})$ holds, we obtain the condensed expression

$$S_{ssb} = \sum_{x \geq y} \ln \Omega_{xy} \tag{4.8}$$

$$= \frac{1}{2} \sum_{x,y} n_x n_y H\left(\frac{e_{xy}}{n_x n_y}\right) \tag{4.9}$$

for the entropy of the standard blockmodel. Here, $H(x)$ is defined as the binary entropy function, see also equation (4.11).

Approximating the logarithm of the binomial coefficient using stirling approximation

The stirling approximation is used to approximate the logarithm of the factorial function. It is given by

$$\ln N! \cong N \ln N - N. \tag{4.10}$$

By employing this approximation, we can simplify the logarithm of the binomial coefficient, given by:

$$\begin{aligned}\ln\binom{N}{r} &= \ln\frac{N!}{r!(N-r)!}\\ &\cong N\ln N - N - r\ln r + r - (N-r)\ln(N-r) + (N-r) = T\end{aligned}$$

Although this expression may not appear immediately useful, we can reframe it by adding and subtracting $r\ln N$:

$$\begin{aligned}T &= N\ln N - r\ln r - (N-r)\ln(N-r)\\ &= N\ln N - r\ln r - (N-r)\ln(N-r) + r\ln N - r\ln N\\ &= (N-r)\ln N + r(\ln N - \ln r) - (N-r)\ln(N-r)\\ &= (N-r)\ln\frac{N}{N-r} + r\ln\frac{N}{r}\end{aligned}$$

By introducing the binary entropy function as:

$$H(x) = -x\ln x - (1-x)\ln(1-x) \tag{4.11}$$

and performing further rearrangements, we arrive at:

$$\begin{aligned}T &= \frac{N}{N}r\ln\frac{N}{r} + \frac{N}{N}(N-r)\ln\frac{N}{N-r}\\ &= N\Big(\tfrac{r}{N}\ln\frac{1}{\frac{r}{N}} + (1-\tfrac{r}{N})\ln\frac{1}{1-\frac{r}{N}}\Big)\\ &= NH(\tfrac{r}{N})\end{aligned}$$

Therefore, we obtain the final approximation for the logarithm of the binomial coefficient as:

$$\ln\binom{N}{r} \cong NH(\tfrac{r}{N}) \tag{4.12}$$

For a full derivation of the stirling approximation and further information see e.g. Information Theory, Inference and Learning Algorithms [71].

4.3 Model selection

As we have already seen earlier, the entropy is a function which is strictly decreasing with the number of communities C. Thus, we cannot use the entropy directly to select the optimal number of communities. Instead, we have to use a different approach to select the optimal number of communities.

Here, the approach of model selection comes into play. It allows us to select the most appropriate model among a set of competing models based on their statistical evidence. In the context of community detection, it helps us determine the optimal number of communities by comparing different community partition models and selecting the one that best explains the data.

Two widely adopted strategies for addressing the challenge of model selection are Bayesian model selection (BMS) and the minimum description length (MDL) criterion, both of which have proven effective in various studies [72–74, 27, 75]. It's worth mentioning that these two approaches converge to identical results [29] given the same model constrains. In the context of this work, we will use MDL for selection the number of communities and Bayesian model selection (BMS) for comparing different model varieties.

In the following we will first introduce the MDL criterion, than we will show how to compute the description length of the standard SBM. Finally, we will introduce the Bayes factor and show how to use it to compare different models variants as this is the approach we use to compare different SBM variants.

4.3.1 Minimum Description Length criterion

The MDL criterion is a general method for model selection that can be applied to any model. It has its origin in information theory but is often seen as an mathematical application of Occam's razor. It suggests that among competing hypotheses or models that explain the same data equally well, the one with the fewest assumptions should be preferred. This concept is often paraphrased as "simpler explanations are more likely to be correct".

Occam's razor

Occam's razor, also known as the principle of parsimony, is a fundamental principle in philosophy and science that aligns closely with the concept of simplicity. It is named after the medieval philosopher and Franciscan friar William of Ockham[a] but is said to predate him.

This principle can be encapsulated by the Latin phrase "Frustra fit per plura quod potest fieri per pauciora," which translates to "It is futile to do with more things that which can be done with fewer". It is applied in a variety of fields including philosophy, economy, law, and statistics.

[a] The spelling here is not wrong. For some reason the community decided to spell it differently.

Following this principle, the most appropriate model is the one that is simple but also explains the data well. The MDL criterion is a formalization of this concept. The description length of the data is the number of bits required to encode the data using the model and is an information theoretic measure of the complexity of the data.

In our case the description length is the entropy of the ensemble given by the model and also the information necessary to describe the model itself. This quantity is than given by

$$\Sigma = -\ln \mathcal{P}(D|\theta) - \ln \mathcal{P}(\theta) \tag{4.13}$$

$$= S + \mathcal{Z} \tag{4.14}$$

where the first term is the entropy of the ensemble S and the second is the information necessary to describe its parameters $\mathcal{Z}$. Here we used the shorthand θ to denote all parameters of the model. In our case the model parameters θ are the block matrix e_{xy} and the partition $\{\hat{c}_i\}$. The minimum value for the description length is thus an upper bound on the information necessary to describe the data [75]. The model with the minimum description length is thus the one that best compresses the data.

This can be seen as an extension to the general framework of SBMs. Instead of minimizing the entropy of the ensemble, we minimize the description length of the ensemble and obtain the optimal partition $\{\hat{c}_i\}$ without being constraint to a fixed number of communities C. Therefore, we can reformulate (4.4) as

$$\hat{\theta} = \arg\min_{\theta} \Sigma \tag{4.15}$$

where $\hat{\theta}$ is the optimal set of parameters for the model.

The most straightforward way to use the MDL criterion to select the optimal number of communities is to perform a parameter swipe over different values of C and select the one with the minimum description length. Even though this approach is useful for evaluation and comparison, it is computationally expensive as we have to fit a model for each value of C (see Fig. 4.4). We will see later on how to approach this problem in a more efficient way (see section 4.4).

Before we dive into the details of inference of communities, let's first look our simple example of the standard stochastic blockmodel and calculate the description length of the model.

4.3.2 Description length of the standard stochastic blockmodel

For the standard stochastic blockmodel, the information necessary to describe the model is relatively straightforward to calculate. It is determined by the number of possible partitions and the number of possible block matrices. One can interpret the block matrix as the adjacency matrix of a graph itself with C nodes and E edges which allows multiple edges between the same pair of nodes. Therefore, the total number of possible block matrices is given by $\left(\!\!\binom{\left(\!\binom{C}{2}\!\right)}{E}\!\!\right)$ and the number of possible partitions is given by C^N.

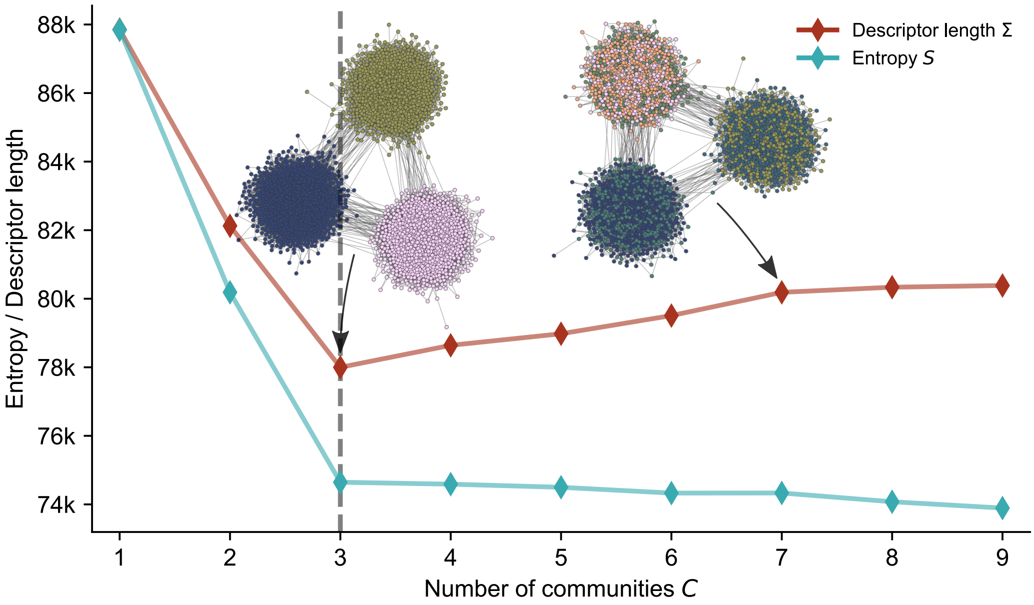


Figure 4.4: The graph demonstrates the relationship between the number of proposed communities, denoted as C, and the corresponding description length, obtained after equilibration. The optimal compression of data occurs when $C = 3$, as it minimizes the description length most effectively. However, it's important to note that as the entropy of the model decreases with increasing C but the necessary information to describe the model actually increases. For the chosen parameters of the generated planted partition graph, refer to Fig. 4.5.

Multiset coefficients

A multiset is a flexible mathematical construct that extends the idea of a set by permitting repeated elements. In contrast to sets, where each element appears only once, multisets embrace the notion of duplicates.

The notation for a multiset coefficient, denoted by $\left(\!\binom{r}{k}\!\right)$, expresses the number of ways to choose k elements from a multiset with r distinct elements. That is the number of ways to choose k elements from a set of r elements with repetition. Multiset coefficients show a connection to binomial coefficients, and their calculation can be simplified as

$$\left(\!\!\binom{r}{k}\!\!\right) = \binom{r+k-1}{k} \tag{4.16}$$

This relationship makes computing multiset coefficients more accessible and reduces complex combinatorial problems to simpler binomial calculations.

The information necessary to describe the standard stochastic blockmodel is then obtained by multiplying both quantities and taking the logarithm. This yields

$$\begin{aligned}
\mathcal{Z}_{ssb} &= \ln\left[C^N \cdot \left(\!\!\binom{\left(\!\binom{C}{2}\!\right)}{E}\!\!\right)\right] && (4.17)\\
&= N \ln C + \ln\left[\left(\!\!\binom{\left(\!\binom{C}{2}\!\right)}{E}\!\!\right)\right] && (4.18)\\
&\cong N \ln C + EH\left(\frac{C(C+1)}{2E}\right) && (4.19)
\end{aligned}$$

where H is the binary entropy function as defined as in (4.11).

4.3.3 Bayes factor

Depending on the data it might sometimes be necessary to compare different model variants to find the most suitable one. For example, in the context of community detection, we might want to compare a model with degree correction to one without degree correction. We can compare two models by considering the posterior distribution of each model.

Let's consider two models denoted by their parameters θ_a and θ_b and the observed data D. The ratio of the posterior distributions of the two models is given by

$$K_{ab} = \frac{\mathcal{P}(\theta_a|D)}{\mathcal{P}(\theta_b|D)} = \frac{\mathcal{P}(D|\theta_a)}{\mathcal{P}(D|\theta_b)} \cdot \frac{\mathcal{P}(\theta_a)}{\mathcal{P}(\theta_b)} \tag{4.20}$$

where we used Bayes theorem to obtain the second equality. This ratio K_{ab} is the posterior odds ration or also called Bayes factor. It quantifies the relative strength of evidence between the two models. If the Bayes factor is greater than 1, the data favors model θ_a and if it is less than 1, the data favors model θ_b. Generally, Bayes factors greater than 3 or 5 are considered substantial evidence in favor of one model over the other [76].

In our context we can even simplify this equation by using the description length of the model. By encapsulating both denominator and numerator with an exponential and using the definition of the description length (4.13), we obtain

$$\begin{aligned}
\mathcal{P}(D|\theta)P(\theta) &= \exp(\ln(\mathcal{P}(D|\theta)\mathcal{P}(\theta))) && (4.21)\\
&= \exp(\ln \mathcal{P}(D|\theta) + \ln \mathcal{P}(\theta)) && (4.22)\\
&= \exp(-\Sigma) && (4.23)
\end{aligned}$$

where Σ is the description length of the model. This allows us to rewrite the Bayes factor as

$$K_{ab} = \frac{\mathcal{P}(D|\theta_a)P(\theta_a)}{\mathcal{P}(D|\theta_b)P(\theta_b)} = \frac{\exp(-\Sigma_a)}{\exp(-\Sigma_b)} \tag{4.24}$$

$$= \exp(\Sigma_b - \Sigma_a) \tag{4.25}$$

$$= \exp(-\Delta\Sigma) \tag{4.26}$$

where $\Delta\Sigma = \Sigma_a - \Sigma_b$ is the difference in description length between the two models. This means that the Bayes factor is proportional to the ratio of the description length of the two models [32]. Therefore, the blockmodel model with the smaller description length is the one that better explains the data.

As we are normally working with multiple samples, i.e. Σ_a^s and Σ_b^t where s and t are the sample indices, we can compute the Bayes factor for each sample combination between the two models. This yields a distribution of Bayes factors which we can use to compute the confidence intervals. Which can be quite compute intensive depending on the number of samples and models. Alternatively, instead of computing all combinations of samples one can use bootstrapping to obtain the distribution of Bayes factors. It is done by randomly drawing from the samples and computing the Bayes factor for each drawn sample combination. This yields a distribution of Bayes factors which we can use to compute the confidence intervals. In the limit both methods converge to the same result [77].

4.4 Inference of communities

Computing the optimal partition $\{\hat{c_i}\}$ for a given graph is generally not tractable, as the exact enumeration to test all possible partitions is not feasible for most bigger networks. The naive testing (via enumeration) scales with $\mathrm{O}\binom{N}{C}$, where N is the number of nodes and C the number of communities. Instead one must rely on approximating methods which are able to sample partitions with a probability given as a function of the entropy S or description length Σ.

Markov chain Monte Carlo (MCMC) methods are chosen for tasks like optimal network partitioning because they offer a practical solution to complex problems where exhaustive enumeration is computationally impractical. Rather than evaluating all possible configurations, MCMC provides a stochastic approach that efficiently explores the solution space by sampling configurations according to a probability distribution, often derived from the problem's entropy or likelihood. This stochasticity enables MCMC to navigate high-dimensional spaces, making it a powerful tool for approximating solutions in various fields, including network analysis and statistical modeling [78].

Markov Chain

A Markov Chain is a collection of random variables $\{X_t\}$, where t is a discrete index representing time. The evolution of the chain is governed by a transition probability $P(X_{t+1}|X_t)$, which embodies the assumption that the future state of the chain depends only on its current state and not on any of its previous states. This is called the Markov property. For a more formal and detailed definition, refer to Siddhartha (2001) [78].

Specifically to approximate the optimal partition, we use the Metropolis-Hastings [79, 80] algorithm to sample partitions according to a probability distribution derived from the entropy of the ensemble. Further, we enhance the naive Metropolis-Hastings algorithm by using a multiflip approach [27] which allows to better explore the parameter space and better escape local minima. Lastly we show the agglomerative heuristics [28] approach which is very similar to ”traditional” merge and split approaches [29]. This approach mainly improves computational efficiency but also helps to avoid metastable configurations. These methods are mostly implemented by the *graph-tool* library [81].

4.4.1 The naive Metropolis implementation

General speaking, MCMC methods are concerned with calculating the likelihood of our model parameters θ given some data D, i.e. $\mathcal{P}(\theta|D)$. This is referred to as the posterior probability. The posterior probability can be computed using Bayes' theorem

$$\mathcal{P}(\theta|D) = \frac{\mathcal{P}(D|\theta)\mathcal{P}(\theta)}{\mathcal{P}(D)}. \tag{4.27}$$

The biggest computational problem here lies in the denominator, i.e. the marginal likelihood $\mathcal{P}(D)$. This is almost never know in analytical form, as it requires integrating over all possible values of θ. The MCMC methods are based on the idea of sampling from the posterior distribution $\mathcal{P}(\theta|D)$ without having to explicitly calculate this marginal likelihood.

The central idea is to construct a Markov chain that has the desired posterior distribution as its equilibrium distribution. Once this chain converges to its equilibrium distribution (i.e., reaches a stationary state), the samples drawn from the chain represent samples from the posterior distribution.

Specifically, we can use a naive Metropolis approach, which resolves around moving the community membership of nodes randomly, and accepting or rejecting those moves. A proposed move is accepted or rejected based on the entropy change caused by the move. These proposed changes are accepted or rejected based on a proposal probability that is proportional to the posterior distribution $\mathcal{P}(\theta|G)$. This process is exact, since it is guaranteed to eventually produce the partitions with the desired probabilities, if the Markov chain is ergodic, satisfies the detailed balance condition and is run for a sufficiently long time.

Ergodicity and detailed balance condition

The MCMC approach relies on the ergodic theorem, which asserts that a Markov chain achieves ergodicity when it is both irreducible and aperiodic. This means the chain can reach any state from any other state and doesn't repeatedly return to a specific state. Essentially, the chain is capable of fully exploring the entire state space.

Ergodicity is guaranteed by satisfying the detailed balance condition, which serves as a sufficient condition. This condition ensures that the probability of transitioning from state x to state y is the same as transitioning from state y to state x. In simpler words, the chain is reversible, and this condition is also referred to as the balance condition.

This process is performed in sweeps, where one sweep consists of N iterations, where N is the number of nodes in the graph. After each sweep, the community membership of each node is updated once. This process is repeated until the Markov chain reaches equilibrium. The number of sweeps required to reach equilibrium is called the mixing time τ. The mixing time is a measure of how long it takes for the Markov chain to produce samples that are independent from the initial state. Once the mixing time is reached, the Markov chain is said to have converged to its equilibrium distribution i.e. samples drawn from the Markov chain after mixing represent samples from the posterior distribution $\mathcal{P}(\theta|G)$.

The naive approach is to propose a move to another community with equal probability i.e. uniform (see algorithm 4.3) and accept or reject the move based on the entropy change of the proposed move. In practice one also incorporates the possibility to add a new community or remove one but we will not consider this here for simplicity. In this case one would use the descriptor length instead of the entropy. The partition entropy change ΔS is given by the difference between the entropy of the proposed move and the current state of the Markov chain. The acceptance probability a is than given by

$$a = \min\left\{e^{-\beta\Delta S}, 1\right\} \tag{4.28}$$

where β is a tuning parameter, often called ”inverse temperature parameter”. This parameter controls the balance between exploration and exploitation. A higher value of β leads to a higher probability of accepting a move that increases entropy. This means that the algorithm is more likely to explore the parameter space. On the other hand, a lower value of β leads to a higher probability of accepting a move that decreases the entropy. This means that the algorithm is more likely to exploit the current state of the Markov chain.

The β parameter can be tuned during the runtime of the algorithm. The idea is to start with a high value of β and gradually decrease it over time. This allows to explore the parameter space in the beginning and then gradually exploit the current state of the Markov chain as it approaches equilibrium and generally helps to escape local minima. This process is called simulated annealing [82].

Even though this algorithm is ergodic and satisfies detailed balance, it can be inefficient. If the number of communities C is large and the structure of the network is well defined (i.e. well constrained by the data), most of the proposed moves are rejected. This can lead to very long mixing times and thus a very long runtime of the algorithm.

```
/* Pick an initial state for the community memberships of all nodes */
$\{c_i^0\} \leftarrow \text{rand}(0, C) \quad \forall i$
$t \leftarrow 0$
while $t < t_{max}$ do
    /* Propse a move $x \rightarrow y$ */
    $i \leftarrow \text{rand}(0, N)$
    $x \leftarrow c_i^t$
    $y \leftarrow \text{rand}(0, C)$
    /* Calculate entropy change of the proposed move and acceptance
       probability */
    $\Delta S \leftarrow S(x \rightarrow y) - S(x)$
    $a \leftarrow \min\{e^{-\beta \Delta S}, 1\}$
    /* Accept or reject the move */
    $c^{t+1} \leftarrow c^t$
    if $a > rand(0, 1)$ then
        $c_i^{t+1} \leftarrow y$
    end
    $t \leftarrow t + 1$
end
```

Algorithm 4.3: Naive Metropolis algorithm for community inference. Note, that an known value or initial value for C is picked here and the transition is proposed from community x to y.

4.4.2 Multiflip Metropolis-Hastings algorithm

A more efficient approach is to propose a move based on the neighborhood of a node as it often results in more meaningful changes in the community structure. This approach is known as the Multiflip Metropolis-Hastings algorithm [27] and consists of attempting to move a node i from community x to y with a probability based on the neighborhood of the node.

Before accepting or rejecting the move $x \rightarrow y$, we first consider the neighborhood of the node i i.e. its nearest neighbors. We choose a random neighbor node j with the community z. Depending on the number of edges e_{zy} between the neighboring community z and the picked community y we accept or reject the move. This conditional probability is then given by

$$\begin{aligned} p(x \to y|z) &= \frac{e_{zy} + \epsilon}{e_z + \epsilon C} && (4.29) \\ &= (1 - R_z)\frac{e_{zy}}{e_z} + R_z \frac{1}{C} && (4.30) \\ \text{with} \quad R_z &= \frac{\epsilon C}{e_z + \epsilon C} && (4.31) \end{aligned}$$

where ϵ is a free parameter and it is normalized probability by the number of half edges $e_z = \sum_x e_{zx}$ and the number of communities C. This is a generalization of the naive approach, as it can be shown that for $\epsilon \to \infty$ we obtain the uniform proposal probability of the naive approach.

For every $\epsilon > 0$ this process is ergodic but it is not guaranteed to satisfy detailed balance. However, this can be enforced using the established Metropolis-Hastings algorithm [83, 79]. A move is accepted with a probability given by

$$a = \min\left\{ e^{-\beta \Delta S} \frac{\sum_z p_z^i p(y \to x|z)}{\sum_z p_z^i p(x \to y|z)}, 1 \right\} \tag{4.32}$$

where p_z^i is the fraction of neighbors to node i which belong to block z and β is a tunning parameter, see also eq. (4.28).

Identical to the naive approach, the Multiflip approach is a general applicable, as it can be applied to any blockmodel where the entropy can be computed. A pseudo code implementation of the Multiflip Metropolis-Hastings algorithm can be found in algorithm 4.4. Considering the mixing time τ, the computational complexity of this algorithm is $\mathrm{O}(\tau N)$ but requires more memory than the naive approach, as one needs to keep track of the neighborhood of a node [27, 28].

