Universität Rostock

Traditio et Innovatio

Public Interest in prevalent Rumors in Online-Communities

Name:	Marc Beller
Matrikelnummer:	220202788
Abgabedatum:	18.3.2026
Betreuer und Gutachter:	Dr. Wolfram Just University of Rostock Institute of Mathematics
Gutachter:	Prof. Jens Starke University of Rostock Institute of Mathematics

Masters Thesis Institute of Mathematics
University of Rostock

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Introduction

Recent advances in Machine Learning have brought a wave of progress to many scientific fields. As Transformer based architectures have enabled context depended machine translations that outperform previous methods [16] and generative models inch closer to passing the once monumental Turing test, it is apparent that a new frontier in natural language processing has been reached. This work aims to utilize this progress and study online communities with a prevalence of topics that draw rumors and possibly misinformation. A core goal is to see whether content in these communities reflects broader public interest and if there are identifiable factors correlated to sentiment of online posts. Selected topics are **Vaccine Skepticism**, the **Contested 2020 Election**, the **Q-Anon** theory and **Flat Earth** theory. While these topics often draw controversy it is not the purpose of this study to marginalize alternative beliefs or to evaluate truthfulness on any of the presented issues.

Embeddings have been used in several ways to study social dynamics, as A. van Loon et al.[10] have shown correlations between factors of racial animus and anti black sentiment in the semantic relation between relevant embeddings. N. Garg et al. [13] have tracked the relation of word embeddings in 10 year periods over the 20th century and have shown changes in relations between social shifts and relations of embeddings. In this work sentence embeddings of telegram messages will be clustered into semantically similar topics with BERTopic[6] and sentiments will be determined with a pretrained RoBERTa model [2]. To understand if prevalence in the relevant communities is aligned with broader public interest in the respective topic, google trends [5] are used as a stand in for public curiosity. With the exception of 20 markers of economic insecurity and a set of 20 random trends, all trends are sampled from the emergent topic descriptions of frequent topics. In contrast to previous methods this prevents researcher bias and creates the starting point for a automated tool to detect how closely a communities interests are aligned with main stream ideas.

2 Data Collection

To collect a sufficiently large dataset there are several requirements the data collection process has to meet. While web data is publicly available many websites limit automated data collection techniques like automated web browsers. Some websites offer official api access but free usability is often limited like in the case of reddits pagination feature or X api. In the following section we will discuss how to handle data in compliance with privacy protection laws and ethical standards, as even though all data used is publicly available it might still contain personal information.

2.1 Telegram Channels

Telegram originated as a private messaging app. Similar to other tools the app allows one-to-one, as well group-chatting but it distinguishes itself with an abundance of so called channels. In a given channel only the channel owner and designated members can share content while most participants consume the shared content like a blog. It has been alleged that this structure lends itself to the spread of conspiratorial thought and is being used for profit [12]. Additionally to text, content creators can share videos, pictures, gifs and links in a format similar to most common messenger apps. Channels allow users to respond with emojis to a message and a corresponding count is listed, some channels also allow a comment section on each message, which functions like a chat group. The app has a search function to seek out channel and user names. A user can preview channel content without ”subscribing” (free) to the channel, but a subscription is necessary to respond with emojis, comments or importantly for web scraping. Some private Telegram groups require requests to join, in this work only public groups have been viewed.

Data Colletion

To seek out relevant channels the following seraches have been performed:

- Vaccines
- Corona
- Corona hoax
- 2020 election
- Stop the steal
- Q anon

- Q drops
- Wwg1wga
- Flat earth
- Flat earth theory

After a core collection of channels has been established the apps ”similar channels” feature and frequently linked channels were used to increase the size of the dataset until sufficiently large. Only majority english channels have been selected for the task. Web scraping was executed slowly between 2025.09.01 and 2026.01.15 due to uncertain api usage limits with the telethon python library.

Privacy

In the following work none of the collected texts are displayed, neither are channel names nor audience sizes. Displayed message amounts are aggregated from more than 100 channels over the respective week and should therefor not reflect any identifiably patterns of individual groups. All collected data was openly available.

2.2 Supplemental Datasets

A key part of extracting insights from data is the relation to external anchors that help categorize different observed behaviors, uncover correlations and narrow down causal relationships. For this work google search trends [5] have been used as a stand in for public interest into a given subject. During the automated compilation of trends possible names of private individuals or groups have been removed. Google trends have been collected via pytrends, since requests of 5 year weekly data were not possible, weekly trend sections were scaled linearly to monthly data of a corresponding time frame.

3 Data Processing

3.1 Natural Language Processing

Having gotten a better understanding of the data structure and the available metadata we will turn our sights on more complex tools of information extraction. Since the AI tools used in this work are in large parts very similar to each other the next part will discuss the fundamental principles enabling natural language processing.

3.1.1 Transformers

While varying in countless ways, current language processing AI is almost always based on the idea of the transformer. It is a mechanism that takes in text and minimizes the error of its output compared to a given goal by utilizing learned patterns, derived during a training phase.

Tokenization and Embedding

The first step for any input text sequence is the split into predefined units from a fixed vocabulary, so called tokens. Any such predefined sequence has an id and will later be turned an embedding vector.

Definition 1. *Token*
An input I generates a sequence of tokens:

$$\underbrace{the}_{t_i}\underbrace{\_trans}_{t_j}\underbrace{form}_{t_k}\underbrace{er}_{t_l}\underbrace{\_is}_{t_m}\underbrace{\_trans}_{t_j}\underbrace{form}_{t_k}\underbrace{ing}_{t_n}... \overset{Tokenization}{\rightarrow} [t_i, t_j, t_k, t_l, t_m, t_j, t_k, t_n...] \tag{3.1}$$

A common way to turn the given token ID into a vector is one hot encoding $OneHot(x) = h_x : h_x \in \{0,1\}^N$, with h_x being the x-th unit vector and N the size of the vocabulary (set of all defined tokens). The later discussed BERT [3] encoder has an english uncased vocabulary size of 30522, so in order to reduce dimensionality these one hot encoded vectors are multiplied with an embedding matrix $E \in \mathbb{R}^{N \times d}$ to create embedding vectors for given tokens $e_x = h_x E : e_x \in \mathbb{R}^d$. With the respective token IDs by a Tokenizer, like "WordPiece" in the case of BERT, the embedding Matrix is the first learned component of the transformer.
Learning in the case of a transformer refers to the process of backpropagation, in which the model makes a prediction during training. This prediction (e.g. the likelihood distribution for filling in a masked token) is compared to the desired output in a loss-function.

With the help of the respective derivatives each learning component (weight) is adjusted according to its contribution to the error by an optimizer, to a set learning rate.

Definition 2. *Positional Encoding*
To encode the Position of an embedded Token $e_i \in \mathbb{R}^d$ in a sequence of length l, a vector $p_i \in \mathbb{R}^d$ of equal dimension is added. This generally happens in a manner so $||(p_i + e_i) - (p_j + e_j)|| >> 0 \forall i, j \in \{1..l\}, i \neq j$, preventing significant overlap with other embeddings.

In Attention is all you need the Authors chose

$$p_{i,2j} = sin(\frac{i}{10000^{\frac{2j}{d}}}), p_{i,2j-1} = cos(\frac{i}{10000^{\frac{2j}{d}}}) \forall j = 1..d/2 \tag{3.2}$$

Attention

With the original input sequence I processed into embeddings, the transformer now creates three different matrices out of X linear projections with learnable components (weihgts), the **Q**uery-, the **K**ey-, and **V**alue- matrices Q, K and V. This is the basis of the so called attention mechanism which has been a key part of modern transformer architecture since its introduction in "Attention is all you need" by Vaswani et al. (2017) [16].
This mechanism, often called attention head, is implemented in many different variants and combinations, depending on model and tasks. The following describes the principle of a single attention head while later in this chapter we will see how the mechanism is implemented in practice.

Definition 3. *Attention Head*
Let $X \in \mathbb{R}^{l \times d}$ denote the input for the attention head, like a sequence of l positionally encoded embeddings:

$$X = \begin{pmatrix} —e_i— \\ —e_j— \\ —e_k— \\ \vdots \end{pmatrix} \tag{3.3}$$

$X \in \mathbb{R}^{l \times d}$ is multiplied with the respective weight matrices $W^Q, W^K \in \mathbb{R}^{d \times d_k}, W^V \in \mathbb{R}^{d \times d_v}$ to produce the query, key and value matrix.

$$\begin{aligned} Q &= XW^Q \\ K &= XW^K \\ V &= XW^V \end{aligned} \tag{3.4}$$

For these matrices we compute the so called "Scaled Dot Product Attention" [16]

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{3.5}$$

Note that $softmax(B)_{i,j} = \frac{e^{B_{i,j}}}{\sum_{m=1}^{l} e^{B_{i,m}}} \forall B \in \mathbb{R}^{l \times l}$ *normalizes each row to a probability distribution. Overall this results in an output matrix* $Y \in \mathbb{R}^{l \times d_v}$.

