

# A Unified Graph-Based Approach to Disinformation Detection using Contextual and Semantic Relations

Marius Paraschiv,<sup>1</sup> Nikos Salamanos,<sup>2</sup> Costas Iordanou,<sup>2</sup> Nikolaos Laoutaris,<sup>1</sup>  
Michael Sirivianos<sup>2</sup>

<sup>1</sup> IMDEA Networks Institute, Spain <sup>2</sup> Cyprus University of Technology  
marius.paraschiv@imdea.org, nik.salaman@cut.ac.cy, costas.iordanou@eecei.cut.ac.cy,  
nikolaos.laoutaris@imdea.org, michael.sirivianos@cut.ac.cy

## Abstract

As recent events have demonstrated, disinformation spread through social networks can have dire political, economic and social consequences. Detecting disinformation must inevitably rely on the structure of the network, on users particularities and on event occurrence patterns. We present a graph data structure, which we denote as a *meta-graph*, that combines underlying users' relational event information, as well as semantic and topical modeling. We detail the construction of an example meta-graph using Twitter data covering the 2016 US election campaign and then compare the detection of disinformation at cascade level, using well-known graph neural network algorithms, to the same algorithms applied on the meta-graph nodes. The comparison shows a consistent 3%-4% improvement in accuracy when using the meta-graph, over all considered algorithms, compared to basic cascade classification, and a further 1% increase when topic modeling and sentiment analysis are considered. We carry out the same experiment on two other datasets, HealthRelease and HealthStory, part of the FakeHealth dataset repository, with consistent results. Finally, we discuss further advantages of our approach, such as the ability to augment the graph structure using external data sources, the ease with which multiple meta-graphs can be combined as well as a comparison of our method to other graph-based disinformation detection frameworks.

## 1 Introduction

Social media have become a primary medium of interaction in modern society – having started as a place for casual discussion and exchange of ideas. Their massive outreach and adaptability to individual users' preferences have made social networks indispensable for a wide range of political and activist groups, companies, governments, and mainstream news outlets. The social importance of these networks stems from the fact that they are more than just a space for public discussion. Social networks such as Twitter and Facebook have become the primary source of information on a global scale (Westerman, Spence, and Van Der Heide 2014). However, the spread of disinformation can have a severe negative social (Chenzi 2020), political (Safieddine 2020), and economic impact (Visentin, Pizzi, and Pichierri 2019; Cheng and Chen 2020), thus it is currently treated as a fast-growing cyberthreat (Belova and Georgieva 2018).

Current disinformation detection methods on social media primarily rely on using part of the available information either regarding users or local network structure (see Section 2), in order to perform the required analysis. Treating such features in isolation may lead to poor results as the manifestation of disinformation is characterized by a plethora of factors, both internal (user characteristics) and external (network characteristics and tweet content).

In this paper, we set out to improve our capacity to detect disinformation in social media by proposing a unified methodology for both internal and external factors. To demonstrate the value of the proposed method, we focus on Twitter content. The primary action on Twitter is message posting by registered users (*tweets*) or message sharing by others within the platform (*retweets*). As such, posting information on a particular topic results in a *root-tweet* (the original tweet that has been retweeted), followed by a series of retweets by other users. This series of retweet events forms a *tree-graph* structure which is known as *retweet cascade*. Moreover, we construct and analyse a very large dataset related to 2016 US Presidential Election, consisting of 152.5M tweets by 9.9M users; In addition, we evaluate our method on the HealthRelease and HealthStory datasets, part of the FakeHealth repository (Dai, Sun, and Wang 2020).

In summary, we propose a graph data structure, named *meta-graph* having the retweet cascades as nodes. Each cascade disseminates a root-tweet together with the web and/or media URL(s) embedded in that tweet. The node features contain: (i) information about the cascade graph structure, obtained by applying a graph embedding algorithm to each individual cascade; (ii) relational user information, extracted from the social network. The edges of the meta-graph represent relationships between cascades, such as structural similarity, number of common users (possibly indicating that the cascades originate from the same community), or content similarity (URL or text of the root tweet). One of the advantages of this approach is that it can be easily expanded with additional information from other datasets. Finally, in the US Elections dataset, many URLs corresponding to the retweet cascades that have been manually labeled as *fake* or *non-fake*, by several external fact-checkers (MBFC 2020; PolitiFacts 2020; FactCheck 2020; Snopes 2020), resulting in a total of 43,989 labeled data points (see Table 4).

The construction of the meta-graph poses a series of non

trivial challenges, such as the choice of content-embedding and graph-embedding algorithms, the selection of relevant features from the raw data or the filtering of initial edges – all of which are described in detail in Section 4. By using the meta-graph approach, we transform the cascade classification problem into a node classification task. In case retweet cascades are treated as independent graphs, then we are dealing with a graph classification task. Data points are represented by individual cascades, labeled by their content. The similarity between cascades is thus a learned feature, inherent to the model itself. There are however advantages to making these relations explicit, such as biasing the algorithm in the desired direction, as well as the capacity to provide additional, external, information to the meta-graph edges, otherwise unavailable to a classification algorithm.

In Section 5 we show a comparison between (i) classifying cascades in isolation (graph classification); (ii) classifying cascades as nodes of the mentioned meta-graph. We demonstrate that the additional relational information contained in the meta-graph leads to consistently higher classification results. For the graph and node classification tasks, we apply four well-known graph neural network algorithms, as they currently represent the state of the art in terms of graph analysis and prediction. For an overview of graph neural networks, we refer the reader to some of the recent review papers, such as (Wu et al. 2020; Zhou et al. 2018; Zhang et al. 2019). As an additional step, to the initial cascade or meta-graph features we append the sentiment analysis scores, as well as the predicted sentiment. This is done so as to avoid instances where a cascade is classified as disinformation, even though the root tweet is highlighting the possible issues with a piece of information, showing clear disagreement. A final set of features comes from topic modeling, where we detect topics, among all tweet texts in the dataset, and assign their corresponding identifier to each cascade or meta-graph node. This provides additional information to the classification algorithm, and we show that this approach offers 1% consistent improvements.

**Data availability:** Part of the US Elections dataset is publicly available (Salamanos et al. 2021) under proper restrictions for compliance with Twitter’s Terms of Service (Tos), General Data Protection Regulation (GDPR). The Fake-Health repository is publicly available (Dai, Sun, and Wang 2020).

## 2 Related Work

Substantial effort has been put into studying false information detection in social media. A first set of approaches concerns performing feature-based detection at user level. (Ahmed, Traore, and Saad 2017) apply various feature extraction methods to the content of social media posts to detect stylistic characteristics that may give away posts containing disinformation. While this is a significant intermediary step, it does not consider relational information between sources and platform users (in the case of Twitter). (Ghosh and Shah 2018) use a similar approach to split the problem into feature extraction and classification steps, while (Zhang, Zhao, and Lecun 2015) use character-level convolutional networks that combine the two. A more recent approach is

Work	User feature-based Detection	Network-based Detection	Content-based Detection
(Ahmed, Traore, and Saad 2017)			✓
(Ghosh and Shah 2018)			✓
(Zhang, Zhao, and Lecun 2015)			✓
(Kaliyar, Goswami, and Narang 2021)			✓
(Castillo, Mendoza, and Poblete 2011)			✓
(Shu et al. 2019c)	✓		
(Tacchini et al. 2017)	✓		
(Guo et al. 2018)	✓		
(Jin et al. 2017)	✓		
(Gupta, Zhao, and Han 2012)	✓	✓	
(Shu et al. 2019a)		✓	✓
(Wu and Liu 2018)		✓	
(Shu, Wang, and Liu 2019)	✓	✓	
(Shu et al. 2019b)		✓	✓
<b>This Work</b>	✓	✓	✓

Table 1: Summary of the related work based on different methodologies that they utilize/combine to identify fake news in social media.

that by (Kaliyar, Goswami, and Narang 2021), which uses the BERT language model (Devlin et al. 2018) to perform feature extraction. A detailed overview of similar methods can be found in (Zhou and Zafarani 2020). (Castillo, Mendoza, and Poblete 2011) assess the credibility of posts based on the past behavior of users, the tweet content, and other external sources. (Shu et al. 2019c) extract features based on particular users’ profiles and determine which type of profile is more inclined to share false information. This is similar to the methods in (Tacchini et al. 2017; Guo et al. 2018; Jin et al. 2017) for exploiting user-type characteristics to classify posted messages as false or not.

