International Journal of Computational Intelligence Systems (2022) 15:86
https://doi.org/10.1007/s44196-022-00147-9

RESEARCH ARTICLE

Argumentation Reasoning with Graph Isomorphism Networks for Reddit Conversation Analysis

Teresa Alsinet[1] · Josep Argelich[1] · Ramón Béjar[1] · Daniel Gibert[2] · Jordi Planes[1]

Received: 2 March 2022 / Accepted: 3 October 2022

Abstract

The automated analysis of different trends in online debating forums is an interesting tool for sampling the agreement between citizens in different topics. In previous work, we have defined computational models to measure different values in these online debating forums. One component in these models has been the identification of the set of *accepted posts* by an argumentation problem that characterizes this accepted set through a particular argumentation acceptance semantics. A second component is the classification of posts into two groups: the ones that agree with the root post of the debate, and the ones that disagree with it. Once we compute the set of accepted posts, we compute the different measures we are interested to get from the debate, as functions defined over the bipartition of the posts and the set of accepted posts. In this work, we propose to explore the use of graph neural networks (GNNs), based on graph isomorphism networks, to solve the problem of computing these measures, using as input the debate tree, instead of using our previous argumentation reasoning system. We focus on the particular online debate forum Reddit, and on the computation of a measure of the polarization in the debate. We explore the use of two different approaches: one where a single GNN model computes directly the polarization of the debate, and another one where the polarization is computed using two different GNNs: the first one to compute the accepted posts of the debate, and the second one to compute the bipartition of the posts of the debate. Our results over a set of Reddit debates show that GNNs can be used to compute the polarization measure with an acceptable error, even if the number of layers of the network is bounded by a constant. We observed that the model based on a single GNN shows the lowest error, yet the one based on two GNNs has more flexibility to compute additional measures from the debates. We also compared the execution time of our GNN-based models with a previous approach based on a distributed algorithm for the computation of the accepted posts, and observed a better performance.

Keywords Reddit · Social networks analysis · Argumentation · Graph neural networks

✉ Ramón Béjar
ramon.bejar@udl.cat

Teresa Alsinet
teresa.alsinet@udl.cat

Josep Argelich
josep.argelich@udl.cat

Daniel Gibert
daniel.gibert@ucd.ie

Jordi Planes
jordi.planes@udl.cat

[1] INSPIRES Research Center, University of Lleida, C/ Jaume II, 69, 25001 Lleida, Spain

[2] CeADAR, University College Dublin, Belfield Office Park, Dublin, Ireland

Abbreviations

DebT	Debate tree
PDebT	Pruned debate tree
WBDebG	Weighted bipartite debate graph
VAF	Value-based abstract argumentation framework
GNN	Graph neural network
GIN	Graph isomorphism network
1GNN model	Single neural network model
2GNN model	Two neural networks model

1 Introduction

In Internet, a wide range of debate forums have been created and rapidly grown in the last years. Debating on Internet commonly occurs as an exchange of messages among several

Published online: 17 October 2022

participants. Debates are often represented as threads, which are initiated by a user posting a starting message (which we refer to as root message), and users replying to any of the posted messages. These sequences are commonly represented as a tree, with the root message on the top. Some previous work has been focused on statistically modeling such discussions as generative models [1] by considering features like node popularity [2] and node novelty, or on surveying statistical graph models for social networks [3]. Instead of focusing on node features or user features, we have focused on the study of structural features of the debates, like discussion polarization, that are based on argumentation reasoning.

Recently, there has been a growing interest in the use of Graph Neural Network (GNN) approaches to model and solve reasoning problems defined via graph inputs [4–6]. The most common approach used by a GNN is to map the feature vector of each node to an embedding representation that also uses (by aggregation) the feature vector of its neighbor nodes. By iterating this scheme k times, the final representation of each node tends to capture structural information within the node's k-hop neighborhood. This scheme can be used to learn any kind of function over graphs that outputs a labeling of its nodes, or that outputs a single value (for graph classification tasks).

In previous work, we have considered the use of argumentation-based models to analyze different characteristics of social network debates. In the argumentation-based approach, we first identify a valued argumentation problem with the debate to be solved, where debate posts are associated with arguments, under a particular acceptance semantics: a set of rules that define what arguments are accepted and what are rejected. The usual acceptance semantics tend to be NP-hard, like the *ideal semantics* [7] we have used in our previous works about measuring discussion polarization with argumentation-based models [8, 9].

In this work, we initiate a line of investigation to study whether a GNN approach can be a good candidate to solve argumentation-based problems with less computational effort. Previous work has already used GNNs for computing argumentation semantics for abstract frameworks. Kuhlmann et al. [10] study the feasibility of using Graph Convolutional Networks (GCN) to solve the set of accepted arguments under preferred semantics. Craandijk and Bex [11] use Argumentation Graph Neural Networks (AGNN) to solve the set of accepted arguments, but with several argumentation semantics. AGNNs represent each node with an embedding that aggregates the state of a node with the state of the neighbors of the node and its input features using a Recurrent Neural Network (RNN) [12]. Our GNN model is based on the Graph Isomorphism Network (GIN) model [6], where node states are also updated with an operator that takes into account the state of the neighbors, and this update is performed through a fixed number of layers on the network.

