Project Report

Topic 1 • Content homophily in a real social network

Online Social Networks and Media • Spring 2022

Giannis Tsimarakis

Abstract

The subject of this study is to measure content homophily in communities of Twitter users affiliated with some chosen Ukrainian and Russian news sources. The dataset used consists of tweets posted during the ongoing war in Ukraine.

We create a vector representation for every examined user based on their posted content, as Doc2Vec text embeddings. We examine the similarities between user vectors in the context of each ego network centered around the Twitter users corresponding to each of the chosen newspapers; both intra-network and inter-network user pairs are examined, with network implementations based on various user interactions.

The conducted experiments in most cases do not offer much meaningful results. Nevertheless, we see some potential in getting more interesting similarity patterns, given some adjustment on the sampling of the users and tweets examined.

1. Introduction

1.1. Purpose

The main purpose of this project is to experiment on a real social network and measure content homophily. Homophily, in this context, can be defined as the tendency of connected nodes to post similar content. We have based our study on the *Twitter* platform and a collection of Tweets.

1.2. Methodology

For the calculation of the similarity between user content, we need a way to construct a vector representation of the text included in their Tweets, also known as text embeddings. For this purpose, we use the Doc2Vec algorithm [2] and a model specifically trained using the dataset under study.

Then, for a number of ego networks centered around a selected set of users, we calculate and report on the cosine similarities between various user pairs.

1.2. Document Layout

This document is laid out as follows:

Chapter 1	provides this introduction.
Chapter 2	provides information about the data used in this study.
Chapter 3	provides information about the construction of the text embeddings and the Doc2Vec model used.
Chapter 4	describes the obtained results regarding similarity calculations.
Chapter 5	provides our conclusions of the study.
Appendix A	lists the references used.
Appendix B	contains an IDEF1X compliant Data Model that the Database implementation was based on; it is given in Attribute Level detail.
Appendix C	provides information about the folder and file structure of the codebase.

2. Dataset

2.1. Description

The dataset used in this study is based on a collection of Tweet IDs given in [1]. The referenced Tweets were posted in the period between 21 February 2022 to 11 May 2022 and their content is related to the war in Ukraine.

2.2. Data Collection

Given that the initial collection only contained Tweet IDs, we proceeded with querying Twitter API for the full Tweet information. A library called *tweepy* is used, which provides some convenient functionality related to this process.

The query is parameterized so that the returned results include information about any referenced Tweets of each initial Tweet:

- the source Tweet if the Tweet was a Retweet
- the replied-to Tweet if the Tweet was a Reply
- the quoted Tweet if the Tweet was a Quote

Starting from the initial collection of Tweets, we recursively extract any Tweets further referenced by those referenced Tweets. For example, for a Tweet that was a Reply, we extract the whole branch of the discussion tree up to the root Tweet.

For all individual Tweets, the results contain information regarding:

- the Author of the Tweet
- the Language of the Tweet, if Twitter was able to identify it
- the Hashtags used in the Tweet
- the user Mentions included in the Tweet
- URLs included in the Tweet
- additional information called Entity Annotations, which are assigned to a Tweet by Twitter based on what is mentioned in the Tweet text. Currently available annotations are of four types; Person; Place; Product; Organization; Other.

Despite initial planning, for the following experiments we used the full Tweet text only and none of the additional information or individual attributes.

2.3. Storage

For easier access, the data acquired is stored in a SQLite database. We query the database:

- to provide some initial statistics.
- to stream the corpus of tweets for the training of the Doc2Vec model.
- to extract the edge information and construct the networks for the various user interactions.

Since the database is not modified afterwards, when the extraction of Tweets is finished we compute some summary information regarding the Tweet count of each user per language. This significantly improves the execution times in the processing that follows.

2.4. Statistics

2.4.1. Tweet Statistics

We collected a total of 41.770.356 distinct Tweet IDs, distinguished in:

- 23.165.040 Retweets
- 18.605.316 Tweets that are not Retweets, and out of which:
    - 2.967.887 are Quotes
    - 10.262.499 are Replies

The majority of the Tweets are written in English. More specifically, the total Tweet count per language, for the top-5 most used languages, is as follows:

English	12.017.538	64.59 %
Unidentified	995.044	5.35 %
Spanish	871.341	4.68 %
German	726.922	3.91 %
French	650.859	3.50 %

For this study we only dealt with the English language Tweets.

The collected Tweets are posted by a total of 6.722.700 distinct authors, and 2.198.270 of them have at least one Tweet posted in the English language.

