A text and GNN based controversy detection method on social media

Samy Benslimane, Jérôme Azé, Sandra Bringay, Maximilien Servajean, Caroline Mollevi

A Text and GNN Based Controversy Detection Method On Social Media

Samy Benslimane[1*], Jérôme Azé[1], Sandra Bringay[1,2], Maximilien Servajean[1,2] and Caroline Mollevi[3,4]

[1*]LIRMM, Univ Montpellier, CNRS, Montpellier, France.
[2]AMIS, Paul-Valéry University, Montpellier, France.
[3]Institut du Cancer Montpellier (ICM), Montpellier, France.
[4]IDESP, UMR Inserm - Univ Montpellier, Montpellier, France.

*Corresponding author(s). E-mail(s): samy.benslimane@lirmm.fr;
Contributing authors: jerome.aze@lirmm.fr;
sandra.bringay@lirmm.fr; maximilien.servajean@lirmm.fr;
caroline.mollevi@chu-montpellier.fr;

Abstract

Expressed opinions on social media frequently cause a controversy. Controversial content refers to content that attracts different opinions and interrogations, implying interaction between communities. Its automatic identification remains a challenging task. Most of the existing approaches rely on the graph structure of discussion and/or the content of messages but did not deeply explore the recent advances on Graph Neural Network (GNN) to predict if a discussion is controversial or not. This paper aims to combine both user interactions present in the graph structure of a discussion and the discussion text features to detect controversy. We rely on sampling techniques to reduce the size of large graphs and augment the graph training set if needed. Our proposed approach relies then on GNN techniques to encode the initial (or sampled) graph in an embedding vector before performing a graph classification task. We propose two controversy detection strategies. The first one is based on a hierarchical graph representation learning to take advantage of hierarchical relationships that could exist between users. The second one is based on the attention mechanism, which allows each user node to give more or less importance to its neighbors when computing node embeddings. We present different experiments conducted with data sources collected from both Reddit and Twitter to show the

applicability of our approach to different social networks. Conducted experiments show the positive impact of combining textual features and structural information in terms of performance and accuracy.

Keywords: Controversy detection, Graph neural networks, Hierarchical graph representation learning, Attention-based graph embedding, Social media

1 Introduction

Social media and their popularity and ease to use increased substantially people connectivity. Opinions on different life and society topics (scientific, ethic, political, religious, etc.) are expressed, shared and discussed more than never. Expressed opinions often trigger fierce and sometimes endless debates, and frequently cause polarization and controversy. A controversial content can simply be defined as any content that attracts both positive and negative feedback [1]. Indeed, polarization stigmatizes user's behaviors in presence of controversial topics [2–4].

Automatic controversy detection can be helpful. For instance, people can be warned by the existence of a controversy to add some nuance to better understand some issues. Objective information could also be brought to people to prevent misinformation or hateful discussions [3]. Detecting a content as non controversial is also helpful as it shows that people agree on a given issue.

Automatic controversy identification is difficult and constitutes a challenging task as it should be done on a large number of continuously evolving posts covering a wide range of topics. This difficulty is increased by the fact that controversy is sometimes time-aware (what is controversial today was not necessarily controversial in the past) and community-aware (what is controversial in a community is not necessarily controversial in another community) [5]. Controversy analysis on Web pages or articles is usually based on features extracted from the content [6, 7]. However, on social media, the interaction between people is widely used to detect controversy. These interactions include social relation (retweet and follow on Twitter, comment on Reddit) [4, 8] and citation relation on Wikipedia [9].

Graphs, as a nonlinear data structure, are often used as a powerful tool to model complex network data in different fields including molecular networks, protein interactions, human interactions in social networks. Using graphs to model complex systems can help to characterize patterns, and then improve the performance of data analysis tasks such as, for instance, protein prediction and face recognition. Unfortunately, classical graph analysis techniques do not scale and present high computation cost in the case of large graphs due to their high dimensionality. Graph embedding techniques [10–12] are recently proposed for the need of transforming large, sparse and high-dimensional graphs into low-dimensional and dense vector spaces while preserving graph structure properties. Each graph node/edge is encoded in a latent low-dimension

vector space with a smaller dimension. Classical downstream graph analytic tasks can then be performed on the learned latent graph embeddings instead of performing them on the original graph. As a consequence, graph analytic are computed much faster and more accurately.

Figure 1 illustrates an example of discussions on two social media. The left side of the figure concerns Reddit and is about death penalty. It shows that a user can directly react to a post or a comment by writing a support/against/neutral text. In some cases, the user text does not include an explicit opinion. The discussion on Reddit can be seen as a post-comment tree with the initial post as the root, and the text is connected to the text it comments. The right side of Figure 1 concerns Twitter and is about Netanyahu speech. Users retweet tweets when they agree on its content. Expressing their own opinion needs a new tweet which is not connected to the first one. Social media can differ in many aspects, including their way to manage privacy (more important on Reddit than on Twitter) or the quality of delivered information (more verification on Reddit than on Twitter). But the most important difference regarding controversy analysis is probably the action proposed to users, which will define their way of conduct. For instance, the retweet is a fundamental action on Twitter and allows a user to express endorsement to tweet content. Such action is not offered by Reddit where only a comment action is allowed to express his/her opinion on a given post/comment. It is then not easy to design a controversy approach that works on any social media, and existing approaches are in general specific to only some social media.

Fig. 1 Toy example of 2 controversial topic situations on Reddit and Twitter. Left: Reddit discussion, where controversial post leads to discussion/debate and users respond to other users under comments. Right: Twitter discussion, where users "agree" with other users on their point of view (when we only refer to the retweet action). Therefore, we have less discussion about POV, but more user standing by lead users of their respective community.

The originality of our approach firstly resides on using very recent state-of-the-art GNN methods to embed user nodes in a low d-dimensional space and take into account structural information (such as local/global information from neighbor nodes), and represent the final graph representation used for the classification task using different pooling approaches. On Reddit, initial text representations of a user are learnt from their comments on the current post. On twitter, tweets are used as input of our GNN-based approaches. To the best of our knowledge, only Zhong et al. started to use GNNs to represent the graph

on one type of social media - Reddit - building their post-level controversy detection method by directly exploiting the comment-tree structure [13]. Our work, on the contrary, exploits the user's interaction graph built from either the comment-tree structure (Reddit case) or retweet graph (Twitter case) and compare multiple state-of-the-art (GNNs) methods to combine both structural and content information.

Contribution. This paper is an extended version of our work published in [14], including an extended version of the literature, a generalisation of the approach to a second social media (twitter) and more experiments using sampling methods on the twitter graph. We consider controversy detection as a graph classification problem when most of existing approaches, except [13], are based on clustering techniques. We explore the use of GNN techniques to classify the entire graph that represents a discussion, but instead of defining a graph of texts (which text is connected to which text, user entity is absent) as done in [13], we opted for a graph of users to represent who interacted with whom. We also associated to each user node the set of texts a user wrote to represent who said what and use BERT model to capture the local information of texts. We then proposed to use two different strategies to embed the entire graph. The first strategy exploits hierarchical links that may exist between user nodes. Graph information is aggregated across the edges iteratively and in a hierarchical way. We rely on the DIFFPOOL approach to encode the whole graph by stacking several pooling layers [15]. The second strategy is based on an attention mechanism [16]. Each user node judges which user neighbor is more or less important than the other, during the node embedding process, according to the structure and the features of the graph. The experiments we conducted on Reddit and Twitter platforms show that embedding user graph by only exploiting user interactions on a given topic gets good performance compared to our baseline. The exploitation of local information via Bert model applied to texts of each user helps to improve the performance of the approach.

The rest of this paper is organized as follows. Section 2 provides an overview of related work. Section 3 describes the different datasets collected from Reddit and Twitter platforms. Section 4 presents our approach to automatically detect controversy on social media. Its different stages are described and formalized. Section 5 presents the performance experiments and discusses the obtained results for both Reddit and Twitter social media. Section 6 concludes the paper and highlights some future work.

2 Related Work

This section presents an overview of the different works related to the controversy analysis problem in the context of social media, and the main GNN-based text classification approaches.

2.1 Graph Information-based Controversy Detection

Graph information-based controversy analysis approaches are mainly based on three steps: graph building to represent a topic discussion in terms of nodes and interactions between these nodes, graph clustering to partition the built graph into two disjoint sets which probably represent the two sides of the discussion, and controversy measure to quantify to what extent a topic is controversial. In [4], Garimella et al. introduced a random-walk-based controversy measure adopted now in different works. It also studied the controversy analysis on four types of graphs: a retweet graph built to represent who agrees with whom on a given topic, a follow graph which focuses on who is connected to whom on the social media, a content graph which links users who share some keywords, and a sentiment graph is built to represent who share with whom the same opinion on the topic. The study shows the usefulness of the retweet graph compared to the others graphs information. Although the proposed approach is language and domain independent and can then be applied easily to any topic discussion, it nevertheless presents the drawback in order not to take advantage of extra information that represents the graph attributes (number of tweets per user, number of followers, etc.).