Our goal in this work is the definition and evaluation of GNN models based on GIN to compute the final measure of interest. This final measure is defined from the set of accepted arguments of the argumentation problem derived from the Reddit debate. Our hypothesis is that even if the worst-case complexity of computing accepted arguments is in general NP-hard, it may be possible to compute, or approximate, the final measure with much less computational effort. In particular, in this work, we focus on the computation of a measure of discussion polarization that is defined as a function of the set of accepted arguments of a discussion, and whether these arguments agree or disagree with the root topic of the discussion. Our discussions come from the social network Reddit.[1]

A Reddit debate is first represented as a debate tree, where edges represent agreement or disagreement relationships between Reddit posts. Then, this debate tree is processed to get a bipartite debate graph where posts are divided in two groups: the ones that agree with the root post of the debate, and the ones that disagree with it. The edges of the bipartite graph represent disagreement between posts of the two groups. There has been recently interested in understanding the behavior of users in social networks like Reddit. For example, in [13] they analyze different dynamic characteristics of the discussions on Reddit, and in [14], they introduce a measure of controversy in discussion threads in Slashdot based on the use of the h-index over the nodes of the discussion tree. More recently, in [15], the authors represent Reddit communities (subreddits) in a high-dimensional behavioral space. Then, each community is positioned along this set of dimensions that represent different social features, like for example political polarization. Finally, for each community, they study how far is positioned with respect to the average value for each dimension. In contrast, our polarization measure is defined over a particular discussion, and it is based on the differences between the set of accepted comments that agree or disagree with the root topic of the discussion. So, it is a measure of polarization with respect to the particular topic of the discussion.

Our results show that we can perform the computation of the final measure with two complementary GIN-based approaches. The first one is focused on an efficient computation based only on the original debate tree, without explicitly computing the set of accepted arguments of the associated argumentation problem. The second one is a more flexible approach, that allows to compute other alternative measures based on the solution (accepted arguments) of the

[1] https://www.reddit.com.

argumentation problem, that is explicitly computed by this second GIN-based model.

The structure of the paper is as follows. In Sect. 2, first, we present the relevant definitions for our argumentation-based Reddit analysis system and second, we briefly survey previous results about GNNs. In Sect. 3, we present the two GNN architectures we have used to model our reasoning system. Finally, in Sect. 4, we present the experimental results we have obtained with a dataset of Reddit debates.

2 Background

In this section, we give the definitions of the different components of the Reddit analysis system introduced in [9] and the most commonly used graph neural networks.

2.1 Debate Analysis System

It is based on two main components: a Reddit debate retrieval system and an argumentation-based reasoning system. The retrieval system takes a Reddit post, which we reference as the root comment, and obtains the complete set of posts or comments generated in the debate on that root comment. From now on, we will refer to Reddit posts as comments.

Definition 1 A *comment* c is a tuple $c = (m, u, sc)$, where m is the text of the comment, u is the user's identifier of the comment, and $sc \in \mathbb{Z}$ is the score of the comment.

Let $c_1 = (m_1, u_1, sc_1)$ and $c_2 = (m_2, u_2, sc_2)$ be two comments. We say that c_1 *answers* c_2 if c_1 is a reply to comment c_2.

Let $r = (m_r, u_r, sc_r)$ be a comment such that m_r contains a link to some news. A *Reddit debate* on the (root) comment r is a non-empty set Γ of Reddit comments such that $r \in \Gamma$ and every comment $c \in \Gamma$, $c \neq r$, c answers some comment in Γ.[2]

Next, we obtain the tree representation of a Reddit debate, where we incorporate edge labels that express the sentiment of the comments.

Definition 2 Let Γ be a *Reddit debate* on a (root) comment r. The *Debate Tree* (DebT) for Γ is a tuple $\mathcal{T} = \langle C, r, E, L \rangle$ such that:

- for every comment in Γ, there is a node in C,
- node $r \in C$ is the root node of $\mathcal{T}$,
- if c_1 answers c_2 then there is a directed edge (c_1, c_2) in E, and
- L is a labeling function $L : E \to [-2, 2]$, where the value assigned to an edge denotes the sentiment of the answer, from highly negative (-2) to highly positive (2).

As argued in [9], because we are interested in considering only comments with enough inclination to either agree or disagree with the root comment, we define a pruned version of a DebT, where we discard any comment $c \in C$ that does not agree or disagree enough with the comment answered, and also the subtree rooted at c. The rationale behind discarding these neutral comments (and their subtrees) is that it is undefined whether a neutral comment agrees or disagrees with the comment to which it responds and, consequently, with the root comment of the debate. For this reason, it is meaningless for a comment to agree or disagree with a neutral comment, so, any comment in the subtree of a neutral comment does not contribute anything relevant with respect to defending or rejecting the root comment of the debate. Next, we formalize the Pruned Debate Tree structure with respect to a pruning threshold.

Definition 3 Let α be a pruning threshold in the real interval [0, 2] and let $\mathcal{T} = \langle C, r, E, L \rangle$ be a DebT. The *Pruned Debate Tree* (PDebT) for $\mathcal{T}$ with respect to α is a tuple $\mathcal{T}_\alpha = \langle C_\alpha, r, E_\alpha, L \rangle$, where both sets of pruned comments $C_\alpha \subseteq C$ and pruned edges $E_\alpha \subseteq E$ are defined as follows:

- the root node (comment) $r \in C_\alpha$,
- r is the root node of $\mathcal{T}_\alpha$ and
- if $(c_1, c_2) \in E$ with $c_2 \in C_\alpha$, then $c_1 \in C_\alpha$ and $(c_1, c_2) \in E_\alpha$, whenever $|L(c_1, c_2)| \geq \alpha$.

Note that for $\alpha = 0$ the pruning threshold has no effect, in the sense that the PDebT obtained corresponds to the original DebT and that for $\alpha = 2$ the PDebT obtained only contains strictly polarized both positive and negative answers. In any case, the PDebT $\mathcal{T}_\alpha$ is a subtree of $\mathcal{T}$ with r being the root node.

Finally, we divide the set of comments into two sets: comments supporting the root comment and comments that disagree with it. Then, the attacks between the comments of both sets are defined as a subset of edges in E_α such that they are negative answers from a comment in one of the sets to a comment in the other set, obtaining a bipartite graph that represents both sides of the debate, and the disagreement between them. Moreover, we also label each node of the graph obtained with a weight that denotes the comments' social acceptance during the debate. Next, we formalize the Weighted Bipartite Debate Graph structure.