Social context-based methods deal both with relations between users sharing news as well as with information related to the user and the news itself (for example, the number of similar postings made by the said user in the past). (Gupta, Zhao, and Han 2012) propose studying the problem from a credibility perspective by performing credibility propagation along a network that comprises events, tweets, and users. The events in this case roughly correspond to the root tweet in our approach. However, their approach does not consider the rich structural information of the cascades as well as cascade similarity. (Shu et al. 2019a) propose a sentence-comment co-attention sub-network, aiming to explain why a piece of news has been detected by an algorithm as false. At the same time, (Wu and Liu 2018) take the approach of detecting false information based on its propagation patterns through the network.

A similar method to the one discussed in this paper is presented by (Shu, Wang, and Liu 2019). The authors use an embedding framework dubbed TriFN, modelling publisher-news relations and users’ interaction with the particular pieces of news simultaneously. Compared to our proposal, it misses relational information between events (cascades), which can prove essential in assessing a user’s past behavior, as well as the semantic features added by topic modeling and sentiment analysis. Another interesting approach is presented in (Shu et al. 2019b) where the authors build a hierarchical propagation network for false information and perform a comparative analysis between false and real news based on linguistic, structural and temporal perspectives.

Root Tweets	46,409	Root Users	8204
Retweets	19,588,072	Retweeters	3,630,992
URLs (web and media)	43,989		

Table 2: US Elections dataset: Retweet cascades with at least 100 unique retweeters

	HealthRelease	HealthStory
tweets	43,245	357,851
replies	1418	23,632
retweets	15,343	105,712
Total	60,006	487,195

Table 3: FakeHealth repository: Tweets, replies and retweets in the collected datasets

The methods outlined above roughly fall into three categories: a) *user feature-based detection* in which the aim is to use information at the user level (be it contextual or behavioral) to assess the type of content a user publishes; b) *network-based detection* in which the relational context is used for content classification; and c) and *tweet content-based detection*, in which linguistic features are exploited for performing classification at the text level. Our proposed meta-graph data structure provides an effective means to combine all three methodologies, fully utilizing content, context, and source information simultaneously, as depicted in Table 1.

### 3 Social Media Data Structure

#### 3.1 US Elections dataset

To analyze disinformation on Twitter, we collected a large number of tweets related to the 2016 US presidential election. During that period, state-sponsored disinformation campaigns are believed to have operated by spreading millions of tweets with ambiguous political content. Hence, Twitter account activity from that period provides us with valuable information for the analysis of disinformation spread in social media. The belief that many of the tweets from that period were spreading fake news has been furthered validated by the fact that Twitter has permanently deleted lots of them as part of its continuous efforts against disinformation and malicious activities on the platform<sup>1</sup>.

**Crawling:** In the period up to the 2016 US presidential election – from September 21<sup>st</sup> to November 7<sup>th</sup>, 2016 (with the exception of October 2<sup>nd</sup> 2016) we collected 152.5M tweets (by 9.9M users) using the Tweepy<sup>2</sup> Python library for accessing the Twitter streaming API. To consider tweets to be politically related, they need to include words from a list of 77 track terms used for the crawling. Track terms as “hillary2016”, “clinton2016”, “trump2016”, “election2016”

<sup>1</sup><https://about.twitter.com/en/our-priorities/civic-integrity>

<sup>2</sup><https://www.tweepy.org/>

etc. would, with a large degree of confidence, return tweets that were indeed part of the intense political polarization and debate that took place during the election period. For each tweet, we stored 27 features related to a tweet (tweet-ID, tweet-text etc.), and to the Twitter account (user-ID, user screen name, number of followers, etc.) who posted that tweet. Moreover, we collected the “Entities” section, which contains several metadata such as the “mentions” and the URLs embedded in the text.

We note that, in this dataset, we have already identified 35.5K tweets from 822 state-sponsored Twitter accounts based on ground-truth data provided by Twitter itself – a large sample of state-sponsored disinformation campaigns from “troll” accounts which operated during that period. Let us note that “troll” is any account that deliberately spreads disinformation, tries to inflict conflict or causes extreme emotional reactions. Hence, our dataset contains valuable information of ground-truth malicious activities which in fact were the subject of a previous study regarding the trolls’ activities during the 2016 US election (Salamanos et al. 2021).

In the retweeting process, there are two actors; (i) the *retweeter*; (ii) the *root-user*, i.e., the user who posted the original tweet (*root-tweet*). We concentrate our attention on retweets that are sufficiently rich in terms of the information transmitted and the population of users that acted as retweeters. For this reason, we concentrate on cascades for which the root-tweet contains at least one web or media URL. We also dropped cascades with less than 100 retweeters. Tweets that very few users have retweeted do not provide enough information, even collectively, regarding their political positions. Following this approach, we analyse 46.4K retweet cascades consisting of 19.6M tweets (Table 2).

**Labeling URLs** To perform supervised/semi-supervised learning, we need to label the collected URLs as “fake”, “non-fake” or “unknown”. Towards that end, we apply the following methodology:

**Step 1 - Unshortening URLs:** The first step involves expanding URLs created from URL shortening services, such as bitly (Bitly 2021), tinyurl (TinyURL 2021), etc. This step is required to identify different short URLs that correspond to the same expanded URL. This step increases the probability of having a URL match with pre-existing annotated URLs from other research projects related to the 2016 US elections, as we explain in the next step. During this step, we use the unshrtn<sup>3</sup> tool to expand the URLs.

**Step 2 - Pre-existing Labels:** Our dataset includes web-pages related to the 2016 US elections, a well-studied dataset with many related research projects available in open-source version control systems, such as GitHub and GitLab. Thus, during this step, we aggregate more than 25 related projects with annotated URLs and combine them into a single database with  $\approx 0.5$ M labeled URLs. Note that we only focused on the “fake” and “non-fake” labels and exclude any other labeling schemes (i.e., humor, satire, etc.) present in the other open-source projects.

**Step 3 - Labeling Consistency check:** During this step we examine the labeling consistency across all the datasets we

<sup>3</sup><https://github.com/DocNow/unshrtn>

Label	Initial Labeling	Manual Labeling	Difference
Fake	4,386	4,556	+170
Non_Fake	915	1,969	+1,054
Unknown	38,688	37,464	-1,224
Total	43,989	43,989	

Table 4: US Elections dataset: The number of labeled URLs obtained at each step of the labeling methodology.

Label	HealthRelease	HealthStory
Fake	198	374
Non_Fake	231	1036
Total News	429	1410

Table 5: FakeHealth repository: Number of news along with their labels in the collected datasets

combine during Step 2. We have kept only consistent labels, while leaving the inconsistent ones for manual validation, as we will explain in the next step.

**Step 4 - Matching URLs and Labels:** During this step, we perform a URL match between the URLs that we extract from our dataset and the one we created by aggregating labels from other open datasets related to the 2016 US elections (Davis 2016; Szpakowski 2020; Macinec 2021).

The output of this step is depicted in Table 4 (second column) “*Initial Labeling*”.