This drawback is addressed in some approaches. For instance, Emamgholizadeh et al. propose in [17] to analyse controversy in attributed graphs, graphs with extra information on the nodes and/or edges. A biased random walks controversy measure is proposed to take into account all added information. In the same way, Hessel and Lee in [1] combine structural features (number of comments, max depth/total comment ratio, average node depth, etc.) of a post-comment tree of a Reddit discussion with textual features outputted by language models such as BERT [18]. In [19], Mendoza et al. considers the entities involved in a given discussion. Comments of users on relevant named entities, eventually polarized, that appear in the news are exploited to infer the tendency nature (positive, negative, neutral) of these users towards the named entities. The user graph generation is then conditioned on the user-named entity relationship. In [20], authors exploit the vocabulary associated to the communities to measure the controversy by detecting if the discussion has one or two principal jargon in use. All these approaches show that the structure of the graph alone is not sufficient to quantify the controversy level in some cases, and using additional extra information can help.

It is known that controversy analysis faces the echo-chamber phenomena, the situations where like-minded people reinforce each other's opinion and may not have the opportunity to be exposed to the opposite views. To limit the impact of such echo-chamber, Garimella et al. propose in [3] to reduce the random walk controversy score by modifying the structure of the retweet graph. New edges are automatically added to materialize connections between users with opposite views. Controversy quantification reduction is formally defined as an edge-recommendation problem and aims to compute a fixed number of new edges between existing nodes that minimise the random-walk controversy score. Experimental evaluation shows that adding edges between nodes with

highest node degree performs the highest decrease of controversy score. The interesting point of this approach is that it succeeded to decrease the impact of echo-chamber phenomena in a language and domain independent manner, just by computing who can be connected to whom from the opposite side. Unfortunately, it is not clear how this approach can be adopted when the graph includes diverse information such as tweets content, user profile, etc. Reducing the echo-chamber phenomena remains an open challenge in controversy analysis approaches that combine structural and textual information. Our proposed approach does not tackle this important issue. Most of the graph-based controversy analysis approaches suppose that polarized discussions are characterized by modular communities. In [21], Guerra et al. reject this hypothesis and consider that the traditional modularity polarization metric is not necessarily a sufficient metric since non-polarized networks may also be divided into modular communities. A community boundary based on polarization metric is defined to better differentiate polarized and non-polarized networks. A concentration of high-degree nodes in the boundary of communities is proposed to characterize polarized and non-polarized communities. They empirically demonstrate that polarized communities are more likely to exhibit low concentration of high-degree nodes along the boundary.

Stance detection is also used to quantify controversy [22]. Stance detection aims to identify the position (Supporting, against, neutral, ...) of a user towards a given entity (topic, person, etc.). A subset of users are labelled with their stances. These labels are obtained manually for some users, the very active ones, and automatically for others by using label propagation method along with user retweet behavior. Labelled users and their different features types are used to train classification models (SVM and FastText) to guess the stances of the unlabeled users. Stances of users are used to cluster users prior to apply controversy measures to quantify controversy. This approach is interesting and exploits recent techniques on classification and label propagation but unfortunately it presents the drawbacks of needing to manually label some users and favor active users over non active ones.

All the above approaches are based on the clustering technique. In [13], Zhong et al. consider controversy detection as a graph classification and rely on Graph Convolutional Networks (GCN). A reddit graph is built in which each node corresponds to text (post or comment) and an edge between two nodes exists when one comments the other one. Convolution operation is used to generate node embeddings. These latters are aggregated to represent the graph embedding. Classification layer determines then if the embedded graph is controversial or not. Authors also argue that exploiting only semantic and structural features of comments attached to the target post is not sufficient and propose to exploit the related posts on the same topic. This approach is the first study which focuses on GNN for controversy analysis and provides good results. Nevertheless, it presents some limits. The comment-tree structure of a post completely ignores the user who wrote the post/comments, and may not

include other types of user behavior such as retweet. The graph embedding strategy is simple (aggregation of all node embeddings) and considers all texts (nodes) at the same level. The use of inter-topic relation might interfere too much with the main detection task.

Our approach considers the controversy detection in the context of social media. It can work on Reddit, Twitter and any other social network. In term of data modeling, interactions between users and exchanged messages/texts are represented as a user graph. Such a graph is extracted from the social media itself, namely from the post-comment tree of a Reddit discussion, the retweet actions of users, etc. Texts that a user exchanges with others are embedded with BERT technique. [13] relies on advanced graph representation learning, like our approache, using GNN techniques to embed the whole graph including the text features. Plus, two different controversy strategies are proposed during the graph embedding: hierarchical graph representation learning and attention-based mechanism.

2.2 GNN-based Text Classification

Controversy detection, as many other domain applications (Fake news detection, product name categorization in e-commerce, etc.), can also be studied from the text classification perspective. Text representation learning is an important phase in text classification and is performed by using different deep learning models such CNN, RNN and more recently GNN.

TEXT GCN, a GNN-based text classification method, is proposed for the first time in [23]. Text classification is viewed as a node classification problem, and a single heterogeneous text graph is built from the entire corpus with two types of nodes (word nodes and document nodes) and two types of weighted edges (word co-occurrence between two word nodes, word frequency between word node and document node). A two-layer GNN exploits the built graph to learn word and document embeddings and to classify documents. Although the use of GNN for text classification is interesting and produces interesting results, unfortunately it suffers from two limitations. The method is not adequate for short texts, as short texts present sparse textual features. The method captures global information present in the entire corpus and unfortunately ignores local information that could be present in each sentence. Short texts related limitation are addressed by mainly integrating additional information: extracted topic information is added to the document representation as a new type of node [24], similarity scores between user behavioral data are added as a new edge between short text sample nodes [25]. Local information-related limitation is addressed in [26] by avoiding to represent the entire documents by a single graph. Authors opt to represent each document by an individual graph and to train GNN on a subset of documents to discover word-word relations. These latters are then generalized to the new documents in the test set. The same representation, one graph per text, is adopted in [27, 28] but the GNN

model is coupled with Bert model for each text to represent both semantic and structural information of each single text.

2.3 Discussion

To the best of our knowledge, [13] is the closest work to ours and the only approach that considers controversy detection as a deep graph classification problem. Our approach is different from the approach presented in [13] from mainly two perspectives: graph data representation perspective and graph embedding perspective. **From the graph data representation perspective**, authors of [13] consider each post or comment as a node in its own, and an edge indicates which comment comments which post/comment. Such graph aims to represent what is said, related to what. The user entity is completely ignored. This means that each node embedding corresponds to a contextual embedding of one and only one text (post or comment), and each text is embedded without taking into account who said what. In our data graph representation, the user entity plays an important role and all texts (post, comment) belonging to the same user are gathered and associated to its user. The texts of the same user are contextually embedded as one text unit. Our graph representation correspond to who said what. Ignoring user entities in graph representation as presented in [13] limits the possibility of the approach to include additional information. It will not be possible for instance to include without redundancy profile and meta information of users in the node features. Our approach allows this and can include any data that could describe users. Authors of [13] do not consider how their graph representation can be adopted for the case of Twitter Platform. We think that the graph representation in [13] is not appropriate for the Twitter platform for two main reasons: (1) it is not obvious to represent Twitter actions such as retweet and follow in the graph representation, and (2) representing each tweet as a node will increase the computation cost of the graph embedding. Our graph representation is appropriate for both Reddit and Twitter platforms. **From graph embedding perspective**, authors of [13] adopted a GCN-based technique to embed nodes and aggregate then the obtained node embeddings to compute the whole graph embedding. In our work, we propose two strategies to embed the whole graph. The first strategy exploits the hierarchical links that could exist between nodes and graph information is aggregated across the edges iteratively and in a hierarchical way. To do so, 2 GNNs are used at each layer. The first one is used to learn the node embeddings and the second one is used to learn how nodes can be hierarchically coarsened. The second strategy we propose exploits the attention mechanism to allow user nodes to select a subset of neighbors during the embedding process according to the structure and the features of the graph. A pooling function is then applied to construct graph representation. The attention mechanism is also used in [13] but only for the need of inter-topics detection to disentangle features related and not related to the topic.

3 Data sources

As we explained in Section 1, each social media has is own code of conducts, features that represent different user behaviors. For example, Reddit is considered a forum, where everyone comments directly on a post, or produces one. Twitter is considered a micro-blog, where everyone can post but also endorse point of views of other users, by sharing their post (e.g Retweet). Moreover, users can have different feelings about those social networks, where communities can have more confidence in one rather than another. That is why we decided here to evaluate the performance of our approaches with different datasets, on 2 different social media. We present, on the next part, the two used social media.

3.1 Reddit dataset

We first exploit different real-world Reddit dataset, in English, released by Hessel and Lee [1]. The same dataset is also covered by Zhong et al. [13].

Reddit is an American social forum. User can submit content to the site such as links, text posts, images, and videos, which are then voted up or down and commented by other members. Posts are organized by subject called subreddits, which cover a variety of topics such as news, politics, religion, science, movies, etc.

The collected data covers a period from 2007 to February 2014. It contains 6 specific online channels (also called subreddit): *AskMen* (AM), *AskWomen* (AW), *Fitness* (FN), *LifeProTips* (LT), *personnalfinance* (PF), *relationships* (RS). On Reddit, each user can comment on a post (threads) which is related to a specific topic (subreddit). Each subreddit contains a set of posts. Meta-data and a tree-comment structure are associated to each post. Only comments that were written at most 3 hours after the concerned post are used. Finally, only posts with a total of at least 30 comments are kept, assuming that less than 30 comments are not enough to build a significant graph. Each post is automatically labelled controversial or not controversial, depending on various post meta-data [1], among them the ratio between up-votes and down-votes [1]. Indeed, here posts that have received many up-votes and down-votes should be considered as the most controversial. The data is separated into 6 datasets, corresponding to each subreddit. Each subreddit s gets a set of user graphs $\mathcal{G}_s$, with one graph per post (each set having at least 1000 posts).