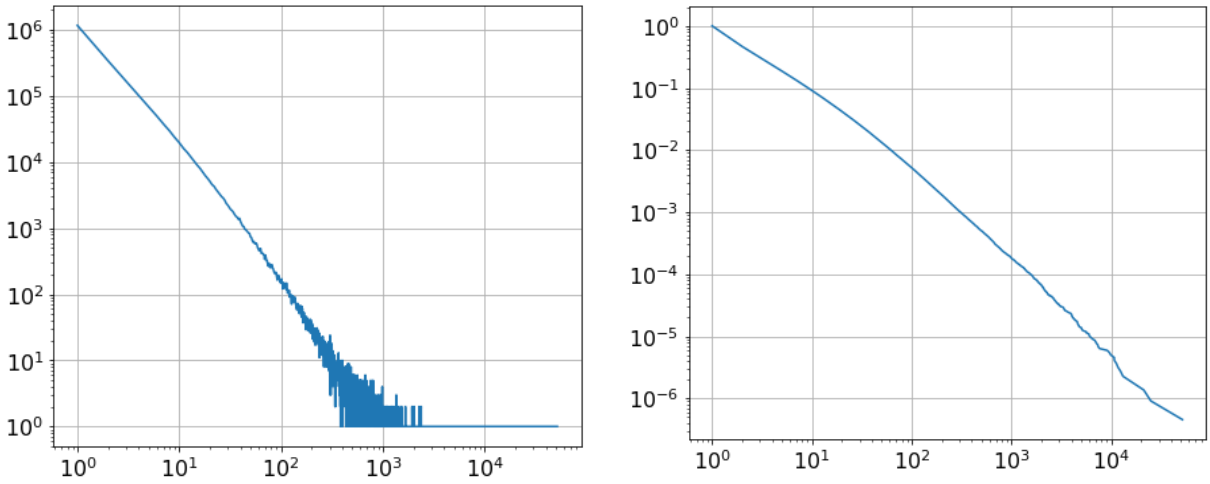

Fig. 1. Log-log plot of the histogram (left) of Tweet counts with respect to the number of authors with that specific tweet count, and the cumulative distribution (right)

In Figure 1, we see that the tweet count distribution clearly follows a power law where most of the users in our dataset have posted a small number of tweets, while a small fraction of them have much bigger activity.

2.4.2. Network Statistics

The networks we experimented with are centered at five ego-nodes, which correspond to the following english-language Ukrainian and Russian newspapers:

- Kyiv Post
- The Kyiv Independent
- The Moscow Times
- Sputnik
- RT

Due to Twitter API usage limitations, we were unable to acquire any following/follower relationships between users. The interactions between users were used as a substitute. That is, for two distinct users A and B, we extracted four types of edges between A and B as:

- A retweets B
- A quotes B
- A replies to B
- A mentions B

The following table shows the number of nodes and edges for the combinations of ego networks and edge types:

	Retweet	Quote	Reply	Mention	
Kyiv Post	7087	1526	1135	2252	**Nodes**
	39269	5223	1343	8069	**Edges**
The Kyiv Independent	70538	17932	21199	36705	**Nodes**
	756833	103631	50253	193454	**Edges**
The Moscow Times	1500	471	484	958	**Nodes**
	5955	1920	554	1683	**Edges**
Sputnik	1007	308	367	1113	**Nodes**
	1955	581	394	2446	**Edges**
RT	6782	2535	4611	9664	**Nodes**
	19359	5553	7350	37453	**Edges**

Table 1. Number of nodes and edges of the different networks.

3. Text Embeddings

To get the vector representations of Tweets, we used an implementation of the Doc2Vec [2] algorithm, included in the *gensim* library.

Training a Doc2Vec model is done by supplying the documents in the corpus, accompanied by a tag for each document. As a result, you get a vector representation for each tag used.

In our case, we supplied the individual Tweets as documents, tagging them with the Account ID of the Tweets' authors. That is, we get a vector representation for every Account, based on their Tweets.

3.1. Preprocessing & vocabulary building

- For tokenizing the Tweets, we tried various approaches on stopword removal and also experimented with removing user Mentions and URLs from the Tweet's text.
- We did not perform any lemmatization or other preprocessing techniques.
- During the vocabulary building phase of Doc2Vec, we tried various values for the minimum allowed term frequency.

The different tuning approaches did not provide any significant differences in the results.

3.2. Model Training

The models were always trained using the Distributed Bag-of-words algorithm for 10 epochs.