The practical choice of d_k and d_v is dependent on the surrounding transformer architecture. Commonly there is a number of n attention heads that all take in the same input $X \in \mathbb{R}^{l \times d}$ and map it to $Y_i \in \mathbb{R}^{l \times \frac{d}{n}}$ for all $i \in \{1..n\}$, so $d_k = d_v = \frac{d}{n}$. Multiplying another learned Matrix $W^O \in \mathbb{R}^{d \times d}$ to our concatenated results yields an output

$$\begin{pmatrix} | & | & | & | \\ Y_1 & Y_2 & ... & Y_n \\ | & | & | & | \end{pmatrix} W^O = O \in \mathbb{R}^{l \times d} \tag{3.6}$$

with the same dimensions as the input and simultaneously describes the concept of multi head attention [16], for now we will assume a single attention head.
This setup encodes the semantic intuition that the meaning of a word (or token) depends on its surrounding context. When each output $Y_{i,:}$ is understood as an updated representation of the respective $X_{i,:}$ input, we see $Q_{i,:}K^T$ produces a score vector that consists of a series of dot products of the i-th query with different keys. After scaling and softmax we gain a vector of attention scores $(softmax(\frac{Q_{i,:}K^T}{\sqrt{d_k}})) \in \mathbb{R}^{1 \times l}$ that define how relevant the values of every token from the key matrix is to the query token in position i. Onto this vector of attention scores we multiply $V \in \mathbb{R}^{l \times d_v}$, which results in the row $Y_{i,:}$ containing information from all tokens, weighted by the corresponding attention score. A crucial benefit is that asymmetric relationships of tokens can be learned.

Feed Forward Networks

Feed forward networks are a classic part of artificial intelligence design and somewhat follow the intuition of real neurons. When the strength of incoming signals crosses a boundary the neuron fires. In natural language processing feed forward layers are often added after multi head attention. With the context of the surrounding tokens added each token passes through the same neural network, which enhances or suppresses features according to non linear dependencies learned during training. This is achieved with the help of activation functions like ReLU (original Transformer [16]) or GELU (BERT [3]), which opposed to a real neurons on-off mechanic send a more differentiated signal.

Definition 4. *Feed Forward Network FFN For an input vector* $Y_{i,:} \in \mathbb{R}^{1 \times d}$ *a feed forward network can be expressed as:*

$$FFN(Y_{i,:}) = \alpha(Y_{i,:}W_1 + b_1)W_2 + b_2 \tag{3.7}$$

With $W_1 \in \mathbb{R}^{d \times s}, W_2 \in \mathbb{R}^{s \times d}, b_1 \in \mathbb{R}^{1 \times s}, b_2 \in \mathbb{R}^{1 \times d}$ *and the activation function* $\alpha : \underset{X_{1,j} \mapsto \alpha(X_{1,j})}{\mathbb{R}^{1 \times s} \to \mathbb{R}^{1 \times s}}$.

In Attention is all you need the choice forα fell on $ReLU(x) = \begin{cases} 0, & x \in (-\infty, 0) \\ x, & x \in [0, \infty) \end{cases}$.

Transformer Architectures

After gaining an understanding of the basic transformer components we will now turn to the different possible combinations of these building blocks. The idea of scaled dot product attention was first introduced in the context of an encoder-decoder architecture, which was trained for a machine translation task. In such a setup the transformer combines an input and its translation up to this point to generate a new token, in this case, another part of the translation.
For the the purpose of this work we will focus on encoder-only models, which use stacked layers of a multi-head attention and a feed forward network (transformer blocks) to produce new token representations of the input sequence, additionally accounting for their surrounding context. As introduced by Vaswani et. al. every individual layer output is combined linearly with the respective layer input, preserving the original token structure for longer. Similarly a dropout regime is applied during training which leads to randomly selected activations are blocked for individual training batches, this process prevents overfitting by reducing over reliance on individual neurons. Additonally each output $Y_{i,:}$ is normalized along its embedding dimension with a Layer Norm operation.

Definition 5. *Layer Norm Operation A Layer Norm is scaling operation based on row wise mean* $\mu = \frac{1}{d}\sum_{k=1}^{d} Y_{i,k}$*, standard deviation and 2 vectors of learnable weights* $w, w' \in \mathbb{R}^{1,d}$.

$$LN(Y_{i,j}) = w_j \frac{Y_{i,j} - \mu}{\sqrt{\frac{1}{d}\sum_{m=1}^{d}(Y_{i,m} - \mu)^2 + \epsilon}} + w'_j \forall j = 1..d \tag{3.8}$$

With a small $\epsilon \in \mathbb{R}_{>0}$

3.1.2 BERT

Bidirectional **E**ncoder **R**epresentations from **T**ransformers is an encoder based language representation model introduced by Devlin et. al.[3]. It uses the WordPiece tokenizer and is set up with L transformer blocks,an embedding dimension d and A heads in multi-head attention. The model has originally been configured in 2 differently sized variants, BERT_{Base} with $(A = 12, d = 768, L = 12)$ and BERT_{Large} with $(A = 16, d = 1024, L = 24)$. In the following we will refer to BERT_{Base} as BERT, both models have received the same training regimen.
The input can be structured as one or two sequences, in the case of one sequences the token series is composed as $[CLS], t_1, t_2..t_n[SEP]$, in the case of 2 sequences we have $[CLS], t_1, t_2, ..t_n, [SEP], t_{n+1}, t_{n+2}, .., t_m, [SEP]$. These sequences are embedded as seen above with a key difference being that positional embeddings are learned during training [1] and summed with learned encodings that delineate whether the input sequence is part of the first sentence or the second.

[1] 90% for a maximum sequence length of 128 and 10% for a sequence legnth of 512, this was done to facilitate inputs up to length 512 tokens while maintaining efficiency in most training steps

BERT Training

The BERT models fundamental training on language encoding (pre-training) consist of a combination of a **M**asked **L**anguage **M**odel task and **N**ext **S**entence **P**rediction. While both of these tasks are common machine learning objectives, the following definitions correspond to the setup in BERT[3].

Definition 6. *MLM Task*
For any input sequence 15% of tokens $t_{i_{MASK}}$ get selected for prediction and masked with the following rule:

- *80% of selected tokens get replaced with a [MASK] token.*
- *10% of selected tokens get replaced with a random token.*
- *10% of selected tokens do not get replaced*

The model then produces a probability distribution by feeding the final hidden state of the masked token $Y_{i_{MASK},:}$ into a learned matrix $W \in \mathbb{R}^{d \times d}$, scaling it and multiplying it with the transposed embedding Matrix $E \in \mathbb{R}^{N \times d}$.

$$softmax(LN(\sigma(Y_{i_{MASK}}W + b_1))E^T + b_2) \tag{3.9}$$

The strategy was designed to reduce the mismatch between the current pretraining and later finetuning steps in which the [MASK] token generally does not appear.
To engrain an understanding for inter-sentence relationships and set up the possibility of multi part input for a variety of downstream tasks like question answering and natural language inference a 2 part input was facilitated.

Definition 7. *NSP Task*
Any input sequence during pre-training is constructed from 2 sentences A and B seperated by the [SEP] token.

- *In 50% of training instances B follows A in the underlying corpus and it is labeled as **IsNext**.*
- *In the other 50% of training instances B is chosen at random and labeled as **NotNext**.*

*The prediction of probabilities of **IsNext** and **NotNext** classification head based of the transformed value of the $[CLS]$ token: $Y_{[CLS],:}$, leading to the prediction $softmax(Y_{[CLS],:}W_C + b), W_C \in \mathbb{R}^{d \times 2}$ [2].*

During the training the loss functions of these tasks are combined while this regimen operates on the BookCorpus and english Wikipeedia[3]. After pre-training the model runs through several steps of finetuning in which several more task heads increase different model abilities.

[2] In the original BERT implementation exists pooling layer with tanh activation function $tanh(Y_{[CLS],:}W + b)$ not mentioned in the paper [8]

3.2 RoBERTa

Robustly **o**ptimized **BERT** pretraining **a**pproach, is as the name suggests, a BERT like encoder and was introduced by Liu et al. [9]. While it largely maintains the BERT architecture it refines training with significant performance impact. It replaces word piece tokenization with a byte pair encoding algorithm [3] Beside increasing the amount of training data RoBERTa changes hyper parameters and removing the NSP task[4].

3.2.1 Classification with Roberta

As seen in the training of BERT, it is possible to train such a model for binary decision making by feeding the transformed $[CLS]$ into a classification head.

Definition 8. *Classification Head*
A classification head is a learned component of a language model that enables the attribution of input into $k \in \mathbb{N}$ predefined classes, corresponding to respective labels during training. In BERT like encoders the input for the classification head mainly falls on the final hidden state of the predefined starting token $Y_{0,:}$ (eg. $Y_{[CLS],:}$ or $Y_{[s],:}$).

$$Class(Y_{0,:}) = softmax(Y_{0,:}W_C + b) \tag{3.10}$$

With $W_C \in \mathbb{R}^{d \times k}$.