Even though the Multiflip Metropolis-Hastings approach is more efficient than the naive one, it still has some problems. Depending on the initial state i.e. the starting point, the mixing time can vary heavily. It is larger if the initial state is far from typical partitions. If the community structure is well defined, this can lead to metastable configurations where the community structure of the graph is only partially discovered (see Fig. 4.5). This is a problem as the algorithm can get stuck in a metastable configuration and can take a very long time to escape such metastable configurations. Additionally, by observing the entropy of the ensemble, one can wrongly derive that the equilibrium has been reached. Resulting in inaccuracy of the final partitions. Therefore, simply considering the entropy of one ensemble is not a good indicator for the quality of the partition.

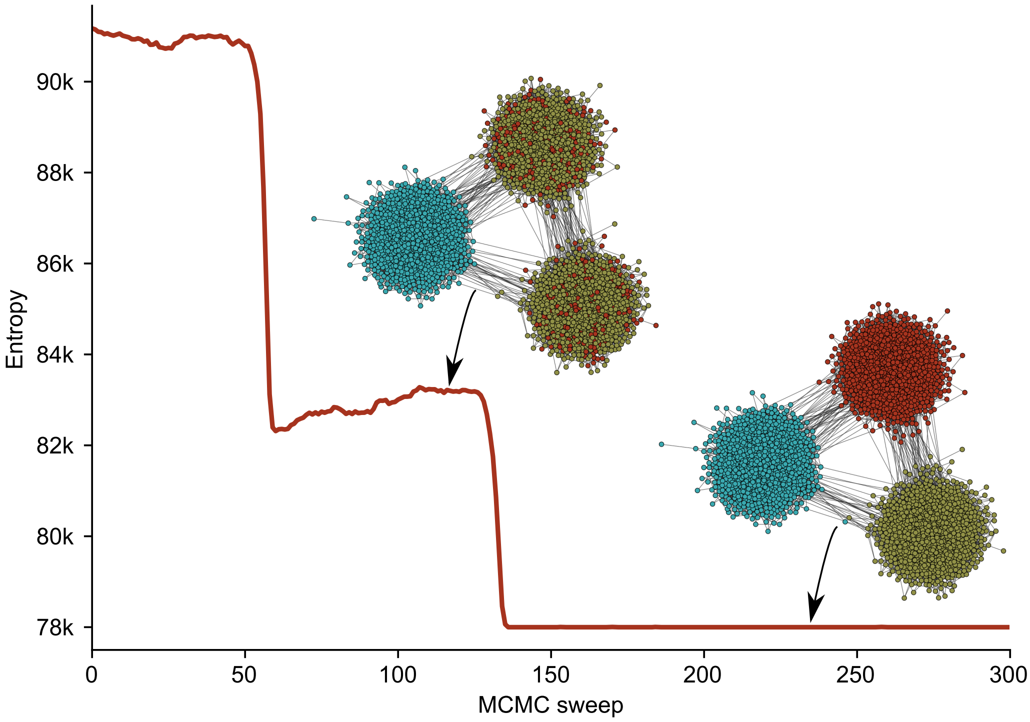


Figure 4.5: During the initial iterations of the MCMC algorithm, the community structure is only partially revealed, but as the algorithm progresses, it becomes capable of identifying the correct partitions. However, a potential issue arises during the early stages of the algorithm, where premature conclusions about reaching equilibrium can be drawn solely from the entropy of the ensemble, this can effect the accuracy of the final results. In this example a randomly generated planted partition graph [84, 22] was used with 3 groups, 1000 nodes in each group, a probability of connecting vertices within a group of 0.85% and a probability of connected vertices between groups of 0.005%. The naive Metropolis algorithm was used to infer the community structure.

4.4.3 Agglomerative heuristics

To avoid the metastable states mentioned earlier, one can exploit the influence of block sizes on their occurrence [28]. One can use a known better configuration for some $C' > C$ and then use it to obtain a better configuration for C. By merging communities together based on their size (node count) and edge counts a better configuration can be obtained. Implementation is done by considering each merge as a community move in the MCMC algorithm.

For each node, m moves are proposed again by considering the neighborhood of the node (4.29). However, now we start with an initial partition for $C' = N$ and progressively reduce

```
/* Pick an initial state for the community memberships */
$\{c_i\} \leftarrow \text{rand}(0, C) \quad \forall i$
$t \leftarrow 0$
while $t < t_{max}$ do
    $i \leftarrow \text{rand}(0, N)$
    $x \leftarrow c_i^t$
    $y \leftarrow \text{rand}(0, C)$
    $j \leftarrow \text{rand-neighbor}(i)$ // Randomly select a neighbor $j$ of node $i$
    $z \leftarrow c_j^t$
    if $R_z < rand(0, 1)$ then
        /* Recject $y$ proposal and choose a neighbor community instead */
        y ← rand-neighbor(x)
    end
    $\Delta S \leftarrow S(x \rightarrow y) - S(x)$
    /* Calculate acceptance probability by considering all neighbor
       blocks $t$ */
    $a \leftarrow \min\left\{ e^{-\beta \Delta S} \frac{\sum_t p_t^i p(y \rightarrow x|z)}{\sum_t p_t^i p(x \rightarrow y|z)}, 1 \right\}$
    if $a > rand(0, 1)$ then
        $c_i^{t+1} \leftarrow y$
    end
    $t \leftarrow t + 1$
end
```

Algorithm 4.4: Multiflip Metropolis-Hastings algorithm for community inference

the value of C to reach the desired partition size. That is, given some σ each iteration we decrease the number of communities by

$$C_i + 1 = \frac{C_i}{\sigma} \tag{4.33}$$

until we reach the desired number of communities C. The value of σ is a parameter that can be tuned to control the number of merges per iteration.

To counter the impact of unfavorable merges in the initial stages, nodes are also allowed to move between merge steps. This is achieved by applying the naive or multiflip MCMC algorithm with $\beta \rightarrow \infty$.

The Agglomerative heuristics approach has been shown to almost always avoid metastable configurations, resulting in improved partition quality. Additionally, the algorithm exhibits excellent computational efficiency, with an overall complexity of approximately $O(N \log^2 N)$, making it suitable for large-scale network analysis.

For additional details on the implementation of the agglomerative heuristics and a number of evaluations see Peixoto, 2014 [28].

4.4.4 Sampling and convergence

For all models, sampling was performed using the *graph-tool* library [81] using a mixture between the multiflip MCMC algorithm and agglomerative heuristics. We initiated 4 chains and run a initial minimization of the descriptor length by using agglomerative heuristics with a β value of ∞. This was done for one sweep because it is relatively cheap and gives a good starting point for the chains. Subsequently we used simulated annealing to further equilibrate/fine tune the chains. We started with a $\beta = 1$ and increased it up to $\beta = 10$ in 100 steps logarithmically spaced (i.e. exponential cooling). We considered the chains equilibrated after there was no change observed in the descriptor length Σ for a windows of 4000 draws with 4 chains. We sampled using an inverse temperature of $\beta = 10$ and a thinning of 4 (i.e. we only kept every 4th sample). This results in 4000 samples per chain and 16000 samples in total per model. In the further analysis, we discarded the first 1000 samples as burn-in for each chain.

Full equilibration of the chains can take a long time and is highly dependent on the proposed model. For instance, the standard multigraph model mgb fitting the full Telegram qraph was achieved after ≈20000 sweeps, here the chains converged to a stable state and the descriptor length did not decrease further (see Fig. 4.6).

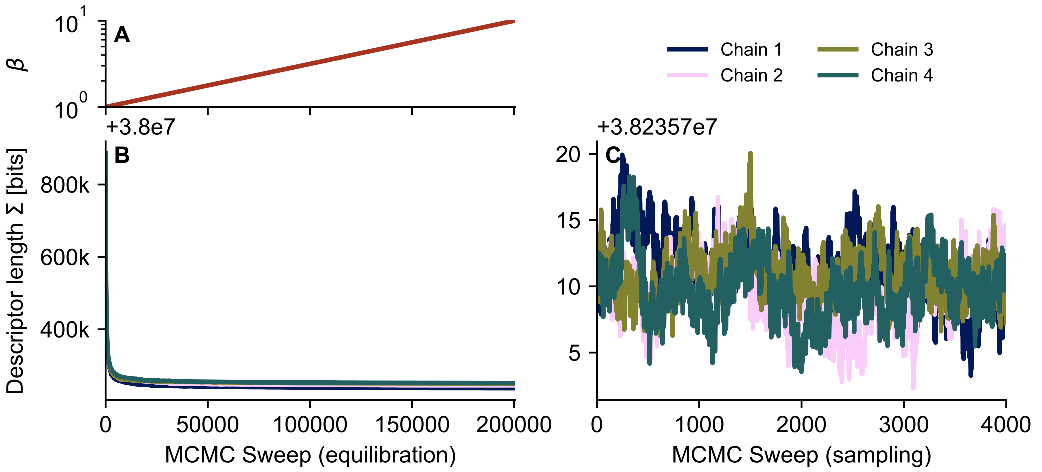


Figure 4.6: **Equilibration and sampling of the standard multigraph model on the Telegram graph.** We employ simulated annealing with a beta value from 1 to 10 (**A**) and a thinning of 4. The descriptor length Σ continuously decreases until it reaches a stable value after about 20k sweeps (**B**). During sampling we observed that the chains are converged to a similar state (**C**).

To evaluate the convergence and sample quality, we use the Gelman-Rubin statistic, denoted as $\hat{R}$ (R-hat) [85, 86]. The utilization of this statistic serves verification to the dependability of our findings and confirming that the Markov chains used for sampling have successfully converged. We find that all models have converged to a stable state as the Gelman-Rubin statistic $\hat{R}$ is close to 1 for all parameters (see Table 4.1). An example of the sample trace of the multigraph model mgb is shown in Fig. 4.6. All models show very similar equilibration and sampling behavior but depending on the model the equilibration can take longer.

Shorthand	maximum $\hat{R}$
mgb	1.119
mgb^d	1.134
mgb^c	1.102
mgb^{dc}	1.159
mgb^h	1.109
mgb^{dh}	1.102
mgb^{ch}	1.064
mgb^{dch}	1.098

Table 4.1: **Table of convergence statistics for all models.** The Gelman-Rubin statistic $\hat{R}$ is close to 1 for all parameters, indicating that the chains have converged to a stable state.

R-hat

The idea is to monitor the convergence of the Markov chain by comparing the variance between chains to the variance within chains and estimating the factor by which the scale of the current distribution for each parameter might be reduced if the chain were to be run for an infinite amount of time.

The between chain variance is estimated by

$$B = \frac{n}{m-1} \sum_{j=1}^{m} (\bar{\theta}_{.j} - \bar{\theta}_{..})^2 \tag{4.34}$$

where m is the number of chains, n is the number of samples per chain, $\bar{\theta}_{.j}$ is the mean of the jth chain and $\bar{\theta}_{..}$ is the mean of all chains. Here θ is a scalar parameter of the model. The within chain variance is estimated by

$$W = \frac{1}{m} \sum_{j=1}^{m} s_j^2 \tag{4.35}$$

where s_j^2 is the variance of the jth chain. Using these two, we can estimate the marginal posterior variance of var$(\theta|y)$ by the weighted average

$$\widehat{\text{var}}^{+}(\theta|y) = \frac{n-1}{n} W + \frac{1}{n} B \tag{4.36}$$

and the potential scale reduction factor by

$$\hat{R} = \sqrt{\frac{\hat{\mathrm{var}}^{+}(\theta|y)}{W}}. \tag{4.37}$$

This factor offers insights into how much the scale of the parameter distribution could be reduced, providing a crucial indication of the chain's convergence. If $\hat{R}$ is close to 1, the chains have converged, thus yielding reliable samples for posterior inference [86].

4.5 Variety of different stochastic blockmodels

We evaluate a variety of different stochastic blockmodels to find the most likely model for our Telegram dataset. We start with the standard stochastic blockmodel and than add features to it to create a more realistic model. We will extend the standard stochastic blockmodel to account for directed edges, multiple edges and implement a degree corrected version. We will also consider hierarchical blockmodels which allow to describe the network on different scales and overcome the resolution limit of the standard stochastic blockmodel. Further we consider the possibility that nodes can belong to multiple communities at the same time (overlap). For each of these models we will derive the entropy and the descriptor length.

This is done in a modular fashion, i.e. we can combine different features to create a model which is suitable for our dataset. For instance, we can combine the directed and degree corrected extensions to obtain a model which allows for directed edges and degree correction.

For naming conventions we proceed as follows. We identified two basic classes of blockmodels, one allowing only singular edges between nodes and one allowing multiple edges between nodes. The first class is the standard stochastic blockmodel (denoted ssb) and the second class is the multigraph blockmodel (denoted mgb). This differentiation is made because of the inherent differences in their assumed data structure, thus model selection between the two classes is not very meaningful. For both classes we define the following extensions: directed edges (denoted with superscript d), degree correction (denoted c), overlapping communities (denoted o) and hierarchical blockmodels (denoted h). Extensions can be combined arbitrary, e.g. a stochastic blockmodel which allows for directed edges and overlap in community assignment is denoted ssb^{do}. For a list of all extensions and their corresponding shorthand see table 4.2. All possible combinations sum up to 32 unique models.

Considering our Telegram dataset, the model should be directed and allow for multiple edges between nodes as forwarding a message is a directed action and multiple forwards between the same two nodes are possible. Nonetheless, we test all models but do not expect reasonable results from models which do not allow for directed edges or multiple edges.

Superscript	Short description
d	Directed edges
c	Degree correction
o	Overlapping communities
h	Hierarchical blockmodel

Table 4.2: Table of different model extensions.

4.5.1 Multigraph blockmodel

We can extend the the standard SBM to consider the case with multiple edges between nodes. This is useful for a variety of reasons, for once the blockmodel itself is a multigraph i.e. with the block matrix e as edges and the community partitions $\{c_i\}$ as nodes. Further, in our Telegram dataset we need to consider a multigraph as multiple messages can be forwarded between the same two channels. This is also the case for other social networks, where multiple interactions between the same two nodes are possible.

For now let's consider the undirected case without degree correction, we will see how to implement these features later on. The total number of different edge choices between two blocks x and y is now given by

$$\Omega_{xy} = \left(\!\!\binom{n_x n_y}{e_{xy}}\!\!\right) \qquad \Omega_{xx} = \left(\!\!\binom{\left(\!\binom{n_x}{2}\!\right)}{e_{xx}/2}\!\!\right) \tag{4.38}$$

where $\left(\!\binom{n}{k}\!\right)$ is the multiset coefficient. Similar to the standard stochastic blockmodel the total number of possible graphs is given by

$$\Omega = \prod_{x \geq y} \Omega_{xy} \tag{4.39}$$

and than again using the Stirling approximation we obtain the entropy

$$S_{mgb} = \ln \Omega \tag{4.40}$$

$$= \frac{1}{2} \sum_{xy} (n_x n_y + e_x y) H\left(\frac{n_x n_y}{n_x n_y + e_{xy}}\right) \tag{4.41}$$

where $H(x) = -x \ln x - (1-x)\ln(1-x)$ is the binary entropy function, see (4.11).

The description length can be derived identical to the standard stochastic blockmodel (4.19) and is thus given by

$$\mathcal{Z}_{mgb} \cong N \ln C + EH\left(\frac{C(C+1)}{2E}\right) \tag{4.42}$$

where E is the total number of edges in the block matrix e and C is the number of communities. The approximation is due to the Stirling approximation used in the derivation of the entropy (4.41).

4.5.2 Directed edges

Let's consider the edges which are represented by forwarded messages in our dataset. These edges are directed, as the message is forwarded from one channel to another. However, as of now we have only considered the undirected case for both, the standard stochastic blockmodel (see subsection 4.2 and 4.3.2) and the multigraph blockmodel (see 4.5.1).

To account for the directional nature of the edges, we can just disregard the earlier symmetry considerations. Thus the entropy resulting in the nearly the same expression as in (4.9) or (4.41) but without the factor of $1/2$. I.e. we can just take the product over all directed x, y pairs i.e. $\Omega = \prod_{xy} \Omega_{xy}$. The entropy is then given by

$$S_{ssb^d} = \sum_{x,y} n_x n_y H\left(\frac{e_{xy}}{n_x n_y}\right) \tag{4.43}$$

$$S_{mgb^d} = \sum_{xy} (n_x n_y + e_{xy}) H\left(\frac{n_x n_y}{n_x n_y + e_{xy}}\right) \tag{4.44}$$

where we have used the same notation as in (4.9) and (4.41) but with the superscript d to denote the directed case.

Moreover, for the directed case, the information necessary to describe the model $\mathcal{Z}_{ssb^d}$ (or $\mathcal{Z}_{mgb^d}$) requires a modification as compared to the undirected case. We need to replace $C(C+1)/2 \rightarrow C^2$ in (4.19) as the orders of the blocks are not interchangeable anymore. Thus, we obtain the following expression for the description length of the directed stochastic blockmodel.

$$\mathcal{Z}_{ssb^d} = \mathcal{Z}_{mgb^d} \cong N \ln C + EH\left(\frac{C^2}{E}\right) \tag{4.45}$$

4.5.3 Degree correction

As shortly mentioned earlier, the observed degree of realistic networks is often not consistent with the expected degree distribution of the standard stochastic blockmodel. To account for this, one imposes an expected degree sequence $\{\kappa_i\}$ on all nodes N of the graph. Each individual κ_i represent the average degree of a node i over all samples in the ensemble. We can now separate different nodes by their degree in blocks. Thus, extending the block formulation to (x, κ) where the first label is the community label and the second is the expected degree label. Here we assume the average degree in block (x, κ) is κ. Using this we can rewrite the the full entropy from (4.9) as,

$$S_{ssb^c} = \frac{1}{2} \sum_{x\kappa y\kappa'} n_{(x,\kappa)} n_{(y,\kappa')} H\left(\frac{e_{(x,\kappa),(y,\kappa')}}{n_{(x,\kappa)} n_{(y,\kappa')}} \right) \tag{4.46}$$

This now accommodates undirected blockmodels with arbitrary degree sequences, but also of arbitrary degree correlation, since it is defined as a function of the full matrix $e_{(x,\kappa),(y,\kappa')}$ [26].

In practice, focusing on the ensemble while limiting the total number of edges between blocks, regardless of their expected degrees, proves to be more advantageous. This is not possible as a closed form solution but one can derive an approximation, as can be seen in [26] or [64].

$$S_{ssb^c} \cong E - \sum_{\kappa} N_\kappa \ln \kappa - \frac{1}{2} \sum_{xy} e_{x,y} \ln\left(\frac{e_{x,y}}{e_x e_y} \right) \tag{4.47}$$

$$\text{with } e_{x,y} = \sum_{\kappa\kappa'} e_{(x,\kappa),(y,\kappa')} \tag{4.48}$$

Hereby E is the total number of edges in the block matrix, N_κ is the number of nodes with expected degree κ and $e_x = \sum_y e_{x,y}$ is the total number of edges in block x.

To obtain the minimal descriptor length we need to add the description length of the expected degree sequence $\{\kappa_i\}$ to the description length of the blockmodel. This is given by

$$\mathcal{Z}_{ssb^c} = \mathcal{Z}_{ssb} + N \sum_{\kappa} p_\kappa \ln p_\kappa \tag{4.49}$$

where p_κ is the fraction of a nodes having degree κ in the ensemble and the information necessary is given by (4.19). If we want to use the directed version of the degree corrected blockmodel we need to use $\mathcal{Z}_{ssb^d}$ instead of $\mathcal{Z}_{ssb}$ and further replace $\kappa \rightarrow (\kappa^-, \kappa^+)$ for differentiation of ingoing and outgoing degree in the previous equations.

Analogously, we can extend the multigraph blockmodel to account for degree correction by using (4.41) as our starting point. The entropy is then given by

$$\mathcal{S}_{mgb^c} = \sum_{x\kappa y\kappa'} (n_{(x,\kappa)} n_{(y,\kappa')} + e_{(x,\kappa),(y,\kappa')}) H\left(\frac{n_{(x,\kappa)} n_{(y,\kappa')}}{n_{(x,\kappa)} n_{(y,\kappa')} + e_{(x,\kappa),(y,\kappa')}} \right) \tag{4.50}$$

and the description length is given by (4.49).

Extending the directed model variants is also done in the same fashion, we separate the edges into blocks depending on the ingoing and outgoing degrees. This is than given by the block labels (x, κ^-, κ^+). If we insert this into (4.43) and (4.44) we obtain the entropy for the directed degree corrected blockmodels.

$$\mathcal{S}_{ssb^{cd}} \cong \sum_{x\kappa^-\kappa^+y\kappa'^-\kappa'^+} n_{(x,\kappa^-,\kappa^+)} n_{(y,\kappa'^-,\kappa^+)} H\left(\frac{e_{(x,\kappa^-,\kappa^+),(y,\kappa'^-,\kappa^+)}}{n_{(x,\kappa^-,\kappa^+)} n_{(y,\kappa'^-,\kappa^+)}}\right) \tag{4.51}$$

$$\begin{aligned}\mathcal{S}_{mgb^{cd}} \cong & \sum_{x\kappa^-\kappa^+y\kappa'^-\kappa'^+} \left(n_{(x,\kappa^-,\kappa^+)} n_{(y,\kappa'^-,\kappa^+)} + e_{(x,\kappa^-,\kappa^+),(y,\kappa'^-,\kappa^+)}\right) \\ & \cdot H\left(\frac{n_{(x,\kappa^-,\kappa^+)} n_{(y,\kappa'^-,\kappa^+)}}{n_{(x,\kappa^-,\kappa^+)} n_{(y,\kappa'^-,\kappa^+)} + e_{(x,\kappa^-,\kappa^+),(y,\kappa'^-,\kappa^+)}}\right)\end{aligned} \tag{4.52}$$

For the approximations as in (4.47) and a full derivation of the entropy for the directed degree corrected multigraph blockmodel see [26].

4.5.4 Overlapping communities

It might be reasonable to assume that nodes can belong to multiple communities at the same time. For instance in social networks, a person can be part of multiple social circles. This is not possible in the standard stochastic blockmodel as each node can only belong to a single community. However, we can extend the standard stochastic blockmodel to allow for overlapping communities. This is relatively straightforward, instead of a singular partition $\{c_i\}$ we now have a partition for each node i we introduce a binary mixture vector $\vec{c}_i$ which is of length C and each entry $b_i^x \in \{0, 1\}$ denotes if node i belongs to community x. Further comparing to the models without overlap we only have to change the computation of the number of blocks which belong to a community x, i.e. $n_x = \sum_i b_i^x$. The entropy is then given by the original model variants (4.9) or (4.41) but with the new definition of n_x. This also works for the directed variants.