3.2 Twitter dataset

Twitter is a micro-blogging and social network, where users post and interact with tweets. Users can post, like, retweet and quote different tweets. As explained earlier, Twitter presents many advantages compared to other social media. First, Retweet is considered as an endorsement, meaning that a user agrees and supports the contain of the original tweet. Moreover, Twitter is one

[1] Up-vote and down-vote indicate agreement and disagreement on the post.

Table 1 Information on data retrieved in [4] for each different topic. The first 9 topics represent controversial topics, where the last 6 non-controversial topics.

Topic	Tweets retrieved	Rate T-found	main #	Description (2015)
LEADERSDEBATE	693 281	60.9%	#leadersdebate	Debate during the U.K. national elections, May 3
RUSSIA_MARCH	51 395	43.3%	#russia_march	Protests after death of Boris Nemtsov ("march"), Mar 1–2
UKRAINE	155 258	54.0%	#ukraine	Ukraine conflict, Feb 27–Mar 2
INDIANA	68 008	58.4%	#indiana	Indiana pizzeria refuses to cater gay wedding, Apr 2–5
BEEFBAN	46 654	55.2%	#beefban	Government of India bans beef, Mar 2–5
NETANYAHU	113 343	44.5%	#netanyahuspeech	Netanyahu's speech at U.S. Congress, Mar 3–5
INDIASDAUGHTER	92 289	55.0%	#indiasdaughter	Controversial Indian documentary, Mar 1–5
BALTIMORE	90 908	41.7%	#baltimoreriots	Riots in Baltimore after police kills a black man, Apr 28–30
NEMTSOV	105 379	57.4%	#nemtsov	Death of Boris Nemtsov, Feb 28–Mar 2
ONEDIRECTION	190 662	38.0%	#1dfamheretostay	OneDirection concert, Mar 27–29
SXSW	233 810	68.0%	#sxsw	SXSW conference, Mar 13–22
MOTHERSDAY	1 033 839	57.5%	#mothersday	Mother's day, May 8
GERMANWINGS	561 836	61.9%	#germanwings	Germanwings flight crash, Mar 24–26
NEPAL	772 618	59.5%	#nepal	Nepal earthquake, Apr 26–29
ULTRALIVE	187 920	51.6%	#ultralive	Ultra Music Festival, Mar 18–20

of the most used social media for public debates and is also used to report news about current events. Therefore, the available data is bigger than any other social network. It allows us to work with more global information and reduce the variance of our data. However, it is complicated to capture only tweets of a particular topic, so the noise (tweets not belonging to the concerned topic) can be a disadvantage.

We retrieved our data from [4]. Indeed, the authors selected 20 different topics (10 controversial and 10 non-controversial), related to different events or public persons. After selecting the common hashtags related to each topic, we proceed to data collection, using these hashtags as queries via the twitter API. The tweets collected by the authors were published between February 27 and Jun 15 2015. However, the authors have released only tweet ids for 15 topics, and 9 topics of these 15 are controversial [2]. Moreover, not all tweets are still present on Twitter databases. Different reasons could explain that, as they could have been deleted by their respective authors or by moderators. As we want to compare our performance with and without incorporating textual features to users, we decided to only create graphs from tweets retrieved, for more fairness. We then collected all tweets that were still available, from the tweets ids released in [4], and create future graphs from it. We got an overall 56.5% rate of tweets found, with a 293 146 average tweets by topic, which is considered good enough to capture the structural information for our task. Different statistics of the data we used are presented on table 1.

From this data, we create user retweet graphs, representing, for each topic, a graph where users (nodes) are related to each other if one has retweeted another at least once (edge). We suppose that this graph representation will help us representing community cluster on controversial graphs, as explained in [4]. Using the same method in [4] with metis [3], graphs are partitioned into 2 communities, to see if the tweets retrieved for each topic is enough to capture the structural information. Figure 2 shows a controversial retweet graph (Netanyahu speech) which presents a well-separated graph, compared to the non-controversial graph about the "sxsw" conference.

[2] Twitter dataset is available at https://github.com/gvrkiran/controversy-detection

[3] library for serial graph partitioning and fill-reducing matrix ordering

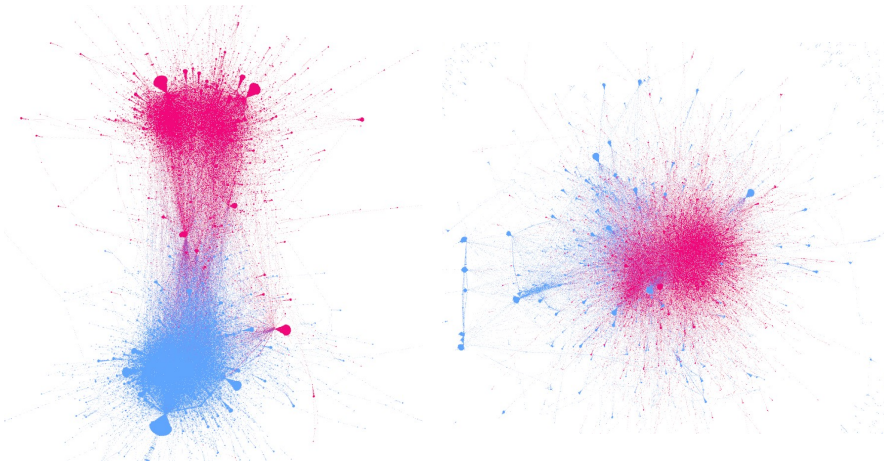

Fig. 2 left: Controversial retweet graph representing topic "Netanyahu". Right: Non-controversial retweet graph representing topic "sxsw". Both graph are represented using ForceAtlas2 [29] algorithm for spatial visualisation.

4 Graph Neural Network-based Controversy Detection Approach

This section describes our post-level controversy detection approach. The main idea is to exploit both text content and user interactions by representing the Reddit or Twitter discussion as a user graph and exploring advanced GNN embedding techniques. The method is based on Graph Neural Network (GNN) to improve the structural feature representation of the graph. Figure 3 presents an overview of our approach. We divide our pipeline into four sequential stages: Graph Building, User Feature Extraction, Graph Embedding, and Graph Classification.

The graph building stage represents data extracted as a user graph. We represent the initial comment-post tree as a graph on Reddit and the Retweet flow as a graph on Twitter. Nodes represent users and edges correspond to the interactions that exist between users. Each node can be represented by different features representing the user, such as user-id, age, location, texts, etc. In our case, only user textual comments/posts will be used. The user feature extraction stage enriches graph nodes by adding textual embedded features. These features are computed by using state-of-the-art NLP techniques. This allows to better interpret the texts that users sent out. The graph embedding stage computes the embedding of the whole graph. Different advanced GNN-based graph representation learning techniques are used, namely DIFFPOOL [15], GCN [10], and GAT-GC [16]. Finally, the graph classification stage predicts the binary label associated to the whole graph, that is to classify a post as controversial or not.

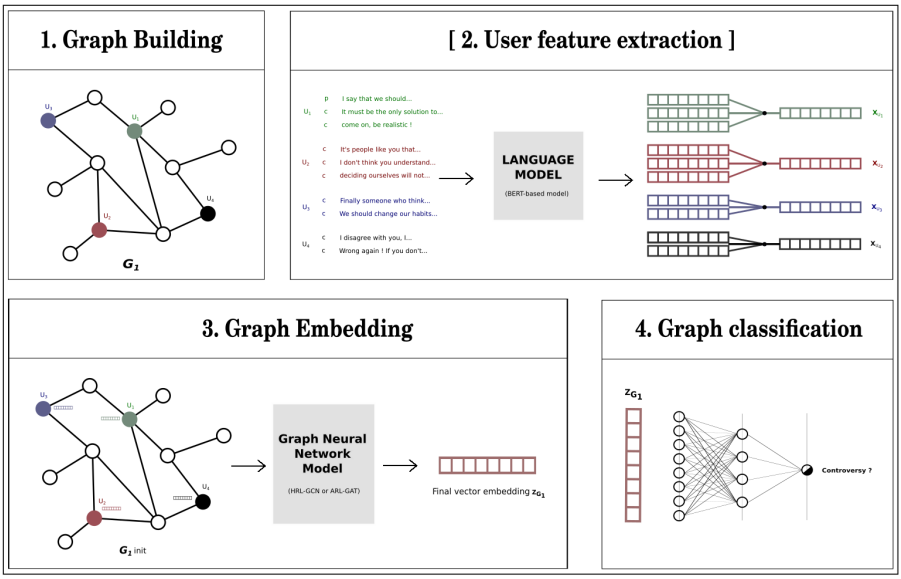

Fig. 3 Overview of the 4 different stages of our controversy detection approach.