Due to the nature of the encoder architecture the final hidden state of the starting token carries a significant amount of information form all following tokens, as for example displayed in the NSP task in BERT training, but it is in theory possible to train for any classification problem. From selecting answers to a given question over author attribution [19] to sentiment detection [2], the hidden state of the starting token can be utilized, and, due to the broad availability of training data and the relative small cost of finetuning, a broad range of text classifiers are freely accessible.

TweetEval

Since the collected dataset of this work consists of short form texts directed at a semi-public audience in a social media like environment it is pertinent to choose a model trained in a similar environment. The RoBERTa based TweetEval model introduced by Barbieri et. al. [2] fits this requirement well due to the large and diverse training set, similar to the collected data. As the name suggest the classifier was trained on a dataset of Twitter posts, which are short form social media posts directed at a public audience like content collected from Telegram channels

[3] The most frequent pair of input symbols is merged into a new symbol which is added to the vocabulary, this can be iterated until vocabulary entries reach a certain length or any other stopping condition.

[4] Token designations also change, for example the starting token is now <s>

3.3 SBERT

Having developed and intuition for the computation of the final hidden states of tokens and the encoded information within, we are now able to frame any type of text reliant classification task as a function of the final hidden state of the starting token in BERT like models, but research shows that for certain tasks there are far more efficient ways to fulfill a given goal.

Text Similarity

To predict whether two sentences are similar it is easy to construct a classification head which either returns the probabilities for a binary result in $\{is\_similar, not\_similar\}$ or a similarity score mappable to a finite set of labels. While this algorithm is easily implemented and aligned with the sentence or sequence wise separated inputs in many BERT like models, this method shows significant drawbacks when a detailed measure is required or many documents have to be put in relation to each other. As demonstrated by Reimers and Gurevych in [14] finding the most similar sentences in a collection 10000 samples can take 50 million inference computations with BERT, but the authors demonstrate how the corresponding computational time of around 65 hours[5] can be cut down to a process of about 5 seconds
A different way to approach semantic similarity relies on the same principle by which learned embeddings of each respective token conveys information about the surrounding context. In the same way many works [13],[10] use the relations of embedded individual tokens it is possible to do so with any input sequence, by leveraging the transformed final hidden states of these respective tokens. While training, an embedding matrix encodes general information from the entire text corpus, such as the synonymous use of words, so the final hidden states of tokens in an encoded input sequence carry semantic information about every other token in the sequence. The evident problem is the sequence length dependent variation in dimensionality any encoded input displays, as according to the introduced transformer architecture any input sequence of length $l \in \mathbb{N}$ will, given embedding dimension $d \in \mathbb{N}$, result in a transformed output $Y \in \mathbb{R}^{l \times d}$. This poses the challenge to create a fixed length representation of any input sequence that conveys relevant meaning.
With **S**entence **BERT** Reimers and Gurevych devised a simple but elegant solution to this problem. Selected from different strategies, taking the mean of each embedding dimension along the sequence length enables, given correct training, better results than methods based on starting tokens both in quality and efficiency.

Definition 9. *Mean Pooling Layer*
A mean pooling layer takes the average of a given transformed token sequence $Y_{i,:} \in \mathbb{R}^{1 \times d}, i = 1..l$ *of length* l *and thereby aggregates patterns in an output vector* $u \in \mathbb{R}^{1 \times d}$

[5] on an NVIDIA V100 GPU

that reflect information within the whole sequence[6].

$$u_{1,j} = \frac{1}{l}\sum_{i=1}^{l} Y_{i,j} \tag{3.11}$$

Of course this vastly reduces dimensionality which emphasis the importance of correct training incentives to reliably distill required information.
While newer models have replaced the original SBERT transformer most still rely the same principle of mean pooling the final hidden states of the token sequence. The following fine tuning task illustrates the training objective of the SBERT like encoder, utilized in this study [15].

Definition 10. ***S**emantic **T**ext **S**imilarity*
There are multiple kinds of STS task structures, due to efficiency they are often performed batch wise, for the pooled outputs of a BERT like encoder $u_{i,:} \in \mathbb{R}^{1\times d} \forall i = 1..n$ with batchsize n. In every batch exist sentence pairs that are similar or not. The following assumes pairs of similar sentences but this principle can be extended to groups or more differentiated gradients of similarity. For these vectors cosine similarity function is computed for every pair $p_{i,j} = (u_{i,:}, u_{j,:}), i \neq j$:

$$cos\_sim(u,v) = \frac{\langle u, v\rangle}{||u||||v||} \tag{3.12}$$

with the euclian norm $||\cdot||$ and the dot product $\langle\cdot,\cdot\rangle$. Every row of the matrix $p \in \mathbb{R}^{n\times n}$ is now treated as its own n-class classification problem in which $p_{i,j} \in [-1,1]$ is the predicted similarity of anchor $u_{i,:}$.[7]

all-MiniLM-L6-v2

all-MiniLM-L6-v2 [18] was used for this work for in part for similar reasons to those allready presented in the case of sentiment classification. Training data consists in large parts of short online sentences like reddit comments [7]. Additionally it brings a significant speed advantage compared to full BERT sized models since it is based on MiniLM-L6-H384-uncased, which implements the self attention knowledge distillation process as proposed by Wenhui Wang et. al.[18].

Definition 11. *Self Attention **K**nowledge **D**istillation*
Knowledge distillation refers to a training process in which during inference outputs of selected layers from a teacher model T get matched by a student Network with fewer layers. In the case of the 6 layered all-MiniLM-L6-v2 it machtes the computed attention outputs of every second layer of the 12 layered teacher MiniLM-L12-H384-uncased[17][8].

[6] In practical implementations so called padding tokens are added to the end of sequences during training to create sequences of fixed length but stay unattended as $-\infty_{1,:} \in \mathbb{R}^{1\times l}$ is added to the respective rows of QK^T before softmax is applied in self attention (attention masking)

[7] This method is often used due to the possibility of parallelizing many computations, a cross entropy loss function is often applied.

[8] MiniLM-L12-H384-uncased itself is a distilled model form the semi BERT like UniLM v2[1] which mainly differentiates itself by adding an attention mask $M_{i,j} = -\infty, : j \geq i \in 1..l$ to the encoding

3.3.1 BERTopic

Given sentence embeddings from a group of documents $D_1, D_2, ...$ represented by fixed length vectors $u \in \mathbb{R}^{1\times d}$ created from an SBERT like encoder it is possible to establish distance relations and therefor form clusters of semantically similar texts. Clustering in high dimensions is computationally expensive and conventional norms ($||\cdot||_i \forall i = 1..\infty$) might not capture local differences well due to the large ammount of contributing factors (vector elements). **BERTopic** [6] introduces a convenient and modular pipeline that allows users to cluster the embeddings of a chosen sentence transformer [9]

To combat the "curse of dimensionality" [6] a dimensionality reduction algorithm is employed. Specifically authors chose the UMAP algorithm over other well known methods like PCA for its ability to better preserve local variations in high dimensional data. This is facilitated by constructing local k nearest neighbor graphs and optimizing a lower dimensional representation of the embeddings so the graph relations are preserved. The clustering algorithm HDBSCAN takes advantage of this structure as it uses kNN distance to estimate local density based on mutual reachability. From this density the algorithm constructs a hierarchy of clusters, while points that can't be reliably attributed are labeled as outliers. This allows for the application on noisy, outlier rich datasets [11]. With the help of the word frequency based TF-IDF algorithm relevant topic descriptors are extracted form a concatenated version of all texts belonging to a topic. This results in a group of one or two word combinations which act as descriptors for their respective topics, simultaneously this word list reflects relevant and otherwise unusual terms related to the topic. [10]

3.3.2 Pre-Processing

To make data analysis more robust several pre-processing measures have been applied to the messages. To match uncased training conditions [18] [2] all characters have been made lowercase. Additionally url addresses have been replaced with spaces [2]. Further all double or multi spaces as well as line breaks have been replaced with a single space. To ensure a minimum of information messages with less than 60 characters or a numeric to alphanumeric ratio of >0.4 have been discarded. While earliest messages go back to before 2016 the analysis time frame has been restricted to 2019-12-09 until 2025-09-01.

---

of the second sequence in any pair $softmax(\frac{QK^T}{\sqrt{d}} + M)$

[9] First introduced with the SBERT framework, the algorithm is applied in this case to all-MiniLM-L6-v2 embeddings

[10] UMAP was initialized with n_neighbors=15,n_components=5,and cosine similarity metric. This yields a relatively fine clustering structure.
HDBSCAN was initialized with min_cluster_size=15, min_samples=5, euclidean metric and eom cluster selection. These settings were chosen to allow relatively small clusters to emerge while still meaningfully distinguishing outliers

4 Data Analysis

In the following chapter we will discuss the analysis of the data and introduce some statistical tools. First we will introduce Pearson Correlation and see how it can be applied to understand local behaviors over long time stretches. Then we will define a score to find significant behavioral similarities between trends.