If one considers degree correction, one also has to analogous to the partition vector introduce the number of half edges incident on a given node i which belong to a community x, i.e. κ_i^x. Similarly the combined degree of node i is denoted as $\vec{\kappa}_i = \{\kappa_i^x\}$. The entropy is then derived analogous to the degree corrected models without overlap, see [30, 26].

$$S_{ssb^{co}} \cong -E - \frac{1}{2}\sum_{xy} e_{xy} \ln\left(\frac{e_{xy}}{e_x e_y}\right) - \sum_{ix} \ln \kappa_i^x! \tag{4.53}$$

In order to perform model selection we need to add the information necessary to describe the overlap structure to the description length of the model. This is done via encoding the parameters by a particular generative process while at the same time averting biases by being noninformative. Deriving this generative process would go beyond the scope of this thesis, but for a detailed derivation see [30]. The conclusion is that the information necessary to describe the overlap structure is given by

$$\mathcal{Z}_{ssb^o} = \ln\left(\!\!\binom{D}{N}\!\!\right) + \sum_d \ln\left(\!\!\binom{\binom{C}{n}}{n_d}\!\!\right) + \ln N! - \sum_{\vec{c}} n_{\vec{c}}! \tag{4.54}$$

where d are the number of mixtures i.e. $d_i = \sum_x c_i^x$ and n_d is the number of nodes with d mixtures and D denotes the maximum number of mixtures.

4.5.5 Hierarchies

All blockmodels considered earlier are based on the assumption that the network is described by a single partition. However, in many cases this is not the most realistic choice. For example, in social networks, one might want to describe the network on different scales, e.g. on the level of close friends, of larger even at the global level of the entire network. Additionally, the resolution limit is a problem for detecting the optimal number of communities in a network by using the minimum description length criterion. It can be shown that the maximum number of blocks which are detectable scale with $C_{max} \sim \sqrt{N}$ and this holds for directed and degree corrected blockmodels [27, 87] and is very similar to the modular optimization limit [24]. This limitation arises out of the lack of a priori assumptions about the network i.e. all realizations are given equal probability by default.

However, using a hierarchy of blockmodels where the upper levels serve as prior information for the lower levels allow to overcome this resolution limit. Using this approach allows to push the scaling to $C_{max} \sim \ln N$ [29].

As the blockmodel formulation can itself be interpreted as a multigraph we can propose a hierarchy where the lowest level is a model on the observed graph (such as a standard SBM) and the following $L - 1$ levels are multigraph blockmodels. At each level $l \in [0, L]$ there are C^{l-1} nodes, which are divided into C^l communities (with $C^l \leq C^{l-1}$). Let's denote the e_{xy}^l as the edge counts at level l and n_x^l as the count of nodes with community x at level l. Additionally we set $C^{-1} \equiv N$, such that the observed network is considered as a level in the hierarchy. We also restrict the number of edges between communities to be constant across all levels i.e. $\sum_{xy} e_{xy}^l = E$.

Using this we can define a combined entropy for all levels as

$$S_{ssb^h/mgb^h} = S_{ssb/mgb}(\{e_{xy}^0\}, \{n_x^0\}) + \sum_{l=1}^{L} S_{mgb}(\{e_{xy}^l\}, \{n_x^l\}) \tag{4.55}$$

where the full entropy corresponds to a nested sequence of network ensembles. Each sample from the ensemble at level l generates a sample at level $l - 1$ and the sample at level 0.

Similarly, to all other blockmodels we can use the minimum description length criterion to find the optimal number of communities at each level. The amount of information necessary to describe a level in the hierarchy is given by

$$\mathcal{Z}_h^l = \ln\left(\!\!\binom{C_l}{C_{l-1}}\!\!\right) + \ln C_{l-1}! - \sum_x \ln n_x^l! \tag{4.56}$$

and the total information to describe the whole hierarchy is than given by

$$\mathcal{Z}_{ssb^h/mgb^h} = \mathcal{Z}_{ssb/mgb} + \sum_{l=1}^{L} \mathcal{Z}_h^l \tag{4.57}$$

where we also need to include the information necessary to describe the lowest level i.e. $\mathcal{Z}_{ssb/dcb}$. For a more rigorous derivation of both the entropy and the information necessary to describe the hierarchy see Peixoto 2014 [29].

Identically to the standard stochastic blockmodel, we can extend the hierarchical blockmodel to account for directed edges and degree correction. This is done by using the same notation as in the previous sections and replacing the corresponding terms in the right hand side of (4.55) and (4.57).

4.6 Diversity Computation

Throughout the thesis, especially during the dataset description, we use the term *diversity* to quantify the variability of different features. We employ a method rooted in information theory and entropy. Even though we call it diversity it is more of a measure of uncertainty as it is derived from the Shannon entropy [88, 71].

The Shannon Entropy H^S for a discrete random variable X with possible n possible outcomes $x_i \forall i \in \{1, \ldots, n\}$ with probabilities $P(X = x_i)$ is defined as:

$$H^S(X) = -\sum_{i=1}^{n} P(X = x_i) \cdot \log_2(P(X = x_i)) \tag{4.58}$$

It describes the uncertainty of the outcome of a random variable. The higher the entropy the more uncertain the outcome. The entropy is maximal if all outcomes are equally likely (see also Section 4.2).

For a meaningful comparisons across different random variables we normalize the entropy, by dividing the entropy by the maximal entropy H^S_{max} of the random variable. This normalization results in our diversity metric D defined as:

$$D = \frac{H^S}{H^S_{max}} \tag{4.59}$$

Here, the maximal entropy H^S_{max} is computed by assuming that one outcome is always observed i.e. with probability 1. This leads to $H^S_{max} = \log_2(n)$.

Crucially, our adoption of the term *diversity* resonates with established principles in ecology. In ecological studies this allows to assess species diversity within ecosystems [89]. Even though we have called this metric (Shannon) diversity it is also known as the Shannon equitability index or more generally normalized Shannon entropy [88, 71].

Acronyms

API	Application programming interface
BMS	Bayesian model selection
CI	Confidence interval
CPU	Central Processing Unit
DSA	Digital services act
IO	Input/Output
MCMC	Markov chain Monte Carlo
MDL	Minimum description length
NPI	Nonpharmaceutical Intervention
ORM	Object Relational Mapper
SBM	Stochastic blockmodel
sfdp	Scalable Force-Directed Placement
SQL	Structured Query Language

Bibliography

1. European Commission, Brussels. *Eurobarometer 96.3 (2022)*. Version 1.0.0. GESIS, 2022. DOI: 10.4232/1.13908. URL: https://search.gesis.org/research_data/ZA7848?doi=10.4232/1.13908.
2. *Telegram Global MAU 2022*. Statista. URL: https://www.statista.com/statistics/234038/telegram-messenger-mau-users/.
3. *Telegram Privacy Policy*. Telegram. URL: https://telegram.org/privacy.
4. C. C. Su, M. Chan, and S. Paik. "Telegram and the Anti-ELAB Movement in Hong Kong: Reshaping Networked Social Movements through Symbolic Participation and Spontaneous Interaction". *Chinese Journal of Communication* 15:3, 2022, pp. 431–448. DOI: 10.1080/17544750.2022.2092167. URL: https://doi.org/10.1080/17544750.2022.2092167.
5. M. Wijermars and T. Lokot. "Is Telegram a "Harbinger of Freedom"? The Performance, Practices, and Perception of Platforms as Political Actors in Authoritarian States". *Post-Soviet Affairs* 38:1-2, 4, 2022, pp. 125–145. DOI: 10.1080/1060586X.2022.2030645. URL: https://www.tandfonline.com/doi/full/10.1080/1060586X.2022.2030645.
6. A. Urman, J. C.-t. Ho, and S. Katz. "Analyzing Protest Mobilization on Telegram: The Case of 2019 Anti-Extradition Bill Movement in Hong Kong". *Plos one* 16:10, 2021, e0256675. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0256675.
7. S. Walther and A. McCoy. "US Extremism on Telegram". *Perspectives on Terrorism* 15:2, 2021, pp. 100–124. JSTOR: 27007298. URL: https://www.jstor.org/stable/27007298.
8. J. Guhl and J. Davey. "A Safe Space to Hate: White Supremacist Mobilisation on Telegram". *Institute for Strategic Dialogue* 26, 2020. URL: https://www.isdglobal.org/wp-content/uploads/2020/06/A-Safe-Space-to-Hate.pdf.
9. M. Hoseini, P. Melo, F. Benevenuto, A. Feldmann, and S. Zannettou. "On the Globalization of the QAnon Conspiracy Theory through Telegram". In: Proceedings of the 15th ACM Web Science Conference 2023. 2023, pp. 75–85. DOI: 10.1145/3578503.3583603. URL: https://doi.org/10.1145/3578503.3583603.
10. M. Zehring and E. Domahidi. "German Corona Protest Mobilizers on Telegram and Their Relations to the Far Right: A Network and Topic Analysis". *Social Media+ Society* 9:1, 2023, p. 20563051231155106. DOI: 10.1177/20563051231155106. URL: https://doi.org/10.1177/20563051231155106.
11. D. L. M. Lummen. "Is Telegram the New Darknet? A Comparison of Traditional and Emerging Digital Criminal Marketplaces", 2023. URL: http://essay.utwente.nl/94687/.

12. *REGULATION (EU) 2022/2065 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 19 October 2022 on a Single Market For Digital Services and Amending Directive 2000/31/EC (Digital Services Act)*. URL: https://eur-lex.europa.eu/eli/reg/2022/2065/oj.
13. *Telegram Misses the New EU Regulation on Tech Platforms for This Time*. Durov˙s Code. 21, 2023. URL: https://durovscode.com/the-dsa-does-not-yet-apply-to-telegram.
14. M. A. Gisondi, R. Barber, J. S. Faust, A. Raja, M. C. Strehlow, L. M. Westafer, and M. Gottlieb. "A Deadly Infodemic: Social Media and the Power of COVID-19 Misinformation". *Journal of Medical Internet Research* 24:2, 1, 2022, e35552. DOI: 10.2196/35552. URL: https://www.jmir.org/2022/2/e35552.
15. M. P. Patel, V. B. Kute, S. K. Agarwal, and O. behalf of COVID. "Infodemic COVID 19: More Pandemic than the Virus". *Indian journal of nephrology* 30:3, 2020, p. 188. DOI: 10.4103/ijn.IJN_216_20. URL: https://journals.lww.com/ijon/fulltext/2020/30030/_Infodemic__COVID_19__More_Pandemic_than_the_Virus.13.aspx.
16. F. A. Rathore and F. Farooq. "Information Overload and Infodemic in the COVID-19 Pandemic". *J Pak Med Assoc* 70:5, 2020, S162–S165. DOI: 10.5455/JPMA.38. URL: https://doi.org/10.5455/JPMA.38.
17. J. Zarocostas. "How to Fight an Infodemic". *The lancet* 395:10225, 2020, p. 676. DOI: 10.1016/S0140-6736(20)30461-X. URL: https://doi.org/10.1016/S0140-6736(20)30461-X.
18. G. Eady, T. Paskhalis, J. Zilinsky, R. Bonneau, J. Nagler, and J. A. Tucker. "Exposure to the Russian Internet Research Agency Foreign Influence Campaign on Twitter in the 2016 US Election and Its Relationship to Attitudes and Voting Behavior". *Nature Communications* 14:1, 9, 2023, p. 62. DOI: 10.1038/s41467-022-35576-9. URL: https://www.nature.com/articles/s41467-022-35576-9.
19. M. La Morgia, A. Mei, and A. M. Mongardini. *TGDataset: A Collection of Over One Hundred Thousand Telegram Channels*. 9, 2023. DOI: 10.48550/arXiv.2303.05345. arXiv: 2303.05345 `[cs]`. URL: http://arxiv.org/abs/2303.05345. preprint.
20. J. Baumgartner, S. Zannettou, M. Squire, and J. Blackburn. "The Pushshift Telegram Dataset". *Proceedings of the International AAAI Conference on Web and Social Media* 14, 26, 2020, pp. 840–847. DOI: 10.1609/icwsm.v14i1.7348. URL: https://ojs.aaai.org/index.php/ICWSM/article/view/7348.
21. T. P. Peixoto. *Descriptive vs. Inferential Community Detection in Networks: Pitfalls, Myths, and Half-Truths*. 31, 2023. DOI: 10.1017/9781009118897. arXiv: 2112.00183 `[physics, stat]`. URL: http://arxiv.org/abs/2112.00183.
22. S. Fortunato. "Community Detection in Graphs". *Physics Reports* 486:3-5, 2010, pp. 75–174. DOI: 10.1016/j.physrep.2009.11.002. URL: https://linkinghub.elsevier.com/retrieve/pii/S0370157309002841.
23. S. Fortunato and D. Hric. "Community Detection in Networks: A User Guide". *Physics Reports*. Community Detection in Networks: A User Guide 659, 11, 2016, pp. 1–44. DOI: 10.1016/j.physrep.2016.09.002. URL: https://www.sciencedirect.com/science/article/pii/S0370157316302964.

24. S. Fortunato and M. Barthélemy. "Resolution Limit in Community Detection". *Proceedings of the National Academy of Sciences* 104:1, 2, 2007, pp. 36–41. DOI: 10.1073/pnas.0605965104. URL: https://pnas.org/doi/full/10.1073/pnas.0605965104.
25. T. P. Peixoto. "Bayesian Stochastic Blockmodeling". In: *Advances in Network Clustering and Blockmodeling*. John Wiley & Sons, Ltd, 2019, pp. 289–332. DOI: 10.1002/9781119483298.ch11. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119483298.ch11.
26. T. P. Peixoto. "Entropy of Stochastic Blockmodel Ensembles". *Physical Review E* 85:5, 30, 2012, p. 056122. DOI: 10.1103/PhysRevE.85.056122. URL: https://link.aps.org/doi/10.1103/PhysRevE.85.056122.
27. T. P. Peixoto. "Parsimonious Module Inference in Large Networks". *Physical Review Letters* 110:14, 5, 2013, p. 148701. DOI: 10.1103/PhysRevLett.110.148701. URL: https://link.aps.org/doi/10.1103/PhysRevLett.110.148701.
28. T. P. Peixoto. "Efficient Monte Carlo and Greedy Heuristic for the Inference of Stochastic Block Models". *Physical Review E* 89:1, 13, 2014, p. 012804. DOI: 10.1103/PhysRevE.89.012804. URL: https://link.aps.org/doi/10.1103/PhysRevE.89.012804.
29. T. P. Peixoto. "Hierarchical Block Structures and High-Resolution Model Selection in Large Networks". *Physical Review X* 4:1, 24, 2014, p. 011047. DOI: 10.1103/PhysRevX.4.011047. URL: https://link.aps.org/doi/10.1103/PhysRevX.4.011047.
30. T. P. Peixoto. "Model Selection and Hypothesis Testing for Large-Scale Network Models with Overlapping Groups". *Physical Review X* 5:1, 25, 2015, p. 011033. DOI: 10.1103/PhysRevX.5.011033. URL: https://link.aps.org/doi/10.1103/PhysRevX.5.011033.
31. D. Hric, T. P. Peixoto, and S. Fortunato. "Network Structure, Metadata, and the Prediction of Missing Nodes and Annotations". *Physical Review X* 6:3, 12, 2016, p. 031038. DOI: 10.1103/PhysRevX.6.031038. URL: https://link.aps.org/doi/10.1103/PhysRevX.6.031038.
32. T. P. Peixoto. "Nonparametric Bayesian Inference of the Microcanonical Stochastic Block Model". *Physical Review E* 95:1, 17, 2017, p. 012317. DOI: 10.1103/PhysRevE.95.012317. URL: https://link.aps.org/doi/10.1103/PhysRevE.95.012317.
33. T. P. Peixoto. "Merge-Split Markov Chain Monte Carlo for Community Detection". *Physical Review E* 102:1, 13, 2020, p. 012305. DOI: 10.1103/PhysRevE.102.012305. URL: https://link.aps.org/doi/10.1103/PhysRevE.102.012305.
34. T. P. Peixoto. "Revealing Consensus and Dissensus between Network Partitions". *Physical Review X* 11:2, 5, 2021, p. 021003. DOI: 10.1103/PhysRevX.11.021003. URL: https://link.aps.org/doi/10.1103/PhysRevX.11.021003.
35. T. P. Peixoto. "Ordered Community Detection in Directed Networks". *Physical Review E* 106:2, 2022, p. 024305. DOI: 10.1103/PhysRevE.106.024305. URL: https://doi.org/10.1103/PhysRevE.106.024305.
36. A. R. Pamfil, S. D. Howison, R. Lambiotte, and M. A. Porter. "Relating Modularity Maximization and Stochastic Block Models in Multilayer Networks". *SIAM Journal on Mathematics of Data Science* 1:4, 2019, pp. 667–698. DOI: 10.1137/18M1231304. URL: https://doi.org/10.1137/18M1231304.
37. P. Zhang and C. Moore. "Scalable Detection of Statistically Significant Communities and Hierarchies, Using Message Passing for Modularity". *Proceedings of the National*

Academy of Sciences 111:51, 2014, pp. 18144–18149. DOI: 10.1073/pnas.1409770111. URL: https://doi.org/10.1073/pnas.1409770111.

38. A. Joulin, E. Grave, P. Bojanowski, M. Douze, H. Jégou, and T. Mikolov. “FastText.Zip: Compressing Text Classification Models”. 2016. arXiv: 1612.03651.
39. *What Are the Top 200 Most Spoken Languages?* Ethnologue (Free Dev). URL: https://www.ethnologue.com/insights/ethnologue200/.
40. T. Hale, N. Angrist, R. Goldszmidt, B. Kira, A. Petherick, T. Phillips, S. Webster, E. Cameron-Blake, L. Hallas, S. Majumdar, and H. Tatlow. “A Global Panel Database of Pandemic Policies (Oxford COVID-19 Government Response Tracker)”. *Nature Human Behaviour* 5:4, 4 2021, pp. 529–538. DOI: 10.1038/s41562-021-01079-8. URL: https://www.nature.com/articles/s41562-021-01079-8.
41. *Overview of the Implementation of COVID-19 Vaccination Strategies and Deployment Plans in the EU/EEA*. URL: https://www.ecdc.europa.eu/en/publications-data/overview-implementation-covid-19-vaccination-strategies-and-deployment-plans.
42. *Timeline of the Russian Invasion of Ukraine (24 February – 7 April 2022)*. In: *Wikipedia*. 16, 2023. URL: https://en.wikipedia.org/w/index.php?title=Timeline_of_the_Russian_invasion_of_Ukraine_(24_February_%E2%80%93_7_April_2022)&oldid=1185382482.
43. E. Mathieu, H. Ritchie, L. Rodés-Guirao, C. Appel, C. Giattino, J. Hasell, B. Macdonald, S. Dattani, D. Beltekian, E. Ortiz-Ospina, and M. Roser. “Coronavirus Pandemic (COVID-19)”. *Our World in Data*, 5, 2020. URL: https://ourworldindata.org/covid-stringency-index.
44. *Reactions, Spoilers, Translation and QR Codes*. Telegram. 30, 2021. URL: https://telegram.org/blog/reactions-spoilers-translations/fa?setln=en.
45. L. Muchnik, S. Pei, L. C. Parra, S. D. S. Reis, J. S. Andrade Jr, S. Havlin, and H. A. Makse. “Origins of Power-Law Degree Distribution in the Heterogeneity of Human Activity in Social Networks”. *Scientific Reports* 3:1, 1 7, 2013, p. 1783. DOI: 10.1038/srep01783. URL: https://www.nature.com/articles/srep01783.
46. A. D. Broido and A. Clauset. “Scale-Free Networks Are Rare”. *Nature Communications* 10:1, 1 4, 2019, p. 1017. DOI: 10.1038/s41467-019-08746-5. URL: https://www.nature.com/articles/s41467-019-08746-5.
47. M. Serafino, G. Cimini, A. Maritan, A. Rinaldo, S. Suweis, J. R. Banavar, and G. Caldarelli. “True Scale-Free Networks Hidden by Finite Size Effects”. *Proceedings of the National Academy of Sciences* 118:2, 12, 2021, e2013825118. DOI: 10.1073/pnas.2013825118. URL: https://www.pnas.org/doi/abs/10.1073/pnas.2013825118.
48. A. Levina, V. Priesemann, and J. Zierenberg. “Tackling the Subsampling Problem to Infer Collective Properties from Limited Data”. *Nature Reviews Physics* 4:12, 12 2022, pp. 770–784. DOI: 10.1038/s42254-022-00532-5. URL: https://www.nature.com/articles/s42254-022-00532-5.
49. S. B. Mohr, A. C. Schneider, and V. Priesemann. *Telegram Graph Data of COVID-19 Related Channels*. Version V2. 2023. DOI: 10.25625/H5JUZG. URL: https://doi.org/10.25625/H5JUZG.
50. H. Jeffreys. *The Theory of Probability*. OuP Oxford, 1998.