[2] Given the structure of a Reddit debate, except for the root comment, each comment answers exactly one previous comment, usually by another user or author.

Definition 4 Let $\mathcal{T}_\alpha = \langle C_\alpha, r, E_\alpha, L\rangle$ be a PDebT for a Reddit debate Γ. A *Weighted Bipartite Debate Graph* (WBDebG) for $\mathcal{T}_\alpha$ is a tuple $G = \langle C_+ \cup C_-, E_-, W\rangle$, where

- C_+ and C_- is a bipartition of C_α. Thus, $C_+ \cup C_- = C_\alpha$ and $C_+ \cap C_- = \emptyset$, where C_+ denotes the set of comments that agree with the root comment r, and C_- denotes the set of comments that disagree with it.
- $E_- = \{(c_1, c_2) \in E_\alpha \mid L(c_1, c_2) < 0\}$ and corresponds with the set of disagreement edges between the comments in C_+ and C_-. Thus, if $(c_1, c_2) \in E_-$, then either $c_1 \in C_+$ and $c_2 \in C_-$ or, $c_1 \in C_-$ and $c_2 \in C_+$.
- W is a weighting scheme $W : C_\alpha \to \mathbb{N}$ of the weight of nodes (comments). The weighting scheme W evaluates the social acceptance of comments by mapping the score sc of a comment $(m, u, sc) \in C_\alpha$ to a value in $\mathbb{N}$.

The WBDebG for a PDebT can be computed with the algorithm that we presented in [8]. The algorithm starts by initializing the sets of nodes C_+ (agreement set) and C_- (disagreement set), and the set of edges E_-, with the empty set. Then, the root comment is put in the set C_+ and a recursive procedure is used to classify its child nodes. This recursive procedure classifies the children of a node n as follows:

- It receives as arguments the set to which the node pertains (which in the first call corresponds to the set C_+), and the opposite set (which in the first call corresponds to the set C_-).
- For every child node c, if the corresponding comment agrees with its parent, the child node is put in the same set as the parent node. Then, the children of the node c are recursively classified by calling this procedure again.
- If, otherwise, the comment of the child node c disagrees with its parent, the child node c is put in the opposite set of the parent node. In this case, a directed edge in E_- is created from the child node c to the parent node n. Finally, the children of the node c are also recursively classified.

2.2 Reasoning System

At this point, we are ready to introduce the argumentation-based reasoning system used to obtain the set of comments, from the two opposite groups of a WBDebG, that are accepted in the sense that this set should represent a kind of consensus among all the comments of the debate. To this end, we use value-based abstract argumentation [16] to model the weighted argumentation problem associated with a WBDebG and ideal semantics [17] to compute its solution (the set of comments that can be accepted).

The *value-based abstract argumentation framework* (VAF) we define for a WBDebG $G = \langle C_+ \cup C_-, E_-, W\rangle$, interprets each comment in $C_+ \cup C_-$ as an argument and defines a *defeat* relation (or effective attack relation) between arguments as follows:

$$defeats = \{(c_1, c_2) \in E_- \mid W(c_2) \not\geq W(c_1)\};$$

i.e., argument c_1 *defeats* argument c_2 if and only if c_1 attacks or disagrees with c_2 and the social acceptance value of c_2 is not preferred over the social acceptance value of c_1, based on the weighting scheme W.

Then, a set of comments $S \subseteq C_+ \cup C_-$ is called *conflict-free* if for all $c_1, c_2 \in S, (c_1, c_2) \notin defeats$, and a conflict-free set of comments $S \subseteq C_+ \cup C_-$ is defined as *maximally admissible* if for all $c_1 \notin S, S \cup \{c_1\}$ is not conflict-free and, for all $c_2 \in S$, if $(c_1, c_2) \in defeats$, then there exists $c_3 \in S$ such that $(c_3, c_1) \in defeats$. Finally, the *solution* or *set of accepted comments* for a debate is the largest admissible conflict-free set of comments $S \subseteq C_+ \cup C_-$ in the intersection of all maximally admissible conflict-free sets, denoted as the ideal extension.

We select this semantics to define the solution for a debate, because it represents a maximally admissible set of conflict-free comments, such that they defend against attacks outside the set with comments inside the set, and they are included in any admissible set of comments. This set, therefore, represents a kind of *maximum consensus* between all the possible admissible sets of comments. For our particular case of an acyclic VAF, the picture is even simpler, as there is a unique maximally admissible set, and thus the solution for ideal semantics coincides with this set. Moreover, for the case of a VAF that is acyclic or bipartite (as in the case of a WBDebG), we can compute its solution in linear time, with respect to the number of comments, for debates of big size with the distributed algorithm we developed in [18]. However, in the worst case, the status of each comment in the solution may depend on the status of the rest of the comments, so that is why we explore in this work a possible GNN-based architecture where nodes (comments) only consider information from nodes at distance bounded by a constant.

Given that the solution for the debate provides us with a consensus point of view, an interesting characteristic to analyze is its degree of polarization.

Definition 5 Let $G = \langle C_+ \cup C_-, E_-, W\rangle$ be a WBDebG and let $S \subseteq C_+ \cup C_-$ be the solution for G. The *polarization degree* of solution S is a measure in the real interval $[-1, 1]$ defined as follows:

$$polarization\,(S) = \frac{\#(S \cap C_+) - \#(S \cap C_-)}{\#S}. \tag{1}$$

We use the polarization degree value as a measure of the bias of the solution S towards comments in C_+ and comments in C_-. The value that indicates total bipartisanship (0) is obtained when the number of comments of S in C_+ equals the number in C_-. The highest positive value is obtained when all the comments of the solution are found in C_+, and analogously for the lowest negative value.

2.3 Polarization Computation

The distributed algorithm introduced in [18] efficiently computes the polarization of a discussion with the ideal extension (solution) of an acyclic VAF. This is possible because an acyclic VAF is cohesive and coherent.