**Step 5 - Manual Labeling / FactCheck:** To increase the total number of labeled URLs, we then turn our attention to the “Unknown” URLs of the Initial Labeling phase. We use four different FactChecking tools, 1. PolitiFacts (PolitiFacts 2020), 2. Media Bias/Fact Check (MBFC) (MBFC 2020), 3. FactCheck (FactCheck 2020), 4. Snopes (Snopes 2020), and we manually annotate a subset of the unknown URLs. Since this step is very time-consuming, we only focused on expanded URLs that correspond to more than one short URL (see Step 1 above) in our dataset.

The final number of each label is depicted in Table 4 (third column - “*Manual Labeling*”), while the final column (“*Difference*”) depicts the difference between the Initial Labeling and the Manual Labeling steps.

### 3.2 FakeHealth datasets

In order to evaluate our method we utilize two datasets of the FakeHealth repository (Dai, Sun, and Wang 2020). Due to the twitter policy of protecting user privacy, the full content of user engagement and network are not allowed to be published by the authors, instead, the authors provide a useful API available at <https://github.com/EnyanDai/FakeHealth>. The API provides the code and details on how to download the full content of users social engagements and network. Using the provided API we collect all related information, a summary is depicted in Tables 3 and 5. Note that the final

numbers reported in the above tables is lower by  $\approx 1.1\%$  than the numbers reported by the original authors. This can be attributed (1) to changes on behalf of the twitter users that choose to disallow public access of their tweets, (2) the tweeter itself delete the tweet due to some internal policies, (3) or the tweeter user account has been deleted.

## 4 The Meta-Graph Approach

Before we give the meta-graph’s construction details, we first provide a formal definition of the data structure.

**Definition 4.1** (Meta-graph). Let  $\mathcal{G} = (V, E)$  be a graph with vertex set  $V$  and edge set  $E$ . Let also  $X_{v_i}$  and  $R_{e_{i,j}}$  be the feature vectors of node  $v_i$  and of edge  $e_{i,j}$  respectively. We call  $\mathcal{G}$  a *meta-graph* constructed from a set of events in a social network, if each event corresponds to a vertex  $v_i \in V$ , and the feature vectors  $X_v$  and  $R_{e_{i,j}}$  encode both user and event relational information.

The node feature vectors in Definition 4.1 encode three types of features: user attributes, tweet content, and cascade structural information. A minimalist node feature vector  $X_v$  can be described as

$$X_v = (C_{\text{emb}} || X_u || T_{\text{emb}} || S \dots), \quad (1)$$

where “ $||$ ” is the vector concatenation operation,  $C_{\text{emb}}$  is the cascade embedding vector (obtained by using some graph embedding method, in our case DeepWalk (Perozzi, Al-Rfou, and Skiena 2014)),  $X_u$  are the concatenated feature vectors of all users present in the cascade, and  $T_{\text{emb}}$  is the text embedding vector (Devlin et al. 2018), provided that the tweet also has the text content.  $S$  is a vector of sentiment analysis scores if one such vector can be constructed, depending on the presence of text within the cascade tweets. The user feature vectors concatenated into  $X_u$  contain information related to the user account itself, such as date of creation, number of followers, number of tweets, geolocation data, topic categories, sentiment analysis label and scores, etc. This formulation allows trivial expansion of node features by concatenating representative vectors from other sources when available.

The meta-graph’s edges are initially constructed based on common users or a common topic (URL or tweet text). The corresponding edge feature attributes encode cascade similarity and tweet content similarity, once more defined in a very general manner. An example of a cascade feature vector is:

$$R_{e_{i,j}} = (N_{u_{i,j}} || V_{i,j} || H_{i,j} \dots), \quad (2)$$

, where  $N_{u_{i,j}}$  represents a one-element vector containing the number of common users in cascades  $i$  and  $j$ ,  $V_{i,j}$  is

the value of a graph similarity metric (Zager and Verghese 2008; Blondel et al. 2004; Zhao et al. 2020; Bisseling 2020) applied to the two cascades.  $H_{i,j}$  stands for a content similarity metric (Gomaa and Fahmy 2013) between the contents of the original root tweets (and possibly retweets) of the two cascades, if available. The edge feature vector can, similar to the node feature vector, be trivially expanded to include additional relational information between cascades.

#### 4.1 The Meta-Graph Construction

There are two main obstacles to be addressed in order to apply our methodology to the data provided by Twitter. First, the actual social network – the follower graph – is mostly unknown; the users’ follower lists are not always accessible. Second, the raw data returned by the Twitter API have limited information (by design) regarding the source of influence in a given retweet cascades. The only available information provided is the root–tweet and the root–user for a given retweet. In other words, all the retweeters have been influenced by the root–user. This star–like cascade structure (Figure 1(a)) does not depict the true chain of retweet events, which in fact is a *tree*, like the one presented in Figure 1(b).

We use the following steps to address the above issues:

**Step 1:** Construct a graph that approximates the Twitter social network.

**Step 2:** Map a cascade to social affiliations of the users that participate in the cascade. Since the cascade is a subgraph of the social network we compute its embedding in a low dimensional space.

**Step 3:** Construct the final meta-graph.

We have to note that, we concentrate our analysis on pure retweet cascades, only. In this way, we ensure that the text that has been diffused by a given cascade (a chain of retweets) was exactly the same with the original/root tweet-text. For this reason, we have excluded the “quotes” from the meta-graph construction. A quoted-tweet is a special case of retweet, where the retweeter has added an additional text above the original one.

**The Social network** We leverage the users’ activity as it is recorded in the data to construct an approximation of the follower graph – the true social network which is not publicly available to a large extent. In the Twitter platform, the interactions between users belong in three categories; *replies*, *retweets*, and *quotes* – a special form of retweet – and *mentions*. Based on these actions, we construct a graph/network of interactions between the users. We map users to nodes and directed edges to interactions. For example, if a user- $i$  has replied to a user tweet- $j$ , then we add the edge  $(i, j)$ . The direction of the edge implies that  $i$  is a *follower* of  $j$ , while the reverse direction represents the information flow from  $j$  to  $i$ . In conclusion, we map users to nodes and use the interactions between users to define the edges. This process outputs a multi-graph, where many edges may connect the same pair of users. For this reason, we discard the duplicate edges keeping only the earliest one.

**US Elections dataset:** The overall graph has 9.32M users/nodes connected with 84.1M directed edges. For the purpose of our analysis, the final social network is repre-

sented by the induced subgraph formed from the 3.63M users – retweeters and root-users who participated in the retweet cascades – who are connected with 61.05M directed edges.

Regarding the two FakeHealth datasets: In the HealthRelease we have 9,055 total users that participate in the retweet cascades. The corresponding social network counts 8,218 users (nodes) connected by 10,510 edges. In the HealthStory the total number of users is 64,593. The corresponding social network consists of 57,851 users and 88,750 edges.

The FakeHealth repository includes the *user-following* adjacency lists who represent the ground-truth social networks. Specifically, in the HealthRelease we have 8,566 users connected by 177,866 edges. The HealthStory consists of 62,011 users and 3,402,241 edges. Having this information available, We compare the “empirical” social networks which we constructed based on the users’ actions with the ground user-following relations that are available for the HealthRelease and HealthStory. In short, for both datasets, we constructed the “empirical” social network based on the actions between the users (mentions, replies, retweets). We restrict our attention only to those users who participated in retweet cascades, since only this graph region is involved in the meta-graph method. Then, we compared the “empirical” edges (i.e. relations) with the ground-truth ones. The 65% and 59% of the “empirical” edges for the HealthRelease and HealthStory respectively, appear in the ground-truth. Although these numbers are not very high they do not affect the overall validity of the meta-graph method. Our goal is not to predict the ground-truth social network but to use past interactions among the users in order to construct a small graph (per cascade) where we can compute the DeepWalk embedding.