4.1 Pearson Correlation

In principle the Pearson Correlation coefficient is derived from the idea of normalizing the covariance of two random variables with the respecive root of their variances $\rho_{X,Y} = \frac{Cov(X,Y)}{\sqrt{Var(X)}\sqrt{Var(Y)}}$. When applied to data we substitute the random variables with estimates of variance and covariance

Definition 12. *Sample Pearson Correlation*
For $(x_i, y_i)_{1..n} = (x, y) \in \mathbb{R}^n$ the Sample Pearson Correlation is defined as:

$$r_{x,y} = \frac{\sum_{i=1}^{n}(x_i - \frac{1}{n}\sum_{i=1}^{n} x_i)(y_i - \frac{1}{n}\sum_{i=1}^{n} y_i)}{\sqrt{\sum_{i=1}^{n}(x_i - \frac{1}{n}\sum_{i=1}^{n} x_i)^2}\sqrt{\sum_{i=1}^{n}(y_i - \frac{1}{n}\sum_{i=1}^{n} y_i)^2}} \tag{4.1}$$

This yields $r \in [-1, 1]$ where positive or negative values correspond to positive or negative linear correlation. Under the assumption of $\begin{pmatrix} X \\ Y \end{pmatrix}$ being jointly bivariate normal distribution with $\rho = 0$ we define the exact probability distribution of r.

$$f_{x,y}(r) = \frac{(1 - r^2)^{\frac{n}{2}-2}}{B(\frac{1}{2}, \frac{n}{2} - 1)} \tag{4.2}$$

By computing $p = 2\int_{|r|}^{1} f_{x,y}(h)dh$ and a p-Value that reflects the probability that the absolute value of an $r_{x',y'}$ drawn from two random normally distributed samples is higher than $r_{x,y}$.

While Pearson Correlation gives a good idea of linear dependencies any suitable dataset it is quite reductive and with increasing data size truly uncorrelated data pairs within the set can become invisible in a correlated majority or vice versa.
Since the reference dataset of google trends will be built from topic related keywords it is reasonable to expect that the selected google trends will not capture the topics complexity and yield relevant but tangential trends. For example a topic may be about a cat show in london and end up with the descriptors ['cat','london','show','cat show','london

show','london cat']. Over a long time the topic and trends might not be correlated but for a certain timeframe searches for 'london show' might be related to our cat show in london.
For this reason rolling Pearson Correlation is employed.

Definition 13. *Rolling Pearson Correlation*
For any point in the respective time series $(x_i, y_i), i \in \mathbb{Z}$ *we take the Pearson Correlation of* $\{(x_j, y_j) : j \in \{w/2 - i + 1, .., w/2 + i\}\}$ *with an even window length* w*.*

The algorithm applied in this work dynamically reduces the size of the correlation window when nearing the start or end of a timeseries to a minimum window length of 20 (asymmetrically around the respective end). To find significant correlations a threshold of r has to be defined, in this case the correlation is allowed to dip under the designated value to account for unrelated spikes in trends.

Definition 14. *Episode of correlation The following time interval* $[t_{start}, .., t_{stop}]$ *will be designated window of correlation if the following conditions are met:*

- $r(t_i)_{x,y}$ *does not drop under a set Value* $\alpha = 0.25$ *for more than one consecutive* t_i
- *A minimum Length of* $l \geq t_{stop} - t_{start}$
- $r(t_i)_{x,y}$ *does not change sign* $\forall t_i \in [t_{start}, .., t_{stop}]$
- *the conditions are met for at least 20 time steps*

To rank the significance of an episode rolling correlations a score is defined for any pair:

$$Score((x_i, y_i)_{i \in \{1..n\}}) = r_m \sqrt{L} * (1 + 0.25N) \tag{4.3}$$

With r_m *being the absolute median of* r *during the longest rolling correlation episode,* L *as the length of the longest episode and* N *as the total number of episodes.*

5 Results

Out of the collected Telegram messages 1645319 have been evaluated after pre-processing. From these messages all were analyzed with the RoBERTa based TweetEvaL hate and sentiment classifiers[2]. These respectively make a prediction about which sentiment a message might reflect. The messages get assigned probabilities to be classified to show: Joy, Optimism, Sadness or Anger and additionally a prediction is made if the regarding content is hateful. For the sake of efficiency and the prevalence of short training data for the classifier messages have been truncated to 512 characters during classification, which shortened 14.2 % of messages.
200000 messages have been sampled in a manor faithful to the prevalence of each channel in the total message amount and clustered with BERTopic as specified above. The embeddings of the remaining were classified by the so created model and assigned one of the 1244 determined topics.
Out of the sample 54.34% were classified as the outlier topic -1 in the sample, for the total preprocessed corpus this rises to 61.29%. [1] To gain a basic reflection of the public interest into prevalent topics in the studied communities, google trends for the topic descriptors from the 100 most frequent topics have been collected. Additionally a set of 20 Google search trends indicating economic insecurity [2] and 20 randomly generated search trends [3] have been selected.

[1] In a run with 100000 similarly selected samples the outlier rate lies by 55.3% in the sample and 63.31% in the preprocessed corpus.

[2] 'homes for sale', 'houses for sale', 'apartments for rent','mortgage rates', 'refinance mortgage', 'refinance calculator', 'rent assistance', 'eviction help', 'food bank near me', 'food stamps','bankruptcy', 'debt consolidation', 'foreclosure', 'unemployment benefits', 'jobless claims', unemployment office', 'payday loan', 'cash advance', 'title loan', 'loan for bad credit',

[3] creditableness, ungentleness, ordinar ,clouter, bloodstones, reprographic, interferometers, distinctions, unveilings, panzer, outgrowth, favus, sliding ,Herbert, subscriber ,trunk, dialling, cacodemon, overalled, het, vaporizer, foisted

	mean	ssd	le%25	le%50	le%75
messages per channel	12754.41	19911.46	1220	4212	14185
messages per topic	1322.60	28597.62	111	215	463
topics per channel	365.65	289.66	122	301	548

Tabelle 5.1: ssd is the standard sample deviation, for x_per_y: le%z means z% of y have less than cell-entry x

Tabelle 5.2: Topics of Interst

topic	Total	Representation	synth	$\frac{correctsynth}{falsesynth}$
-1	1008543	['and' 'the' 'to' 'of' 'in' 'is' 'we' 'this' 'are' 'that']	42	0.500
14	5344	['flat' 'the flat' 'earther' 'flat earther' 'flat earth' 'nasa' 'earth' 'tmetheflateartherr' 'earther tmetheflateartherr' 'moon']	10	0.400
17	4776	['mask' 'masks' 'wear' 'wearing' 'mask mandate' 'to wear' 'mandate' 'mask mandates' 'face masks' 'wear masks']	3	0.667
22	3706	['maricopa' 'arizona' 'maricopa county' 'audit' 'az' 'county' 'ballots' 'the arizona' 'election' 'the maricopa']	1	0.000
23	2907	['ballots' 'election' 'mailin' 'voting' 'machines' 'ballot' 'voter' 'elections' 'fraud' 'pennsylvania']	4	0.250
48	2130	['qanon' 'anons' 'drops' 'qs' 'the drops' 'movement' 'tippy' 'the qanon' 'exposedn' 'tippy top']	12	0.500
129	1141	['trafficking' 'human trafficking' 'child trafficking' 'child' 'human' 'trafficking is' 'children' 'sex trafficking' 'sex' 'child sex']	2	0.500
173	869	['autism' 'aluminum' 'vaccines' 'autismnn' 'rfk' 'cdc' 'and autism' 'rfk jr' 'autistic' 'cause']	2	0.500
252	609	['pcr' 'sarscov2' 'virus' 'pcr test' 'test' 'het' 'de' 'tests' 'viruses' 'pcr tests']	3	0.667
315	540	['omicron' 'variant' 'omicron variant' 'the omicron' 'booster' 'overton window' 'overton' 'vaccine' 'mild' 'vaccinated']	1	1.000
599	258	['mules' '2000 mules' '2000' 'dinesh' 'premiere' 'movie' 'film' 'dsouza' 'the film' 'the movie']	2	0.500

5.1 Core Topics

To identify topics related to the initially formulated question, different statements (synthetic statements) with relevant content are embedded like the full corpus and categorized into the existing topics. [4].

There are visible differences in the ratio of true and false statements embedded into each topic, nonetheless it is clear from the list of synthetic statements that even entirely contradictory statements can belong to the same topic. Due to the small sample size of synthetic statements these differences should not be weighed too heavily. A more representative view can be gained regarding the tone of posts as classified by the introduced RoBERTa based models. It seems that while out of the relevant topics number 23 has the largest fraction of messages classified as angry the flat earth related topic 14 draws by far the most joy 5.4. As seen in table 5.3 the noticeable correlation between angry sentiment and average reactions per message holds over the whole dataset. This reflects a broader trend noticed in social media environments.