In the distributed model implemented in Pregel [19], these properties were utilized to design an algorithm where a node is accepted in the solution if and only if all its defeating nodes are not accepted (or if it has no defeating nodes). This recursive acceptance condition is well-defined for the case of an acyclic VAF. In the algorithm, each node can be in three states: undefined, accepted or rejected. A node keeps track of the number of accepted defeaters and the number of rejected defeaters of the node in a given state of the algorithm. Initially, every node starts in the undefined state and with its counters set to zero.

2.4 Graph Neural Networks

In the last years, there has been an increasing interest in solving graphs problems with machine learning (ML) techniques [20, 21]. Because of the immense expressive power of graphs, they can be used to model the interaction between complex structures such as proteins [22], mRNA [23], particles in physics models [24], etc. One key factor is the ability of such ML-based methods to deal with graphs of different sizes and shapes.

There have been various attempts in the literature of using graph neural networks (GNNs), mainly by: (i) focusing on learning node embeddings by aggregating the nodes, and (ii) by mapping from the node neighborhood domain (adjacency matrix) to spectral domain. From the first type, we highlight the Generalizing Aggregation GraphSage [4] used for node classification. This method focuses on learning node embeddings, and then it aggregates the resulting embeddings to handle size-varying neighborhoods. These neural networks follow a neighborhood aggregation strategy, where they iteratively update the representation of a node by aggregating representations of its neighbors. After k iterations of aggregation, the representation of a node captures the structural information in its k-hop network neighborhood. From the second type, we feature the Spectral Graph Convolution Model [5] used for the classification of nodes using their adjacency matrix. In addition, it uses Chebyshev filters (passband filters) and Laplacian regularization in the loss function.

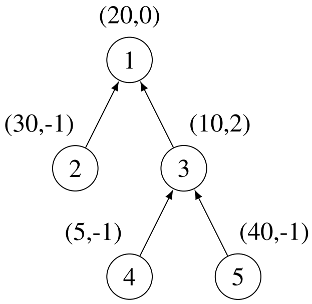

Fig. 1 Graph for Example 3.1. Nodes are indexed from 1 to 5, and every node is labeled with a pair (*score*, *sentiment*)

A recent improvement over the first method is the Graph Isomorphism Networks (GINs), presented in a study of GNN expressively [6], which is inspired by the Wesfeiler–Lehman (WL) test [25] of graph isomorphism. The authors proposed a WL equivalent aggregator, which achieves the maximum discriminative power among the GNNs in the literature.

3 Graph Isomorphism Network Models

We propose two different GNN approaches to approximately compute the polarization degree (see Definition 1) of a Reddit debate, both based on Graph Isomorphism Networks (GIN) [6]: in the first approach the value is computed by a single GIN; and in the second approach the value is computed with two different GINs, which output the set of accepted nodes (the solution) and the bipartition set.

Two key factors we have considered in the design of the networks are: (a) the node embedding, i.e., how the nodes are modeled to properly be fed to the network, and (b) the activation function, which is performed at the output of any layer in the network, is needed to introduce nonlinearity, and be able to solve complex problems.

In our GIN-based models, the following two lists are required to be given to the neural network: one list for the information associated with every node, and one list for the edges, with input and output adjacent nodes for every node. In particular, in computing the polarity of a Reddit conversation, the lists are: a list of pairs (*score*, *sentiment*), and a list of pairs of input and output node lists $(input\_nodes, output\_nodes)$. We have used the python libraries pytorch[3] and pytorch geometric[4] to implement all the GIN models in our work.

[3] https://pytorch.org/.

[4] https://pytorch-geometric.readthedocs.io/.

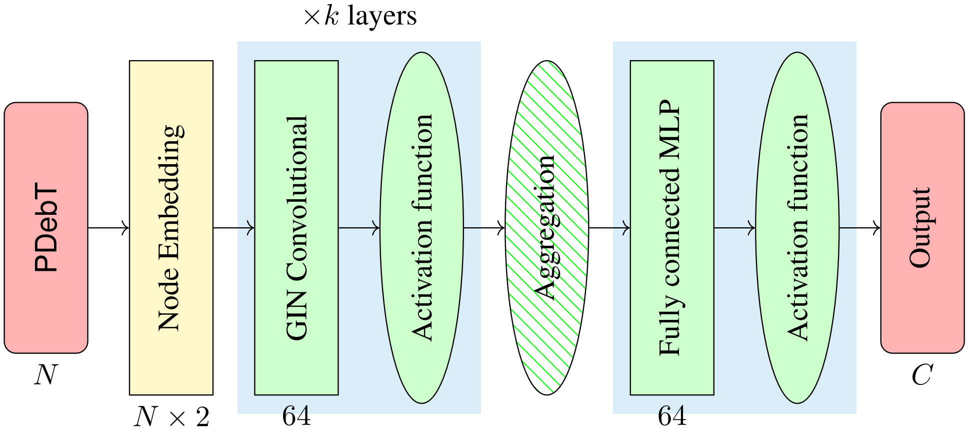

Fig. 2 1GNN model: GNN architecture for computing the polarization degree of a Pruned Debate Tree

Example 3.1 Having a graph with 5 nodes, as in Fig. 1, the introduced information to the GIN is the following: a list of node information as $\{(20,0),(30,-1),(10,2),(5,-1),(40,-1)\}$; and a list of edges as $\{((2,3),\emptyset), (\emptyset,(1)), ((4,5),(1)), (\emptyset,(3)), (\emptyset,(3))\}$. Notice that the node number corresponds to the position in the list of node information and to the position in the list of edges.

3.1 Single Neural Network Model

In our first approach, which we reference as 1GNN model, we use a single GIN that receives as input a PDebT $\mathcal{T}_\alpha = \langle C_\alpha, r, E_\alpha, L\rangle$ with $|C_\alpha| = N$ nodes, obtained from a Reddit debate as explained in [8], and outputs the polarization degree.