**From Cascades to Graph Embeddings** As we mentioned previously, a retweet cascade is a series of chain events upon the same root–tweet. Some of the users have directly retweeted the root–tweet, whereas some others have retweeted a retweet of a friend on the same root–tweet (retweet of a retweet). This tree–like structure is the true retweet cascade (see Figure 1(b)) and reflects the diffusion path of information that has been transmitted by the users in the social network. The problem we face here is that the data provided by Twitter do not represent the true cascades. The raw data contain only the retweet & retweeter IDs and the root–tweet & the root–user IDs. Hence, it is unknown who was influenced by whom during the retweeting process. The raw–data correspond to a star–like graph where all the retweeters are connected with the root–user. As a result, this form does not provide sufficient structural information. This is a well-known problem in the literature and many methods have been proposed to estimate the true diffusion path (Goel et al. 2015).

To address this problem, we leverage the social network we have constructed. We construct a subgraph formed by the interactions that the retweeters of this cascade have had in the past. Specifically, for a given retweeter  $i$  who performed her retweet at date  $t$ , we identify which friends she had before the date  $t$  and which of them belong to the set of

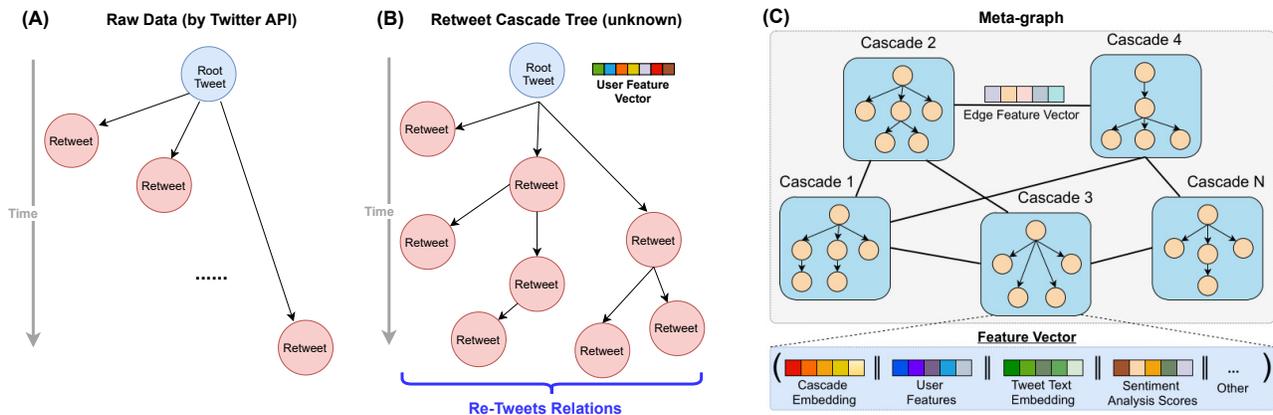


Figure 1: (a) The raw data that are provided by Twitter API correspond to a star graph structure. (b) The true retweet cascade tree, which is actually unknown, highlights the root tweet and subsequent retweets i.e the true series of retweet events. (c) The proposed meta-graph data structure: every node represents a Twitter cascade. The cascade features are given by the cascade vector embedding; the user features vector; the tweet text embedding and sentiment analysis scores (if retweet text is available). The edge features represent cascade structural similarity, i.e number of common users. This can easily be expanded with additional attributes, either from current or external datasets.

retweeters of this cascade. Then we append this set of edges (if any) with the star-like structure, where each retweeter is connected with the root-user. Finally, we discard any duplicate edges and produce the undirected version. By this method, the subgraph is always connected. The extreme case occurs when the retweeters did not have any previous interaction. Then, the resulted graph coincides with the raw data collected from Twitter. In summary, the subgraph is just the star-like graph (i.e., the raw Twitter data) enhanced by the retweeters’ social relations. This construction per cascade represents the social structure that the participants of the cascade had before being activated and be able to perform their retweet.

Each cascade should use the corresponding subgraph as a feature that will reflect the structural (social) relation that the cascades’ participants had. To achieve that, in a consistent way, we produce the embedding of each subgraph to a low dimensional space. We use the DeepWalk algorithm (Perozzi, Al-Rfou, and Skiena 2014) and specifically the *Karate Club* extension library’s implementation for NetworkX (Rozemberczki, Kiss, and Sarkar 2020). The DeepWalk embedding is a  $N \times 128$  matrix, where  $N$  is the number of users that participated in the cascade. We applied the default parameters of this implementation, that is: (i) Number of random walks = 10; (ii) Length of random walks = 80; (iii) Dimensionality of embedding = 128.

**Topics and Sentiment Analysis** While the previous set of features were concerned with user similarity and relational information extracted from the social network, the final features, which complete the meta-graph, are focused on semantics and subject-based grouping of tweet-retweet cascades. For this purpose and for the US Elections dataset only, we carry out two tasks, topic detection and sentiment analysis of the root-tweet content. For the former, as the initial

number of topics covered by our dataset is unknown, we employ a Hierarchical Dirichlet Process (HDP) (Teh et al. 2005; Newman et al. 2009), using the Tomotopy library (bab2min and Fenstermacher 2021). Three parameters of the HDP model are changed from the default values, namely  $\text{min\_cf} = 5$ ,  $\text{gamma} = 1$  and  $\text{alpha} = 0.1$ , as these produce the best results. For term weighing, we produce three instances of the model with equal term weighing, Point-wise Mutual Information-based weighing and Inverse Frequency term weighing (low weights for terms with high occurrence and vice-versa). After training the three versions of the HDP model on our tweet content, we use them in order to assign a topic ID to each tweet-retweet cascade.

The second step in the process is to perform sentiment analysis on the root-tweet of each cascade. This is done using a pre-trained BERT language model, with the help of the Transformers library (Wolf et al. 2020). Along with the three topic identifiers obtained above, the sentiment label and sentiment score form the semantic features used in our approach.

The semantic information, together with the embeddings obtained in the previous section, represent the features of each individual cascade, in the case of graph-level classification, or of each individual node, in the meta-graph, for the node-classification task. In the case of cascades, it is assumed that retweets share the topic and sentiment of the root tweet.

**The meta-graph** The final step is the construction of the meta-graph itself. That is, a meta-structure represented by a graph that has the retweet cascades as nodes. The edges of the meta-graph are defined by the proximity among the cascades in terms of their retweeter population.

In particular, two cascades  $i$  and  $j$  are connected by an edge if  $|RT_i \cap RT_j| \geq 1$ , where  $RT_i$  and  $RT_j$  are the set of retweeters of  $i$  and  $j$ . In other words, two cas-

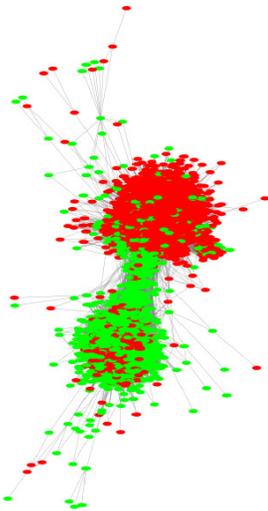


Figure 2: Meta-graph visualization for the US Elections dataset: Giant component of the meta-graph produced by the disparity filtering with  $\alpha=0.1$ . “Fake” nodes/cascades have been colored red, while the “non-fake” nodes are green. The meta-graph consists of 8291 cascades connected by 1,363,702 undirected edges and it has 3 connected components. The giant component counts 8282 nodes and 1,363,680 edges.

cades are connected by an edge when they share at least one retweeter. In conclusion, the meta-graph is always an undirected weighted graph, where the weight of each edge is equal to the number of retweeters that participate in both nodes/cascades. In case that the meta-graph is very dense, we apply the disparity filtering method for undirected weighted graphs (Serrano, Boguñá, and Vespignani 2009). Given a significance level  $\alpha$ , the disparity filter identifies which edges should be preserved in the graph (see Equation-2 in (Serrano, Boguñá, and Vespignani 2009)).