[4]To avoid personal bias the relevant statements were created with GPT-5.2 [4]. The prompt is to create pairs of true and untrue sentences about: The 2020 election, Flat Earth, Covid 19 pandemic, Vaccination, Qanon

col1	col2	r	p
rel_anger	rel_reacts	0.413	0.000
rel_hate	rel_anger	0.247	0.000
rel_hate	rel_reacts	-0.002	0.947
rel_sadness	rel_reacts	-0.024	0.395
rel_hate	rel_sadness	-0.028	0.332
rel_joy	rel_optimism	-0.053	0.062
rel_optimism	rel_reacts	-0.100	0.000
rel_hate	rel_optimism	-0.131	0.000
rel_hate	rel_joy	-0.160	0.000
rel_anger	rel_sadness	-0.198	0.000
rel_sadness	rel_optimism	-0.209	0.000
rel_anger	rel_optimism	-0.341	0.000
rel_joy	rel_reacts	-0.347	0.000
rel_joy	rel_sadness	-0.387	0.000
rel_anger	rel_joy	-0.689	0.000

Tabelle 5.3: Pearson correlation of rel_reacts (average reactions per message) and rel_(sentiment/hate) per topic

For every topic the weekly message amount has been put into relation with all selected Google trends in rolling correlation windows of 26, 52 and 104 weeks. The weekly message count has been chosen to prevent overly noisy or heavily skewed behavior while still preserving distinctive temporal patterns. Disregarding the outlier topic the there are on average 512.29 messages per topic over 314 weeks. Correlation windows have been defined 14 and correlations have been ranked by the introduced score based on the median r and duration of the longest window as well as the amount of windows.

5.1.1 Flat Earth Theory

The Flat Earth Theory related topic 14 has the most joyous tone out of the presented topics and is within the top 10% of the most joyous topics overall. Over time its only strongly correlated over broad rolling windows with seemingly unrelated trends[5].

5.1.2 Covid19 Pandemic and Vaccine Skepticism

Within the topics 17,171,252 and 315 associated with Vaccination and the Covid19 Pandemic there is clear trend of strong to moderate correlation with public interest as reflected by google trends. An outlier is topic 173 which encompasses messages with a context of the cause off autism and vaccination.[6] While the topic seems to still have

[5]Trends created from descriptors of unrelated topics

[6]By individually sampling messages it has been verified that some of these messages allege a causal link.

topic	total_messages	rel_reacts	rel_hate	rel_anger	rel_optimism	rel_joy	rel_sadness	description
	73804.000	201.713	0.048	0.394	0.159	0.172	0.275	average over all topics
-1	1008543	228.878	0.061	0.434	0.142	0.252	0.171	['and' 'the' 'to' 'of' 'in' 'is' 'we' 'this' 'are' 'that']
14	5344	66.268	0.038	0.142	0.092	0.655	0.111	['flat' 'the flat' 'earther' 'flat earther' 'flat earth' 'nasa' 'earth' '[CHANNEL NAME REDACTED]' 'earthe [CHANNEL NAME REDACTED]' 'moon']
17	4776	149.851	0.037	0.527	0.096	0.131	0.246	['mask' 'masks' 'wear' 'wearing' 'mask mandate' 'to wear' 'mandate' 'mask mandates' 'face masks' 'wear masks']
22	3706	221.385	0.010	0.532	0.167	0.114	0.186	['maricopa' 'arizona' 'maricopa county' 'audit' 'az' 'county' 'ballots' 'the arizona' 'election' 'the maricopa']
23	2907	395.504	0.020	0.762	0.079	0.032	0.127	['ballots' 'election' 'mailin' 'voting' 'machines' 'ballot' 'voter' 'elections' 'fraud' 'pennsylvania']
48	2130	232.202	0.048	0.515	0.175	0.253	0.058	['qanon' 'anons' 'drops' 'qs' 'the drops' 'movement' 'tippy' 'the qanon' 'exposedn' 'tippy top']
129	1141	176.345	0.105	0.491	0.121	0.039	0.350	'human trafficking' 'child trafficking' 'child' 'human' 'trafficking is' 'children' 'sex trafficking' 'sex' 'child sex']
173	869	280.854	0.005	0.155	0.086	0.084	0.674	['autism' 'aluminum' 'vaccines' 'autismnn' 'rfk' 'cdc' 'and autism' 'rfk jr' 'autistic' 'cause']
171	699	136.608	0.050	0.219	0.202	0.089	0.491	['mrna' 'spike' 'protein' 'spike protein' 'mrna vaccines' 'vaccines' 'the mrna' 'the spike' 'cells' 'rna']
252	609	97.736	0.151	0.402	0.143	0.130	0.325	['pcr' 'sarscov2' 'virus' 'pcr test' 'test' 'het' 'de' 'tests' 'viruses' 'pcr tests']
315	540	85.111	0.081	0.217	0.230	0.078	0.476	['omicron' 'variant' 'omicron variant' 'the omicron' 'booster' 'overton window' 'overton' 'vaccine' 'mild' 'vaccinated']
599	258	362.473	0.000	0.419	0.097	0.395	0.089	['mules' '2000 mules' '2000' 'dinesh' 'premiere' 'movie' 'film' 'dsouza' 'the film' 'the movie']

Tabelle 5.4: rel_reacts are the average amount of all emoji-reactions per message for the given topic, rel_(sentiment/hate) are the fractions of messages classified as the given label for the specified topic

topic	google_trend	window	score	longest_episode_weeks	longest_ep_median(r)	frac_t_|r|>.25	sign_flip_count	episode_intervals
14	zoo	104	8.970	118	0.554	0.717	0	[['2020-05-17', '2022-08-14'], ['2022-11-06', '2024-08-11']]
14	sex	104	8.500	132	0.507	0.734	1	[['2020-06-28', '2023-01-01'], ['2023-06-25', '2024-07-21']]
14	eight	104	7.240	112	-0.486	0.539	2	[['2020-06-14', '2022-07-31']]
14	benjamin netanyahu	104	6.954	106	0.531	0.395	3	[['2020-06-07', '2022-06-12']]
14	three	104	6.794	125	-0.360	0.791	1	[['2020-07-12', '2022-11-27'], ['2023-05-07', '2024-08-18']]
14	israeli prime minister	104	6.760	106	0.531	0.364	2	[['2020-06-07', '2022-06-12']]
14	the pope	104	6.669	114	-0.495	0.380	2	[['2020-06-07', '2022-08-07']]
14	strip	104	6.516	114	0.488	0.380	0	[['2020-05-31', '2022-07-31']]
14	qsi	104	6.371	111	0.456	0.404	0	[['2020-06-21', '2022-07-31']]
14	the armed	104	6.364	118	-0.457	0.394	1	[['2020-06-21', '2022-09-18']]
14	flat earth	104	0.000	0	NaN	0.061	3	[]

Tabelle 5.5: Rolling correlation windows of the flat earth theory related topic

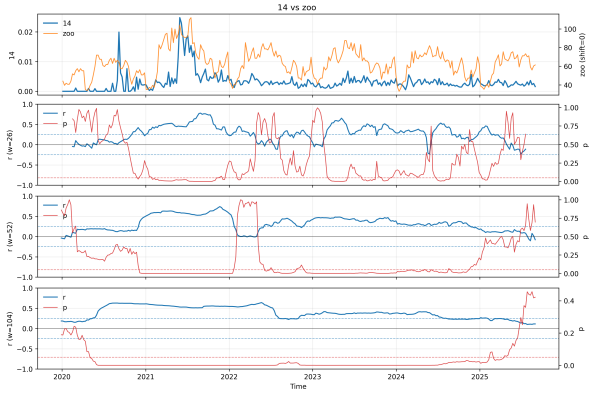

Abbildung 5.1: The Flat Earth related topic shows some correlations with events with strong seasonal components

some moderate windows of correlation with public interest in pope Francis and Fulton county frequent sign changes and behavior outside of correlation windows point to spurious correlations.

5.1.3 Q Anon Theory

For the Q Anon associated topic 48 we see strong correlation windows with public interest in vaccination and related searches for a time frame beginning in early 2020 up to march 2022. This correlation holds in the 26 week correlation window for extended periods around 2021 as well. Since the activity level of topic 48 rises for the first time in strong continuous correlation with the public interest in vaccines this could suggest a common driver behind the spike in public interest in vaccine and the mass spread of Q Anon related messages. Interestingly the google search trend for COVID shows weak to no correlation with a median $r = -0.11$ and no correlation windows (for none of the applied rolling correlation lengths). Topic 129 seems to correlate to public interest in media and figures associated with the political right. Additionally it correlates moderately with the google trend for q anaon.