51. R. E. Kass and A. E. Raftery. "Bayes Factors". *Journal of the american statistical association* 90:430, 1995, pp. 773–795. DOI: 10.1080/01621459.1995.10476572. URL: https://doi.org/10.1080/01621459.1995.10476572.
52. Y. Hu. "Efficient and High Quality Force-Directed Graph Drawing". *Mathematica Journal* 10, 1, 2005, pp. 37–71. URL: https://www.mathematica-journal.com/issue/v10i1/contents/graph_draw/graph_draw.pdf.
53. D. Holten. "Hierarchical Edge Bundles: Visualization of Adjacency Relations in Hierarchical Data". *IEEE Transactions on Visualization and Computer Graphics* 12:5, 2006, pp. 741–748. DOI: 10.1109/TVCG.2006.147. URL: http://ieeexplore.ieee.org/document/4015425/.
54. F. Crameri, G. E. Shephard, and P. J. Heron. "The Misuse of Colour in Science Communication". *Nature Communications* 11:1, 1 28, 2020, p. 5444. DOI: 10.1038/s41467-020-19160-7. URL: https://www.nature.com/articles/s41467-020-19160-7.
55. J. Ramos. "Using Tf-Idf to Determine Word Relevance in Document Queries". In: Proceedings of the First Instructional Conference on Machine Learning. Vol. 242. 1. Citeseer, 2003, pp. 29–48.
56. M. Thoma. *Wili-2018 - Wikipedia Language Identification Database*. Zenodo, 7, 2018. DOI: 10.5281/ZENODO.841984. URL: https://zenodo.org/record/841984.
57. H. M. Wallach. "Topic Modeling: Beyond Bag-of-Words". In: *Proceedings of the 23rd International Conference on Machine Learning*. ICML '06. Association for Computing Machinery, New York, NY, USA, 25, 2006, pp. 977–984. DOI: 10.1145/1143844.1143967. URL: https://dl.acm.org/doi/10.1145/1143844.1143967.
58. W. A. Qader, M. M. Ameen, and B. I. Ahmed. "An Overview of Bag of Words;Importance, Implementation, Applications, and Challenges". In: *2019 International Engineering Conference (IEC)*. 2019 International Engineering Conference (IEC). 2019, pp. 200–204. DOI: 10.1109/IEC47844.2019.8950616. URL: https://ieeexplore.ieee.org/abstract/document/8950616.
59. *Pyrogram: Elegant, Modern and Asynchronous Telegram MTProto API Framework in Python for Users and Bots*. URL: https://github.com/pyrogram/pyrogram.
60. M. Barker, N. P. Chue Hong, D. S. Katz, A.-L. Lamprecht, C. Martinez-Ortiz, F. Psomopoulos, J. Harrow, L. J. Castro, M. Gruenpeter, P. A. Martinez, and T. Honeyman. "Introducing the FAIR Principles for Research Software". *Scientific Data* 9:1, 14, 2022, p. 622. DOI: 10.1038/s41597-022-01710-x. URL: https://www.nature.com/articles/s41597-022-01710-x.
61. *SQLAlchemy: The Python SQL Toolkit and Object Relational Mapper*. URL: https://www.sqlalchemy.org.
62. MariaDB Foundation. *MariaDB: The Open Source Relational Database*. URL: https://mariadb.org/.
63. T. A. Snijders and K. Nowicki. "Estimation and Prediction for Stochastic Blockmodels for Graphs with Latent Block Structure". *Journal of Classification* 14:1, 1, 1997, pp. 75–100. DOI: 10.1007/s003579900004. URL: http://link.springer.com/10.1007/s003579900004.

64. B. Karrer and M. E. J. Newman. “Stochastic Blockmodels and Community Structure in Networks”. *Physical Review E* 83:1, 21, 2011, p. 016107. DOI: 10.1103/PhysRevE.83.016107. URL: https://link.aps.org/doi/10.1103/PhysRevE.83.016107.
65. P. K. Gopalan, S. Gerrish, M. Freedman, D. Blei, and D. Mimno. “Scalable Inference of Overlapping Communities”. In: *Advances in Neural Information Processing Systems*. Vol. 25. 2012.
66. E. M. Airoldi, D. Blei, S. Fienberg, and E. Xing. “Mixed Membership Stochastic Blockmodels”. *Advances in neural information processing systems* 21, 2008.
67. K. S. Xu and A. O. Hero. “Dynamic Stochastic Blockmodels: Statistical Models for Time-Evolving Networks”. In: Social Computing, Behavioral-Cultural Modeling and Prediction: 6th International Conference, SBP 2013, Washington, DC, USA, April 2-5, 2013. Proceedings 6. Springer, 2013, pp. 201–210.
68. C. DuBois, C. Butts, and P. Smyth. “Stochastic Blockmodeling of Relational Event Dynamics”. In: Artificial Intelligence and Statistics. PMLR, 2013, pp. 238–246.
69. X. Tang and C. C. Yang. “Detecting Social Media Hidden Communities Using Dynamic Stochastic Blockmodel with Temporal Dirichlet Process”. *ACM Transactions on Intelligent Systems and Technology* 5:2, 30, 2014, 36:1–36:21. DOI: 10.1145/2517085. URL: https://dl.acm.org/doi/10.1145/2517085.
70. J. D. Wilson, N. T. Stevens, and W. H. Woodall. “Modeling and Detecting Change in Temporal Networks via the Degree Corrected Stochastic Block Model”. *Quality and Reliability Engineering International* 35:5, 2019, pp. 1363–1378. DOI: 10.1002/qre.2520. URL: https://onlinelibrary.wiley.com/doi/10.1002/qre.2520.
71. D. J. MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge university press, 2003.
72. L. Wasserman. “Bayesian Model Selection and Model Averaging”. *Journal of Mathematical Psychology* 44:1, 2000, pp. 92–107. DOI: 10.1006/jmps.1999.1278. URL: https://linkinghub.elsevier.com/retrieve/pii/S0022249699912786.
73. M. Mariadassou, S. Robin, and C. Vacher. “Uncovering Latent Structure in Valued Graphs: A Variational Approach”, 2010.
74. P. Latouche, E. Birmele, and C. Ambroise. “Variational Bayesian Inference and Complexity Control for Stochastic Block Models”. *Statistical Modelling* 12:1, 2012, pp. 93–115.
75. M. Rosvall and C. T. Bergstrom. “An Information-Theoretic Framework for Resolving Community Structure in Complex Networks”. *Proceedings of the National Academy of Sciences* 104:18, 2007, pp. 7327–7331. DOI: 10.1073/pnas.0611034104. URL: https://www.pnas.org/doi/full/10.1073/pnas.0611034104.
76. C. M. Bishop and N. M. Nasrabadi. *Pattern Recognition and Machine Learning*. Vol. 4. 4. Springer, 2006.
77. B. Efron and R. Tibshirani. *An Introduction to the Bootstrap*. 0^{th} ed. Chapman and Hall/CRC, 15, 1994. DOI: 10.1201/9780429246593. URL: https://www.taylorfrancis.com/books/9781000064988.
78. S. Chib. “Markov Chain Monte Carlo Methods: Computation and Inference”. In: *Handbook of Econometrics*. Vol. 5. Elsevier, 2001, pp. 3569–3649. DOI: 10.1016/S1573-4412(01)05010-3. URL: https://linkinghub.elsevier.com/retrieve/pii/S1573441201050103.

79. N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. “Equation of State Calculations by Fast Computing Machines”. *The journal of chemical physics* 21:6, 1953, pp. 1087–1092. DOI: 10.1063/1.1699114. URL: https://doi.org/10.1063/1.1699114.
80. S. Chib and E. Greenberg. “Understanding the Metropolis-Hastings Algorithm”. *The american statistician* 49:4, 1995, pp. 327–335.
81. T. P. Peixoto. *The Graph-Tool Python Library*. figshare, 2017. DOI: 10.6084/M9.FIGSHARE.1164194.V14. URL: https://figshare.com/articles/dataset/graph_tool/1164194/14.
82. P. J. Van Laarhoven, E. H. Aarts, P. J. van Laarhoven, and E. H. Aarts. *Simulated Annealing*. Springer, 1987. DOI: 10.1007/978-94-015-7744-1_2.
83. W. K. Hastings. “Monte Carlo Sampling Methods Using Markov Chains and Their Applications”, 1970. DOI: 10.1093/biomet/57.1.97.
84. A. Condon and R. M. Karp. “Algorithms for Graph Partitioning on the Planted Partition Model”. *Random Structures & Algorithms* 18:2, 2001, pp. 116–140. DOI: 10.1002/1098-2418(200103)18:2<116::AID-RSA1001>3.0.CO;2-2.
85. S. P. Brooks and A. Gelman. “General Methods for Monitoring Convergence of Iterative Simulations”. *Journal of computational and graphical statistics* 7:4, 1998, pp. 434–455. DOI: 10.1080/10618600.1998.10474787. URL: https://doi.org/10.1080/10618600.1998.10474787.
86. A. Vehtari, A. Gelman, D. Simpson, B. Carpenter, and P.-C. Bürkner. “Rank-Normalization, Folding, and Localization: An Improved R-hat for Assessing Convergence of MCMC (with Discussion)”. *Bayesian analysis* 16:2, 2021, pp. 667–718. DOI: 10.1214/20-BA1221.
87. D. S. Choi, P. J. Wolfe, and E. M. Airoldi. “Stochastic Blockmodels with a Growing Number of Classes”. *Biometrika* 99:2, 2012, pp. 273–284. DOI: 10.1093/biomet/asr053. URL: https://doi.org/10.1093/biomet/asr053.
88. C. E. Shannon. “A Mathematical Theory of Communication”. *The Bell system technical journal* 27:3, 1948, pp. 379–423.
89. R. K. Peet. “Relative Diversity Indices”. *Ecology* 56:2, 1975, pp. 496–498. DOI: 10.2307/1934984. URL: https://esajournals.onlinelibrary.wiley.com/doi/10.2307/1934984.
90. Sebastian B. Mohr. *Supplementary Repository for Master Thesis*. URL: https://github.com/semohr/master_thesis_src.
91. J. Dehning, S. B. Mohr, S. Contreras, P. Dönges, E. N. Iftekhar, O. Schulz, P. Bechtle, and V. Priesemann. “Impact of the Euro 2020 Championship on the Spread of COVID-19”. *Nature Communications* 14:1, 18, 2023, p. 122. DOI: 10.1038/s41467-022-35512-x. URL: https://www.nature.com/articles/s41467-022-35512-x.
92. S. Contreras, J. Dehning, M. Loidolt, J. Zierenberg, F. P. Spitzner, J. H. Urrea-Quintero, S. B. Mohr, M. Wilczek, M. Wibral, and V. Priesemann. “The Challenges of Containing SARS-CoV-2 via Test-Trace-and-Isolate”. *Nature Communications* 12:1, 15, 2021, p. 378. DOI: 10.1038/s41467-020-20699-8. URL: https://www.nature.com/articles/s41467-020-20699-8.
93. E. N. Iftekhar, V. Priesemann, R. Balling, S. Bauer, P. Beutels, A. Calero Valdez, S. Cuschieri, T. Czypionka, U. Dumpis, E. Glaab, E. Grill, C. Hanson, P. Hotulainen, P. Klimek, M. Kretzschmar, T. Krüger, J. Krutzinna, N. Low, H. Machado, C. Martins, M. McKee, S. B. Mohr, A. Nassehi, M. Perc, E. Petelos, M. Pickersgill, B. Prainsack,

J. Rocklöv, E. Schernhammer, A. Staines, E. Szczurek, S. Tsiodras, S. Van Gucht, and P. Willeit. "A Look into the Future of the COVID-19 Pandemic in Europe: An Expert Consultation". *The Lancet Regional Health - Europe* 8, 2021, p. 100185. DOI: 10.1016/j.lanepe.2021.100185. URL: https://linkinghub.elsevier.com/retrieve/pii/S2666776221001629.

94. S. Bauer, S. Contreras, J. Dehning, M. Linden, E. Iftekhar, S. B. Mohr, A. Olivera-Nappa, and V. Priesemann. "Relaxing Restrictions at the Pace of Vaccination Increases Freedom and Guards against Further COVID-19 Waves". *PLOS Computational Biology* 17:9, 2, 2021. Ed. by C. J. Struchiner, e1009288. DOI: 10.1371/journal.pcbi.1009288. URL: https://dx.plos.org/10.1371/journal.pcbi.1009288.

95. S. Contreras, J. Dehning, S. B. Mohr, S. Bauer, F. P. Spitzner, and V. Priesemann. "Low Case Numbers Enable Long-Term Stable Pandemic Control without Lockdowns". *Science Advances* 7:41, 8, 2021, eabg2243. DOI: 10.1126/sciadv.abg2243. URL: https://www.science.org/doi/10.1126/sciadv.abg2243.

96. M. Linden, S. B. Mohr, J. Dehning, J. Mohring, M. Meyer-Hermann, I. Pigeot, A. Schöbel, and V. Priesemann. "Case Numbers Beyond Contact Tracing Capacity Are Endangering the Containment of COVID-19". *Deutsches Ärzteblatt international*, 13, 2020. DOI: 10.3238/arztebl.2020.0790. URL: https://www.aerzteblatt.de/10.3238/arztebl.2020.0790.

97. P. Dönges, J. Wagner, S. Contreras, E. N. Iftekhar, S. Bauer, S. B. Mohr, J. Dehning, A. Calero Valdez, M. Kretzschmar, M. Mäs, K. Nagel, and V. Priesemann. "Interplay Between Risk Perception, Behavior, and COVID-19 Spread". *Frontiers in Physics* 10, 15, 2022, p. 842180. DOI: 10.3389/fphy.2022.842180. URL: https://www.frontiersin.org/articles/10.3389/fphy.2022.842180/full.

98. K. Y. Oróstica, S. Contreras, S. B. Mohr, J. Dehning, S. Bauer, D. Medina-Ortiz, E. N. Iftekhar, K. Mujica, P. C. Covarrubias, S. Ulloa, A. E. Castillo, R. A. Verdugo, J. Fernández, Á. Olivera-Nappa, and V. Priesemann. *Mutational Signatures and Transmissibility of SARS-CoV-2 Gamma and Lambda Variants*. 23, 2021. arXiv: 2108.10018. URL: http://arxiv.org/abs/2108.10018. preprint.

99. J. Dehning, F. P. Spitzner, M. C. Linden, S. B. Mohr, J. P. Neto, J. Zierenberg, M. Wibral, M. Wilczek, and V. Priesemann. *Model-Based and Model-Free Characterization of Epidemic Outbreaks*. preprint. Epidemiology, 18, 2020. DOI: 10.1101/2020.09.16.20187484. URL: http://medrxiv.org/lookup/doi/10.1101/2020.09.16.20187484.

100. K. Sherratt et al. "Predictive Performance of Multi-Model Ensemble Forecasts of COVID-19 across European Nations". *eLife* 12, 21, 2023, e81916. DOI: 10.7554/eLife.81916. URL: https://elifesciences.org/articles/81916.

101. A. Gelman, B. Goodrich, J. Gabry, and A. Vehtari. "R-Squared for Bayesian Regression Models". *The American Statistician*, 2019. DOI: 10.1080/00031305.2018.1549100. URL: https://doi.org/10.1080/00031305.2018.1549100.

102. *Usage Statistics and Market Share of Content Languages for Websites, October 2023*. URL: https://w3techs.com/technologies/overview/content_language.

Appendices

A Acknowledgements

I express my sincere gratitude to Andreas Schneider, whose unwavering guidance and support were instrumental throughout the entire process of composing this thesis. Despite it being a side project during his PhD, he consistently made himself available for insightful discussions and provided invaluable assistance. I am truly grateful for the many engaging conversations and his consistently encouraging and positive approach.

I extend my appreciation to my supervisor, Dr. Viola Priesemann, for her support and for providing me with the opportunity to start this project. Her encouragement has been pivotal to the development and completion of this work.

Special thanks are also due to all the members of the Telegram subgroup within the Priesemann group, whose steadfast support and simultaneous efforts in handling the Telegram data greatly contributed to the project's success.

I would like to express my heartfelt gratitude to Janine Schönefeld for her unwavering support throughout this journey. Her encouragement, understanding, and positivity have been a constant source of inspiration and motivation.

Lastly, I would also like to thank the dedicated members of the Priesemann group for their continuous input and feedback, which further enriched the overall quality of this project.

D Data availability

As of now, there does not exist a public version of the full dataset but it is already used internally for multiple projects. However, the graph data of the collected Telegram channels is available on *GRO.data*[49]. This data joined with the supplementary GitHub repository [90] should allow to reproduce most of the results presented in this thesis.

The software developed for snowball crawling is also available on github[1]. We are currently working on publishing it and depending on the time of reading this thesis it might already be available publicly.

[1] https://github.com/Priesemann-Group/telegram_crawler

P Publications

Even though this thesis stands for itself, during my master studies I was able to publish and contribute to a number of papers under the supervision of Dr. Viola Priesemann. The following list contains all publications in which I was involved during my master studies. Other than the use of MCMC these publications are mostly unrelated to the topic of this thesis.

- J. Dehning, S. B. Mohr, S. Contreras, P. Dönges, E. N. Iftekhar, O. Schulz, P. Bechtle, and V. Priesemann. "Impact of the Euro 2020 Championship on the Spread of COVID-19". *Nature Communications* 14:1, 18, 2023, p. 122. doi: 10.1038/s41467-022-35512-x. url: https://www.nature.com/articles/s41467-022-35512-x

- S. Contreras, J. Dehning, M. Loidolt, J. Zierenberg, F. P. Spitzner, J. H. Urrea-Quintero, S. B. Mohr, M. Wilczek, M. Wibral, and V. Priesemann. "The Challenges of Containing SARS-CoV-2 via Test-Trace-and-Isolate". *Nature Communications* 12:1, 15, 2021, p. 378. doi: 10.1038/s41467-020-20699-8. url: https://www.nature.com/articles/s41467-020-20699-8

- E. N. Iftekhar, V. Priesemann, R. Balling, S. Bauer, P. Beutels, A. Calero Valdez, S. Cuschieri, T. Czypionka, U. Dumpis, E. Glaab, E. Grill, C. Hanson, P. Hotulainen, P. Klimek, M. Kretzschmar, T. Krüger, J. Krutzinna, N. Low, H. Machado, C. Martins, M. McKee, S. B. Mohr, A. Nassehi, M. Perc, E. Petelos, M. Pickersgill, B. Prainsack, J. Rocklöv, E. Schernhammer, A. Staines, E. Szczurek, S. Tsiodras, S. Van Gucht, and P. Willeit. "A Look into the Future of the COVID-19 Pandemic in Europe: An Expert Consultation". *The Lancet Regional Health - Europe* 8, 2021, p. 100185. doi: 10.1016/j.lanepe.2021.100185. url: https://linkinghub.elsevier.com/retrieve/pii/S2666776221001629

- S. Bauer, S. Contreras, J. Dehning, M. Linden, E. Iftekhar, S. B. Mohr, A. Olivera-Nappa, and V. Priesemann. "Relaxing Restrictions at the Pace of Vaccination Increases Freedom and Guards against Further COVID-19 Waves". *PLOS Computational Biology* 17:9, 2, 2021. Ed. by C. J. Struchiner, e1009288. doi: 10.1371/journal.pcbi.1009288. url: https://dx.plos.org/10.1371/journal.pcbi.1009288

- S. Contreras, J. Dehning, S. B. Mohr, S. Bauer, F. P. Spitzner, and V. Priesemann. "Low Case Numbers Enable Long-Term Stable Pandemic Control without Lockdowns". *Science Advances* 7:41, 8, 2021, eabg2243. doi: 10.1126/sciadv.abg2243. url: https://www.science.org/doi/10.1126/sciadv.abg2243

- M. Linden, S. B. Mohr, J. Dehning, J. Mohring, M. Meyer-Hermann, I. Pigeot, A. Schöbel, and V. Priesemann. "Case Numbers Beyond Contact Tracing Capacity Are Endangering the

Containment of COVID-19". *Deutsches Ärzteblatt international*, 13, 2020. DOI: 10.3238/arztebl.2020.0790. URL: https://www.aerzteblatt.de/10.3238/arztebl.2020.0790

- P. Dönges, J. Wagner, S. Contreras, E. N. Iftekhar, S. Bauer, S. B. Mohr, J. Dehning, A. Calero Valdez, M. Kretzschmar, M. Mäs, K. Nagel, and V. Priesemann. "Interplay Between Risk Perception, Behavior, and COVID-19 Spread". *Frontiers in Physics* 10, 15, 2022, p. 842180. DOI: 10.3389/fphy.2022.842180. URL: https://www.frontiersin.org/articles/10.3389/fphy.2022.842180/full

- K. Y. Oróstica, S. Contreras, S. B. Mohr, J. Dehning, S. Bauer, D. Medina-Ortiz, E. N. Iftekhar, K. Mujica, P. C. Covarrubias, S. Ulloa, A. E. Castillo, R. A. Verdugo, J. Fernández, Á. Olivera-Nappa, and V. Priesemann. *Mutational Signatures and Transmissibility of SARS-CoV-2 Gamma and Lambda Variants*. 23, 2021. arXiv: 2108.10018. URL: http://arxiv.org/abs/2108.10018. preprint

- J. Dehning, F. P. Spitzner, M. C. Linden, S. B. Mohr, J. P. Neto, J. Zierenberg, M. Wibral, M. Wilczek, and V. Priesemann. *Model-Based and Model-Free Characterization of Epidemic Outbreaks*. preprint. Epidemiology, 18, 2020. DOI: 10.1101/2020.09.16.20187484. URL: http://medrxiv.org/lookup/doi/10.1101/2020.09.16.20187484

- K. Sherratt et al. "Predictive Performance of Multi-Model Ensemble Forecasts of COVID-19 across European Nations". *eLife* 12, 21, 2023, e81916. DOI: 10.7554/eLife.81916. URL: https://elifesciences.org/articles/81916

S Supplementary material

In this supplementary material, we present additional figures and analyses that did not fit the natural flow of the main manuscript due to their repetitive nature or their size. For instance we present equilibration and sampling statistics as shown in the main manuscript, but for the other models.

♦

Chapter contents

S1 Collected dataset

Type	Count	Fraction (%)
URL	641,841,647	31.24372
BOLD	476,324,393	23.18664
MENTION	343,394,833	16.71586
TEXT_LINK	263,795,376	12.84110
HASHTAG	166,869,199	8.12290
ITALIC	80,063,614	3.89736
CUSTOM_EMOJI	26,110,749	1.27103
CODE	14,750,606	0.71803
EMAIL	12,780,496	0.62213
UNDERLINE	11,760,118	0.57246
PHONE_NUMBER	5,195,145	0.25289
BOT_COMMAND	2,872,671	0.13984
PRE	2,214,984	0.10782
TEXT_MENTION	1,999,717	0.09734
BANK_CARD	1,775,444	0.08643
STRIKETHROUGH	1,218,981	0.05934
CASHTAG	985,301	0.04796
SPOILER	352,531	0.01716
Total	2,054,305,805	100.00000

Table S1.1: **The number of message entities per type** and their respective fraction to the total number of message entities.