The overall architecture is presented in Fig. 2. The input and output are represented as red rectangles, the node embedding is represented as a yellow rectangle, the layers are represented as green rectangles, and the functions performed after the layers are represented as green ellipses. The details are described as follows:

Node embedding The input layer contains a two-dimensional vector for each non-root comment $c_i = (m_i, u_i, sc_i)$ that contains the score of the comment sc_i and the sentiment from the label $L(c_i, c_j)$, where c_j is the unique comment such that $(c_i, c_j) \in E_\alpha$.

GIN Convolutional (k layers). Every layer combines the node embedding of the previous layer, considering the node close neighbors. The aggregator in the layer l is the following:

$$\mathbf{x_i}^{(l)} = MLP\left((1+\epsilon)\cdot \mathbf{x_i}^{(l-1)} + \sum_{j\in\mathcal{N}(i)} \mathbf{x_j}^{(l-1)}\right), \qquad (2)$$

where $\mathbf{x_i}^{(l)}$ is the embedding of node i in the layer l, ϵ is a learnable parameter, and MLP is a multi-layer perceptron with nonlinearity, and $\mathcal{N}(i)$ is the set of neighbors of node i.

The first GIN layer has an input dimension of 2 and an output dimension of 64, which has been determined empirically to balance learning time and accuracy. The following layers have input and output dimensions of 64. Globally, this GIN block maps the two-dimensional vector of each node to a vector of 64 values that tries to capture the information from nodes k hops away from it. As an activation function, we insert a Rectified Linear Unit (ReLU) layer after each GIN layer, to help encode non-linear outputs in the network.

Observe that the node embedding in step k captures the graph structure within k hops from the original node [6], since it aggregates information from adjacent nodes [$\mathcal{N}(i)$ in (2)]. Having the graph in Example 3.1, the node embedding for node 3, after the computation in the first GIN layer, captures the overall structure of node 3 together with its set of neighbors (1, 4 and 5). The dimension of the output embedding after each aggregation step is larger (64 in our models) than the dimension of the input vectors, to be able to encode enough information from the set of node neighbors.

Normalization We give also the option to apply layer normalization between consecutive GIN layers, because previous work suggests that it may speed up the learning process [26].

Aggregation The aggregation layer creates the final graph representation using the mean operator, aggregating all the node embeddings into one graph embedding, as a vector with the same dimension (64).

Fully connected MLP This block maps the final aggregated embedding representation of the graph into the polarization of the debate. In this case, the *Output* size C is 1. The activation function in this layer is the hyperbolic tangent function, or tanh, because we would like the GNN to output a value between −1 and 1, indicating the polari-

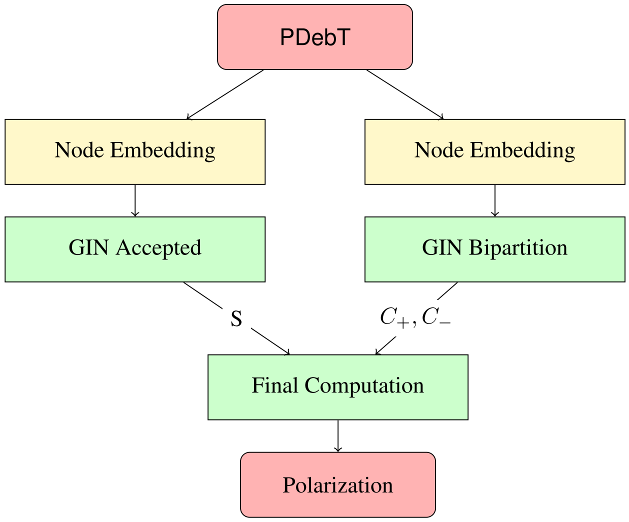

Fig. 3 2GNN model: Two GNN model architecture for computing the polarization degree of a Pruned Debate Tree

zation degree of the Reddit debate encoded as a Pruned Debate Tree. The tanh function is as follows:

$$\tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}.$$

After every ReLU layer and at the end of the fully connected MLP, a dropout of 0.1 is applied to prevent overfitting [27].

3.2 Two Neural Networks Model

In our second approach (cf. Fig. 3), which we reference as 2GNN model, we use two individual GINs that both receive the same input as the previous approach: the PDebT. The first GIN is used to compute the solution S of the debate, i.e., the set of accepted comments as defined in Sect. 2; and the second GIN is used to compute the bipartition C_+, C_- of the corresponding WBDebG. Then, once we have the solution and the bipartition of the debate, we compute the debate polarization (cf. Definition 1).

The architectures of these two GINs differ from the previous one in two points:

- The aggregation layer is removed, since we need the state of every node.
- In the final layer, the state of each node is mapped into a vector of dimension equal to the number of node classes, C, in our case is $C = 2$ (node accepted/not accepted, node in C_+/node in C_-). The activation function in this layer is the *softmax*, instead of *tanh*, to output the probability distribution among the set of C node classes. The softmax function is as follows:

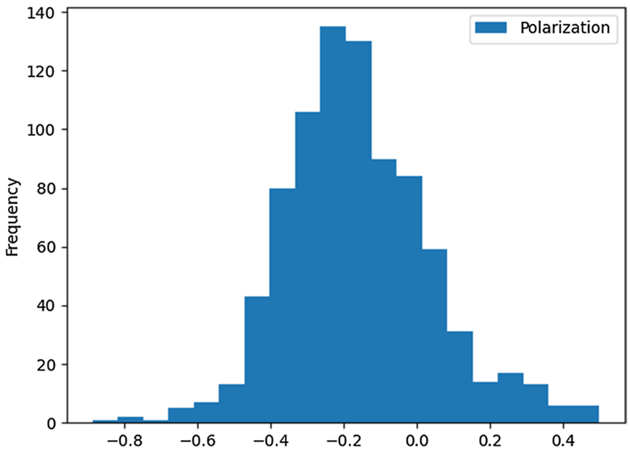

Fig. 4 Dataset polarization distribution

$$\text{softmax}\,(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \quad \text{for } i \text{ } in \text{ } 1, \ldots, C.$$

The rationale of creating this second architecture is its flexibility regarding the polarization computation. If a small change is introduced in such a computation (e.g., increasing the weight for the set C_+), the architecture 1GNN needs to be retrained. Rather, the architecture 2GNN does not need a retraining in such a case, and the polarization can be computed straight away.