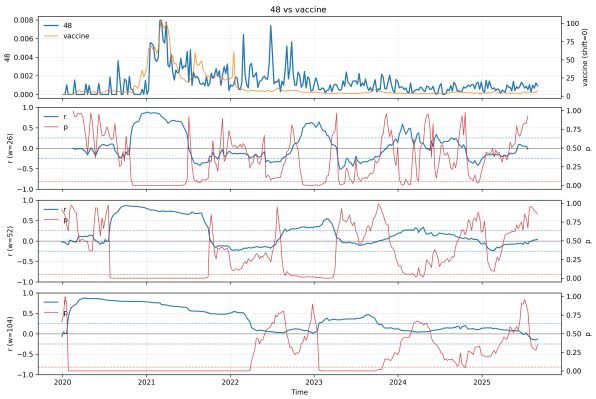

Abbildung 5.2: Q Anon related topic 48

topic	google_trend	window	score	longest_episode_weeks	longest_ep_median(r)	frac_t_|r|>.25	sign_flip_count	episode_intervals
17	mask mandate	104	10.475	218	0.468	0.966	0	[['2019-12-29', '2021-03-21'], ['2021-07-04', '2025-08-31']]
17	covid	104	9.467	171	0.506	0.865	0	[['2019-12-29', '2021-02-14'], ['2021-09-12', '2024-12-15']]
17	masks	104	8.795	247	0.427	0.855	0	[['2019-12-29', '2024-09-15']]
17	mask	104	8.412	175	0.502	0.593	8	[['2019-12-29', '2023-04-30']]
17	homes for sale	104	8.309	182	0.523	0.653	1	[['2019-12-29', '2023-06-18']]
17	houses for sale	104	8.200	179	0.515	0.599	5	[['2019-12-29', '2023-05-28']]
17	mandate	104	7.700	163	0.450	0.700	1	[['2021-08-01', '2024-09-08']]
17	mask mandates	104	7.380	89	0.528	0.641	0	[['2021-07-04', '2023-03-12'], ['2023-07-30', '2024-09-15']]
17	refinance mortgage	104	7.195	99	0.524	0.586	3	[['2021-04-04', '2023-02-19'], ['2024-03-24', '2025-07-06']]
17	mask mandate	52	7.143	84	0.442	0.879	0	[['2021-02-07', '2022-09-11'], ['2022-10-23', '2024-03-17'], ['2024-07-14', '2025-07-27']]
171	julian calendar	52	7.040	104	0.543	0.538	3	[['2021-07-11', '2023-07-02'], ['2024-07-07', '2025-07-06']]
171	injured	104	5.877	104	0.454	0.357	12	[['2022-01-09', '2023-12-31']]
171	injured	52	5.684	56	0.625	0.279	14	[['2022-06-19', '2023-07-09']]
171	calendar	52	5.279	55	0.582	0.259	4	[['2022-06-26', '2023-07-09']]
171	speaker	104	5.217	104	0.407	0.387	3	[['2022-01-09', '2023-12-31']]
171	injured	26	5.058	59	0.526	0.482	14	[['2022-04-03', '2023-05-14']]
171	good morning	52	4.962	82	-0.438	0.401	9	[['2020-11-15', '2022-06-05']]
171	the pope	104	4.947	83	0.452	0.276	4	[['2022-06-12', '2024-01-07']]
171	speaker	52	4.937	52	0.555	0.286	11	[['2022-07-10', '2023-07-02']]
171	jair bolsonaro	104	4.889	105	0.398	0.384	3	[['2022-01-09', '2024-01-07']]
173	pope francis	104	7.089	71	0.665	0.239	11	[['2024-04-28', '2025-08-31']]
173	francis	104	6.624	71	0.610	0.239	10	[['2024-04-28', '2025-08-31']]
173	jesus	104	6.618	72	0.581	0.242	7	[['2024-04-21', '2025-08-31']]
173	fulton	52	6.606	79	0.494	0.337	11	[['2019-12-29', '2021-06-27']]
173	fulton county	52	6.582	53	0.844	0.340	5	[['2019-12-29', '2020-12-27']]
173	the vatican	104	6.396	88	0.551	0.296	10	[['2023-12-31', '2025-08-31']]
173	fulton	104	6.245	64	0.630	0.293	3	[['2019-12-29', '2021-03-14']]
173	christ	104	6.146	72	0.548	0.242	9	[['2024-04-21', '2025-08-31']]
173	fulton county	104	6.093	64	0.587	0.253	5	[['2019-12-29', '2021-03-14']]
173	vatican	104	5.893	88	0.495	0.296	4	[['2023-12-31', '2025-08-31']]
252	covid	104	7.473	179	0.341	0.609	1	[['2021-04-11', '2024-09-08']]
252	coronavirus	104	7.242	67	0.612	0.756	2	[['2020-02-09', '2021-04-11'], ['2021-10-24', '2023-01-29']]
252	masks	104	7.229	118	0.606	0.424	4	[['2021-08-08', '2023-11-05']]
252	the vaccine	104	6.999	162	0.407	0.603	6	[['2020-01-19', '2023-02-19']]
252	vaccine	104	6.974	188	0.373	0.630	3	[['2020-01-19', '2023-08-20']]
252	cdc	104	6.551	116	0.464	0.397	1	[['2021-06-13', '2023-08-27']]
252	vaccines	104	6.409	159	0.426	0.532	5	[['2019-12-29', '2023-01-08']]
252	mask	104	6.065	81	0.554	0.306	2	[['2021-08-08', '2023-02-19']]
252	rent assistance	104	6.027	75	0.462	0.589	1	[['2020-01-05', '2021-04-25'], ['2021-09-12', '2023-02-12']]
252	refinance calculator	104	5.945	85	0.555	0.276	5	[['2021-08-15', '2023-03-26']]
315	covid	104	8.121	190	0.427	0.636	3	[['2021-01-10', '2024-08-25']]
315	maxwell	104	7.549	110	0.660	0.374	3	[['2020-12-06', '2023-01-08']]
315	aid	104	7.378	111	0.548	0.461	1	[['2020-12-06', '2023-01-15']]
315	covid	52	7.214	139	0.464	0.560	3	[['2021-07-04', '2024-02-25']]
315	ghislaine	104	7.191	106	0.639	0.360	7	[['2021-01-03', '2023-01-08']]
315	ghislaine maxwell	104	7.172	106	0.641	0.360	5	[['2021-01-03', '2023-01-08']]
315	epsteins	104	7.087	75	0.668	0.249	3	[['2021-08-08', '2023-01-08']]
315	santa claus	52	6.879	54	0.768	0.313	7	[['2021-06-13', '2022-06-19']]
315	elephant	52	6.852	55	0.749	0.333	4	[['2021-06-06', '2022-06-19']]
315	christmas	52	6.672	55	0.741	0.299	7	[['2021-06-06', '2022-06-19']]

Tabelle 5.6: Rolling correlation episodes of vaccine and pandemic related topics

topic	google_trend	window	score	longest_episode_weeks	longest_ep_median(r)	frac_t_|r|>.25	sign_flip_count	episode_intervals
48	vaccine	104	9.044	115	0.704	0.508	6	[['2020-01-26', '2022-04-03']]
48	vaccines	104	8.705	114	0.675	0.397	5	[['2020-01-26', '2022-03-27']]
48	the vaccine	104	7.793	114	0.615	0.492	8	[['2020-01-26', '2022-03-27']]
48	vaccine	52	7.132	63	0.727	0.350	13	[['2020-07-26', '2021-10-03']]
48	vaccines	52	6.759	62	0.668	0.279	12	[['2020-07-26', '2021-09-26']]
48	rent assistance	104	5.962	116	0.357	0.562	4	[['2020-02-09', '2021-02-07'], ['2022-04-24', '2024-07-07']]
48	the vaccine	52	5.886	62	0.575	0.323	15	[['2020-07-26', '2021-09-26']]
48	apartments for rent	104	5.876	84	0.447	0.603	3	[['2020-03-01', '2021-10-03'], ['2022-10-02', '2023-12-24']]
48	sound of freedom	104	5.820	112	0.448	0.593	3	[['2020-03-08', '2022-04-24']]
48	tax	104	5.749	104	0.474	0.468	2	[['2020-02-23', '2022-02-13']]
48	qanon	52	0.000	0	NaN	0.118	3	[]
129	sound of freedom	104	7.677	108	0.567	0.512	6	[['2022-07-10', '2024-07-28']]
129	caviezel	104	7.308	108	0.540	0.364	5	[['2022-07-10', '2024-07-28']]
129	bannon	104	7.255	86	0.623	0.290	7	[['2019-12-29', '2021-08-15']]
129	steve bannon	104	7.239	86	0.624	0.290	7	[['2019-12-29', '2021-08-15']]
129	sound of freedom	52	6.632	73	0.629	0.419	8	[['2022-09-11', '2024-01-28']]
129	trafficking	104	6.510	106	0.481	0.377	4	[['2022-07-24', '2024-07-28']]
129	qanon	104	6.507	87	0.537	0.456	1	[['2019-12-29', '2021-08-22']]
129	caviezel	52	6.331	56	0.746	0.263	8	[['2023-01-08', '2024-01-28']]
129	qanon	52	6.289	57	0.627	0.382	3	[['2020-01-26', '2021-02-21']]
129	bannon	52	6.262	52	0.690	0.239	11	[['2020-02-23', '2021-02-14']]

Tabelle 5.7: Rolling correlation episodes of Q anon related topics

5.1.4 Contested 2020 Election

From the election related topics topic 23 shows the clearest correlation with general interest in elections, as well as strong political motivators. Topic 22 strongly reflects a public interest in audits around mid 2021, the year after the election. Topic 599 seems unrelated to public interest in the election and the collected google trends of the 100 most prevalent topics, but due to the relative small message volume a definite stance is hard to take.