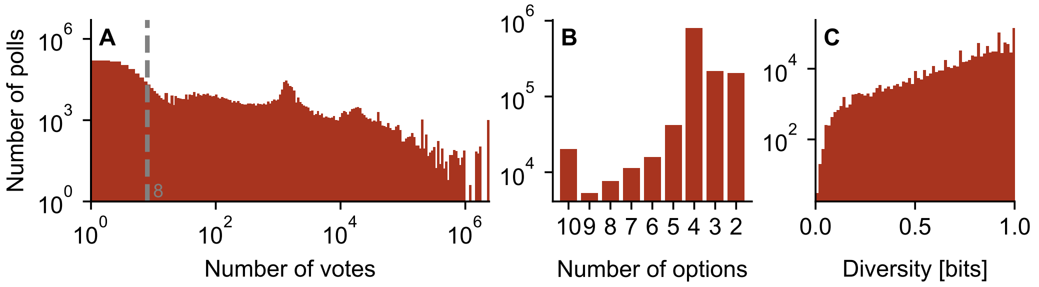


Figure S1.1: **Characteristics of the recorded polls**. The distribution of total votes per poll does not follow a visible distribution, here the median value is indicated by a gray dashed line (**A**). The majority of polls have 4 options, it is noteworthy that polls with 10 options are selected more frequently than those with more than 5 options (**B**). The distribution of diversity in poll answers shows a trend towards greater diversity (**C**).

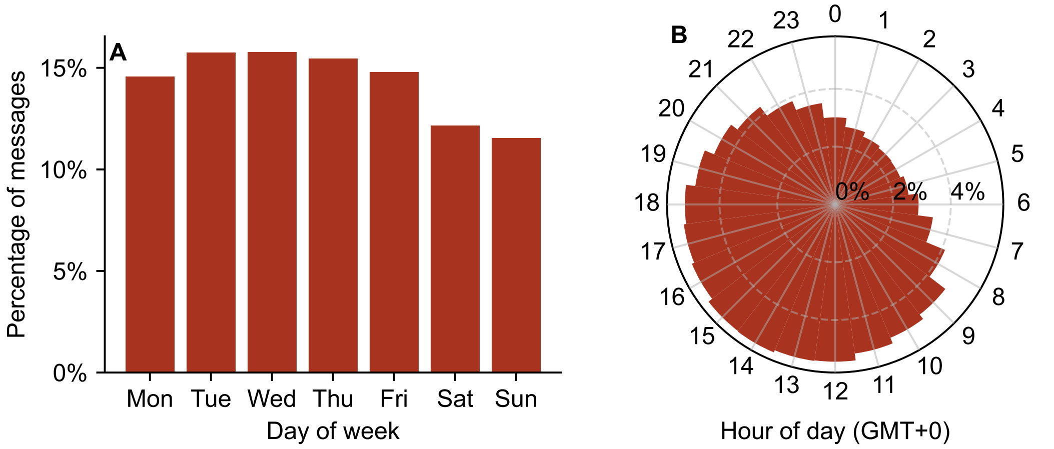


Figure S1.2: **Number of messages recorded by weekday and hour.** The number of messages recorded per day slightly decreases over the weekends **A**. During the night and early hours of the day less messages are posted as during the day **B**.

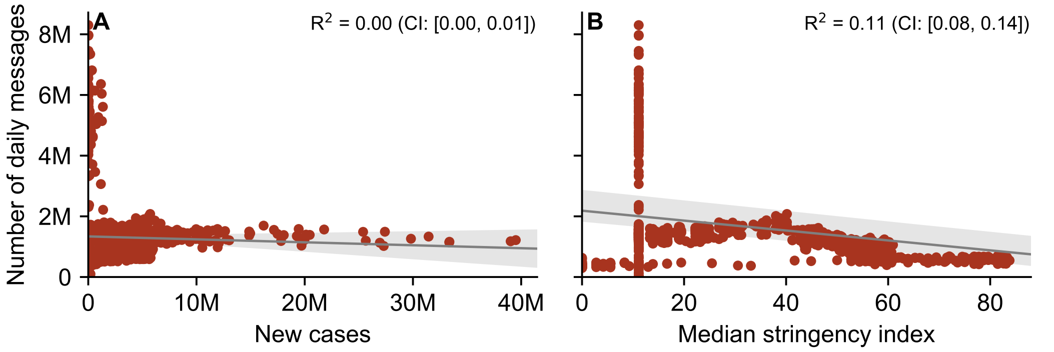


Figure S1.3: **Correlation of number of daily messages with new cases and Nonpharmaceutical Interventions (NPIs).** The number of messages recorded per day does not show significant correlation with the number of new cases or stringency index i.e. ths NPIs. Note that the R-squared values [101] are very low, indicating that the number of messages does not explain much of the variance in the number of new cases nor NPIs. The linear bayesian regression is done identical to the one in Dehning, Mohr et. al. 2023 [91]. The shaded area represents the 95% Confidence interval (CI) of the regression.

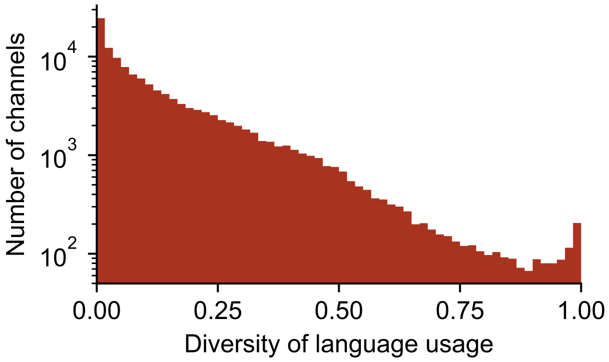


Figure S1.4: **Diversity in language usage.** The diversity in language usage per channel is skewed towards low values. This indicates that the majority of channels use a single or a few languages.

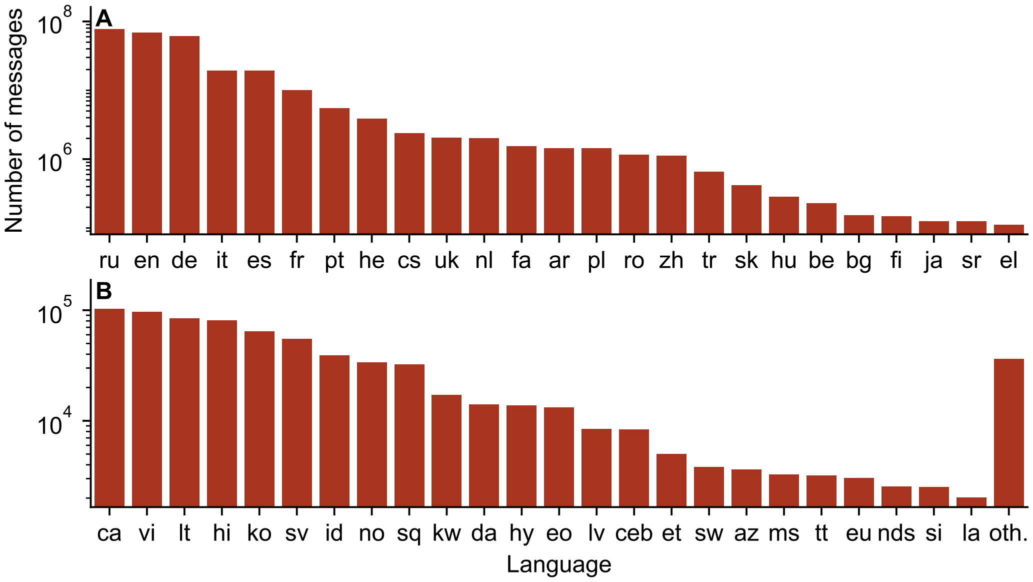


Figure S1.5: **Number of messages per language.** The majority of messages are in Russian, English and German. We show the 25 most frequent languages (**A**) and the 25-50 most frequent languages (**B**). The remaining languages are grouped into *other*. We only included messages where the language was detected with a confidence of at least 90%. Detection was done with FastText [38].

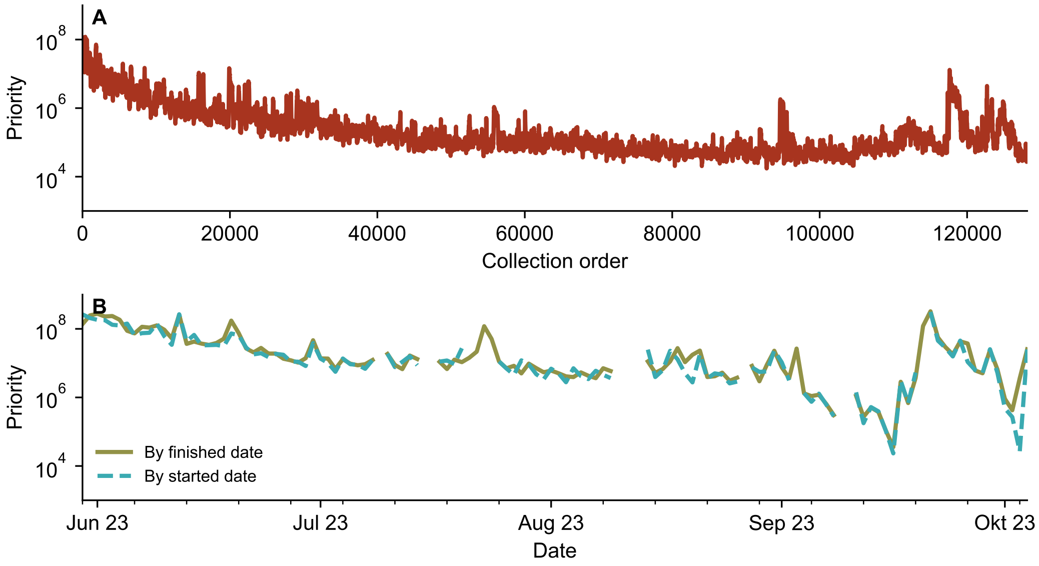


Figure S1.6: **The priority of each channel significantly decreased over the runtime of the crawler.** During the collection process, the channels chronologically showed a major decrease in priority (**A**). This indicates that the crawler was able to transition from the guided to a more random walk phase. Similarly this is visible with the median daily priority of finished or started channels (**B**). Finish dates in red and starting dates in blue. The missing days are times when the crawler was not running due to maintenance. In panel **A** priority values are smoothed with a rolling average of 50 channels.

S1.1 Early subset

During the collection of the dataset we already created some of the dataset figures as seen in the main manuscript. These show different distinctive patterns, therefore we include the old versions here. Overall it seems that the full dataset might not be as restricted to the chosen keywords as the early subset was.

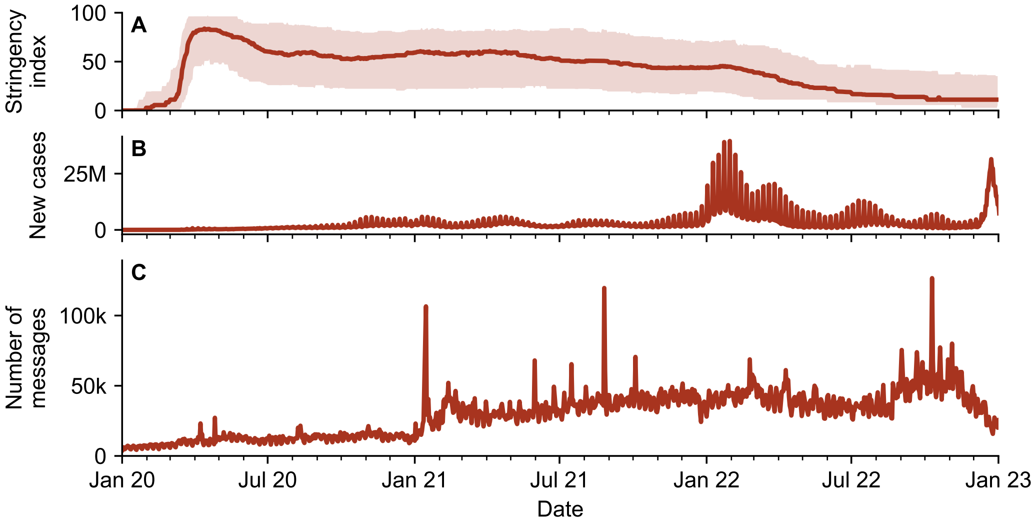


Figure S1.7: **Old data: Number of daily messages and the relation to the *COVID-19* pandemic.** We find no significant correlation between the number of daily messages (**C**) and the number of new cases (**A**) nor the stringency index (**B**). The stringency index is a measure of the strictness of NPIs and is estimated by the *Oxford COVID-19 Government Response Tracker* [40]. The new cases are aggregated by *Our World in Data* [43].

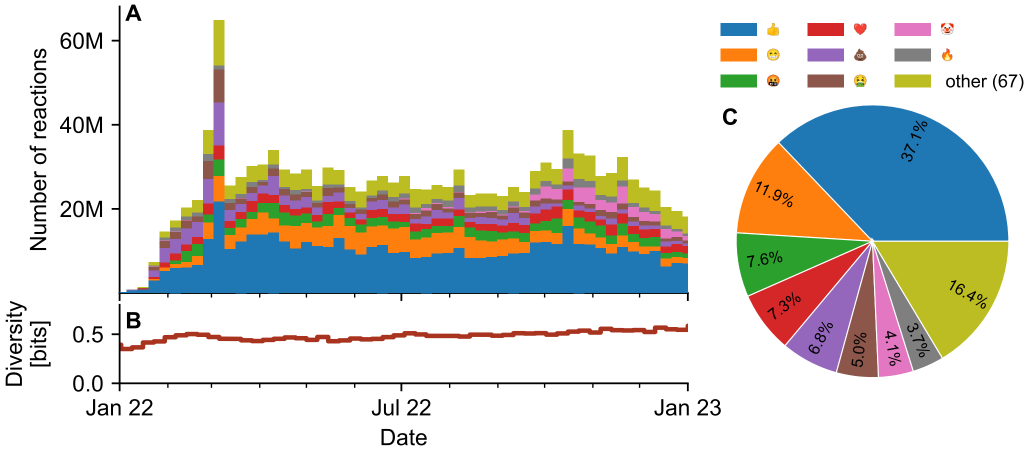


Figure S1.8: **Old data: Number of different reactions in the collected dataset.**

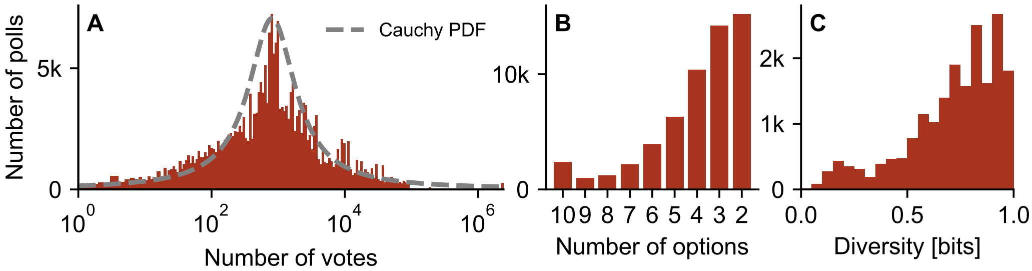


Figure S1.9: **Old data: Characteristics of the recorded polls**. The distribution of total votes per poll follows a logarithmic cauchy distribution (**A**). The distribution of diversity in poll answers shows a small bimodal pattern near zero (**C**).

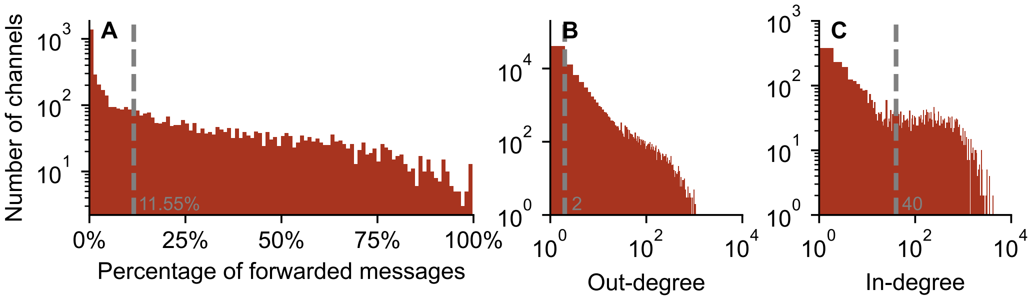


Figure S1.10: **Old data: Summary of forwarded messages and the resulting network structure**. The distribution of the number of forwards relative to the total number of messages in the chat. The majority of messages are not forwards (**A**). The distribution of the number of outgoing forwards per channel (out-degree) (**B**) and the distribution of incoming forwards (in-degree) does only to a limited (**C**) roughly follows a power-law distribution with an unknown modulation. Median values are indicated by gray dashed lines.

S2 Structural analysis

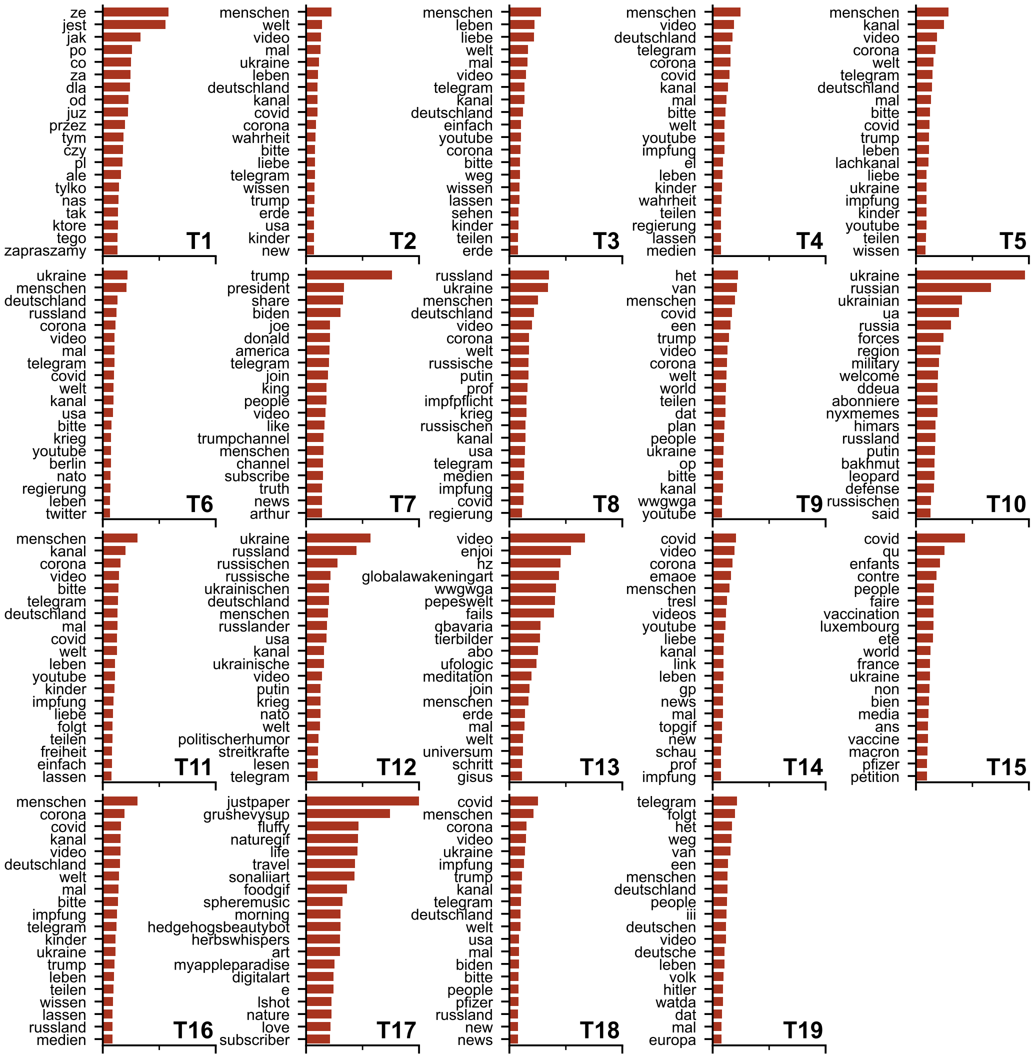


Figure S2.11: **Topics of the modules in the German branch within layer 5.** The topics encompass discussions on the Ukraine war (**T6, T12, T10**), American geopolitics (**T7**), and a predominant focus on COVID-related subjects (**T15, T16, T18**). Additionally, some topics do not have a clearly identifiable focus(**T2, T13**) or include non-German content (**T1,T17**). The modules here are of the German branch as seen in Fig. 2.11 (green).

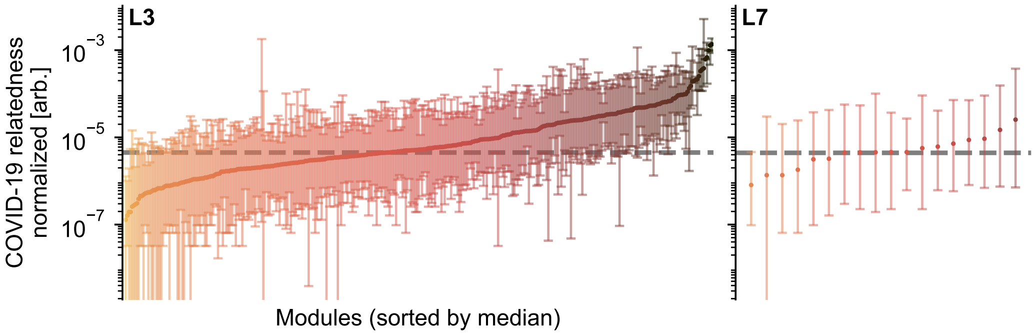


Figure S2.12: **The separation of COVID-19 related channels from non COVID-19 related channels is lost in modules of the higher layers.** The modules of layer 3 (**L3**) encode the COVID-19 relatedness of the channels. This is less the case for the modules of the higher layers where this separation is lost or washed out. For instance note that the median values of the modules in layer 7 (**L7**) are mostly within the 68% CI of the median values of all other modules in this layer. Therefore the modules of this layer do not encode the COVID-19 relatedness of the channels. Whiskers indicate the 68% CI of the median.