US Elections dataset: The meta-graph of 8323 labeled cascades counts 8323 nodes/cascades connected by 15,946,910 undirected edges. Since the graph is extremely dense, we apply the disparity filter for several  $\alpha$  values. In this way, we produce one filtered meta-graph for each  $\alpha$  value. Then, we perform the classification in each meta-graph independently. We report the results in Table 6. As an example, Figure 2 presents the giant component of the meta-graph that has been produced by the disparity filter for  $\alpha = 0.1$ . 8291 nodes/cascades out of the 8323 are connected by 1,363,702 undirected edges. The graph has 3 connected components. The giant component counts 8282 nodes and 1,363,680 edges.

FakeHealth datasets: In the HealthRelease the full version of the meta-graph counts 1969 cascades connected by 11,581 undirected edges (Figure 3, left-hand figure). In the HealthStory the full meta-graph consists of 13,263 cascades connected by 100,001 edges (Figure 3, right-hand figure). Since these graphs are not very large, we use the full version for the classification.

$\alpha$	Number of Nodes	Number of Edges	Accuracy of GCNConv (%)
0.01	8171	300,330	83.96
0.02	8221	434,643	84.06
0.03	8246	555,934	84.20
0.04	8260	673,595	84.46
0.05	8262	788,212	84.44
0.06	8268	902,797	84.56
0.07	8277	1,016,344	84.68
0.08	8285	1,131,124	85.04
0.09	8290	1,247,597	85.17
0.1	8291	1,363,702	85.34
0.2	8310	2,606,347	86.52
0.3	8311	4,079,032	86.60
0.4	8315	5,742,849	87.64
<b>0.5</b>	<b>8319</b>	<b>7,763,018</b>	<b>87.70</b>
0.6	8319	9,899,401	87.66
0.7	8319	12,536,442	86.91
0.8	8320	14,866,835	87.12
0.9	8320	15,829,764	86.45

Table 6:  $\alpha$  parameter selection using GCNConv as the best performing model.

**Node features:** The features of a given node/cascade with  $N$  users (the retweeters plus the root-user) are the following: (1) The DeepWalk embedding of the correspondent subgraph – a  $N \times 128$  matrix; (2) The dates of users’ account creation; (3) The maximum value of users’ followers count; (4) The maximum value of users’ friends count; (5) The maximum value of users’ statuses count; (6) The maximum value of users’ favorites count; (7) A Boolean identifier whether the users’ account is verified; (8) The language of the users’ account. Note that features 2 to 8 comprise an array of length  $N$ . The maximum values per user are based on the retweets that the user has posted; (9) The most representative topic of the root-tweet based on the three topic models – three values. Regarding the two FakeHealth datasets, we set as topics the news’ “tags” and “category”. This information is provided by the “reviews” in the FakeHealth repository; (10) The sentiment (*label, score*) of the root-tweet – two values, either (*2, score*) for ‘Positive’ label or (*1, score*) ‘Negative’ label.

Regarding the US Elections dataset, we note that we analyse the retweet cascades in which at least one URL is embedded in the root-tweet. Since the labeled part of the data is the URLs, we use these labels as ground-truth. We focus our attention on the 6525 URLs that have been labeled as “fake” and “non-fake” (Table 4). For each URL, we collect the cascades that had this URL embedded in their root-tweets. Moreover, there is no one-to-one correspondence between the cascades and the URLs – multiple cascades may have spread the same URL. This is why the number of labeled cascades is larger than the number of labeled URLs. Moreover, a cascade might contain several URLs, “fake” and

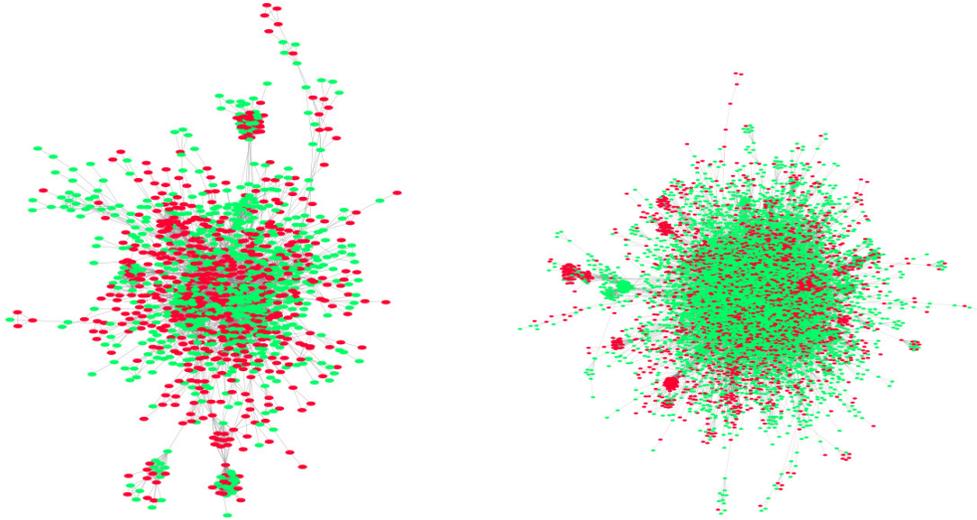


Figure 3: HealthRelease (left-hand figure) and HealthStory meta-graphs. “Fake” nodes/cascades have been colored red, while the “non-fake” nodes are green. Both are connected graphs.

Model	Cascade Classification		Metagraph Node Classification	
2016 US Presidential Elections Dataset				
Without Topics and Sentiment Analysis	Accuracy (%)	F1 Score	Accuracy (%)	F1 Score
GCNConv	<b>83.30</b>	<b>0.721</b>	<b>86.75</b>	<b>0.812</b>
GATConv	83.28	0.716	85.52	0.793
HypergraphConv	82.17	0.693	85.67	0.791
SAGEConv	82.23	0.702	85.12	0.787
With Topics and Sentiment Analysis	Accuracy (%)	F1 Score	Accuracy (%)	F1 Score
GCNConv	<b>84.23</b>	<b>0.748</b>	<b>87.70</b>	<b>0.830</b>
GATConv	83.74	0.732	86.62	0.822
HypergraphConv	83.91	0.731	86.67	0.825
SAGEConv	83.42	0.727	86.84	0.810
Health Release Dataset				
With Topics and Sentiment Analysis	Accuracy (%)	F1 Score	Accuracy (%)	F1 Score
GCNConv	<b>86.54</b>	<b>0.890</b>	<b>88.07</b>	<b>0.936</b>
GATConv	84.43	0.753	87.81	0.903
HypergraphConv	85.76	0.794	86.60	0.892
SAGEConv	84.82	0.758	86.74	0.896
Health Story Dataset				
With Topics and Sentiment Analysis	Accuracy (%)	F1 Score	Accuracy (%)	F1 Score
GCNConv	59.23	0.616	60.16	0.751
GATConv	<b>60.18</b>	<b>0.632</b>	<b>61.03</b>	<b>0.757</b>
HypergraphConv	56.80	0.587	58.70	0.735
SAGEConv	57.36	0.590	57.36	0.726

Table 7: Classification results obtained with four different graph neural network types, on three datasets: the 2016 Presidential Election dataset and the FakeHealth package, composed of the Health Release and Health Story datasets.

“non-fake” ones. In this case, we discard these cascades.