4.1 Controversy detection Problem Definition

Controversial content refers to any content that attracts both positive and negative feedback. Considering that data related to a discussion is represented as a graph (graph of texts, graph of users), controversy detection aims to classify any new graph based on graphs already labeled as controversial or not controversial. More formally, we define controversy detection problem as follows:

Definition 1 Let G be a set of graphs, and $G^L = \{(G_1, l_1), \ldots, (G_i, l_i), \ldots (G_n, l_n)\}$ be a set of labeled graphs: $G_i \in G$ is a graph representation of a given discussion, and l_i its corresponding label (1 if controversial, 0 otherwise). The controversy detection problem is defined as a classification problem, and our objective is to learn a mapping function $f : G \rightarrow L = \{0, 1\}$ that assigns a label (0 or 1) to each unlabeled graph based on the labeled graphs G^L.

4.2 Graph building

Existing controversy detection methods [1, 13] on **Reddit** use the classic comment-post tree representation as they mainly focus on the structure of the discussion. However, many research works have established that user interaction can be helpful to extract different features on social media that can improve the controversy detection. In this work, we adopt a graph representation of a discussion to highlight these user interactions. Given a discussion on a post (thread) p extracted from a subreddit s, we build an undirected graph

where a node u_i represents a user involved in the discussion. An edge (u_i, u_j) is created when a user u_j responds to the post p or any comment posted by u_i. It is worth mentioning that our graph representation does not allow multi-edges. This means that when users u_i and u_j respond to each other, only one undirected edge is represented. Indeed, some investigations we conducted (visualization of the tree-structure, textual comparison of post-comments and comment-comments) showed that differentiating different edges between two nodes do not necessary add more useful information that we can expect it.

Concerning **Twitter** data, multiple user graphs can be created for the data made available by the API, such as follow graph, retweet graph, quoted graph, or even hybrid graph combining multiple types of edges. To get closer to experiments on Reddit, we decided to stay with the creation of retweet (RT) graphs. To remain consistent with Reddit notation, we build, for each topic p, an undirected graph where a user node u_i is linked to another user u_j if one have retweeted the other at least once. RT graphs have interesting properties, such as good representation of user interaction and community separation, as figure 2 shows. As explained in the section 3, Twitter graphs are too large to be processed correctly on our models, so we sample each of our graph, in order to reduce the sample size. Sampling experimentation will be explained in more details in the experimentation section.

More formally, a post (topic for twitter) p is represented as a graph $G = (\mathcal{U}, \mathcal{E}, X)$ where $\mathcal{U} = \{u_1, u_2, ..., u_n\}$ denotes the user nodes, $\mathcal{E} = \{(u_i, u_j)\}_{1 \leq i,j \leq n}$ denotes the edges of the graph, and $X \in \mathbb{R}^{n \times e}$, e being the feature dimension, denotes the user node features matrix. Each node corresponds to a unique user, and an edge between two nodes exists if there are interaction links between the corresponding users. The computation of the matrix X is described in section 4.3.

4.3 User feature extraction

In order to bring valuable information to the graph representation, user features are extracted from the posted texts by using advanced NLP techniques. Recently, different NLP language models pre-trained on a large corpus have been proposed to improve the dynamic text representation, such as BERT-based models [18]. The user features extraction is performed for each user as follows.

For **Reddit**, each message (post or comment) a user u_i posts is firstly cleaned (reddit tags and url link removed). and is then embedded in an e-dimensional vector by using a language model based on BERT. Then, as Figure 4 shows, the text will be split into different tokens, using a special tokenizer [4] with an already set vocabulary list, plus special tokens [CLS] and [SEP]. The [CLS] tokens represent the classification token, the one used when applying classification task with this model. Then all tokens will go through the BERT-based language model with multiple self-attention layers [18] and

[4]We used the 'bert-base-uncased' tokenizer stored by https://huggingface.co/

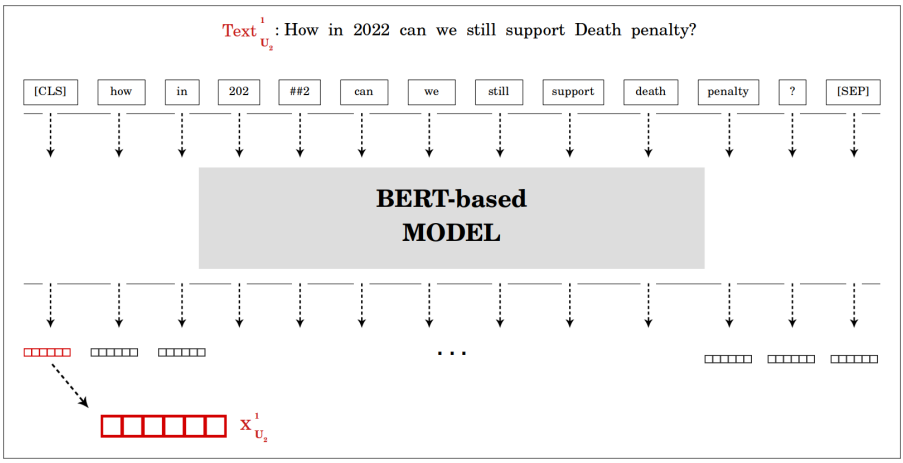

Fig. 4 Example of how a comment c of a user u_2 is embedded into a vector of dimension e.

return an e-dimensional embedded vector for each of this token. Finally, we use the [CLS] token embedding as the vector representation of the text. The embedded vectors obtained from the different messages posted by a user u_i are aggregated to form the final user features x_{u_i} as shown in equation 1.

$$x_{u_i} = \text{AGGREGATION}\left([x^0_{u_i}, x^1_{u_i}, ..., x^m_{u_i}]\right) \tag{1}$$

where $x^j_{u_i} \in \mathbb{R}^e$ is the individual e-dimensional embedded vector computed from the j^{th} message of user u_i and m is the number of messages a user u_i posted. The aggregation of the embedded text vectors is performed via the Max-pooling function, but any other aggregation function can be used. Concerning **Twitter** graphs, each user u_i is represented only by its original tweets. After pre-processing and cleaning those original tweets, we passed them through the language model and, as for Reddit users, aggregate them to get the user feature vector x_{u_i}. If a user does not have any original tweet on the current topic (he only retweeted tweets of other users), we set a default vector which will be represented as their features. Features of each user u_i is stacked on a matrix $X \in \mathbb{R}^{n \times e}$. The user graph $G = (\mathcal{U}, \mathcal{E}, X)$ is now fully represented and includes node textual features. It will also be referenced by (A, X) where A represents its adjacency matrix.

We can now focus on adding structural information in our graph representation. For the next couple sections, we consider Reddit and Retweet graphs as the same.

4.4 Graph embedding

The graph embedding stage aims to encode the whole user graph in a low-dimensional vector. This latter will fed the graph classification stage to predict if a post is controversial or not. Recently, different GNN-based approaches were

proposed to adapt deep learning architectures to the graph structured data [10, 11]. The main idea is to consider each graph node as a computation node, and to learn classical neural network primitives that compute node embeddings.

This stage relies on GNN architectures with the objective to exploit both user node features computed in the previous stage and the user graph structure. The output is the embedding of the whole graph denoted by z_G. Learning individual node embeddings denoted by z_{u_i} is also performed as an intermediate stage. We propose in this paper two main strategies to embed the whole graph for the controversy detection needs. These strategies rely on hierarchical representations of graphs, convolutional networks, and attention-based graph representations.

4.4.1 Hierarchical graph representation learning-based strategy.

Classical GNN for graph encoding are known to be flat. They only propagate information through edges before using a simple aggregation-pooling layer (such as MAX or MEAN pooling) of all resulted node embeddings. The strategy we propose in this section exploits hierarchical structure that may exist in the user graph structure. Thus, in the whole graph encoding process, graph information are aggregated across the edges iteratively and in a hierarchical way. At each iteration, user nodes who interact with each others are grouped. By doing so, we give more importance to the interaction information held between users. This could help the model to understand some behaviors, and capture important features to classify the topic as controversial or not. We rely on the DIFFPOOL [15] approach which encodes the whole graph by stacking several pooling layers. Each pooling layer is composed of two distinct GNN: one, called GNN_{embed}, learns user nodes embeddings H, and the other, called $\text{GNN}_{pooling}$, learns an assignment matrix S that indicates which user nodes are assigned to which cluster. The matrix S is used to coarsen the graph.