4 Experimental Results

In this section, we present the results obtained when learning GNN models following the two GNN-based approaches introduced in Sect. 3 to compute the polarization for a set of Reddit debates. All the experiments were performed in a Linux Ubuntu 20.04 with Intel i7 at 3.20GHz and 32GB of RAM memory, and a GPU GeForce GTX 1080 Ti. Finally, we compare the GNN-based approaches with the distributed solver approach when computing the polarization.

4.1 Data Description

To train and test our models, we use a dataset with 843 Reddit debates.[5] The computation of a debate polarization is automatically computed by the distributed solver [18]. The size of the debates (number of comments) in the dataset ranges from 50 to 500, and the debates are from four different topics: *bitcoin, funny, politics*, and *travel*. These topics have been selected with the goal to have a broad interval of

[5] The data set can be downloaded from https://greia-git.udl.cat/josep/OpenData/src/branch/master/data/2022-IJCIS/raw_data.tgz.

Table 1 Experimental results for polarization computation with our 1GNN model

Num GIN layers	No normalization		With normalization	
	300 epochs	600 epochs	300 epochs	600 epochs
2	(0.01, 0.03, 0.03)	(0.01, 0.03, 0.04)	(0.01, 0.03, 0.03)	(0.01, 0.03, 0.05)
4	(0.01, 0.02, 0.02)	(0.01, 0.02, 0.03)	(0.01, 0.02, 0.03)	(0.01, 0.02, 0.03)
6	(0.01, 0.02, 0.03)	(0.01, 0.03, 0.03)	(0.01, 0.03, 0.04)	(0.03, 0.02, 0.03)
8	(0.01, 0.03, 0.03)	(0.05, 0.03, 0.05)	(0.05, 0.03, 0.04)	(0.03, 0.02, 0.03)

For each case, the triplet of values (train, validation, test) shows the average loss for the train, validation and test sets

Table 2 Experimental results for accepted nodes computation with our 2GNN model

Num GIN layers	No normalization		With normalization	
	300 epochs	600 epochs	300 epochs	600 epochs
2	(0.39, 0.41, 0.39)	(0.38, 0.40, 0.39)	(0.39, 0.39, 0.38)	(0.38, 0.39, 0.38)
4	(0.36, 0.36, 0.36)	(0.35, 0.36, 0.36)	(0.37, 0.37, 0.37)	(0.37, 0.37, 0.37)
6	(0.36, 0.37, 0.36)	(0.35, 0.36, 0.36)	(0.37, 0.37, 0.37)	(0.36, 0.36, 0.36)
8	(0.37, 0.37, 0.36)	(0.36, 0.37, 0.36)	(0.39, 0.39, 0.38)	(0.37, 0.37, 0.37)

For each case, the triplet of values (train, validation, test) shows the average loss for the train, validation and test sets

different polarization values. After analyzing the polarization in the debates, we observe that the distribution of polarization ranges from − 0.89 to 0.5, with a mean of − 0.16 and with 75% of the debates with a polarization below − 0.05. Although we selected different topics, expecting some to be more positive than others, in general they tend to have a more negative polarization value. We can observe the complete distribution of the polarization of the debates in the histogram shown on Fig. 4.

To download the set of comments for each Reddit debate, we use the Python Reddit API Wrapper (PRAW).[6] Then, in the PDebT $\mathcal{T}_\alpha$ obtained from each Reddit debate, the label for each edge (c_1, c_2) is computed with the sentiment analysis software of [28]. It uses the text of the comment c_1, where the value assigned denotes the sentiment of the answer, from highly negative (− 2) to highly positive (2). The pruning parameter α is set to the value 0.15. We have tried four different values for the number of GIN layers (2, 4, 6, 8) and also experimented with either using a normalization layer after each GIN layer or not. The number of GIN layers is kept low, compared with the size of the debates, to explore whether bounding the neighborhood size used by the GNN still allows a reasonable approximation of the right output value.

4.2 Experimental Training Analysis for 1GNN and 2GNN Models

From all the debates, 80% have been used for training, 10% for validation, and 10% have been used for testing. The training set is used to train the GNN models during a certain number of epochs of the learning algorithm, the Adam optimizer [29]. The resulting models are then run with the validation set, and the best performing model with the validation set is the one selected to be evaluated with the test set.

The experimental results with the 1GNN model for the average loss for the training, validation and test set are shown in Table 1. The loss is computed with the mean square error (MSE):

$$\frac{1}{N}\sum_{i=1}^{N}(p_s(i) - p_n(i))^2,$$

where N is the number of debates in the set, $p_s(i)$ is the correct polarization value of debate i (computed by the distributed solver), and $p_n(i)$ is the polarization value of debate i computed by the neural network.

The results show that, in general, the validation loss is quite similar to the test loss, and both slightly higher than the training loss, that could indicate a slight overfitting problem. The results obtained with different number of GIN layers do not seem to have a significant impact on the loss. Analogously, the use of normalization layers between GIN layers

[6] https://github.com/praw-dev/praw.