S3 Keyword lists

We translated the main keyword list from English (see Listing 4.1) into 44 different languages, i.e. Czech S.1, French S.2, Lithuanian S.3, German S.4, Swahili S.5, Norwegian S.6, Russian S.7, Dutch S.8, Estonian S.9, Italian S.10, Belarusian S.11, Polish S.12, Hindi S.13, Danish S.14, Thai S.15, Korean S.16, Ukrainian S.17, Zulu S.18, Finnish S.19, Turkish S.20, Kurdish (kurmanji) S.21, Kannada S.22, Vietnamese S.23, Hausa S.24, Romanian S.25, Portuguese S.26, Croatian S.27, Spanish S.28, Catalan S.29, Chinese (simplified) S.30, Greek S.31, Persian S.32, Slovenian S.33, Japanese S.34, Arabic S.35, Hebrew S.36, Slovak S.37, Serbian S.38, Swedish S.39, Latvian S.40, Urdu S.41, Hungarian S.42, Indonesian S.43, and Bulgarian S.44. These languages are picked because the are the predominant languages used on the internet [102]. Additionally, we included some languages spoken in Africa as the continent is often overlooked in research. We removed all duplicate values from the full list which occurred because of the translation process.

```
keywords = ['covid', 'korona', 'virus', 'pandemický',
↪  'izolování', 'zdraví', 'maska', 'distancování', 'nákaza',
↪  'příznak', 'karanténa', 'chřipka', 'vakcína', 'očkování',
↪  'ventilátor', 'izolace', 'imunita', 'nemocnice', 'icu',
↪  'jednotka intenzivní péče', 'léčba', 'virolog', 'klinika',
↪  'homeopatie', 'nemocný', 'farmacie', 'polymerázová řetězová
↪  reakce', 'pcr', 'černý kašel', 'nouzový', 'injekce',
↪  'lékaři', 'buněčný', 'práce na dálku', 'přední linie',
↪  'covid-19', 'sars', 'kanalizace', 'patogen', 'propaganda',
↪  'choroba', 'epidemický', 'průjem', 'adjuvans', 'respirační',
↪  'hygiena', 'protein', 'lék', 'nové případy', 'pozitivní',
↪  'vědec', 'nakažlivý', 'mandát', 'varianta', 'infikovat',
↪  'úmrtí', 'kašel', 'opatření', 'virový', 'mrna', 'zabránit',
↪  'zdravotní péče', 'smlouva', 'vypnout', 'neštovice',
↪  'posilovač', 'protilátka', 'dávka', 'důkaz', 'dezinformace',
↪  'pozorováno', 'povinné', 'alergický', 'alergie', 'imunní',
↪  'nedostatek', 'syndrom', 'čínština', 'test', 'omezení',
↪  'šíření', 'vax', 'experimentální']
```

Listing S.1: List of Czech keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['couronne', 'pandémie', 'confinement', 'santé',
↪  'masque', 'distanciation', 'épidémie', 'symptôme',
↪  'quarantaine', 'grippe', 'vaccin', 'vaccination',
↪  'ventilateur', 'isolement', 'immunité', 'hôpital', 'unité de
↪  soins intensifs', 'traitement', 'virologue', 'clinique',
↪  'homéopathie', 'malade', 'pharmaceutique', 'réaction en
↪  chaîne par polymérase', 'coqueluche', 'urgence', 'injection',
↪  'médecins', 'cellulaire', 'travail à distance', 'première
↪  ligne', 'covid-19 feminine', 'assainissement', 'agent
↪  pathogène', 'la propagande', 'maladie', 'diarrhée',
↪  'adjuvants', 'respiratoire', 'hygiène', 'protéine',
↪  'médecine', 'nouveaux cas', 'positif', 'scientifique',
↪  'contagieux', 'mandat', 'une variante', 'infecter', 'décès',
↪  'je vais', 'toux', 'mesure', 'viral', 'arnm', 'prévenir',
↪  'soins de santé', 'contracter', 'fermer', 'variole',
↪  'amplificateur', 'anticorps', 'dose', 'preuve',
↪  'désinformation', 'observé', 'obligatoire', 'allergique',
↪  'allergie', 'immunitaire', 'pénurie', 'syndrome',
↪  'médicament', 'chinois', 'restriction', 'propagé',
↪  'expérimental']
```

Listing S.2: List of French keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['virusas', 'pandemija', 'izoliacija', 'sveikata',
↪  'kaukė', 'atitolimas', 'protrūkis', 'simptomas',
↪  'karantinas', 'gripas', 'vakcina', 'vakcinacija',
↪  'ventiliatorius', 'isolation', 'imunitetas', 'ligoninė',
↪  'intensyviosios terapijos skyriuje', 'gydymas',
↪  'virusologas', 'homeopatija', 'serga', 'pharma', 'polimerazės
↪  grandininė reakcija', 'pkr', 'kokliušo', 'skubus atvėjis',
↪  'injekcija', 'gydytojai', 'ląstelinis', 'nuotolinis darbas',
↪  'fronto linija', 'sanitarija', 'patogenas', 'liga',
↪  'epidemija', 'viduriavimas', 'adjuvantai', 'kvėpavimo',
↪  'higiena', 'baltymas', 'vaistas', 'naujų atvejų',
↪  'teigiamas', 'mokslininkas', 'užkrečiama', 'mandatas',
↪  'variantas', 'užkrėsti', 'mirtys', 'nesveikas', 'kosulys',
↪  'matuoti', 'virusinis', 'ponia', 'užkirsti kelią', 'sveikatos
↪  apsauga', 'sutartis', 'išjungti', 'raupai', 'stiprintuvas',
↪  'antikūnas', 'dozę', 'įrodymai', 'dezinformacija',
↪  'pastebėjus', 'privalomas', 'alergiškas', 'alergija',
↪  'imuninis', 'trūkumas', 'sindromas', 'narkotikų', 'kinų',
↪  'bandymas', 'apribojimas', 'plisti', 'eksperimentinis']
```

Listing S.3: List of Lithuanian keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['corona', 'pandemie', 'sperrung', 'gesundheit',
 ↪  'maske', 'distanzierung', 'ausbruch', 'symptom',
 ↪  'quarantäne', 'impfstoff', 'impfung', 'ventilator',
 ↪  'isolierung', 'immunität', 'krankenhaus', 'intensivstation',
 ↪  'behandlung', 'virologe', 'klinik', 'homöopathie', 'krank',
 ↪  'polymerase kettenreaktion', 'keuchhusten', 'notfall',
 ↪  'injektion', 'ärzte', 'zellular', 'heimarbeit', 'frontlinie',
 ↪  'hygiene', 'erreger', 'krankheit', 'epidemie', 'durchfall',
 ↪  'adjuvantien', 'atemwege', 'eiweiß', 'medizin', 'neue fälle',
 ↪  'positiv', 'wissenschaftler', 'ansteckend', 'variante',
 ↪  'infizieren', 'todesfälle', 'husten', 'messen', 'verhindern',
 ↪  'gesundheitspflege', 'vertrag', 'abschalten', 'pocken',
 ↪  'booster', 'antikörper', 'dosis', 'beweis',
 ↪  'fehlinformationen', 'beobachtet', 'obligatorisch',
 ↪  'allergisch', 'immun', 'mangel', 'arzneimittel',
 ↪  'chinesisch', 'prüfen', 'beschränkung', 'verbreiten',
 ↪  'experimental-']
```

Listing S.4: List of German keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['virusi', 'janga kubwa', 'kusitishwa katikhuli za
 ↪  kawaida', 'afya', 'mask', 'umbali', 'mkurupuko', 'dalili',
 ↪  'karantini', 'mafua', 'chanjo', 'kipumuaji', 'kujitenga',
 ↪  'kinga', 'hospitali', 'kitengo cha wagonjwa mahututi',
 ↪  'matibabu', 'daktari wa virusi', 'zahanati', 'homeopathy',
 ↪  'mgonjwa', 'dawa', 'mmenyuko wa mnyororo wa polymerase',
 ↪  'pertussis', 'dharura', 'sindano', 'madaktari', 'simu za
 ↪  mkononi', 'kazi ya mbali', 'mstari wa mbele', 'usafi wa
 ↪  mazingira', 'pathojeni', 'ugonjwa', 'janga', 'kuhara',
 ↪  'wasaidizi', 'kupumua', 'usafi', 'protini', 'kesi mpya',
 ↪  'chanya', 'mwanasayansi', 'ya kuambukiza', 'mamlaka',
 ↪  'lahaja', 'kuambukiza', 'vifo', 'kikohozi', 'kipimo', 'bw',
 ↪  'kuzuia', 'huduma ya afya', 'mkataba', 'kuzimisha', 'ndui',
 ↪  'nyongeza', 'kingamwili', 'ushahidi', 'habari potofu',
 ↪  'kuzingatiwa', 'lazima', 'mzio', 'uhaba', 'wachina',
 ↪  'mtihani', 'kizuizi', 'kuenea', 'vaksi', 'majaribio']
```

Listing S.5: List of Swahili keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['pandemi', 'nedstengning', 'helse', 'distansere',
↪ 'utbrudd', 'karantene', 'influensa', 'vaksine',
↪ 'vaksinasjon', 'isolering', 'immunitet', 'sykehus',
↪ 'intensivavdeling', 'behandling', 'klinikk', 'homeopati',
↪ 'syk', 'polymerase kjedereaksjon', 'kikhoste',
↪ 'nødsituasjon', 'injeksjon', 'leger', 'mobilnettet',
↪ 'fjernarbeid', 'frontlinjen', 'sykdom', 'epidemi', 'diaré',
↪ 'hjelpestoffer', 'luftveiene', 'medisin', 'nye saker',
↪ 'positivt', 'forsker', 'smittsom', 'variant', 'infisere',
↪ 'dødsfall', 'jeg vil', 'hoste', 'måle', 'forhindre',
↪ 'helsevesen', 'kontrakt', 'skru av', 'kopper', 'antistoff',
↪ 'bevis', 'feilinformasjon', 'observert', 'påbudt, bindende',
↪ 'allergisk', 'allergi', 'legemiddel', 'kinesisk',
↪ 'begrensning', 'spre', 'eksperimentell']
```

Listing S.6: List of Norwegian keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['корона', 'вирус', 'пандемия', 'карантин',
↪ 'здоровье', 'маска', 'дистанцирование', 'вспышка', 'симптом',
↪ 'грипп', 'вакцина', 'вакцинация', 'аппарат искусственной
↪ вентиляции легких', 'изоляция', 'иммунитет', 'больница',
↪ 'отделение интенсивной терапии', 'уход', 'вирусолог',
↪ 'клиника', 'гомеопатия', 'больной', 'фармацевтика',
↪ 'полимеразной цепной реакции', 'пцр', 'коклюш', 'чрезвычайная
↪ ситуация', 'инъекция', 'врачи', 'сотовая связь', 'удаленная
↪ работа', 'линия фронта', 'орви', 'санитария', 'патоген',
↪ 'пропаганда', 'болезнь', 'эпидемия', 'диарея', 'адъюванты',
↪ 'респираторный', 'гигиена', 'белок', 'лекарство', 'новые
↪ дела', 'позитивный', 'ученый', 'заразный', 'мандат',
↪ 'вариант', 'заразить', 'летальные исходы', 'кашель', 'мера',
↪ 'популярный', 'мрна', 'предотвращать', 'здравоохранение',
↪ 'договор', 'неисправность', 'оспа', 'усилитель', 'антитело',
↪ 'доза', 'доказательство', 'дезинформация', 'наблюдаемый',
↪ 'обязательный', 'аллергический', 'аллергия', 'нехватка',
↪ 'синдром', 'китайский', 'тест', 'ограничение',
↪ 'распространение', 'вакс', 'экспериментальный']
```

Listing S.7: List of Russian keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['lockdown', 'gezondheid', 'masker', 'afstand nemen',
↪  'uitbraak', 'symptoom', 'influenza', 'vaccinatie',
↪  'isolatie', 'immuniteit', 'ziekenhuis', 'intensive care
↪  afdeling', 'behandeling', 'viroloog', 'kliniek',
↪  'homeopathie', 'ziek', 'farma', 'polymerasekettingreactie',
↪  'kinkhoest', 'noodgeval', 'injectie', 'artsen', 'mobiel',
↪  'afstandswerk', 'sanitaire voorzieningen', 'pathogeen',
↪  'ziekte', 'diarree', 'adjuvantia', 'ademhalingswegen',
↪  'hygiëne', 'eiwit', 'geneesmiddel', 'nieuwe gevallen',
↪  'positief', 'wetenschapper', 'besmettelijk', 'mandaat',
↪  'infecteren', 'sterfgevallen', 'hoest', 'meeteenheid',
↪  'viraal', 'voorkomen', 'gezondheidszorg', 'contract',
↪  'afsluiten', 'pokken', 'aanjager', 'antilichaam', 'bewijs',
↪  'desinformatie', 'opgemerkt', 'verplicht', 'immuun',
↪  'tekort', 'syndroom', 'medicijn', 'chinese', 'beperking',
↪  'spreiding', 'experimenteel']
```

Listing S.8: List of Dutch keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['koroona', 'viirus', 'pandeemia', 'täielik
↪  sulgemine', 'tervist', 'distantseerumine', 'haiguspuhang',
↪  'sümptom', 'karantiin', 'gripp', 'vaktsiin',
↪  'vaktsineerimine', 'ventilaator', 'isolatsioon',
↪  'puutumatus', 'haiglasse', 'intensiivravi osakonnas', 'ravi',
↪  'kliinik', 'homöopaatia', 'haige', 'polümeraasi
↪  ahelreaktsioon', 'tk', 'läkaköha', 'hädaolukord',
↪  'süstimine', 'arstid', 'rakuline', 'kaugtöö', 'eesliinil',
↪  'kanalisatsioon', 'patogeen', 'propagandat', 'haigus',
↪  'epideemia', 'kõhulahtisus', 'adjuvandid', 'hingamisteede',
↪  'hügieen', 'valk', 'ravim', 'uued juhtumid', 'positiivne',
↪  'teadlane', 'nakkav', 'nakatada', 'surmad', 'köha', 'mõõta',
↪  'viiruslik', 'ära hoida', 'tervishoid', 'leping', 'lülita
↪  välja', 'rõuged', 'võimendaja', 'antikeha', 'annust',
↪  'tõendid', 'desinformatsioon', 'täheldatud', 'kohustuslik',
↪  'allergiline', 'allergia', 'immuunne', 'puudus', 'sündroom',
↪  'hiina keel', 'katsetada', 'piirang', 'levik',
↪  'eksperimentaalne']
```

Listing S.9: List of Estonian keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['pandemia', 'confinamento', 'salute', 'maschera',
 ↪  'distanziamento', 'epidemia', 'sintomo', 'quarantena',
 ↪  'vaccino', 'vaccinazione', 'ventilatore', 'isolamento',
 ↪  'immunità', 'ospedale', 'terapia intensiva', 'unità di
 ↪  terapia intensiva', 'trattamento', 'virologo', 'clinica',
 ↪  'omeopatia', 'malato', 'farmaceutico', 'reazione a catena
 ↪  della polimerasi', 'pertosse', 'emergenza', 'iniezione',
 ↪  'medici', 'cellulare', 'lavoro a distanza', 'prima linea',
 ↪  'servizi igienico-sanitari', 'agente patogeno', 'malattia',
 ↪  'diarrea', 'adiuvanti', 'respiratorio', 'igiene', 'proteina',
 ↪  'medicinale', 'nuovi casi', 'positivo', 'scienziato',
 ↪  'contagioso', 'mandato', 'infettare', 'deceduti', 'tosse',
 ↪  'misurare', 'virale', 'impedire', 'assistenza sanitaria',
 ↪  'contrarre', 'fermare', 'vaiolo', 'ripetitore', 'anticorpo',
 ↪  'prova', 'disinformazione', 'osservato', 'obbligatorio',
 ↪  'allergico', 'immune', 'carenza', 'sindrome', 'farmaco',
 ↪  'cinese', 'restrizione', 'diffusione', 'sperimentale']
```

Listing S.10: List of Italian keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['каранавірусная інфекцыя covid', 'карона', 'вірус',
 ↪  'пандэмія', 'каранцін', 'здароўя', 'дыстанцыяванне',
 ↪  'успышка', 'сімптом', 'грып', 'вакцына', 'вакцынацыя', 'швл',
 ↪  'ізаляцыя', 'імунітэт', 'бальніца', 'іку', 'аддзяленне
 ↪  інтэнсіўнай тэрапіі', 'лячэнне', 'вірусолаг', 'паліклініка',
 ↪  'гамеапатыя', 'хворы', 'фарм', 'палімеразнай ланцуговая
 ↪  рэакцыя', 'пкр', 'надзвычайная сітуацыя', "ін'екцыі",
 ↪  'лекары', 'сотавы', 'выдаленая праца', 'прыфрантавая лінія',
 ↪  'врві', 'санітарыя', 'ўзбуджальнік', 'прапаганда', 'хвароба',
 ↪  'эпідэмія', 'дыярэя', 'органы дыхання', 'гігіена', 'бялок',
 ↪  'лекі', 'новыя выпадкі', 'становучы', 'вучоны', 'інфекцыйны',
 ↪  'варыянт', 'заразіць', 'лятальныя вынікі', 'кашаль',
 ↪  'вірусны', 'прадухіліць', 'ахова здароўя', 'кантракт',
 ↪  'адключэнне', 'натуральная воспа', 'паскаральнік',
 ↪  'антыцелы', 'доказы', 'дэзінфармацыя', 'назіраецца',
 ↪  'абавязковым', 'алергічныя', 'алергія', 'імунная', 'дэфіцыт',
 ↪  'сіндром', 'наркотык', 'кітайскі', 'тэст', 'абмежаванне',
 ↪  'распаўсюджванне', 'эксперыментальны']
```

Listing S.11: List of Belarusian keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['wirus', 'izolacja', 'zdrowie', 'dystansowanie',
↪  'wybuch', 'objaw', 'kwarantanna', 'grypa', 'szczepionka',
↪  'wentylator', 'odporność', 'szpital', 'oddział intensywnej
↪  terapii', 'leczenie', 'wirusolog', 'homeopatia', 'chory',
↪  'farmaceutyczny', 'reakcja łańcuchowa polimerazy', 'szt',
↪  'krztusiec', 'nagły wypadek', 'zastrzyk', 'lekarze',
↪  'komórkowy', 'praca zdalna', 'linia frontu', 'urządzenia
↪  sanitarne', 'biegunka', 'adiuwanty', 'oddechowy', 'białko',
↪  'medycyna', 'nowe przypadki', 'pozytywny', 'naukowiec',
↪  'zakaźny', 'wariant', 'infekować', 'zgony', 'kaszel',
↪  'mierzyć', 'wirusowy', 'mrn', 'zapobiegać', 'opieka
↪  zdrowotna', 'zamknięcie', 'ospa', 'wzmacniacz',
↪  'przeciwciało', 'dawka', 'dowód', 'mylna informacja',
↪  'zauważony', 'obowiązkowy', 'uczulony', 'alergia', 'odporny',
↪  'niedobór', 'zespół', 'lek', 'chiński', 'ograniczenie',
↪  'rozpowszechnianie się', 'eksperymentalny']
```

Listing S.12: List of Polish keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

कोविड, कोरोना, वायरस, महामारी, लॉकडाउन, स्वास्थ्य, नकाब, दूरी, प्रकोप, लक्षण, संगरोधन, इंफ्लुएंजा,
टीका, टीकाकरण, पंखा, एकांत, रोग प्रतिरोधक क्षमता, अस्पताल, आईसीयू, गहन देखभाल इकाई,
इलाज, विषाणु विज्ञानी, क्लिनिक, होम्योपैथी, बीमार, फार्मा, पोलीमरेज श्रृंखला अभिक्रिया, पीसीआर,
काली खांसी, आपातकाल, इंजेक्शन, डॉक्टरों, सेलुलर, दूरदराज के काम, सीमावर्ती, सार्स, स्वच्छता,
रोगज़नक़, प्रचार करना, बीमारी, दस्त, गुणवर्धक औषधि, श्वसन, प्रोटीन, दवा, नये मामले, सकारात्मक,
वैज्ञानिक, संक्रामक, शासनादेश, प्रकार, संक्रमित, मौतें, खाँसी, उपाय, वायरल, एमआरएनए, रोकना,
स्वास्थ्य देखभाल, अनुबंध, शट डाउन, चेचक, बूस्टर, एंटीबॉडी, खुराक, प्रमाण, झूठी खबर, देखा,
अनिवार्य, एलर्जी, प्रतिरक्षा, कमी, सिंड्रोम, दवाई, चीनी, परीक्षा, प्रतिबंध, फैलाना, वैक्स, प्रयोगात्मक

Listing S.13: List of Hindi keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['nedlukning', 'sundhed', 'distancering', 'udbrud',
↪  'karantæne', 'vaccine', 'hospital', 'intensivafdeling',
↪  'homøopati', 'syg', 'polymerase kædereaktion', 'kighoste',
↪  'nødsituation', 'indsprøjtning', 'læger', 'cellulære',
↪  'fjernarbejde', 'frontlinje', 'sanitet', 'sygdom', 'diarré',
↪  'adjuvanser', 'respiratoriske', 'hygiejne', 'medicin', 'nye
↪  sager', 'videnskabsmand', 'smitsom', 'inficere', 'dødsfald :
↪  døde', 'sundhedsvæsen', 'lukke ned', 'antistof', 'beviser',
↪  'misinformation', 'observeret', 'obligatorisk', 'prøve',
↪  'begrænsning', 'spredning', 'eksperimentel']
```

Listing S.14: List of Danish keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

โควิด, โคโรนา, ไวรัส, การระบาดใหญ่, การปิดพื้นที่, สุขภาพ, หน้ากาก, เว้นระยะห่าง, การระบาด, อาการ, การกักกัน, ไข้หวัดใหญ่, วัคซีน, การฉีดวัคซีน, เครื่องช่วยหายใจ, การแยกตัว, ภูมิคุ้มกัน, โรงพยาบาล, ห้องไอซียู, หน่วยดูแลผู้ป่วยหนัก, การรักษา, นักไวรัสวิทยา, คลินิก, โฮมีโอพาธีย์, ป่วย, ยา, ปฏิกิริยาลูกโซ่โพลีเมอเรส, พีซีอาร์, ไอกรน, ภาวะฉุกเฉิน, การฉีด, แพทย์, เซลล์, ทำงานระยะไกล, แนวหน้า, โควิด 19, โรคซาร์ส, สุขาภิบาล, เชื้อโรค, การโฆษณาชวนเชื่อ, โรค, ท้องเสีย, ผู้ช่วย, ระบบทางเดินหายใจ, สุขอนามัย, โปรตีน, กรณีใหม่, เชิงบวก, นักวิทยาศาสตร์, โรคติดต่อ, อาณัติ, ตัวแปร, ติดเชื้อ, ผู้เสียชีวิต, ไอ, วัด, คุณ, ป้องกัน, ดูแลสุขภาพ, สัญญา, ปิดตัวลง, ไข้ทรพิษ, บูสเตอร์, แอนติบอดี, ปริมาณ, หลักฐาน, ข้อมูลที่ผิด, สังเกต, บังคับ, แพ้, โรคภูมิแพ้, มีภูมิคุ้มกัน, การขาดแคลน, ซินโดรม, ชาวจีน, ทดสอบ, ข้อ จำกัด, การแพร่กระจาย, แว็กซ์, ทดลอง