As depicted in Figure 5, the functioning of the pooling layer at level (l) is described as follows:

1. *Node embedding generation.* We first apply the $\text{GNN}^{(l)}_{embed}$ to the graph obtained at level (l) represented by its adjacency matrix $A^{(l)}$, and its node features matrix $H^{(l)}$. As described in equation 2, the result is an intermediate node embeddings $H'^{(l)} \in \mathbb{R}^{m \times d'}$, with m number of nodes of the initial graph of the layer, and d' the new dimensional features.

$$H'^{(l)} = \text{GNN}^{(l)}_{embed}\left(A^{(l)}, H^{(l)}\right) \tag{2}$$

2. *Matrix cluster assignment learning.* We then use the $\text{GNN}^{(l)}_{pooling}$ to learn a new assignment matrix $S^{(l)}$ to indicate which nodes of the graph at layer (l) will be clustered together to form a new coarser node at layer (l). The

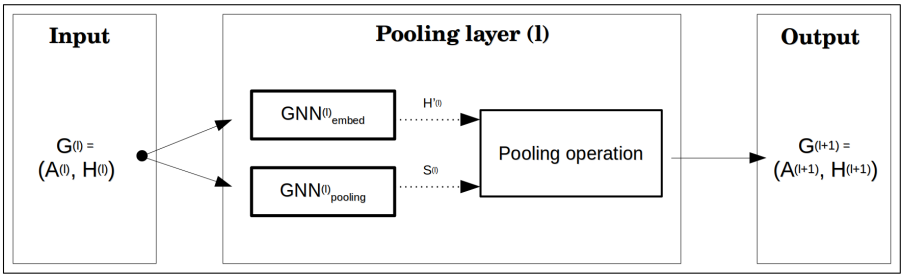

Fig. 5 Diffpool-based Pooling layer architecture. $A^{(l)}$ and $H^{(l)}$ represent the adjacency and feature matrices of the input graph at layer (l) respectively. GNN_{embed} and $\text{GNN}_{pooling}$ are the two GNN blocks used to respectively compute node embeddings $H'^{(l)}$ and assignment matrix $S^{(l)}$. The pooling operation block converts the input graph $(A^{(l+1)}, H^{(l+1)})$ into a new coarsened graph $(A^{(l+1)}, H^{(l+1)})$.

matrix assignment is represented by the equation 3:

$$S^{(l)} = \text{GNN}^{(l)}_{pooling}\left(A^{(l)}, H^{(l)}\right) \tag{3}$$

3. *Nodes and features pooling.* We finally aggregate nodes belonging to the same cluster and their features from $H'^{(l)}$ using the assignment matrix $S^{(l)}$ to output a new coarser graph represented by its adjacency matrix $A^{(l+1)}$ and features matrix $H^{(l+1)}$. This pooling operation is done as follows:

$$A^{(l+1)} = S^{(l)^T} A^{(l)} S^{(l)} \tag{4}$$

$$H^{(l+1)} = S^{(l)^T} H'^{(l)} \tag{5}$$

Figure 6 shows how an example of a graph is represented through the different steps of a layer in our Hierarchical graph representation learning-based strategy. The new output coarsened graph represents the graph input for the next layer, where node embeddings has been computed using the H and S matrix of the current layer.

We can notice that the number of nodes is decreasing at each new layer (l). At the first layer (l=0), A^0 and H^0 correspond to the adjacency matrix A and the feature matrix X of the initial user graph respectively. The last layer L is a single cluster node, and represents the final vector embedding z_G of the whole graph.

In our work, only one kind of GNN is used: Graph Convolutional Networks (GCN) [10].

GCN. Graph Convolutional Networks (GCN) [10] achieved state-of-the-art performance in several benchmark dataset, and can accurately represents on a d-dimensional space local neighborhood of a node, in addition to encoding node features. GCN, based on spectral graph theory, aggregate neighbors representation with itself using Adjacency and Degree matrix (respectively A and D) of our user graph G to get a latent representation of each of its node. We

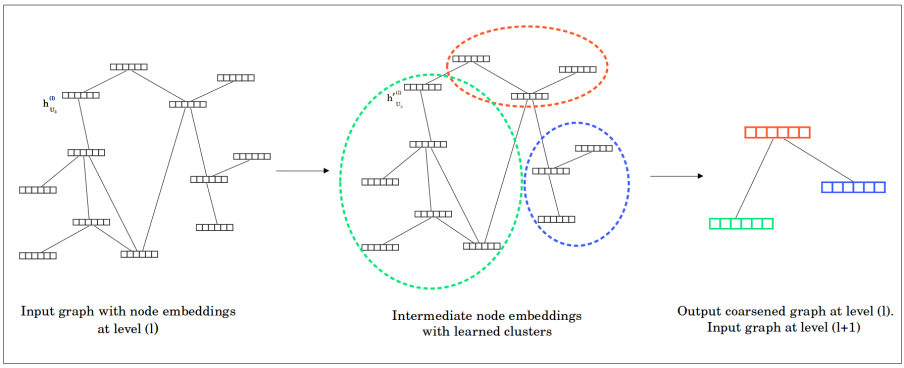

Fig. 6 Example of how the pooling aggregation works on one layer for a given graph.

apply here a linear transformation to learn a weight matrix $W^{(l')}$ for each layer l'. We update all node representation using the following propagation rule:

$$H^{(l'+1)} = \sigma(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(l')}W^{(l')}) \tag{6}$$

where $\tilde{A}$ and $\tilde{D}$ are respectively the adjacency and degree matrix summed up with the identity matrix (with a self-loop connection to also represent its old embedding). $H^{(l'+1)}$ is the current new representation of all nodes, and $H^{(l')}$ the previous one (at the first layer, $H^{(0)} = X$). σ is an activation function ($ReLU$ in this case, except for the last layer where a $Softmax$ function is used).

In our work, we apply a 2-layer GCN, to have a good balance between both local and global information in our representation. $H^{(2)}$ will be our final node matrix representation, where $H^{(2)} \in \mathbb{R}^{N \times d'}$, d' being the new node feature dimension.

4.4.2 Attention mechanism-based user pooling strategy

This strategy is based on attention-based node embedding and allows each user node to judge which user neighbors are more important than the others during the node embedding process, according to the structure and features of the graph. Once node embeddings are generated, they are aggregated to produce the node embedding of the whole graph. The attention mechanism (GAT) with cardinality preservation [16] is used to differentiate user neighbors by assigning them different scores. This attention mechanism is similar to the Transformers block used in BERT [18] for language modelling.

Let $\tilde{\mathcal{N}}_{(u_i)}$ be the multi-set of first-order neighbors of node u_i, including u_i itself. This second strategy is described as follows:

1. *Neighbors attention score.* For each node $u_i \in \mathcal{U}$, and each user neighbor $u_j \in \tilde{\mathcal{N}}_{(u_i)}$, we first compute the attention score $e_{u_i u_j}$ by using an attention function a on transformed features represented by the matrix $W^{(l)}$ of the current layer (l) for both nodes, as described in equation 7:

$$e_{u_i u_j}^{(l)} = a\left(\mathbf{W}^{(l)} h_{u_i}^{(l)}, \mathbf{W}^{(l)} h_{u_j}^{(l)}\right) \tag{7}$$

2. *Attention scores normalization.* We then normalize scores using a softmax function to get a probability distribution of each score.

$$\alpha_{u_i u_j}^{(l)} = softmax(e_{u_i u_j}^{(l)}) = \frac{\exp(e_{u_i u_j}^{(l)})}{\sum_{u_k \in \tilde{\mathcal{N}}_{(u_i)}} \exp(e_{u_i u_k}^{(l)})} \tag{8}$$

3. *User node embedding.* The normalized scores are then used to compute the new user node representation $h_{u_i}^{(l+1)}$

$$h_{u_i}^{(l+1)} = \sigma\left(\sum_{u_j \in \tilde{\mathcal{N}}_{(u_i)}} \alpha_{u_i u_j}^{(l)} \ \mathbf{W}^{(l)} h_{u_j}^{(l)}\right) \tag{9}$$

with σ a non-linear activation function, $h_{u_i}^{(l+1)} \in \mathbb{R}^{n \times d}$ with d feature dimension of the layer. Note that the cardinality preservation allows in 9 to scale the result before the use of the activation function σ. The final node representation h_{u_i} corresponds to the output of the last layer $h_{u_i}^{(L)}$.

4. *Graph embedding.* Finally, we compute the final graph embedding z_G by applying the READOUT function. A simple graph-level pooling function is used: we sum up each node representation at each iteration layer, and then concatenate them as shown in equation 10.

$$z_G = \Big|\Big|_{l=0}^{(L)} \left(\text{READOUT}\left(\{h_{u_i}^{(l)} \Big| u_i \in U\}\right)\right) \tag{10}$$

This attention mechanism presents multiple advantages, in addition to state-of-the-art results in many benchmark graph classification tasks and interpretability. It allows the use of fixed number of parameters, and therefore does not depend on the graph size. It also presents transductive and inductive capabilities.

4.5 Graph Classification

The graph classification stage aims to classify the post represented by its graph embedding as controversial or non-controversial. To do so, we simply rely on a classic multi-layer perceptron classifier with the vector z_G as input. A Softmax activation on the output layer of dimension 2 is used.

5 Experimental Evaluation

We implemented our approach and conducted experiments on both Reddit and Twitter to show the applicability of our approach to different social media.

5.1 Reddit experiment

5.1.1 Datasets preparation

Data we collected from Reddit and presented in Section 3.1 are used for the need of our Reddit experiments. We first separate our data according to the 6 subreddits. For each subreddit s, we create a set of user graphs $\mathcal{G}_s$, one graph per post, each set having at least 1000 posts. We then define $\mathcal{G}_{s,train}$ and $\mathcal{G}_{s,val}$ as our train and validation graph set respectively, all equally balanced between controversial and non-controversial posts. Considering all aspects and few more experiments, we only evaluate our approach on the same validation set. The accuracy metric is used to compare the performance of our approach to some existing ones as the dataset is equally balanced. We separately train the NLP model for user texts representation learning and the GNN for structural information learning.