5.2 Sentiment Analysis

In the same way the presented methodology lets us detect possible common drivers of public interest and topic prevalence in the given dataset, we can set our external google trend data in relation to the detected sentiment.
From the data it becomes apparent that out of all factors economic insecurity seems to be most correlated with angry sentiment over larger time windows. This correlation remains significant with shorter rolling correlation windows (bankruptcy remains top correlation with window size set to w=26). Vice versa the joy sentiment seems to have several significant negative correlations with economic indicators. Sadness seems to only have moderate to little correlation with relevant public interests or economic indicators. Optimism also is somewhat impacted by economic indicators but seems mostly correlated with discussed topics.

topic	google_trend	window	score	longest_episode_weeks	longest_ep_median(r)	frac_t_|r|>.25	sign_flip_count	episode_intervals
22	audit	104	8.688	127	0.652	0.481	3	[['2020-04-26', '2022-09-25']]
22	president donald	104	8.610	80	0.495	0.673	3	[['2020-04-26', '2021-06-20'], ['2022-01-30', '2023-08-06'], ['2024-06-30', '2025-08-31']]
22	paul	104	7.943	91	0.688	0.441	2	[['2023-12-10', '2025-08-31']]
22	president donald	52	7.362	53	0.799	0.536	7	[['2022-05-22', '2023-05-21']]
22	zoo	104	7.359	113	0.548	0.377	4	[['2020-05-03', '2022-06-26']]
22	infowars	104	7.243	94	0.616	0.424	10	[['2023-11-19', '2025-08-31']]
22	audit	52	7.091	102	0.580	0.471	11	[['2020-05-17', '2022-04-24']]
22	paul	52	6.828	53	0.742	0.303	12	[['2024-05-19', '2025-05-18']]
22	apartments for rent	104	6.437	86	0.453	0.549	6	[['2020-05-10', '2021-12-26'], ['2024-07-28', '2025-08-31']]
22	god	52	6.416	63	0.657	0.418	11	[['2022-05-22', '2023-07-30']]
23	voter	104	10.496	96	0.716	0.569	0	[['2020-09-13', '2021-11-21'], ['2023-11-05', '2025-08-31']]
23	roe	104	9.999	95	0.812	0.320	3	[['2023-11-12', '2025-08-31']]
23	abortion	104	9.691	96	0.796	0.323	1	[['2023-11-05', '2025-08-31']]
23	president donald	52	9.514	53	0.867	0.536	13	[['2022-05-22', '2023-06-11'], ['2024-05-12', '2025-05-11']]
23	maga	104	9.384	100	0.620	0.653	0	[['2019-12-29', '2021-11-21'], ['2023-11-12', '2025-08-31']]
23	ballots	104	9.362	168	0.745	0.606	0	[['2022-06-19', '2025-08-31']]
23	election	104	9.298	168	0.753	0.572	0	[['2022-06-19', '2025-08-31']]
23	elections	104	9.258	166	0.754	0.562	0	[['2022-07-03', '2025-08-31']]
23	fraud	104	9.105	105	0.717	0.515	0	[['2019-12-29', '2021-12-26']]
23	roe wade	104	9.009	95	0.744	0.320	3	[['2023-11-12', '2025-08-31']]
599	tehran	104	8.357	113	0.654	0.469	2	[['2021-05-16', '2023-07-09']]
599	tehran	52	6.482	63	0.654	0.310	2	[['2021-11-07', '2023-01-15']]
599	cardinal	104	5.478	110	0.410	0.510	4	[['2021-04-25', '2023-05-28']]
599	gaza	52	5.463	76	0.534	0.338	5	[['2022-12-18', '2024-05-26']]
599	hamas	52	5.193	56	0.580	0.389	2	[['2023-04-16', '2024-05-05']]
599	benjamin netanyahu	52	5.159	66	0.528	0.405	2	[['2023-01-08', '2024-04-07']]
599	israel	52	5.154	56	0.570	0.254	6	[['2023-04-16', '2024-05-05']]
599	gaza strip	52	5.081	58	0.569	0.341	4	[['2023-04-16', '2024-05-19']]
599	la county	52	5.040	56	0.532	0.385	2	[['2021-11-14', '2022-12-04']]
599	la county	26	4.941	53	0.544	0.503	1	[['2022-02-06', '2023-02-05']]

Tabelle 5.8: Rolling correlation episodes for election related topics

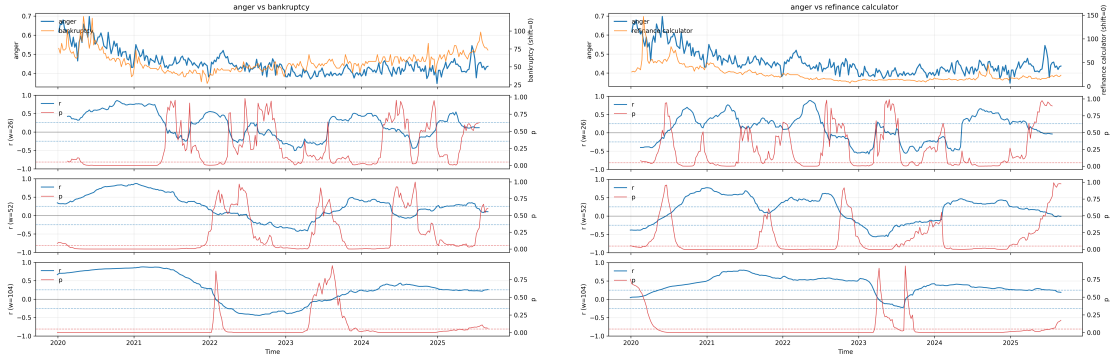

Abbildung 5.3: Angry Sentiment shows large correlation windows with indicators of economic insecurity