## 5 Evaluation of Performance Benefits

In order to evaluate the benefits of the meta-graph method, we perform a series of experiments aiming to address the following research questions: How effective is the meta-graph structure for the cascade classification? Will the topics (broad subject shared by a set of tweets) and sentiment labels (positive or negative sentiment, together with the confidence score) improve the separation of classes if they are included in the features?

In order to address these questions, we follow two classification strategies: (1) First, we ignore the information that is represented by the meta-graph edges and we reduce the problem to a graph classification task. Each retweet cascade is represented by a sub-graph of the social network (see Subsection 4.1), namely a small graph which consists of those users who have been involved in the cascade. The users’ features are those we presented in Subsection 4.1. Moreover, we assume that each user inherits the topic and sentiment label from the root-tweet. In other words, for a given cascade, all users have the same topic and sentiment label as feature. (2) The second classification approach is node-classification in the meta-graph level, where each node corresponds to a cascade (see Figure 1(c)).

In other words, we classify the nodes of the meta-graph based on the meta-graph structure together with the nodes/cascades features. Now the nodes correspond to individual cascades. In both approaches, we rely on the same sub-graphs from the constructed meta-graph. In addition to these and in order to justify the need for topic modeling and sentiment analysis, we compare the two classification strategies with and without the use of topics and sentiments as nodes features. The intuition is that the topic modeling and sentiment analysis introduce semantic information in the meta-graph structure, providing a quantifiable measure of

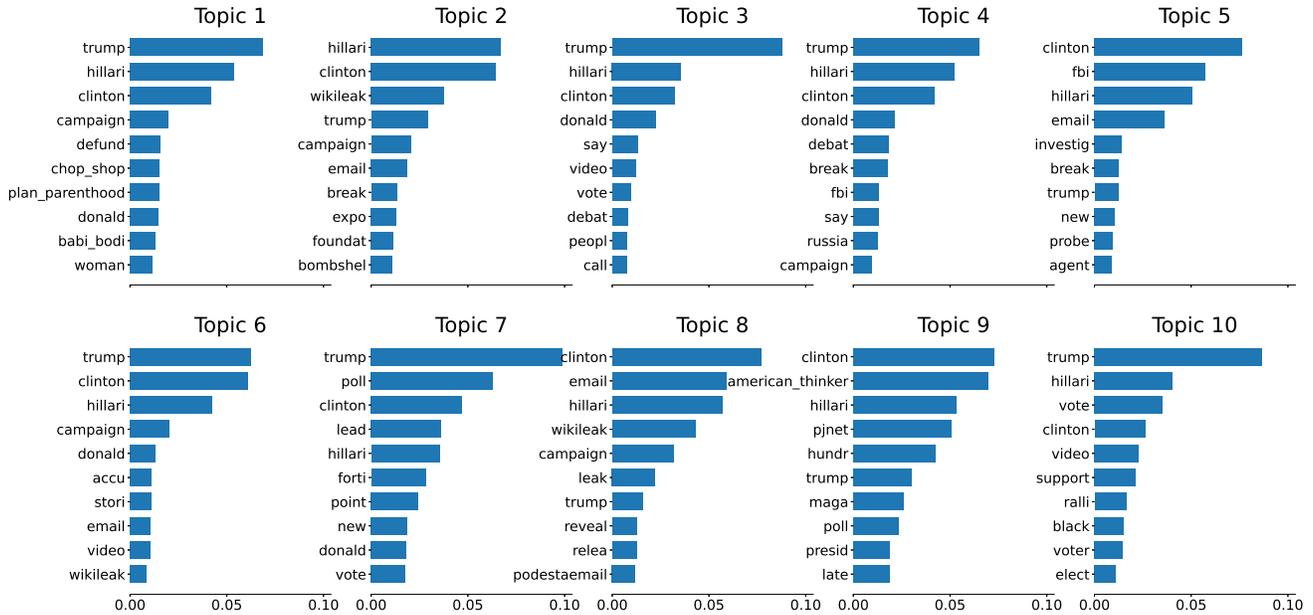


Figure 4: US Elections dataset: **Top**: The top-10 Topics by the Hierarchical Dirichlet Process (HDP) when all terms weighted equally. **Bottom**: The corresponding web-articles of each topic.

agreement as well as topical relational information to other nodes.

Our findings show that: (1) This method of connecting cascades/nodes is effective enough to improve by 3%-4% the classification accuracy. In the meta-graph two cascades are connected by an edge if they have more than 10 retweeters in common. (2) When we include the topics and sentiment labels as features, the accuracy of all classifiers is increased by approximately 1%.

Regarding the actual methods, we applied the following state-of-the-art graph neural networks (GNNs) for both graph and node classification: *GCNConv* (Kipf and Welling 2016), *SAGEConv* (Hamilton, Ying, and Leskovec 2017), *HypergraphConv* (Bai, Zhang, and Torr 2021) and *GATConv* (Veličković et al. 2017).

For the graph classification task, we have applied the algorithms to the 8318 sub-graphs (after filtering contradictory labels). As the treated graphs are relatively small, and GNNs generally do not benefit from an increase in the number of layers (Oono and Suzuki 2020; Li, Han, and Wu 2018; NT and Maehara 2019; Alon and Yahav 2020), we restrict the models to one graph feature extraction layer and one dense layer. We also use a 60-20-20 train-validation-test split. The GNNs have been implemented in PyTorch Geometric (Fey

and Lenssen 2019), maintaining the following hyperparameters constant across experiments:

$$\# \text{ of epochs} = 50, \text{ dropout} = 0.8, \text{ learning rate} = 10^{-4}$$

In Table 7 we present the obtained results, which show a 3%-4% gain in accuracy in favor of the meta-graph method, i.e. for the node classification task. We get the best performance when we include the topics and sentiment labels. For state-of-the-art algorithms this represents a significant gain, and primarily depends on the inclusion of relational information between retweet cascades. Regarding the topic modeling and sentiment analysis, we observe that the 1% gain in accuracy, is nonetheless an indicator that the discovered topics and sentiment labels indeed lead to an improvement to the separation of the classes.

Due to the imbalance of the dataset, we also present F1 scores for all our experiments. We observe a general consistency in terms of the advantage provided by the meta-graph method compared to regular cascade classification. In particular, the extra relational information present in the meta-graph does contribute to the reduction of the number of false positives and false negatives.

In Table 7 we also present results for two health-related misinformation datasets, namely HealthRelease and Health-

Story. These smaller datasets have the added disadvantage of simpler cascade structures. While most cascades present contain a small number of users (sometimes just a tweet-retweet pair), larger cascades tend to have a star-graph shape. This reduces the structural information that our proposed method uses, with the help of graph embedding algorithms, such as DeepWalk.

We conclude the analysis by portraying a representative sample of the topic modeling in order to emphasize the effectiveness of this approach. Figure 4 depicts the top-10 topics based on the HDP-model (Hierarchical Dirichlet Process) when all terms weighted equally. For each topic in the top-10 we plot the top-10 terms by their log-likelihood values. In addition, in Figure 4 (Bottom),

for each top-10 topic we show a representative news article related to the period of the 2016 Presidential elections. For instance, the top terms of Topic-1 and Topic-2 reflect the "Planned Parenthood" debate and the *Wikileaks* source related to Hillary's Clinton leaked emails, respectively.

## 6 Discussion

Our proposed method addresses the detection of disinformation in social networks, by exploiting network structure as well as post content and user sentiment. The example datasets focus on the Twitter social network, whose characteristic is given by the shortness of exchanged user messages. This has an effect on two components of our method: the sentiment analysis component and the text embedding. As some tweets may not contain any original content, and represent just links (retweets) to other user's posts, we have designed our method to be robust to this particular type of information scarcity, by exploiting network structure wherever possible.