5.1.2 Baseline

We compared our approach with the following representative works on controversy detection. These works are selected for mainly two reasons. First, we share the same controversy detection objective with the aim to cover both textual and structural aspects. Secondly, the experiments are based on the same Reddit datasets. Note that those methods perform a k-fold to evaluate their performance, using average accuracy as their metric.

- **(POST (Text+Time))** [1]. It only focuses on the posts content. It uses language modelling based on BERT [18] and extra-features based on the post timestamp of the post.
- **(C-{Text_Rate_Tree} + Post)** [1]. It is based on a simple binary classifier. Textual embeddings of a post are combined with structural features of the comment-tree (average representation of text comments, depth of the tree, etc.) of the post and are used as input of the classifier. We compare post with comment during the first hour and the first three hours.
- **(DTPC-GCN)** [13]. It is based on a Disentangled Topic-Post-Comment Graph Convolutional Network. Controversial posts are identified by using GCN model and by learning structural features. The model is experimented for inter-topic detection, where all posts are gathered together in one dataset.

5.1.3 First experiment: Controversy detection based on structural information

We implemented our GNN-based controversy detection approach in PyTorch. The hierarchical graph representation learning is based on DIFFPOOL [15] and GCN [10]. We refer to it as **HRL-GCN** (Hierarchical Representation Learning based on GCN). We also tested our strategy by using one and two pooling layers, respectively. For each pooling layer, a 3-layer GCN is used. We rely on the same loss and optimizer functions used in DIFFPOOL experimentation [15]. The attention-based node learning is based on GAT [11], using GAT-GC method,

Table 2 Statistics on the 6 real-world balanced Reddit datasets.

	AM	AW	FN	LS	PF	RS
Number of posts	3305	2969	3934	1573	1004	2248
Average number of users by post	72	67	76	79	47	48
Average number of comments by post	144	141	159	132	95	98
Average number of words by comment	41	42	34	28	52	61
Ratio of comments with max tokens ≥ 256	2.68	2.64	1.61	1.03	4.1	6.17

Table 3 Performance comparison of our GNN-based controversy detection with baseline. Performance is evaluated using accuracy of the validation set. Values in bold represent the best accuracies among all methods (including baseline), while underlined values represent the best accuracies among our proposed methods.

	AM	AW	FN	LS	PF	RS
POST (TEXT+TIME)	68.1	65.4	65.5	66.2	66.5	69.3
DTPC-GCN	67.6					
POST + C-{TEXT_RATE_TREE} < 1 hour	71.1	70	68.1	67.9	66.1	65.5
POST + C-{TEXT_RATE_TREE} < 3 hours	**74.3**	72.3	70.5	**71.8**	**69.3**	**67.8**
ARL-GAT (MEAN-aggr)	65.7	69.2	**72.4**	58.4	53.7	62.9
ARL-GAT (SUM-aggr)	67.5	71	72.2	67	63.7	51.8
HRL-GCN (pool=2)	69	72.2	71.7	68.3	65.7	63.6
HRL-GCN (pool=1)	69.6	**74.6**	72.2	67.9	68.2	66.7

with the same hyperparameters used in [16]. We refer to it as **ARL-GAT**. We also tested this strategy with two different node aggregators to compute the whole graph embedding, namely MEAN and SUM pooling functions. Both GNN-based strategies are trained with a learning rate at 0.01, a batch size at 32 during 100 epochs. Table 2 shows statistics on the different datasets.

First experiments are performed without text representation to underline the importance of structural interaction between users in controversial discussion. Table 3 reports the accuracy results where the first four lines correspond to the baseline and the last four lines correspond to our experiments' results. The accuracy of the different methods are shown for each subreddit (AM, AW, FN, LS, PF, RS) we described in section 3, except for the DTPC-GCN method for which the accuracy is related to the whole dataset.

As shown in Table 3, our hierarchical approach HRL-GCN gets the best results among our experiments, with a weighted average accuracy at 70.6 using only one pooling layer. Our Attention-based approach ARL-GAT reaches 66.8 and 66.2 with the SUM and MEAN aggregator, respectively. HRL-GCN beats the DTPC-GCN [13] method and the hybrid method proposed by Hessel and Lee [1] with comments of the first hour for almost every dataset. Our proposed method (HRL-GCN, pool=1) gets around state-of-the-art results on several datasets, even going up to 74.6 accuracy in the AW dataset, beating results in C-{TEXT_RATE_TREE} + POST with comments of more than the first three hours. As the AM dataset is the biggest dataset, it could mean that our approach generalizes better when data are abundant, and less when data are sparce. As explained in [1], not enough comments are available for each dataset of the baseline. Indeed, when the subreddit *AskMen* (AM) has in average 10 comments after 45min, *Relationships* (RS) does not even have those after 3

hours. With more available comments, we might have a better embedding representation of our graph, which could lead to a better performance.

Table 3 shows that our attention-based approach ARL-GAT, combined with the MEAN-aggregator, performs well in the three first datasets, beating our best baseline method on FN, with an accuracy of 72.4 on the validation set. On the other hand, it under-performs on the other three, falling to 53.7 on PF. PF and RS already have low results on our baseline, which means that the data is difficult to understand. This could also be explained by the fact that those 3 subreddits have the least average number of comments (as shown in Table 2), and therefore each user node has less neighbors. Attention scores are in fact less useful in these cases. In general, higher average degree of nodes could lead to a better performance.

5.1.4 Second experiment: Controversy detection based on textual content and structural information

We conducted a second experiment to study the impact of adding textual node features to our GNN-based architecture. Instead of considering all options of our GNN-based architecture shown in Table 3, we only considered our hierarchical representations strategy HRL-GCN, with one pooling layer, as it realises the best accuracy scores.

The Text features of comments and posts are extracted using different language model based on BERT, and are aggregated by user to be used as the initial features of our user nodes. We use different models to extract those features:

- **PT** model. It only uses the pre-trained features to get the message representation. The last layer (768 dimensions) is outputted as our text embeddings.
- **FT_itself** model. We fine-tune [30] a BERT model using comments and posts of our train set $\mathcal{G}_{s,train}$, with an extra-layer of 64 neurons on top, in addition to the classifier layer. We label each comment with the controversy label of its respective post. Note that each subreddit is fine-tuned separately as we suppose that different communities express themselves differently, and texts can be interpreted differently.
- **FT_sentiment** model. We fine-tune a BERT model using sentiment analysis with another Reddit dataset of comments (hosted on kaggle.com), labeled as negative, positive or neutral. Indeed, we suppose here that sentiments can outline users' behavior on controversial posts.
- **PT_lstm** model. To counter the fact that Reddit messages have no limitation and are long texts, we create chunks of fixed-size (200 words with a fifty-word overlap) for each message, and extract their embedding from the pre-trained BERT model. Based on the idea in [31], we then train a bi-LSTM model with each chunk embedding, following the same principle as in FT_ITSELF to finally get an overall message embedding.

Table 4 Performance of our best GNN approach enriched with different user text embeddings as initial node features. Values in bold represent the best accuracies among all methods (including baseline), while underlined values represent the best accuracies among our proposed methods.

	AM	AW	FN	LS	PF	RS
HRL-GCN (pool=1)	69.6	74.6	72.2	67.9	68.2	66.7
+ FT_SENTIMENT	69.1	72.9	70.5	68.6	66.7	64
+ FT_ITSELF	67.3	73.9	71.8	68.3	70.6	63.8
+ PT	70.8	73.7	71	65.4	0.6	64.7
+ PT_LSTM	69.6	**74.8**	72.3	68.9	**70.7**	65.1

In all cases, we use the 'base-bert-uncased' version (with its corresponding tokenizer), with 12 transformer layers and 110 millions parameters. For time and memory performance, we only use 256 tokens max per text (instead of 512, as Table 2 shows that in average, less than 3% of the messages are represented by more than 256 tokens). For fine-tuning models, we used the same hyperparameters used in [30]. Table 4 shows the new accuracy scores obtained by incorporating text features in our HRL-GCN strategy.

For five of the six datasets, adding textual features improve, even slightly, controversy detection results. Our HRL-GCN strategy, merged with features from the pre-trained BERT combined with a bi-LSTM (PT_LSTM), gets better results when using AW, FN, LS, PF datasets, with 74.8, 72.3, 68.9 and 70.7 accuracy result respectively. However, the combination does not increase performance on the AM dataset, where adding only the pre-trained (PT) BERT features improve accuracy to 70.8. Adding sentiment features from FT_SENTIMENT allows us also to increase accuracy from 67.9 to 68.6 on LS. However, it does not perform as well as the PT_LSTM. The complexity of the data and the fact that datasets might be too small compared to the number of features (which goes up to 768 when using PT model) could also explain why our GNN-based models over-fit on some datasets, and therefore do not improve accuracy results. The dimension reduction applied in the last layer of PT_LSTM helps to reduce this problem.

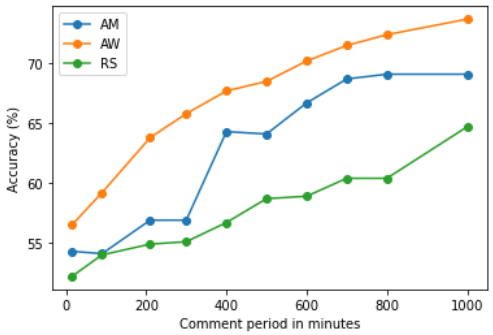

Fig. 7 Impact of comments availability on controversy detection performance.