Listing S.15: List of Thai keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

코로나, 바이러스, 감염병 세계적 유행, 폐쇄, 건강, 마스크, 거리두기, 발생, 징후, 건강격리, 인플루엔자, 백신, 백신 접종, 송풍기, 격리, 면역, 병원, 중환자실, 중환자 실, 치료, 바이러스학자, 진료소, 동종 요법, 아픈, 제약, 폴리 메라 제 연쇄 반응, 백일해, 비상, 주입, 의사들, 세포의, 원격 근무, 최전선, 코로나 19, 사스, 위생, 병원체, 선전, 질병, 감염병 유행, 설사, 보조제, 호흡기, 단백질, 약, 새로운 사례, 긍정적인, 과학자, 전염성, 위임, 변종, 감염시키다, 사망자, 기침, 측정하다, 바이러스의, 예방하다, 보건 의료, 계약, 일시 휴업, 천연두, 부스터, 항독소, 정량, 증거, 오보, 관찰됨, 필수적인, 알레르기가 있는, 알레르기, 면역성 있는, 부족, 증후군, 의약품, 중국인, 시험, 제한, 확산, 백스, 실험적인

Listing S.16: List of Korean keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['пандемія', 'закриття', "здоров'я", 'дистанціювання',
 ↪  'спалах', 'грип', 'щеплення', 'вентилятор', 'ізоляція',
 ↪  'імунітет', 'лікарня', 'відділення інтенсивної терапії',
 ↪  'лікування', 'вірусолог', 'клініка', 'гомеопатія', 'хворий',
 ↪  'полімеразна ланцюгова реакція', 'надзвичайна ситуація',
 ↪  "ін'єкція", 'лікарі', 'стільниковий', 'віддалена робота',
 ↪  'лінія фронту', 'covid 19', 'санітарія', 'збудник',
 ↪  'захворювання', 'епідемія', 'діарея', "ад'юванти",
 ↪  'дихальний', 'гігієна', 'білок', 'ліки', 'нові справи',
 ↪  'позитивний', 'науковець', 'заразний', 'варіант', 'заразити',
 ↪  'смерті', 'міра', 'вірусний', 'запобігти', "охорона
 ↪  здоров'я", 'договір', 'закрити', 'віспа', 'бустер',
 ↪  'антитіло', 'докази', 'дезінформація', 'спостерігається',
 ↪  "обов'язковий", 'алергічний', 'імунний', 'дефіцит',
 ↪  'китайський', 'обмеження', 'поширення', 'експериментальний']
```

Listing S.17: List of Ukrainian keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['i-covid', 'ikhoroni', 'igciwane', 'ubhubhane',
↪  'ukuvalwa thaqa kwezwe', 'impilo', 'imaski', 'ukuqhela',
↪  'ukubheduka', 'uphawu', 'ukuzivalela', 'umkhuhlane',
↪  'umgomo', 'ukugoma', 'i-ventilator', 'ukuzihlukanisa',
↪  'ukungatheleleki', 'esibhedlela', 'igumbi labagula kakhulu',
↪  'ukwelashwa', 'i-virologist', 'umtholampilo', 'i-homeopathy',
↪  'abagulayo', 'ikhemisi', 'ukusabela kwe-polymerase chain',
↪  'i-pertussis', 'izimo eziphuthumayo', 'umjovo', 'odokotela',
↪  'iselula', 'umsebenzi kude', 'phambili', 'i-covid-19',
↪  'ukuthuthwa kwendle', 'i-pathogen', 'inkulumo-ze', 'isifo',
↪  'umqedazwe', 'isifo sohudo', 'ama-adjuvants',
↪  'zokuphefumula', 'inhlanzeko', 'amaprotheni', 'umuthi',
↪  'amacala amasha', 'positive', 'usosayensi', 'iyathathelana',
↪  'igunya', 'okuhlukile', 'ukuthelela', 'abashonile',
↪  'ngiyagula', 'ukukhwehlela', 'isilinganiso', 'vimbela',
↪  'ukunakekela impilo', 'isivumelwano', 'vala shaqa',
↪  'ingxibongo', 'i-booster', 'amasosha omzimba', 'umthamo',
↪  'ubufakazi', 'imininingwane engamanga', 'kubhekwe', 'impoqo',
↪  'ukungezwani komzimba', 'ukungezwani komzimba nezinto
↪  ezithile', 'ukushoda', 'i-syndrome', 'amashayina',
↪  'ukuvinjelwa', 'ukubhebhetheka', 'i-vax', 'okokuhlola']
```

Listing S.18: List of Zulu keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['pandeeminen', 'sisälle suojautuminen', 'terveys',
↪  'naamio', 'etäisyyttä', 'taudin puhkeaminen', 'oire',
↪  'karanteeni', 'influenssa', 'rokote', 'rokotus', 'tuuletin',
↪  'eristäytyminen', 'immuniteetti', 'sairaala', 'teho-osasto',
↪  'hoitoon', 'virologi', 'klinikka', 'sairas',
↪  'polymeraasiketjureaktio', 'kpl', 'hinkuyskä', 'hätä',
↪  'injektio', 'lääkärit', 'solu', 'etätyötä', 'etulinjassa',
↪  'sanitaatio', 'taudinaiheuttaja', 'sairaus', 'epideeminen',
↪  'ripuli', 'adjuvantit', 'hengitys', 'hygienia', 'proteiinia',
↪  'lääke', 'uusia tapauksia', 'positiivinen', 'tiedemies',
↪  'tarttuva', 'mandaatti', 'variantti', 'tartuttaa',
↪  'kuolemat', 'yskä', 'mitata', 'estää', 'terveydenhuolto',
↪  'sopimus', 'sammuttaa', 'isorokko', 'tehostin', 'vasta-aine',
↪  'annos', 'todisteita', 'väärää tietoa', 'havaittu',
↪  'pakollinen', 'allerginen', 'immuuni', 'puute', 'oireyhtymä',
↪  'huume', 'kiinalainen', 'testata', 'rajoitus', 'levitän',
↪  'kokeellinen']
```

Listing S.19: List of Finnish keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['kovid', 'virüs', 'karantina', 'sağlık', 'uzaklaşma',
 ↪  'salgın', 'belirti', 'grip', 'aşı', 'vantilatör',
 ↪  'izolasyon', 'bağışıklık', 'hastane', 'yoğun bakım', 'yoğun
 ↪  bakım ünitesi', 'tedavi', 'hasta', 'ilaç', 'polimeraz
 ↪  zincirleme reaksiyonu', 'boğmaca', 'acil durum',
 ↪  'enjeksiyon', 'doktorlar', 'hücresel', 'uzaktan çalışma',
 ↪  'cephe hattı', 'sanitasyon', 'patojen', 'hastalık', 'ishal',
 ↪  'adjuvanlar', 'solunum', 'hijyen', 'yeni vakalar', 'pozitif',
 ↪  'bilim adamı', 'bulaşıcı', 'yetki', 'değişken', 'enfekte
 ↪  etmek', 'ölümler', 'öksürük', 'ölçüm', 'önlemek', 'sağlık
 ↪  hizmeti', 'sözleşme', 'kapat', 'çiçek hastalığı',
 ↪  'yükseltici', 'antikor', 'doz', 'kanıt', 'yanlış bilgi',
 ↪  'gözlemlendi', 'zorunlu', 'alerjik', 'alerji', 'kıtlık',
 ↪  'sendromu', 'çince', 'ölçek', 'kısıtlama', 'yaymak',
 ↪  'deneysel']
```

Listing S.20: List of Turkish keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['vîrus', 'pandemî', 'tecrît', 'tendûrûstî', 'berrû',
 ↪  'dûrkirin', 'şewbe', 'xûya', 'qarantîna', 'bapêş', 'dermanê
 ↪  perpûnê', 'perpûn', 'cudakirin', 'zixtî', 'nexweşxane',
 ↪  'yekîneya lênêrîna giran', 'demankirinî', 'virologist',
 ↪  'homeopatî', 'nexweş', 'reaksiyona zincîra polymerase',
 ↪  'acîlîyet', 'derzîkirinî', 'doktoran', 'cellular', 'karê
 ↪  dûr', 'xeta pêşîn', 'paqijî', 'pathogen', 'propaxanda',
 ↪  'nexweşî', 'epîdemîk', 'navçûyin', 'respiratory', 'proteîn',
 ↪  'derman', 'dozên nû', 'pozîtîf', 'zanistvan', 'perok',
 ↪  'emrê', 'derbaskirin', 'mirinan', 'kûxîn', 'pîvan',
 ↪  'bergirtin', 'parastina saxlemîyê', 'peyman', 'temirandin',
 ↪  'pisîka', 'antibody', 'delîl', 'dezînformasyon', 'dîtin',
 ↪  'bicî', 'alerjîk', 'alerjî', 'lênakev', 'kêmasî', 'tevazok',
 ↪  'çînî', 'îmtîhan', 'tengkirinî', 'belavbûn', 'ceribandin']
```

Listing S.21: List of Kurdish (kurmanji) keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

ಕರೋನಾ, ವೈರಸ್, ಪಿಡುಗು, ಮುಚ್ಚುವುದು, ಆರೋಗ್ಯ, ಮುಖವಾಡ, ದೂರಮಾಡುವುದು, ಸ್ಫೋಟ, ಲಕ್ಷಣ,
ದಿಗ್ಬಂಧನ, ಇನ್ಫ್ಲುಯೆನ್ಸ, ಲಸಿಕೆ, ವ್ಯಾಕ್ಸಿನೇಷನ್, ವೆಂಟಿಲೇಟರ್, ಪ್ರತ್ಯೇಕತೆ, ವಿನಾಯಿತಿ, ಆಸ್ಪತ್ರೆ, ಐಸಿಯು,
ತೀವ್ರ ನಿಗಾ ಘಟಕ, ಚಿಕಿತ್ಸೆ, ವೈರಾಲಜಿಸ್ಟ್, ಕ್ಲಿನಿಕ್, ಹೋಮಿಯೋಪತಿ, ಅನಾರೋಗ್ಯ, ಫಾರ್ಮಾ,
ಪಾಲಿಮರೇಸ್ ಸರಣಿ ಕ್ರಿಯೆಯ, ಪೆರ್ಟುಸಿಸ್, ತುರ್ತು, ಇಂಜೆಕ್ಷನ್, ವೈದ್ಯರು, ಸೆಲ್ಯುಲಾರ್, ದೂರಸ್ಥ ಕೆಲಸ,
ಮುಂಚೂಣಿ, ಸಾರ್ಸ್, ನೈರ್ಮಲ್ಯ, ರೋಗಕಾರಕ, ಪ್ರಚಾರ, ರೋಗ, ಸಾಂಕ್ರಾಮಿಕ, ಅತಿಸಾರ, ಸಹಾಯಕಗಳು,
ಉಸಿರಾಟದ, ಪ್ರೋಟೀನ್, ಔಷಧಿ, ಹೊಸ ಪ್ರಕರಣಗಳು, ಧನಾತ್ಮಕ, ವಿಜ್ಞಾನಿ, ಅಂಟುರೋಗ, ಆದೇಶ, ಭಿನ್ನ,
ಸೋಂಕು ತಗುಲುತ್ತವೆ, ಸಾವುಗಳು, ಕೆಮ್ಮು, ಅಳತೆ, ವೈರಲ್, ತಡೆಯುತ್ತವೆ, ಆರೋಗ್ಯ, ಒಪ್ಪಂದ,
ಮುಚ್ಚಲಾಯಿತು, ಸಿಡುಬು, ಬೂಸ್ಟರ್, ಪ್ರತಿಕಾಯ, ಡೋಸ್, ಪುರಾವೆ, ತಪ್ಪು ಮಾಹಿತಿ, ಗಮನಿಸಿದೆ, ಕಡ್ಡಾಯ,
ಅಲರ್ಜಿ, ಪ್ರತಿರಕ್ಷಣಾ, ಕೊರತೆ, ಸಿಂಡ್ರೋಮ್, ಔಷಧ, ಚೈನೀಸ್, ಪರೀಕ್ಷೆ, ನಿರ್ಬಂಧ, ಹರಡುವಿಕೆ, ವ್ಯಾಕ್ಸ್,
ಪ್ರಾಯೋಗಿಕ

Listing S.22: List of Kannada keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['hào quang', 'vi-rút', 'dịch bệnh', 'lệnh đóng cửa',
↪  'sức khỏe', 'mặt nạ', 'khoảng cách', 'sự bùng phát', 'triệu
↪  chứng', 'cách ly', 'bệnh cúm', 'vắc xin', 'tiêm chủng', 'máy
↪  thở', 'sự cách ly', 'miễn dịch', 'bệnh viện', 'đơn vị chăm
↪  sóc đặc biệt', 'sự đối đãi', 'nhà virus học', 'phòng khám',
↪  'vi lượng đồng căn', 'đau ốm', 'dược phẩm', 'phản ứng chuỗi
↪  polymerase', 'bệnh ho gà', 'khẩn cấp', 'mũi tiêm', 'nhiều bác
↪  sĩ', 'di động', 'làm việc từ xa', 'tiền tuyến', 'vệ sinh',
↪  'mầm bệnh', 'tuyên truyền', 'bệnh', 'bệnh dịch', 'bệnh tiêu
↪  chảy', 'chất bổ trợ', 'hô hấp', 'chất đạm', 'thuốc', 'trường
↪  hợp mới', 'tích cực', 'nhà khoa học', 'dễ lây lan', 'thi
↪  hành', 'khác nhau', 'lây nhiễm', 'cái chết', 'ốm', 'ho', 'đo
↪  lường', 'nổi tiếng', 'thưa ngài', 'ngăn chặn', 'chăm sóc sức
↪  khỏe', 'hợp đồng', 'tắt', 'bệnh đậu mùa', 'tăng cường',
↪  'kháng thể', 'liều lượng', 'chứng cớ', 'thông tin sai lệch',
↪  'được quan sát', 'bắt buộc', 'dị ứng', 'thiếu', 'hội chứng',
↪  'người trung quốc', 'bài kiểm tra', 'sự hạn chế', 'lây lan',
↪  'thực nghiệm']
```

Listing S.23: List of Vietnamese keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['cutar covid', '□wayar cuta', 'annoba', 'hana fita
 ↪  waje', 'lafiya', 'abin rufe fuska', 'nisantar da kai',
 ↪  '□arkewa', 'alama', 'killace masu cuta', 'mura', 'rigakafi',
 ↪  'maganin alurar riga kafi', 'injin iska', 'ka□aici',
 ↪  'asibiti', 'ku', 'sashin kulawa mai zurfi', 'magani',
 ↪  'likitan dabbobi', 'mara lafiya', 'kantin magani',
 ↪  'polymerase sarkar dauki', 'gaggawa', 'allura', 'likitoci',
 ↪  'salon salula', 'aikin nesa', 'layi na gaba', 'cutar covid
 ↪  19', 'tsaftar muhalli', 'farfaganda', 'cuta', 'gudawa',
 ↪  'numfashi', 'tsafta', 'furotin', 'sababbin lokuta',
 ↪  'tabbatacce', 'masanin kimiyya', 'mai ya□uwa', 'umarni',
 ↪  'bambancin', 'harba', 'mutuwa', 'rashin lafiya', 'tari',
 ↪  'auna', 'kwayar cuta', 'hana', 'kiwon lafiya', 'kwangila',
 ↪  'rufewa', 'cutar sankarau', 'mai kara kuzari', 'kashi',
 ↪  'shaida', 'rashin fahimta', 'lura', 'wajibi', 'rashin
 ↪  lafiyan', 'rashin lafiyar jiki', 'karanci', 'ciwo',
 ↪  'sinanci', 'gwadawa', '□untatawa', 'ya□a', 'na gwaji']
```

Listing S.24: List of Hausa keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['pandemic', 'carantină', 'sănătate', 'masca',
 ↪  'distanțare', 'simptom', 'gripa', 'vaccinare', 'izolare',
 ↪  'imunitate', 'spital', 'unitate de terapie intensiva',
 ↪  'tratament', 'bolnav', 'reacție în lanț a polimerazei', 'de
 ↪  urgență', 'injectare', 'medicii', 'celular', 'lucru la
 ↪  distanță', 'prima linie', 'salubrizare', 'propagandă',
 ↪  'boala', 'diaree', 'adjuvanţi', 'respirator', 'igienă',
 ↪  'proteină', 'medicament', 'cazuri noi', 'pozitiv', 'om de
 ↪  stiinta', 'contagios', 'variantă', 'infecta', 'decese',
 ↪  'tuse', 'măsura', 'împiedica', 'contracta', 'închide',
 ↪  'variolă', 'rapel', 'anticorp', 'doza', 'dovezi',
 ↪  'dezinformare', 'observat', 'obligatoriu', 'alergic', 'imun',
 ↪  'deficit', 'sindrom', 'chinez', 'restricţie', 'răspândire',
 ↪  'experimental']
```

Listing S.25: List of Romanian keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['coroa', 'vírus', 'saúde', 'mascarar', 'distanciar',
↪  'surto', 'sintoma', 'quarentena', 'gripe', 'vacina',
↪  'vacinação', 'ventilador', 'imunidade', 'uti', 'unidade de
↪  tratamento intensivo', 'tratamento', 'virologista',
↪  'clínica', 'doente', 'farmacêutica', 'reação em cadeia da
↪  polimerase', 'emergência', 'injeção', 'médicos', 'trabalho
↪  remoto', 'linha de frente', 'sarna', 'saneamento',
↪  'patógeno', 'doença', 'diarréia', 'adjuvantes',
↪  'respiratório', 'higiene', 'proteína', 'medicamento', 'novos
↪  casos', 'cientista', 'transmissível', 'infectar', 'mortes',
↪  'medir', 'evitar', 'assistência médica', 'contrato',
↪  'desligar', 'varíola', 'reforço', 'evidência',
↪  'desinformação', 'observado', 'obrigatório', 'alérgico',
↪  'imune', 'falta', 'síndrome', 'chinês', 'teste', 'restrição',
↪  'espalhar']
```

Listing S.26: List of Portuguese keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['izolacija', 'zdravlje', 'udaljavanje', 'izbijanje
↪  epidemije', 'karantena', 'cjepivo', 'cijepljenje',
↪  'imunitet', 'bolnica', 'jedinica intenzivne njege',
↪  'liječenje', 'virusolog', 'bolestan', 'farmacija', 'lančana
↪  reakcija polimeraze', 'hripavac', 'hitan slučaj',
↪  'liječnici', 'stanični', 'rad na daljinu', 'linija fronta',
↪  'sanitacija', 'uzročnik bolesti', 'bolest', 'proljev',
↪  'pomoćna sredstva', 'dišni', 'higijena', 'lijek', 'novih
↪  slučajeva', 'pozitivan', 'znanstvenik', 'zarazan',
↪  'varijanta', 'zaraziti', 'smrtni slučajevi', 'kašalj',
↪  'mjera', 'virusni', 'spriječiti', 'zdravstvene zaštite',
↪  'ugovor', 'ugasiti', 'velike boginje', 'pojačivač',
↪  'antitijelo', 'dokaz', 'promatranom', 'obavezna',
↪  'alergičan', 'nedostatak', 'droga', 'kineski', 'ograničenje',
↪  'širenje', 'vosak', 'eksperimentalni']
```

Listing S.27: List of Croatian keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['aislamiento', 'salud', 'mascarilla',
 ↪  'distanciamiento', 'brote', 'síntoma', 'cuarentena',
 ↪  'vacuna', 'vacunación', 'inmunidad', 'uci', 'unidad de
 ↪  cuidados intensivos', 'tratamiento', 'virólogo',
 ↪  'homeopatía', 'enfermo', 'farmacéutica', 'reacción en cadena
 ↪  de la polimerasa', 'tos ferina', 'emergencia', 'inyección',
 ↪  'doctores', 'trabajo remoto', 'primera línea', 'saneamiento',
 ↪  'enfermedad', 'adyuvantes', 'nuevos casos', 'científico',
 ↪  'fallecidos', 'tos', 'medida', 'prevenir', 'cuidado de la
 ↪  salud', 'cerrar', 'viruela', 'refuerzo', 'anticuerpo',
 ↪  'evidencia', 'desinformación', 'obligatorio', 'inmune',
 ↪  'escasez', 'chino', 'prueba', 'restricción', 'desparramar']
```

Listing S.28: List of Spanish keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['pandèmia', 'confinament', 'salut', 'màscara',
 ↪  'distanciament', 'brot', 'símptoma', 'vacunació',
 ↪  'aïllament', 'immunitat', 'unitat de cures intensives',
 ↪  'tractament', 'viròleg', 'malalt', 'reacció en cadena de la
 ↪  polimerasa', 'emergència', 'injecció', 'metges', 'cel·lular',
 ↪  'treball a distància', 'primera línia', 'sanejament',
 ↪  'malaltia', 'epidèmia', 'respiratori', 'proteïna', 'nous
 ↪  casos', 'positiu', 'científic', 'contagiós', 'defuncions',
 ↪  'mesura', 'atenció sanitària', 'contracte', 'tancar',
 ↪  'verola', 'amplificador', 'anticossos', 'dosi', 'proves',
 ↪  'desinformació', 'obligatòria', 'al·lèrgic', 'al·lèrgia',
 ↪  'escassetat', 'xinès', 'restricció', 'propagació']
```

Listing S.29: List of Catalan keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

冠状病毒, 电晕, 病毒, 大流行, 封锁, 健康, 面具, 疏远, 暴发, 症状, 隔离, 流感, 疫苗, 疫苗接种, 呼吸机, 免疫, 医院, 重症监护病房, 重症监护室, 治疗, 病毒学家, 诊所, 顺势疗法, 生病的, 制药公司, 聚合酶链式反应, 聚合酶链反应, 百日咳, 紧急情况, 注射, 医生, 细胞的, 远程工作, 前线, 新冠肺炎, 非典, 卫生, 病原, 宣传, 疾病, 流行性, 腹泻, 佐剂, 呼吸系统, 蛋白质, 药品, 新病例, 积极的, 科学家, 传染性, 授权, 变体, 感染, 死亡人数, 患病的, 咳嗽, 措施, 病毒性的, 姆纳, 防止, 卫生保健, 合同, 关闭, 天花, 助推器, 抗体, 剂量, 证据, 误传, 观察到的, 强制的, 过敏的, 过敏, 短缺, 综合症, 中国人, 测试, 限制, 传播, 瓦克斯, 实验性的