Table 3 Experimental results for nodes bipartition computation with our 2GNN model

Num GIN layers	No normalization		With normalization	
	300 epochs	600 epochs	300 epochs	600 epochs
2	(0.61, 0.63, 0.62)	(0.60, 0.63, 0.62)	(0.63, 0.63, 0.61)	(0.64, 0.63, 0.61)
4	(0.60, 0.64, 0.62)	(0.59, 0.64, 0.62)	(0.64, 0.64, 0.62)	(0.64, 0.64, 0.62)
6	(0.60, 0.64, 0.62)	(0.59, 0.64, 0.62)	(0.64, 0.63, 0.62)	(0.64, 0.63, 0.62)
8	(0.60, 0.65, 0.63)	(0.59, 0.64, 0.63)	(0.60, 0.63, 0.62)	(0.59, 0.63, 0.62)

For each case, the triplet of values (train, validation, test) shows the average loss for the train, validation and test sets

do not seem to have a significant impact, as the results are almost the same.

For the 2GNN model, we show the loss results for the models obtained for the first component (GNN for computing accepted nodes) on Table 2 and the loss results for the models obtained for the second component (GNN for computing the bipartition of the nodes) on Table 3. In this case, the loss is computed using the *cross-entropy* or *logloss* function between the output of the GNN model (p) and the target values for each node (y). The cross-entropy loss for a binary classification problem (as is the case for our two component GNNs of the 2GNN model) with target classes encoded with the set $\{0, 1\}$ is calculated as follows:

$$logloss(y,p) = -\frac{1}{N}\sum_{i=1}^{N}\frac{1}{M_i}\sum_{j=1}^{M_i}\big(y_{i,j}\log(p_{i,j}) + (1-y_{i,j})\log(1-p_{i,j})\big), \tag{3}$$

where N is the number of debates, M_i is the number of nodes for debate i, $y_{i,j}$ is the true label of the node j of the debate i (0 or 1), $p_{i,j}$ is the predicted probability that node j of the debate i belongs to class 1 (so $1 - p_{i,j}$ is the probability to belong to class 0). The reason behind using this loss function is that the log value offers less penalty for small differences between predicted probability and corrected probability, while the larger the difference, the higher the penalty.

For this model, the loss values for training, validation and test show very similar values, even when modifying the number of GIN layers or when using or not using normalization layers. The *logloss* value obtained is always at least 0.3, that indicates that the probability for the right class label is at most 0.5. This seems to indicate that trying to predict the exact state of each node (comment in the debate) with respect to its acceptance or with respect to its side in the bipartition is a challenging task.

4.3 Performance Comparison with the Distributed Solver Approach

Finally, we have performed one final experiment with the goal to analyze the performance of our GNN models as the size of the debates increases, with respect to their execution times, when computing the polarization. We study the performance of our two proposed GNN approaches: the one that uses the single GNN model to directly compute the polarization (1GNN model) and the one that computes the polarization with the formula from Definition 1 from the approximation of the accepted nodes and node bipartition computed with two different GNN components (2GNN model).

First, we compare the execution time of our previous distributed solver [18], that is used in our previous works about measuring discussion polarization with argumentation-based models [8, 9], with the 1GNN and 2GNN models[7] in a test set of 31 debates with sizes in the range [100, 13, 048], obtained from the subreddit *worldnews* filtering the debates by size. The distributed solver approach first takes the PDebT input to transform it to its corresponding WBDebG, and then it computes its set of accepted comments (following the ideal semantics), and finally it computes the polarization. In the 1GNN and 2GNN approaches, the input is always the PDebT and the final output is always the polarization.

Figure 5 shows these results. The left side of the Figure shows the results with the y-axis in linear scale. We can see that there is a very big difference between the distributed solver and the GNN models, being the two GNN models much faster. We cannot appreciate the difference between the two GNN models on linear scale, so on the right side of the figure we have the same plot in log scale. In this second plot, we can see that the 1GNN model is slightly faster than the 2GNN. This is an expected result, as the 2GNN model is the combination of two GNN models. In this second plot, we can also see that the difference between the GNN models and the distributed solver is around 3 orders of magnitude. However, it is clear that in a final application using GNNs, one has to take into account the training time. In our case, for the datasets used, the training time for a model has been around 8 minutes. Nevertheless, this training will be performed only once, or each time there is a significant change in the dataset considered for training.

[7] The models used are the best validation models obtained with 8 layers and normalization layers in the training performed with the first dataset.

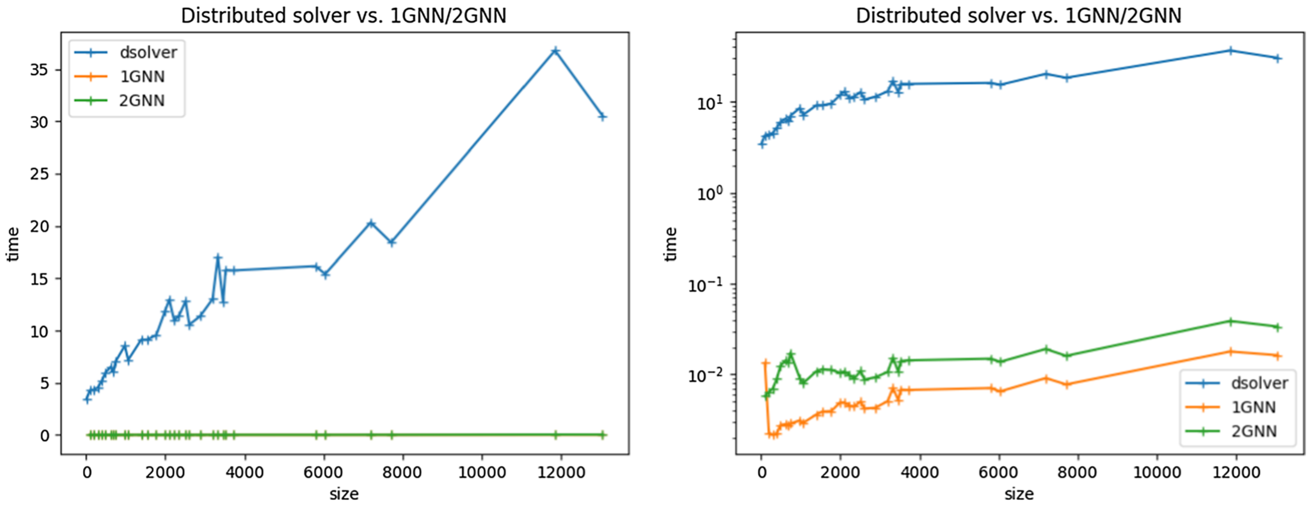

Fig. 5 Execution time of the distributed solver versus the execution time of the 1GNN and 2GNN models. Plot on the left side in linear scale on the *y*-axis, and plot on the right side in log scale