Figure 7 shows the importance of comments availability. It reports accuracy results evolution over time (minutes) of three datasets when using our best HRL-GCN strategy combined with text features from PT. It clearly shows that the more available comments we have, the easier it is to detect controversy. Moreover, regarding Table 2, it could also explain why our method gets lower results on subreddit LT and RS compared to other datasets.

5.2 Twitter experiments

We also conducted experiments of our approach on a second social twitter, namely Twitter.

5.2.1 Datasets preparation

Data we collected from Twitter and presented in section 3.2 are used for the need of our Twitter experiments. Our main concern on Twitter experiments are the large size of the retweet graphs on one hand and the few number of graphs we can obtain (only 15 available topics). To address these two issues, we relied on graph sampling and data augmentation techniques.

Graph sampling methods. The different 15 graphs gathered in the set $\mathcal{G}_s$ are unfortunately too large to be handled by our models, even with a small batch size. At each step (epoch) of the learning process, the learning model takes the whole graph as input. On Reddit post-comment tree, the average number of user is 65. With an adjacency matrix of size n^2 (with n number of users), the memory can support it. However, we work on a different scale on Twitter, with an average number of user around 52 000 on our graphs. Therefore, the RAM of our server can not support it, even using one only batch, so we quickly got out of memory. Therefore, graph sampling techniques are needed to reduce the size of our different large graphs. Graph sampling methods aim to derive a small and similar graph from an original huge graph [32]. The sampling operation consists in selecting a subset of vertices and/or edges of the original graph. In our conducted experiments, we relied on three sampling methods - ISRW, FF and TIES - to reduce the size of large graphs generated from twitter [5].

1. Induced Subgraph Random Walk Sampling (ISRW). It is an oriented sampling node and it is based on the classic Random-walk sampling algorithm. It first randomly picks a starting node. Then, it runs a random walk, by selecting, randomly, a neighbor node on the graph. The algorithm continues until we reach the required node sample size. To avoid obtaining nodes with smaller degrees than in the original nodes, ISRW adds a graph induction step to select additional edges between sampled nodes with the aim to restore connectivity and get a better structure similarity with the original graph
2. Forest Fire (FF). It is also a node oriented sampling method. It randomly picks a seed node and begins what is called "burning" outgoing edges, and

[5] https://github.com/Ashish7129/Graph_Sampling

their corresponding nodes. If a link gets burned, the node at the other endpoint gets a chance to burn its own links. This process is recursively repeated for each burnt neighbor until no new node is selected. This process is run again with a new random node as starting node, until the desired sample size is reached.

3. Total Induction Edge Sampling (TIES). Contrarily to the two methods above, TIES is an edge oriented sampling method. It runs iteratively, by picking an edge at random from the original graph and adding both nodes to the sampled node set at each iteration. It stops adding nodes once the target fraction φ of nodes is collected. After this, the algorithm proceeds to the graph induction step where it walks through all the edges in the graph and forms the induced graph by adding all the edges which already have both end-points in the sampled node set.

Graph samplings are different in their strategies, the resulted graphs are therefore not similar. Figure 8 shows that the ISRW and TIES sampling methods seem to be better to represent a similar general view of the original controversial graph, as we can see that the two communities are well represented and separated. However, as TIES works with edge sampling, it requires access to the whole edge set. As in many large graph examples, the number of edges could be very important compared to the number of nodes, TIES can become less computing efficient when applied to very large networks. Therefore even if we test both options, we make the preliminary assumption that if we have to choose between those two methods, ISRW would be more suitable. On the other hand, FF method seems to focus on representing other aspects of the graph. Moreover, Figure 9 shows the same conclusion on non-controversial graph, as it seems to be less separated.

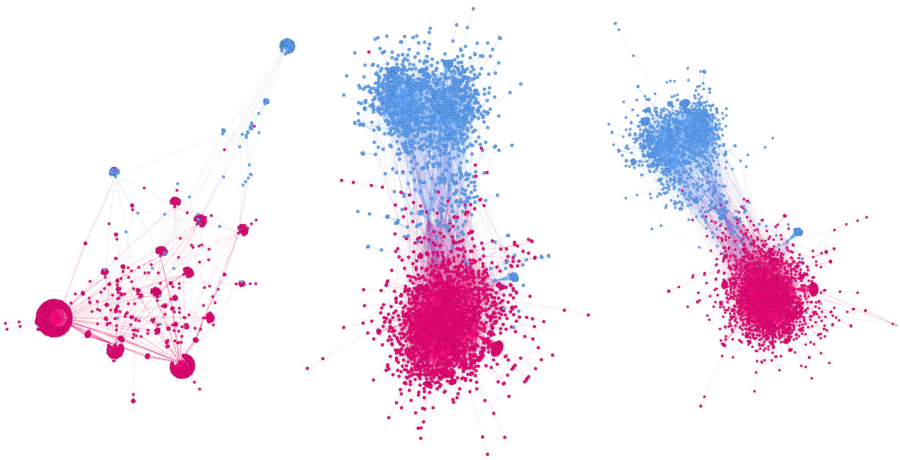

Fig. 8 Different sampled retweet graph of the **controversial topic 'netanyahu'**. Left: FF sampled graph. Middle: TIES sampled graph. Right: ISRW sampled graph.

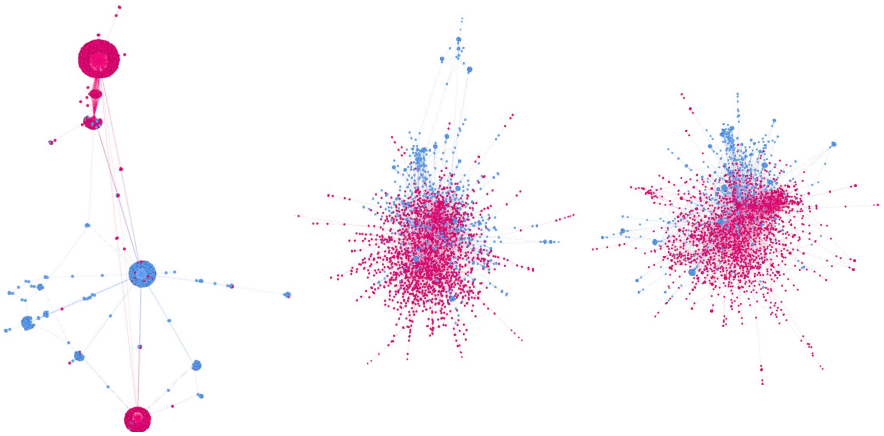

Fig. 9 Different sampled retweet graph of the **non-controversial topic 'sxsw'**. Left: FF sampled graph. Middle: TIES sampled graph. Right: ISRW sampled graph.

Graph Data augmentation. Contrary to Reddit, where it is easy to collect many topics around a subreddit, it is complicated to locate topics on Twitter, and therefore retrieve them. As we got too few training samples, we used the leave-one-out method to split the data into 15 folds. Therefore, only one graph will be used in the test set $\mathcal{G}_{i,val}$ and the 14 others will be used as training samples $\mathcal{G}_{i,train}$ at each time i. The accuracy of our model is computed as the average accuracy computed from the different folds. The test graph will be reduced as well by using the different sampling methods ISRW, FF and TIES. This implies that we train 15 different models for each experiment. To counter the too few number of graphs available in the full set $\mathcal{G}$, we relied on data augmentation techniques and mainly on the same sampling methods ISRW, FF and TIES to augment our training set. We created 10 sub-graphs for each training graph in $\mathcal{G}_{i,train}$. We then equally balanced our training dataset by selecting the same number of sampled graphs from both classes.

Concerning the textual features, we used two BERT-based models to embed tweets into a vector in a latent space:

1. **PT** model. It is the same pre-trained BERT model we used in section 5.1.4 on our REDDIT datasets.
2. **BERTweet** model. It is a language model specially pre-trained for English Tweets [33]. BERTweet is trained based on the RoBERTa pre-training procedure, another method built on BERT but with specific and own hyperparameters

We experimented the models with and without textual features, as we did for Reddit, to see if textual information has any impact on the performance of our approach when twitter is used with large generated graphs. Note that we only

Table 5 Performance accuracy of our GNN-based controversy detection method on Twitter. The text model column represents the model used for computing user textual features. Values in bold represent the best accuracies among our proposed methods.

Sampling method	Text model		
	no features	PT	BERTweet
ISRW	0.5	0.47	0.47
TIES	0.53	0.53	0.5
FF	**0.67**	**0.67**	0.5

tested the model that got the best results (without textual features) on the Reddit dataset, which corresponds to our HRL-GCN, with 1 pooling layer.

5.2.2 Third experiment: Controversy detection on Retweet graphs

We conducted experiments with using the 3 sampling methods presented in section 5.2.1 to reduce the graph size and augment the number of sampled graphs as well. For the random-walk (ISRW) and Forest-Fire (FF) methods, we set the max number of node sample at 2930, which corresponds to 1% of the average number of nodes of topic graphs. Concerning the edge sampling method (TIES), we set φ in a way that the max number of sampled nodes corresponds to 1% of the original graph. We tested out approach with and without node textual features on our HRL-GCN approach. As we used leave-one-out as our model validation technique, we estimated the average accuracy of each fold run.