Network structure is an important aspect with a clear impact on classification results. The meta-graph is constructed from individual tweet-retweet cascades. As such, the structural information of the cascade is contained within a part of the corresponding features of the meta-graph node. These features are obtained by using graph embedding algorithms, for example DeepWalk, in order to capture cascade structure and connectivity patterns. In the case of very small cascades, or when the majority of cascades have star-graph shapes, the embedding vectors are very similar. The effects of this on the performance of our method are visible in Table 7, for the specific case of HealthStory.

In the present paper we adopt a simple method for transferring the URL labels, namely the labels of the root tweet content, to the entire cascade, by assuming that the label of the cascade is given by the label of the content of its root tweet. This can, in some situations, create problems, for instance if the root tweet and all following retweets share a URL containing false information, with the explicit purpose of exposing it, the entire cascade will be labeled as disinformation. We deal with this potential issue by removing "quote tweets", containing additional text. If retweet contents are available, then one can easily expand this trivial labeling to take the already computed sentiment analysis scores into account. If the sentiment scores are strongly negative, a disagreement between the tweets and URL content can be de-

tected and the label of the cascade can be adjusted accordingly.

Due to the nature and structure of our considered datasets, a direct benchmark comparison with similar algorithms such as FANG (Nguyen et al. 2020) and Hierarchical Propagation Networks (Shu et al. 2019b) is not possible, as the mentioned algorithms require the construction of different types of graphs, for which the considered datasets do not contain the relevant information. An example is the heterogeneous graph required for the FANG algorithm, where both articles/posts and users are interconnected nodes.

The strength of our method relies on its capacity to treat information-scarce datasets as long as structure can be exploited, due to the graph embedding features playing a central role. This also displays the inadequacy of the method to datasets which have very limited graph structure, such as the HealthStory dataset, on which the performance both in terms of accuracy and F1 score is very low. In limit-cases, for example in treating isolated nodes, the node classification task naturally reduces to a simple graph (cascade) classification task. As such, the meta-graph approach does not need to remove or provide special treatment to isolated nodes.

## 7 Conclusion

Recent false information detection and classification methodologies rely on user features extracted from the social network, network structure, or the posting (in our case, tweet) content itself. This paper aims to unify these approaches via the use of a single data structure, which we call a *meta-graph*. The meta-graph node features represent retweet cascades, containing information about the tweet-retweet event and the individual users taking part in it. Encoded within the node features, we also add the tweet content, where available. At the same time, the edge features contain feature vectors whose elements are similarity metrics between cascades. This information is beneficial when only a small number of cascades in the meta-graph are labeled.

By combining all available information about a social-network event into a single data structure, we provide a graph-specific classification algorithm with an informational-rich data format that allows it to outperform, in terms of classification accuracy values, approaches based on isolated elements (such as individual graphs). The additional similarity information among pairs of cascades is beneficial in semi-supervised classification settings because labels are routinely hard to obtain, and only a small fraction of the data may have them.

Another advantage of the method is the size of the dataset itself. Even if the meta-graph contains considerably more information than the individual constituent cascades, the storage cost is not prohibitive. In our case, for 8008 nodes and 1,979,031 undirected edges, the dataset (either in meta-graph or cascade sub-graph form) occupies approximately 3 GB.

The presented formalism opens the door for a wide range of future extensions. The meta-graph can be generalized to include not just bipartite relations between events (cascades), but also multipartite ones, thus converting the data

structure into a hypergraph. Another possible research direction is concerned with finding the optimal features to include for nodes and edges. This is far from trivial, as there is a large variety of parameters available, characterizing a tweet or a particular user, as well as many ways in which the graph structure of the cascades can be encoded. Finally, relation learning, similar to the case of knowledge graphs, can be considered, and learned edge features be added to existing ones.

### Acknowledgments

This project has been funded by: (i) the European Union's Horizon 2020 Research and Innovation program under the Cybersecurity CONCORDIA project (Grant Agreement No. 830927) and under the Marie Skłodowska-Curie INCOGNITO project (Grant Agreement No. 824015); (ii) the TV-HGGs project (OPPORTUNITY/0916/ERC-CoG/0003), co-funded by the European Regional Development Fund and the Republic of Cyprus through the Research and Innovation Foundation.

### References

- Ahmed, H.; Traore, I.; and Saad, S. 2017. Detection of Online Fake News Using N-Gram Analysis and Machine Learning Techniques. 127–138. ISBN 978-3-319-69154-1. doi:10.1007/978-3-319-69155-8.9.
- Alon, U.; and Yahav, E. 2020. On the Bottleneck of Graph Neural Networks and its Practical Implications URL arXiv: 2006.05205.
- bab2min; and Fenstermacher, D. 2021. bab2min/tomotopy. doi:10.5281/zenodo.4719813. URL <https://doi.org/10.5281/zenodo.4719813>.
- Bai, S.; Zhang, F.; and Torr, P. 2021. Hypergraph convolution and hypergraph attention. *Pattern Recognition* 110: 107637. doi:10.1016/j.patcog.2020.107637.
- Belova, G.; and Georgieva, G. 2018. Fake News as a Threat to National Security. *International conference KNOWLEDGE-BASED ORGANIZATION* 24: 19–22. doi: 10.1515/kbo-2018-0002.
- Bisseling, R. 2020. *Graph matching*, 291–358. ISBN 9780198788348. doi:10.1093/oso/9780198788348.003.0005.
- Bitly. 2021. Bitly - A URL shortener built with powerful tools to help you grow and protect your brand. <https://bitly.com/>.
- Blondel, V.; Gajardo, A.; Heymans, M.; Senellart, P.; and Van Dooren, P. 2004. A Measure of Similarity between Graph Vertices: Applications to Synonym Extraction and Web Searching. *SIAM Review* 46: 647–666. doi:10.2307/20453570.
- Castillo, C.; Mendoza, M.; and Poblete, B. 2011. Information credibility on Twitter. 675–684. doi:10.1145/1963405.1963500.
- Cheng, Y.; and Chen, Z. F. 2020. The Influence of Presumed Fake News Influence: Examining Public Support for Corporate Corrective Response, Media Literacy Interventions, and Governmental Regulation. *Mass Communication & Society* doi:10.1080/15205436.2020.1750656.
- Chenzi, V. 2020. Fake news, social media and xenophobia in South Africa Fake news, social media and xenophobia in South Africa. *African Identities* doi:10.1080/14725843.2020.1804321.
- Dai, E.; Sun, Y.; and Wang, S. 2020. Ginger Cannot Cure Cancer: Battling Fake Health News with a Comprehensive Data Repository. *Proceedings of the International AAAI Conference on Web and Social Media* 14(1): 853–862.
- Davis, M. 2016. BuzzFeed News - Facebook Pages Are Publishing False And Misleading Information At An Alarming Rate. <https://www.buzzfeednews.com/article/craigsilverman/partisan-fb-pages-analysis>.
- Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding .
- FactCheck. 2020. FactCheck.org - A Project of The Annenberg Public Policy Center. <https://www.factcheck.org/>.
- Fey, M.; and Lenssen, J. E. 2019. Fast Graph Representation Learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.
- Ghosh, S.; and Shah, C. 2018. Towards automatic fake news classification. *Proceedings of the Association for Information Science and Technology* 55: 805–807. doi: 10.1002/pra2.2018.14505501125.
- Goel, S.; Anderson, A.; Hofman, J.; and Watts, D. J. 2015. The structural virality of online diffusion. *Management Science* 62(1).
- Gomaa, W.; and Fahmy, A. 2013. A Survey of Text Similarity Approaches. *international journal of Computer Applications* 68. doi:10.5120/11638-7118.
- Guo, H.; Cao, J.; Zhang, Y.; Guo, J.; and Li, J. 2018. Rumor Detection with Hierarchical Social Attention Network. 943–951. doi:10.1145/3269206.3271709.
- Gupta, M.; Zhao, P.; and Han, J. 2012. Evaluating Event Credibility on Twitter. *Proceedings of the 12th SIAM International Conference on Data Mining, SDM 2012* 153–164. doi:10.1137/1.9781611972825.14.
- Hamilton, W.; Ying, R.; and Leskovec, J. 2017. Inductive Representation Learning on Large Graphs .
- Jin, Z.; Cao, J.; Guo, H.; Zhang, Y.; and Luo, J. 2017. Multimodal Fusion with Recurrent Neural Networks for Rumor Detection on Microblogs. 795–816. doi:10.1145/3123266.3123454.
- Kaliyar, R.; Goswami, A.; and Narang, P. 2021. FakeBERT: Fake news detection in social media with a BERT-based deep learning approach. *Multimedia Tools and Applications* doi:10.1007/s11042-020-10183-2.
- Kipf, T.; and Welling, M. 2016. Semi-Supervised Classification with Graph Convolutional Networks .
- Li, Q.; Han, Z.; and Wu, X.-M. 2018. Deeper Insights into Graph Convolutional Networks for Semi-Supervised Learning URL arXiv:1801.0760.