For the training parameters, we only experimented with some different vector sizes in (50, 100, 200), again with more or less the same results.

4. Experiments

In the following, we provide the results of a model with a vector size of 200, trained on a dataset simply tokenized without any stopword removal.

For the similarity measurements between vectors, we used the Cosine Similarity.

4.1. Ego node similarities

For an initial assessment of the trained model, we calculated the similarities between the chosen ego nodes. The results are shown in Table 1. As expected, we see that:

- The two Russian affiliated media, *Sputnik* and *RT* are almost identical with a similarity score of 0.99.
- The two Ukrainian newspapers, *Kyiv Post* and *The Kyiv Independent* are also very similar with a score of 0.84.
- The similarities between the Russian and Ukrainian media is low in the range of 0.26-0.33.
- Regarding *The Moscow Times*, a russian-originated newspaper that is currently banned in Russia, we see a much bigger similarity with the Ukrainian newspapers (0.78-0.82) than with the Russian ones (0.32)

Further tests between Organizations and Political figures of different affiliations have given results consistent with the above.

Although this is not a proof of validity, these results give some confirmation that the vectors obtained from the trained model can be effective in measuring the similarity between accounts.

	Kyiv Post	**The Kyiv Independent**	**The Moscow Times**	**Sputnik**	**RT**
Kyiv Post	1	0.84	0.78	0.27	0.26
The Kyiv Independent	0.84	1	0.82	0.33	0.31
The Moscow Times	0.78	0.82	1	0.32	0.32
Sputnik	0.27	0.33	0.32	1	0.99
RT	0.26	0.31	0.32	0.99	1

Table 2. Cosine Similarity matrix between the vectors of Twitter users chosen as the ego nodes.

4.2. Network similarities

We proceeded with calculating the mean similarities of node pairs in the different network types.

For intra-network node pairs, we calculated the mean similarities:

- of all node pairs of each network
- of node pairs between connected nodes of each network
- of node pairs between unconnected nodes of each network

For the similarities between the different ego networks, we calculated the mean similarity of every inter-network node pair.

In every case, for any ego network and edge type, we receive the same results, showing mean similarity scores in the range of 0.3 - 0.4 with a maximum standard deviation around 0.15.

Since the previous results offer no interesting similarity patterns, even though we already acquired some promising similarity scores when comparing just the ego nodes, this led us to experiment on some more constrained datasets, regarding the type of nodes we include in the similarity measurements. In that regard, we measured the similarities on networks that contained only the nodes with a minimum amount of Tweets posted, expecting that their vector representations would be more meaningful. We run some experiments on users with a minimum count of 10, 50 and 100 Tweets.

Tables 3 & 4 list the scores we received when using only the nodes with at least 100 Tweets.

Although not definitely clear, we start seeing some more meaningful similarity patterns with regards to the intra-network similarities of Table 3:

- For the Retweet and Quote networks of *Kyiv Post*, *The Kyiv Independent* and *The Moscow Times*, we see a bigger similarity between users that retweeted or quoted each other, than pairs of users without such interactions between them.
- On the other hand, regarding the reply network of the Russian *Sputnik* and *RT*, we see a drop in the similarity between users replying to each other, probably showing a tendency of more intense argumentation between the related users.

The inter-network similarities of Table 4, while more diverse than the ones in the initial experiments, still offer no meaningful results.

	Retweet	Quote	Reply	Mention	
Kyiv Post	0.48	0.54	0.37	0.41	**All**
	0.56	0.63	0.41	0.47	**Neighbors**
	0.47	0.53	0.37	0.41	**Non-neighbors**
The Kyiv Independent	0.48	0.52	0.44	0.44	**All**
	0.59	0.63	0.52	0.52	**Neighbors**
	0.48	0.52	0.44	0.44	**Non-neighbors**
The Moscow Times	0.52	0.57	0.37	0.43	**All**
	0.61	0.67	0.43	0.51	**Neighbors**
	0.52	0.56	0.37	0.42	**Non-neighbors**
Sputnik	0.51	0.53	0.41	0.43	**All**
	0.47	0.52	**0.23**	0.41	**Neighbors**
	0.51	0.53	**0.42**	0.43	**Non-neighbors**
RT	0.52	0.53	0.46	0.44	**All**
	0.53	0.57	**0.38**	0.46	**Neighbors**
	0.52	0.53	**0.46**	0.44	**Non-neighbors**

Table 3. Intra-network similarity scores; per ego network, edge type and node pair type. Including nodes with a minimum Tweet count of 100.