topic	google_trend	window	score	longest_episode_weeks	longest_ep_median(r)	frac_t_|r|>.25	sign_flip_count	episode_intervals
anger	bankruptcy	104	11.373	108	0.797	0.727	2	[['2019-12-29', '2022-01-16'], ['2022-03-27', '2023-04-02']]
anger	refinance calculator	104	10.117	149	0.545	0.798	2	[['2020-05-17', '2023-03-19'], ['2023-10-01', '2025-03-02']]
anger	refinance mortgage	104	9.304	143	0.499	0.811	2	[['2020-06-07', '2023-02-26'], ['2023-10-01', '2025-01-26']]
anger	foreclosure	104	9.030	134	0.715	0.569	2	[['2019-12-29', '2022-07-17']]
anger	ai	104	8.876	160	-0.466	0.859	4	[['2020-05-24', '2023-06-11'], ['2023-11-05', '2025-08-31']]
anger	distinctions	104	8.752	114	0.725	0.448	2	[['2019-12-29', '2022-02-27']]
anger	mask	104	7.707	161	0.520	0.569	2	[['2020-07-19', '2023-08-13']]
anger	unemployment office	104	7.632	119	0.422	0.700	1	[['2020-05-31', '2022-09-04'], ['2024-03-17', '2025-08-31']]
anger	the vaccine	104	7.630	79	-0.609	0.556	3	[['2019-12-29', '2021-06-27'], ['2021-10-10', '2022-12-18']]
anger	crypto	104	7.625	92	-0.551	0.542	6	[['2019-12-29', '2021-09-26'], ['2022-02-27', '2023-05-14']]
joy	ai	104	10.905	133	0.673	0.761	2	[['2021-04-11', '2023-10-22'], ['2024-02-11', '2025-08-31']]
joy	refinance mortgage	104	10.409	163	-0.532	0.902	2	[['2019-12-29', '2023-02-05'], ['2023-10-29', '2025-06-22']]
joy	masks	104	10.248	208	-0.584	0.697	3	[['2019-12-29', '2023-12-17']]
joy	ai is	104	10.218	117	0.648	0.847	2	[['2021-07-25', '2023-10-15'], ['2024-02-11', '2025-08-31']]
joy	unemployment office	104	10.000	162	-0.515	0.747	9	[['2019-12-29', '2023-01-29'], ['2024-05-12', '2025-07-13']]
joy	mask	104	9.879	202	-0.562	0.744	1	[['2019-12-29', '2023-11-05']]
joy	refinance calculator	104	9.686	157	-0.538	0.795	2	[['2020-03-22', '2023-03-19'], ['2023-11-26', '2025-05-25']]
joy	department of	104	9.577	115	-0.713	0.697	4	[['2019-12-29', '2022-03-06'], ['2024-02-25', '2025-08-03']]
joy	department	104	9.504	141	-0.592	0.875	0	[['2019-12-29', '2022-09-04'], ['2024-03-17', '2025-08-31']]
joy	the vaccine	104	9.420	126	-0.582	0.852	1	[['2021-07-11', '2023-12-03'], ['2024-03-03', '2025-07-27']]
sadness	aid	52	6.803	135	0.460	0.468	14	[['2019-12-29', '2022-07-24']]
sadness	bankruptcy	104	6.533	109	-0.507	0.485	0	[['2019-12-29', '2022-01-23']]
sadness	distinctions	104	6.468	109	-0.508	0.411	2	[['2019-12-29', '2022-01-23']]
sadness	aid	104	6.274	158	0.374	0.529	4	[['2019-12-29', '2023-01-01']]
sadness	shot	52	6.109	92	0.511	0.337	6	[['2020-10-11', '2022-07-10']]
sadness	bankruptcy	52	6.105	131	-0.393	0.495	9	[['2019-12-29', '2022-06-26']]
sadness	border	26	6.009	59	-0.626	0.468	13	[['2021-06-27', '2022-08-07']]
sadness	kjv	104	5.932	75	0.585	0.300	4	[['2019-12-29', '2021-05-30']]
sadness	the border	26	5.867	59	-0.637	0.458	17	[['2021-07-04', '2022-08-14']]
sadness	fraud	52	5.866	86	-0.409	0.505	8	[['2020-11-08', '2022-06-26'], ['2024-05-12', '2025-07-20']]
optimism	the vaccine	104	10.169	158	0.569	0.855	0	[['2019-12-29', '2023-01-01'], ['2023-09-10', '2025-07-27']]
optimism	crypto	104	9.273	152	0.523	0.815	4	[['2019-12-29', '2022-11-20'], ['2023-12-10', '2025-08-31']]
optimism	vaccines	104	9.135	158	0.496	0.862	0	[['2019-12-29', '2023-01-01'], ['2023-09-10', '2025-08-10']]
optimism	files	104	9.036	154	-0.498	0.774	1	[['2019-12-29', '2022-12-04'], ['2024-02-04', '2025-07-27']]
optimism	text	104	8.519	92	0.540	0.758	1	[['2020-03-15', '2021-06-13'], ['2021-06-27', '2022-10-30'], ['2023-11-19', '2025-08-17']]
optimism	intelligence	104	8.512	89	0.533	0.727	2	[['2020-04-05', '2021-04-11'], ['2021-07-11', '2022-12-04'], ['2023-11-19', '2025-07-27']]
optimism	bankruptcy	104	8.174	100	-0.601	0.690	3	[['2019-12-29', '2021-11-21'], ['2024-03-03', '2025-07-06']]
optimism	ai	104	8.129	92	0.613	0.646	7	[['2020-05-31', '2021-06-20'], ['2023-11-19', '2025-08-17']]
optimism	xrp	104	7.972	94	0.584	0.811	0	[['2019-12-29', '2021-10-03'], ['2023-11-19', '2025-08-31']]
optimism	vaccine	104	7.844	158	0.520	0.795	1	[['2019-12-29', '2023-01-01']]

Tabelle 5.9: Top rolling correlation episodes for sentiments

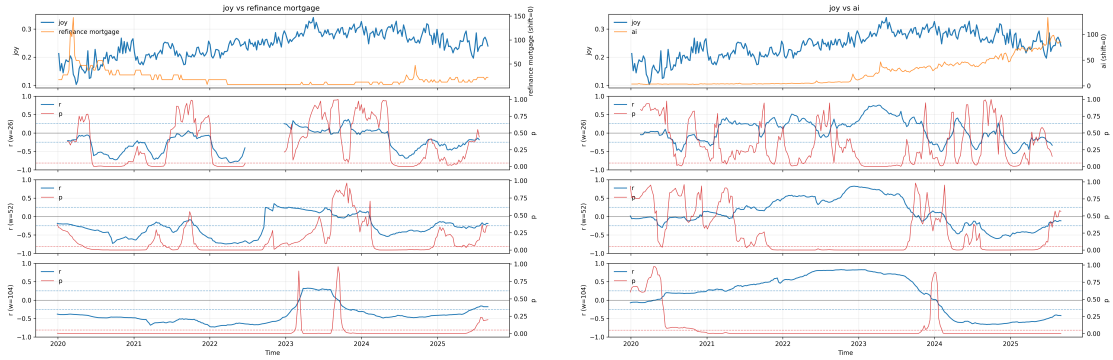

Abbildung 5.4: Due to constant stretches in the refinance mortgages trend Pearson Correlation cant be computed everywhere.

5.3 Summary

By leveraging modern natural language processing techniques this work shows differences in the prevalence of fringe beliefs and related public interest depend on the regarded topic. While the discussion of flat earth theory seems to show some fluctuations akin to seasonal interests [7] it is uncorrelated with public interest in the topic. Most discussion of the covid19 epidemic and vaccination show context dependent behavior, but conversations that relate to a topic that encompasses both the cause of autism and vaccines are seemingly uncorrelated to public interest in vaccination. In the Q Anon related topics there occurs a similar split: one topic correlates with associated media while the other might initially share a common driver with public interests but sustains itself independently after. The more frequent election related topics seem to be clearly related to public interests.
Overall this work suggests that there are topic dependent factors that modulate how public interest drives the discussion of fringe ideas.
Additionally this work shows empirical evidence that the prevalence of sentiment classified as angry is strongly connected to factors of economic insecurity. It is suggested that economic insecurity should be reduced for a more equitable and less angry future.

[7] longterm trends correlate to googlesearches for Zoo

Literatur

[1] Hangbo Bao u. a. *UniLMv2: Pseudo-Masked Language Models for Unified Language Model Pre-Training*. 2020. arXiv: 2002.12804 [cs.CL].

[2] Francesco Barbieri u. a. *TweetEval: Unified Benchmark and Comparative Evaluation for Tweet Classification*. 2020. arXiv: 2010.12421 [cs.CL].

[3] Jacob Devlin u. a. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL].

[4] DuckDuckGo. *Duck.ai (GPT-5.2) chat transcript*. Large language model output; accessed 2026-03-17. 17. März 2026. URL: https://duck.ai.

[5] Google. *Google Trends*. https://trends.google.com/trends/. Accessed 2026-01; filters: US; time range: 2019.12.29-2025.09.01.

[6] Maarten Grootendorst. *BERTopic: Neural topic modeling with a class-based TF-IDF procedure*. 2022. arXiv: 2203.05794 [cs.CL].

[7] Matthew Henderson u. a. *A Repository of Conversational Datasets*. 2019. arXiv: 1904.06472 [cs.CL].

[8] Contribution from ïliaschalkidis".

[9] Yinhan Liu u. a. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. 2019. arXiv: 1907.11692 [cs.CL].

[10] van Loon A Giorgi S Willer R et al. „Negative associations in word embeddings predict anti-Black bias across regions—but only via name frequency“. In: *Proceedings of the International AAAI Conference on Web and Social Media* (2022).

[11] Claudia Malzer und Marcus Baum. „A Hybrid Approach To Hierarchical Density-based Cluster Selection“. In: *2020 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*. IEEE, Sep. 2020, S. 223–228. DOI: 10.1109/mfi49285.2020.9235263.

[12] Vincenzo Imperati Massimo La Morgia Alessandro Mei Alberto Maria Mongardini und Francesco Sassi. „"The Conspiracy Money Machine: Uncovering Telegram's Conspiracy Channels and their Profit Model."“ In: *In 34th USENIX Security Symposium (USENIX Security 25), pp. 5229-5246. 2025* ().

[13] Garg Nikhil u. a. „Word embeddings quantify 100 years of gender and ethnic stereotypes“. In: *Proceedings of the National Academy of Sciences* 115.16 (Apr. 2018). DOI: 10.1073/pnas.1720347115.

[14] Nils Reimers und Iryna Gurevych. *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks*. 2019. arXiv: 1908.10084 [cs.CL].

[15] *SBERT*.

[16] Ashish Vaswani u. a. „Attention Is All You Need“. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2017. DOI: 10.48550/arXiv.1706.03762.

[17] Wenhui Wang u. a. *MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers*. 2020. arXiv: 2002.10957 [cs.CL].

[18] Wenhui Wang u. a. *MiniLMv2: Multi-Head Self-Attention Relation Distillation for Compressing Pretrained Transformers*. 2021. arXiv: 2012.15828 [cs.CL].

[19] Xiangyu Wang und Mizuho Iwaihara. „Integrating RoBERTa Fine-Tuning and User Writing Styles for Authorship Attribution of Short Texts“. In: *Web and Big Data*. Hrsg. von Leong Hou U u. a. Cham: Springer International Publishing, 2021, S. 413–421.