- Macinec, P. 2021. Fake News Datasets. <https://github.com/pmaginec/fake-news-datasets>.
- MBFC. 2020. Media Bias/Fact Check (MBFC) - The most comprehensive media bias resource on the internet. <https://mediabiasfactcheck.com/>.
- Newman, D.; Asuncion, A.; Smyth, P.; and Welling, M. 2009. Distributed algorithms for topic models. 1801–1828.
- Nguyen, V.-H.; Sugiyama, K.; Nakov, P.; and Kan, M.-Y. 2020. *FANG: Leveraging Social Context for Fake News Detection Using Graph Representation*, 1165–1174. New York, NY, USA: Association for Computing Machinery. ISBN 9781450368599. URL <https://doi.org/10.1145/3340531.3412046>.
- NT, H.; and Maehara, T. 2019. Revisiting Graph Neural Networks: All We Have is Low-Pass Filters URL arXiv:1905.09550.
- Oono, K.; and Suzuki, T. 2020. Graph Neural Networks Exponentially Lose Expressive Power for Node Classification. In *International Conference on Learning Representations*. URL <https://openreview.net/forum?id=S1ldO2EFPr>.
- Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. DeepWalk: Online Learning of Social Representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14*, 701–710. New York, NY, USA: Association for Computing Machinery. ISBN 9781450329569.
- PolitiFacts. 2020. The Principles of the Truth-O-Meter: PolitiFact's methodology for independent fact-checking. <https://www.politifact.com/>.
- Rozemberczki, B.; Kiss, O.; and Sarkar, R. 2020. Karate Club: An API Oriented Open-source Python Framework for Unsupervised Learning on Graphs. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM '20)*, 3125–3132. ACM.
- Safieddine, F. 2020. *Political and Social Impact of digital Fake news in an era of Social Media*, 43–58. ISBN 9781786614223.
- Salamanos, N.; Jensen, M. J.; Iordanou, C.; and Sirivianos, M. 2021. Did State-sponsored Trolls Shape the 2016 US Presidential Election Discourse? Quantifying Influence on Twitter. *CoRR* abs/2006.09938. URL <https://arxiv.org/abs/2006.09938>.
- Serrano, M. Á.; Boguñá, M.; and Vespignani, A. 2009. Extracting the multiscale backbone of complex weighted networks. *Proceedings of the National Academy of Sciences* 106(16): 6483–6488. ISSN 0027-8424. doi:10.1073/pnas.0808904106.
- Shu, K.; Cui, L.; Wang, S.; Lee, D.; and Liu, H. 2019a. dFEND: Explainable Fake News Detection. ISBN 978-1-4503-6201-6. doi:10.1145/3292500.3330935.
- Shu, K.; Mahudeswaran, D.; Wang, S.; and Liu, H. 2019b. Hierarchical Propagation Networks for Fake News Detection: Investigation and Exploitation.
- Shu, K.; Wang, S.; and Liu, H. 2019. Beyond News Content: The Role of Social Context for Fake News Detection. doi:10.1145/3289600.3290994.
- Shu, K.; Zhou, X.; Wang, S.; Zafarani, R.; and Liu, H. 2019c. The role of user profiles for fake news detection. 436–439. doi:10.1145/3341161.3342927.
- Snopes. 2020. Snopes - Snopes is the internet's definitive fact-checking resource. <https://www.snopes.com/>.
- Szpakowski, M. 2020. Fake News Corpus. <https://github.com/several27/FakeNewsCorpus>.
- Tacchini, E.; Ballarin, G.; Della Vedova, M.; Moret, S.; and Alfaro, L. 2017. Some Like it Hoax: Automated Fake News Detection in Social Networks.
- Teh, Y. W.; Jordan, M. I.; Beal, M. J.; and M., B. D. 2005. Sharing clusters among related groups: Hierarchical Dirichlet processes. NIPS'05, 1385–1392. *Advances in Neural Information Processing Systems*.
- TinyURL. 2021. TinyURL - Making over a billion long URLs usable! <https://tinyurl.com/>.
- Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2017. Graph Attention Networks .
- Visentin, M.; Pizzi, G.; and Pichierri, M. 2019. Fake News, Real Problems for Brands: The Impact of Content Truthfulness and Source Credibility on consumers' Behavioral Intentions toward the Advertised Brands. *Journal of Interactive Marketing* 45: 99–112. doi:10.1016/j.intmar.2018.09.001.
- Westerman, D.; Spence, P. R.; and Van Der Heide, B. 2014. Social Media as Information Source: Recency of Updates and Credibility of Information. *J. Comp.-Med. Commun.* 19(2): 171–183. doi:10.1111/jcc4.12041.
- Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; Davison, J.; Shleifer, S.; von Platen, P.; Ma, C.; Jernite, Y.; Plu, J.; Xu, C.; Scao, T. L.; Gugger, S.; Drame, M.; Lhoest, Q.; and Rush, A. M. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 38–45. Online: Association for Computational Linguistics.
- Wu, L.; and Liu, H. 2018. Tracing Fake-News Footprints: Characterizing Social Media Messages by How They Propagate. doi:10.1145/3159652.3159677.
- Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; and Yu, P. 2020. A Comprehensive Survey on Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems* PP: 1–21. doi:10.1109/TNNLS.2020.2978386.
- Zager, L.; and Verghese, G. 2008. Graph similarity scoring and matching. *Applied Mathematics Letters* 21: 86–94. doi:10.1016/j.aml.2007.01.006.
- Zhang, S.; Tong, H.; Xu, J.; and Maciejewski, R. 2019. Graph convolutional networks: a comprehensive review. *Computational Social Networks* 6. doi:10.1186/s40649-019-0069-y.

Zhang, X.; Zhao, J.; and Lecun, Y. 2015. Character-level Convolutional Networks for Text Classification.

Zhao, C.; Yin, Z.; Wang, H.; Zhao, X.; and Niu, D. 2020. A Novel Method for Graph Matching. 177–181. doi:10.1145/3422713.3422730.

Zhou, J.; Cui, G.; Zhang, Z.; Yang, C.; Liu, Z.; and Sun, M. 2018. Graph Neural Networks: A Review of Methods and Applications.

Zhou, X.; and Zafarani, R. 2020. A Survey of Fake News: Fundamental Theories, Detection Methods, and Opportunities. *ACM Computing Surveys* 53. doi:10.1145/3395046.