According to results on Table 5, ISRW and TIES do not achieve good performances, which means that compared to the approach used in [4], our approach does not perform well when it tries to capture structural information of two distinct communities (as shown in figures 8 and 9), and rather focuses on other structural features. Forest Firing sampling method (FF) is based on the spread of fire, which could correspond in controversial topic on stance or idea on a user graph. Moreover, the fact that we only train our model on 14 real user graphs could explain why the model does not perform well.

Concerning the textual representation of users, pre-trained models seems to fail capturing positions or stances of users on controversial topics among communities, especially with BERTweet. The noise in our data could also explain why it is hard to capture representative textual embeddings on the current topic. However, table 5 shows that using Forest Firing (FF) sampling to reduce and augment graphs gets better performance. In other words, it better captures the structural information of our Retweet graph. We get 0.67 accuracy with and without text features from PT model. Results are close to what we got on the previous experiments on the Reddit datasets. Therefore, it means that our approach can be generalized to other social networks, despite the fact that we got less graphs.

6 Conclusion

Automatic controversy detection on social media remains a challenging task. In this paper, we considered controversy detection as a classification problem. We adopted very recent GNN techniques along with machine learning-based natural language processing to combine both structural information (interactions between people) and discussion contents. Discussion between people on any social media is represented as a graph of users. Graph embedding is performed by incorporating textual content features computed from BERT and LSTM models. The proposed approach supports two controversy detection strategies. The first strategy exploits hierarchical structure that may exist in the user graph. The second strategy allows each user to select its neighbors in the embedding nodes process. Experimental evaluation was performed on different datasets collected from Reddit and Twitter platforms. This evaluation confirms that structural information is useful to detect controversy and clearly shows that text content features can improve its accuracy results in some cases on Reddit, while our experiments on Twitter suggest that classic pre-trained models do not fit well. Fine-tuning a BERT-based model on a specific task could be an interesting perspective, as sentiment analysis or stance detection. In terms of future work, we would like to examine the appropriateness of other GNN techniques for controversy detection. For instance, it could be interesting to study the impact, in terms of performance, of using GNN architectures that take into account nodes and edges properties [34] on heterogeneous graphs, where user nodes could be linked in different ways. Concerning Twitter, mixing our approach on graph neural networks, with the idea of quantifying controversy [4] on a complete graph could also be an interesting avenue to explore. Indeed, we could focus on only structural information learning with GNNs, and study which measures could help to understand the graph representation and to quantify controversy. In order to have a better overview of different structures, a more complete dataset of controversial and non-controversial topics is needed and could improve the performance of our apporach.

Acknowledgement

This work was supported by grants from Janssen Horizon endowment fund. It was granted access to the HPC resources of IDRIS under the allocation AD011012604 made by GENCI.
Conflict of Interest. The authors declare that they have no conflict of interest.

References

[1] Hessel, J., Lee, L.: Something's brewing! early prediction of controversy-causing posts from discussion features. In: Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, pp. 1648–1659 (2019)

[2] Beelen, K., Kanoulas, E., van de Velde, B.: Detecting controversies in online news media. In: 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1069–1072 (2017)

[3] Garimella, K., Morales, G.D.F., Gionis, A., Mathioudakis, M.: Reducing controversy by connecting opposing views. In: Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI, pp. 5249–5253 (2018)

[4] Garimella, K., Morales, G.D.F., Gionis, A., Mathioudakis, M.: Quantifying controversy on social media. ACM Trans. Soc. Comput. **1**(1), 3–1327 (2018)

[5] Jang, M., Dori-Hacohen, S., Allan, J.: Modeling controversy within populations. In: Proceedings of the SIGIR International Conference on Theory of Information Retrieval, ICTIR, pp. 141–149 (2017)

[6] Sznajder, B., Gera, A., Bilu, Y., Sheinwald, D., Rabinovich, E., Aharonov, R., Konopnicki, D., Slonim, N.: Controversy in context. CoRR (2019)

[7] Jang, M., Foley, J., Dori-Hacohen, S., Allan, J.: Probabilistic approaches to controversy detection. In: 25th ACM International Conference on Information and Knowledge Management, CIKM, pp. 2069–2072 (2016)

[8] Morales, A.J., Borondo, J., Losada, J.C., Benito, R.M.: Measuring political polarization: Twitter shows the two sides of venezuela. CoRR (2015)

[9] Jang, M., Allan, J.: Improving automated controversy detection on the web. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR, pp. 865–868 (2016)

[10] Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: 5th International Conference on Learning Representations, ICLR (2017)

[11] Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: 6th International Conference on Learning Representations, ICLR (2018)

[12] Hamilton, W.L., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems, pp. 1024–1034 (2017)

[13] Zhong, L., Cao, J., Sheng, Q., Guo, J., Wang, Z.: Integrating semantic and structural information with graph convolutional network for controversy detection. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL, pp. 515–526 (2020)

[14] Benslimane, S., Azé, J., Bringay, S., Servajean, M., Mollevi, C.: Controversy Detection: a Text and Graph Neural Network Based Approach. In: 22nd International Conference on Web Information Systems Engineering. Lecture Notes in Computer Science, vol. 13080, pp. 339–354. Melbourne, Australia (2021). https://hal.archives-ouvertes.fr/hal-03464243

[15] Ying, Z., You, J., Morris, C., Ren, X., Hamilton, W.L., Leskovec, J.: Hierarchical graph representation learning with differentiable pooling. In: Annual Conference on Neural Information Processing Systems, NeurIPS, pp. 4805–4815 (2018)

[16] Zhang, S., Xie, L.: Improving attention mechanism in graph neural networks via cardinality preservation. In: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020, pp. 1395–1402 (2020)

[17] Emamgholizadeh, H., Nourizade, M., Tajbakhsh, M.S., Hashminezhad, M., Esfahani, F.N.: A framework for quantifying controversy of social network debates using attributed networks: biased random walk (BRW). Soc. Netw. Anal. Min. **10**(1), 90 (2020)

[18] Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT Conference: Human Language Technologies, Volume 1, pp. 4171–4186 (2019)

[19] Mendoza, M., Parra, D., Soto, Á.: GENE: graph generation conditioned on named entities for polarity and controversy detection in social media. Inf. Process. Manag. **57**(6), 102366 (2020)

[20] Zarate, J.M.O.D., Feuerstein, E.: Vocabulary-based method for quantifying controversy in social media. In: Ontologies and Concepts in Mind and Machine - 25th International Conference on Conceptual Structures, ICCS. Lecture Notes in Computer Science, vol. 12277, pp. 161–176 (2020)

[21] Guerra, P.H.C., Jr., W.M., Cardie, C., Kleinberg, R.: A measure of polarization on social media networks based on community boundaries. In: Seventh International Conference on Weblogs and Social Media (2013)

[22] Rashed, A., Kutlu, M., Darwish, K., Elsayed, T., Bayrak, C.: Embeddings-based clustering for target specific stances: The case of a polarized turkey. CoRR **abs/2005.09649** (2020)

[23] Yao, L., Mao, C., Luo, Y.: Graph convolutional networks for text classification. In: The Thirty-Third AAAI Conference on Artificial Intelligence. AAAI Press, pp. 7370–7377 (2019)

[24] Ye, Z., Jiang, G., Liu, Y., Li, Z., Yuan, J.: Document and word representations generated by graph convolutional network and BERT for short text classification. In: 24th European Conference on Artificial Intelligence, 2020. Frontiers in Artificial Intelligence and Applications, vol. 325, pp. 2275–2281 (2020)

[25] Tayal, K.: Short text classification using graph convolutional network. In: NeurIPS Graph Representation Learning Workshop (2019)

[26] Zhang, Y., Yu, X., Cui, Z., Wu, S., Wen, Z., Wang, L.: Every document owns its structure: Inductive text classification via graph neural networks. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, pp. 334–339

[27] Yiping, Y., Cui, X.: Bert-enhanced text graph neural network for classification. Entropy **23**(11) (2019)

[28] Lu, Z., Du, P., Nie, J.: VGCN-BERT: augmenting BERT with graph embedding for text classification. In: 42nd European Conference on IR Research. Lecture Notes in Computer Science, vol. 12035, pp. 369–382 (2020)

[29] Jacomy, M., Venturini, T., Heymann, S., Bastian, M.: Forceatlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software. PloS one **9**, 98679 (2014)

[30] Sun, C., Qiu, X., Xu, Y., Huang, X.: How to fine-tune BERT for text classification? In: Chinese Computational Linguistics - 18th China National Conference, CCL. Lecture Notes in Computer Science, vol. 11856, pp. 194–206 (2019)

[31] Pappagari, R., Żelasko, P., Villalba, J., Carmiel, Y., Dehak, N.: Hierarchical transformers for long document classification, pp. 838–844 (2019)

[32] Hu, P., Lau, W.C.: A Survey and Taxonomy of Graph Sampling. arXiv:1308.5865 [cs, math, stat] (2013)

[33] Nguyen, D.Q., Vu, T., Nguyen, A.: Bertweet: A pre-trained language model for english tweets, pp. 9–14 (2020)

[34] Simonovsky, M., Komodakis, N.: Dynamic edge-conditioned filters in convolutional neural networks on graphs. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2017) (2017)