	Kyiv Post	The Kyiv Independent	The Moscow Times	Sputnik	RT	
Kyiv Post	-	0.46	0.50	0.45	0.44	**Retweet**
	-	0.51	0.55	0.52	0.50	**Quote**
	-	0.43	0.38	0.38	0.39	**Reply**
	-	0.42	0.42	0.40	0.41	**Mention**
The Kyiv Independent	0.46	-	0.51	0.45	0.45	**Retweet**
	0.51	-	0.51	0.49	0.49	**Quote**
	0.43	-	0.41	0.41	0.42	**Reply**
	0.42	-	0.43	0.41	0.42	**Mention**
The Moscow Times	0.50	0.51	-	0.48	0.47	**Retweet**
	0.55	0.51	-	0.52	0.51	**Quote**
	0.38	0.41	-	0.37	0.38	**Reply**
	0.42	0.43	-	0.43	0.41	**Mention**
Sputnik	0.45	0.45	0.48	-	0.47	**Retweet**
	0.52	0.49	0.52	-	0.51	**Quote**
	0.38	0.41	0.37	-	0.42	**Reply**
	0.40	0.41	0.43	-	0.44	**Mention**
RT	0.44	0.45	0.47	0.47	-	**Retweet**
	0.50	0.49	0.51	0.51	-	**Quote**
	0.39	0.42	0.38	0.42	-	**Reply**
	0.41	0.42	0.41	0.44	-	**Mention**

Table 4. Inter-network similarity scores; per ego network combination and edge type. Including nodes with a minimum Tweet count of 100.

5. Conclusions

In the end, we did not manage to get any clear results through the conducted experiments.

- We got some confirmation that the trained Doc2Vec model could be successfully used to measure the content homophily between users, as shown when comparing the similarity between the chosen ego nodes.
    - Note that these nodes post properly structured Tweets, with a very few of them being Replies or Quotes, so we expect that their text embeddings will be among the most representative.
- The experiments on the more constrained networks give as a hint that the reason for failure may be the lack of proper vectorization for, most probably, a significant amount of nodes, which mostly Reply to or Quote other Tweets, and rarely post “plain” Tweets with significant content.
- Most probably we are getting text embedding based on a large amount of meaningless Tweets, without much of a content. Further tests should be run, where we use only “plain” Tweets, excluding those being Replies or Quotes and consequently a number of nodes with only such limited interaction.

A. References

[1] Ehsan-Ul Haq, Gareth Tyson, Lik-Hang Lee, Tristan Braud, Pan Hui. Twitter Dataset for 2022 Russo-Ukrainian Crisis, 2022. Available at https://arxiv.org/abs/2203.02955

[2] Quoc V. Le, Tomas Mikolov. Distributed Representations of Sentences and Documents. ICML'14: Proceedings of the 31st International Conference on International Conference on Machine Learning, Volume 32, Pages II-1188–II-1196, June 2014. Available at https://arxiv.org/abs/1405.4053

B. Data Model

This model is included in the separate file *AppendixB_DataModel.pdf*.

C. File Structure

`data/`	contains any files related with the dataset extraction and storage.
`russo_ukraine_dataset/`	contains the initial dataset files of Tweet IDs

`tweet_tables.sql`
`tweet_indexes.sql`
`tweet_views.sql`
	contain the SQL scripts for the database Table and View creation, and additional indexing.

`summary.sql`	contains a SQL script to calculate and insert some aggregated data after loading the data into the database tables.
`create_db.py`	is a simple script to execute the database creation procedure.
`extract_tweets.py`	contains the extraction procedure; requests data from the Twitter API and stores it in the database.
`tweet.db`	is the SQLite database file.
`embeddings/`	contains the Doc2Vec related files.
`models/`	contains the trained Doc2Vec models.
`corpus.py`	contains an iterator class that streams the documents from the database one-by-one to avoid any memory issues of loading the whole corpus in RAM.
`doc2vec.py`	is the Doc2Vec model training script.
`networks/`	contains edge-list files and the network extraction script.
`extract_networks.py`	extracts the edge-lists of each network. The edge data for the whole network is queried from the database and then each ego subnetwork is created.
`{min}_{type}_{ego}.edges`	the edgelist of a network: ● for nodes with `{min}` tweet count ● for `{type}` of network in {retweet, quote, reply, mention} ● for the specified `{ego}` node
`statistics.ipynb`	a notebook containing the statistics mentioned in the report.
`similarities.py` `ego_similarities.ipynb`	contains the scripts used to measure the similarities between the various node pairs.