Listing S.30: List of Chinese (simplified) keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

στέμμα, ιός, πανδημία, απαγόρευση κυκλοφορίας, υγεία, μάσκα,
αποστασιοποίηση, έξαρση, σύμπτωμα, καραντίνα, γρίπη, εμβόλιο,
εμβολιασμός, εξαεριστήρας, απομόνωση, ασυλία, ανοσία, νοσοκομείο, μονάδα
εντατικής θεραπείας, θεραπεία, ιολόγος, κλινική, οποιοπαθητική, άρρωστος,
αλυσιδωτή αντίδραση πολυμεράσης, κοκκίτης, επείγον, ένεση, γιατρούς,
κυτταρικός, απομακρυσμένη εργασία, πρώτης γραμμής, σάρς, υγιεινή,
παθογόνο, προπαγάνδα, ασθένεια, επιδημία, διάρροια, ανοσοενισχυτικά,
αναπνευστικός, πρωτεΐνη, φάρμακο, νέες περιπτώσεις, θετικός, επιστήμονας,
μεταδοτικός, εντολή, παραλαγή, μολύνω, θάνατοι, εγώ θα, βήχας, μετρήστε,
ιογενής, αποτρέψει, φροντίδα υγείας, σύμβαση, τερματισμος λειτουργιας,
ευλογιά, αρωγός, αντίσωμα, δόση, απόδειξη, κακή πληροφορία,
παρατηρήθηκε, επιτακτικός, αλλεργικός, αλλεργία, απρόσβλητος, έλλειψη,
σύνδρομο, κινέζικα, δοκιμή, περιορισμός, εξάπλωση, πειραματικός

Listing S.31: List of Greek keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

,هنیطنرق, تمالع, عویش, نتفرگ هلصاف, کسام, یتمالس, یلیطعت, یمدناپ, سوریو, انورک, دیووک
سوریو, راتفر, هژیو یاه تبقارم دحاو, ناتسرامیب, تینوصم, اوزنا, شکاوه, نویسانیسکاو, نسکاو, ازنالوفنآ
,یرارطضا, هفرس هایس, زارمیلپ یا هریجنز شنکاو, یزاسوراد, رامیب, یتاپویموه, هاگنامرد, سانش
,از یرامیب, یتشادهب سیورس, سراس, 19 دیووک, مدقم طخ, رود هار زا راک, یلولس, ناکشزپ, قیرزت
,دنمشناد, تبثم, دیدج دراوم, وراد, نیئتورپ, تشادهب, یسفنت, اه یکمک, لاهسا, یمدیپا, یرامیب, تاغیلبت
,ندرک یریگولج, یسوریو, نتفرگ هزادنا, هفرس, ناگدش200cتوف, ندرک هدولآ, هنوگ, روتسد, یرسم
تاعالطا, کرادم و دهاوش, زود, نتداپ, هدننک تیوقت, هلبا, ندش شوماخ, دادرارق, یتشادهب یاه تبقارم
,شرتسگ, تیدودحم, تست, اه ینیچ, مردنس, دوبمک, نوصم, یژرلآ, یتیساسح, یرابجا, دش هدهاشم, طلغ
یبرجت, سکاو

Listing S.32: List of Persian keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['zaprtje', 'zdravje', 'distanciranje', 'izbruh',
↪  'cepivo', 'cepljenje', 'imunost', 'bolnišnica', 'enoti za
↪  intenzivno nego', 'zdravljenje', 'bolan', 'verižna reakcija
↪  polimeraze', 'oslovski kašelj', 'nujnost', 'injekcijo',
↪  'zdravniki', 'celični', 'delo na daljavo', 'frontline',
↪  'sanitarij', 'bolezen', 'driska', 'adjuvansi', 'dihalni',
↪  'beljakovine', 'zdravilo', 'novih primerov', 'pozitivno',
↪  'nalezljiv', 'okužiti', 'smrti', 'kašelj', 'ukrep',
↪  'virusno', 'preprečiti', 'skrb za zdravje', 'pogodba',
↪  'ugasniti', 'črne koze', 'pospeševalnik', 'protitelesa',
↪  'odmerek', 'dokazi', 'napačne informacije', 'opazili',
↪  'obvezno', 'alergičen', 'imunski', 'pomanjkanje', 'kitajski',
↪  'omejitev', 'širjenje', 'eksperimentalno']
```

Listing S.33: List of Slovenian keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

コロナ, ウイルス, パンデミック, ロックダウン, マスク, 距離を置く, アウトブレイク, 検疫, インフルエンザ, ワクチン, 予防接種, 人工呼吸器, 分離, 病院, 集中治療室, 処理, ウイルス学者, 診療所, ホメオパシー, 病気, 製薬, ポリメラーゼ連鎖反応, 緊急, 医師, セルラー, リモートワーク, 最前線, covid-19（新型コロナウイルス感染症, サーズ, 衛生, 病原体, 宣伝, 伝染病, 下痢, アジュバント, 呼吸器系, タンパク質, 薬, 新しい症例, ポジティブ, 科学者, 伝染性の, 委任, 変異体, 感染する, 死亡者（数）, 咳, 測定, バイラル, 防ぐ, 健康管理, 契約, シャットダウン, 天然痘, 増幅器, 用量, 証拠, 誤報, 観察された, 必須, アレルギー性, アレルギー, 不足, 症候群, テスト, 制限, 広める, ヴァックス, 実験的な

Listing S.34: List of Japanese keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

رجحلا, ضرم ةمالع, يشفتلا, دعابت, عانق, ةحص, قالغإ, ةحئاج, سورياف, انوروك, انوروك سوريف ضرم
,جالع, ةزكرملا ةيانعلا ةدحو, بفشتسم, ةناصح, لزع, يعانصلا سفنتلا, حيقلت, لصم, ازنولفنالا, يحصلا
لاعسلا, لسلستملا زاريميلوبلا لعافت, امراف, ضيرم, ةيعيبطلا داوملاب جالع, ةدايع, تاسوريفلا ملاع
لماوعلا, يحصلا فرصلا, 19-ديفوك, ةهجاوملا طخ, دعب نع لمعلا, ةيولخلا, ءابطألا, ةنقح, ئراط, يكيدلا
,يباجيإ, ةديدج تالاح, ءاودلا, نيتورب, يسفنت, ةدعاسملا داوملا, لاهسإ, ءابو, ضرم, ةياعد, ةضرمملا
ةاعرلا, عنمي, انرم, رشتنم, سيقي, لاعس, فوس, ةافولا تالاح, بيصت, ليدبلا, ضيوفت, يدعم, ملاع
,ةيساسحلا, يمازلإ, ظحال, ةئطاخ تامولعم, ةداهش, ةعرج, داضم مسج, معادلا, يردج, قلغا, دقع, ةيحصلا
يبيرجت, راشتنالا, دييقت, ناحتما, بنيص, ءاود, ةمزالتم, صقن, عينم, ةيساسح

Listing S.35: List of Arabic keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

,תעַפַּשַׁ, דודיב, סוטפמיס, תוצרפתה, תוקחרתה, הכסמ, תוּאירְבּ, רגס, הפגמ, ףּיגנ, הרָטָעֲ, דיבוק
,סחַיַ, ץָרמנ לופיטל הדיחי, הקלחמה, סילוח תיב, תוּניסִחֲ, ררֵוְאַמְ, ןוסיח, ביכּרתַ
,הקירז, סוריח, תלעש, זארמילופ תרשרש תבוגת, המראפ, הלֶוֹח, היָתִפָּוֹאימֵוֹה, האפרמ, גולוריו
,הלָחֲמַ, הלָוּמעֲתַ, הלָחֲמַ ללֵוֹחמְ, האָוּרבְתַ, סראס, תיזְחֲהַ וקַ, קוחרמ הדובע, יאִתָ, סיאפור
,קבדמ, ןעָדְמַ, יבִוּיחִ, סישדח סירקמ, הפורת, ןוֹבּלְחֶ, תוּהגֵ, המישנה תכרעמ, סיפסות, לוֹשׁלשְׁ
,תואירב, עַוֹנמְלְ, יפִיגְנְ, הדימ, לעֵתַשְׁהִלְ, הלוח, ורטפנש סישנא, קיבְּדהַלְ, תרֶחֶאַ הסָרְגִ, טדָנמַ
,היָגְרלֶאָ, יגִרלֶאַ, הבוח, סיפצנ, קרָבּוּפמְ עדָימֵ, תוּדעֵ, הנָמָ, ןוֹגְדָוָ, רבֵּגמַ, תוֹעוּבּעבַאֲ, תובכל, הזֶוֹח
ינִוֹיסְנִ, סקוו, תוטשפתה, הלָבָּגהַ, ןחָבְמִ, תינִיסְ, תרוּפָּהַ, תנֶוֹמסְתִ, רוסחמ, ןיסִחֲ

Listing S.36: List of Hebrew keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['koróna', 'obmedzenie pohybu', 'zdravie',
↪   'dištancovanie', 'epidémia', 'symptóm', 'chrípka',
↪   'očkovanie', 'izolácia', 'nemocnica', 'jednotka intenzívnej
↪   starostlivosti', 'liečbe', 'virológ', 'poliklinika', 'chorý',
↪   'farmácia', 'polymerická reťazová reakcia', 'núdzový',
↪   'injekciou', 'lekárov', 'bunkový', 'práca na diaľku',
↪   'frontovej línii', 'sanitácia', 'patogén', 'hnačka',
↪   'dýchacie', 'proteín', 'liek', 'nové prípady', 'pozitívne',
↪   'vedec', 'nákazlivý', 'infikovať', 'úmrtia', 'kašeľ',
↪   'opatrenie', 'vírusový', 'zabrániť', 'zdravotná
↪   starostlivosť', 'zmluvy', 'vypnúť', 'kiahne', 'posilňovač',
↪   'dôkazy', 'dezinformácie', 'pozorované', 'imúnna',
↪   'nedostatok', 'syndróm', 'obmedzenie', 'šírenie',
↪   'experimentálne']
```

Listing S.37: List of Slovak keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['ковид', 'цорона', 'пандемија', 'изолација',
↪   'здравље', 'дистанцирање', 'избијање', 'грипа',
↪   'вакцинација', 'вентилатор', 'имунитет', 'болница', 'ицу',
↪   'интензивне неге', 'третмана', 'виролог', 'хомеопатија',
↪   'болестан', 'пхарма', 'полимеразе ланчана реакција',
↪   'пертуссис', 'хитан', 'ињекција', 'лекари', 'ћелијски', 'рад
↪   на даљину', 'фронтлине', 'сарс', 'санитација', 'патогена',
↪   'болест', 'епидемија', 'дијареја', 'помоћна средства',
↪   'респираторни', 'хигијена', 'беланчевина', 'лек', 'нови
↪   случајеви', 'позитивним', 'научник', 'заразан', 'мандата',
↪   'варијанта', 'преминуле особе', 'кашаљ', 'мерити', 'вирусна',
↪   'спречити', 'здравствена заштита', 'уговор', 'искључити',
↪   'велике богиње', 'боостер', 'доказ', 'дезинформације',
↪   'посматрано', 'обавезна', 'алергични', 'алергија', 'имуни',
↪   'несташица', 'синдрома', 'дрога', 'кинески', 'ограничење',
↪   'ширење', 'вак', 'експериментални']
```

Listing S.38: List of Serbian keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['pandemisk', 'nedstängning', 'hälsa',
↪  'avståndstagande', 'utbrott', 'karantän', 'fläkt', 'sjukhus',
↪  'intensivvårdsavdelning', 'sjuk', 'polymeraskedjereaktion',
↪  'kikhosta', 'nödsituation', 'läkare', 'cellulär',
↪  'distansarbete', 'sanering', 'sjukdom', 'diarre',
↪  'andningsorganen', 'hygien', 'nya fall', 'forskare',
↪  'smittsam', 'infektera', 'dödsfall', 'hosta', 'mäta',
↪  'förhindra', 'sjukvård', 'avtal', 'stänga av', 'smittkoppor',
↪  'antikropp', 'dos', 'felaktig information', 'observerade',
↪  'brist', 'läkemedel', 'kinesiska', 'testa', 'restriktion',
↪  'sprida', 'experimentell']
```

Listing S.39: List of Swedish keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['vīruss', 'pandēmija', 'blokāde', 'veselība',
↪  'distancēšanās', 'uzliesmojums', 'simptoms', 'karantīna',
↪  'vakcīna', 'vakcinācija', 'ventilators', 'izolācija',
↪  'imunitāte', 'slimnīca', 'intensīvās terapijas nodaļā',
↪  'ārstēšana', 'virusologs', 'klīnika', 'homeopātija', 'slims',
↪  'polimerāzes ķēdes reakcija', 'gab', 'garā klepus',
↪  'ārkārtas', 'ārstiem', 'šūnu', 'attālināts darbs', 'frontes
↪  līnija', 'sanitārija', 'patogēns', 'slimība', 'epidēmija',
↪  'caureja', 'palīgvielas', 'elpošanas', 'higiēna',
↪  'olbaltumvielas', 'medicīna', 'jauni gadījumi', 'pozitīvs',
↪  'zinātnieks', 'infekciozs', 'pilnvaras', 'variants',
↪  'inficēt', 'nāves gadījumi', 'slim', 'klepus', 'mērs',
↪  'vīrusu', 'novērst', 'veselības aprūpe', 'līgums', 'izslēgt',
↪  'bakas', 'pastiprinātājs', 'antivielu', 'devu',
↪  'pierādījumi', 'dezinformāciju', 'novērotā', 'obligāts',
↪  'alerģisks', 'alerģija', 'imūns', 'trūkums', 'sindroms',
↪  'narkotiku', 'ķīniešu', 'pārbaude', 'ierobežojums',
↪  'izplatība', 'eksperimentāls']
```

Listing S.40: List of Latvian keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

نشینیسکیو, نیسکیو, ازنئولفنا, ہنیطنرق, ؤالیھپ, یرود, تحص, نؤاڈ کال, ءابو یملاع, سرئاو, انوروک,
کنیلک, ٹسجولورئاو, تشادہگن ئاہتنائےارب ٹنوی, وی یس یئآ, لاتپسہ, تعفادم توق, یگدحیلع, رٹیلیٹنیو,
رود, رلولیس, ہورٹکاڈ, نشکجنا, یسنجرمیا, یسناھک یلاک, رآ یس یپ, لمعدر نیچ زیرمیلوپ, یھتیپویموہ
نیٹورپ, تحص ناظفح, یک سناس, نواعم, لاہسا, اڈنگیپورپ, قنزگور, یئافص, نئال ٹنرف, ماک اک زارد,
تحص, انکور, لرئاو, شئامیپ, یسناھک, تاوما, رثاتم, ریغتم, ٹیڈنیم, یدعتم, نادسنئاس, تامدقم ےئن, یئاود
ایک ہدہاشم, تامولعم طلغ, توبث, کاروخ, یڈاب یٹنیا, رٹسوب, کچیچ, دنب, ہدہاعم, لاھب ھکید یک,
یتابرجت, یدنباپ, ھکرپ, ینیچ, اود, مورڈنس, تلق, یتعفادم, یجرلا, یمزال

Listing S.41: List of Urdu keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['világjárvány', 'lezárás', 'egészség', 'maszk',
 ↪  'távolságtartó', 'kitörés', 'tünet', 'karantén', 'oltás',
 ↪  'elkülönítés', 'immunitás', 'kórház', 'intenzív osztályon',
 ↪  'kezelés', 'virológus', 'homeopátia', 'beteg', 'polimeráz
 ↪  láncreakció', 'szamárköhögés', 'vészhelyzet', 'injekció',
 ↪  'orvosok', 'sejtes', 'távmunka', 'frontvonal', 'higiénia',
 ↪  'kórokozó', 'betegség', 'járvány', 'hasmenés', 'adjuvánsok',
 ↪  'légúti', 'fehérje', 'gyógyszer', 'új esetek', 'pozitív',
 ↪  'tudós', 'fertőző', 'megbízás', 'változat', 'megfertőzni',
 ↪  'halálozások', 'köhögés', 'intézkedés', 'vírusos',
 ↪  'megakadályozni', 'egészségügyi ellátás', 'szerződés',
 ↪  'leállítás', 'himlő', 'gyorsító', 'ellenanyag', 'dózis',
 ↪  'bizonyíték', 'félretájékoztatás', 'megfigyelt', 'kötelező',
 ↪  'allergiás', 'immunis', 'hiány', 'szindróma', 'drog',
 ↪  'kínai', 'teszt', 'korlátozás', 'terjedés', 'kísérleti']
```

Listing S.42: List of Hungarian keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['kuncitara', 'kesehatan', 'menjauhkan diri', 'wabah',
 ↪  'gejala', 'vaksin', 'vaksinasi', 'isolasi', 'kekebalan',
 ↪  'rsud', 'unit perawatan intensif', 'perlakuan', 'ahli virus',
 ↪  'homoeopati', 'sakit', 'farmasi', 'reaksi berantai
 ↪  polimerase', 'pertusis', 'keadaan darurat', 'injeksi',
 ↪  'dokter', 'seluler', 'kerja jarak jauh', 'garis depan',
 ↪  'kebersihan', 'penyakit', 'diare', 'bahan pembantu',
 ↪  'pernafasan', 'obat', 'kasus baru', 'ilmuwan', 'menular',
 ↪  'varian', 'menulari', 'meninggal', 'batuk', 'ukuran', 'tuan',
 ↪  'mencegah', 'kontrak', 'matikan', 'cacar', 'pemacu',
 ↪  'antibodi', 'bukti', 'keterangan yg salah', 'diamati',
 ↪  'wajib', 'alergi', 'kekurangan', 'sindroma', 'cina', 'tes',
 ↪  'larangan', 'menyebar', 'eksperimental']
```

Listing S.43: List of Indonesian keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

```
keywords = ['затваряне', 'здраве', 'дистанциране', 'избухване',
 ↪  'карантина', 'ваксина', 'ваксинация', 'изолация', 'интензивно
 ↪  отделение', 'лечение', 'хомеопатия', 'болен', 'фармация',
 ↪  'полимеразна верижна реакция', 'спешен случай', 'инжекция',
 ↪  'клетъчен', 'дистанционна работа', 'фронтова линия',
 ↪  'заболяване', 'епидемия', 'диария', 'адюванти', 'дихателна',
 ↪  'хигиена', 'протеин', 'нови случаи', 'положителен', 'учен',
 ↪  'заразен', 'заразяват', 'смъртни случаи', 'аз ще', 'кашлица',
 ↪  'мярка', 'вирусен', 'предотвратявам', 'здравеопазване',
 ↪  'изключвам', 'едра шарка', 'антитяло', 'доказателства',
 ↪  'наблюдаваното', 'задължителен', 'алергия', 'имунен',
 ↪  'недостиг', 'китайски', 'разпространение', 'експериментален']
```

Listing S.44: List of Bulgarian keywords used in the guidance mechanism of the crawler. For the base list in English see Listing 4.1.

S4 Seed List

```
chat_names = ['COVID19_SPb','msiavaxfenomena',⌋
↪  'dannicollateralivaccinocovid','covid_med','anti_covid21',⌋
↪  'covidvaccinevictims','studiscientificivaccini','COVID19Up',⌋
↪  'vaccinecovid19news','Karnataka_KoViD19_Broadcast',⌋
↪  'covid_world21','resistenza_liberta','covid19Law',⌋
↪  'JScovid1984','covid19bulgaria','vaccinationcovid',⌋
↪  'Pierre_Kory','MyGovCoronaNewsdesk','worlddoctorsalliance',⌋
↪  'GrapheneAgenda','noalvaccino','truthpills',⌋
↪  'cdc_nhs_vaccination_cards','impfpass_impfzertifikat',⌋
↪  'DigitalQRcode','corona','digitaler_Impfpass_QR_Covid_MMR',⌋
↪  'impfausweis4','phcoronavirus','COVID19_ImpfpassQR_Digital',⌋
↪  'antivaxclinic','digitalcovide19card','CovidRedPills',⌋
↪  'covidvaccineinjuries','Covid_vaccine_victims','thuletide',⌋
↪  'SvetInWorld','loscaballerosdelzodiacovideo',⌋
↪  'vaccinationcovid','nederlandvaccinatie','covid19vaccinec',⌋
↪  'medicine_t','cdcvaccinescards','corona_atila',⌋
↪  'covid_vaccine_certificate','certificado_covid','young_med',⌋
↪  'Vrach_covid','medic_news_tg','vaccinecovid19news',⌋
↪  'koronavirusj','noalvaccino','coronachecknederlandqr',⌋
↪  'freedomfromcovidscam','covidizh',⌋
↪  'Karnataka_KoViD19_Broadcast/stat','shixun160',⌋
↪  'CoronavirusNewsIta','bulledeveil','covid19Law',⌋
↪  'sputnik_vaccine','coronahockey','JScovid1984',⌋
↪  'Unionforpeace','jobzoid','resistenza_liberta',⌋
↪  'real_hero_official','Impfschaeden_Suedtirol_Corona',⌋
↪  'CovidHealer','HeikoSchrangTV','reinfocovid974',⌋
↪  'corona2019ncov','coronabildirim','covid19golosiiv',⌋
↪  'covid19indonesia','HKFIGHTSCOVID19','JScovid1984',⌋
↪  'Coronavirus_ye','Virus_Info','corona2019ncov',⌋
↪  'Impfschaeden_Suedtirol_Corona','covid19_moldova',⌋
↪  'ftaroexpcovid1984','notizie19','Sputnik_M',⌋
↪  'UnityProjectUSA','glogerok','Online_Forms','reinfocovid974',⌋
↪  'uglyprescribtions','TelegramTipsAR',⌋
↪  'PathshalaClassesJaipur','stresultofficial',⌋
↪  'QArmyJapanFlynn','mainnews','morefaternews','CENmedia',⌋
↪  'FLM888','CENmedia']
```

Listing S.45: The seed list used to in the snowball crawling. The list contains 100 different channels which are used to start the crawling process i.e. they are the starting point for the data acquisition. If a included channels still exist they can be accessed online by using the following link https://t.me/<chat_name>.