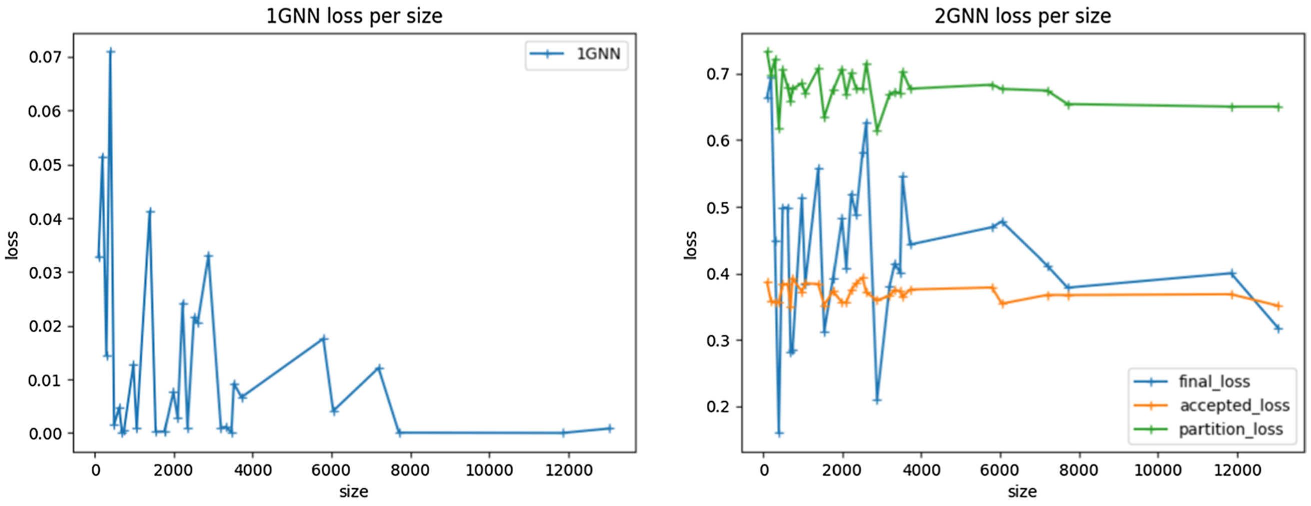

Fig. 6 MSE loss of the 1GNN model (left) and logloss of the two GNNs of the 2GNN model together with the MSE loss (final_loss) of the polarization computed with the 2GNN model (right). The *accepted loss* is the logloss of the first GIN used to compute the set of accepted comments (cf. Sect. 2). The *partition loss* is the logloss of the second GIN used to compute the bipartition of the corresponding WBDebG. The *final loss* is the MSE loss of the polarization computation

4.4 Polarization Error Analysis

Next, we analyze the errors (test loss) of our 1GNN and 2GNN models, when computing the polarization, on the same test set used for the time performance comparison. Observe that the MSE error for the 1GNN model and the MSE error for final loss of the 2GNN model are actually giving us a measure of the distance between the polarization computed by the GNN models and the correct one computed by our baseline method, i.e., using the distributed solver.

Figure 6 shows the MSE loss for the 1GNN model (left part) and the right part shows the logloss of the 2GNN model for its two building GNNs (the one for computing the accepted nodes and the one for computing the bipartition of the nodes) and the final MSE loss (the MSE loss of the polarization computed with its defining formula from the information of these two GNNs). For the 1GNN model, the results show that the error is bounded in a small interval ([0, 0.07]) with an average value of 0.0127 and as the size of the debates increases, it seems to stabilize in a smaller range. For the 2GNN model, the results show a different average loss for each different component: 0.37 for the computation of the accepted nodes, 0.67 for the bipartition computation and 0.44 for the final value (MSE loss for the polarization).

So, the MSE loss for the polarization of the 2GNN model is bigger than in the 1GNN model.

However, the fact that the 2GNN model offers more flexibility regarding its possible use to compute other measures, as we discussed previously in Sect. 3.2, still makes this model an interesting approach for computing different measures defined from the accepted nodes and their bipartition. Also, the fact that the error seems to not increase as the debate size increases may give at least the option to bound the uncertainty in the final computed value.

5 Conclusions and Future Work

In this paper, we have presented two GNN systems, based on graph isomorphism networks, to solve the problem of computing a polarization measure from a Reddit debate. In the first GNN model (1GNN), we take as input a tree representation of the debate to analyze, and outputs the value of a polarization measure that is originally defined as a function of the solution of the corresponding argumentation problem and of the bipartite graph representation of the debate. Although the 1GNN model does not explicitly compute the solution of the argumentation problem, it is able to approximate the final polarization measure, that it is originally defined from that solution. This happens even if our GNN model aggregates information in each node considering always a neighborhood with bounded distance, given that the number of GIN layers is kept constant. In the second GNN model (2GNN) we have considered instead the intermediate computation of the solution of the argumentation problem and the bipartition corresponding to the bipartite graph representation of the debate. This second model offers a more flexible system, that can be used to compute other measures of interest apart from the polarization measure we have considered here. However, the error we obtain with this second model is higher than with the first GNN model.

An interesting direction for future work is to consider the computation of other argumentation-based measures that consider as input author graphs, instead of debate trees. Author graphs come from the aggregation of comments from the same author in a single node, such that the resulting graph may contain cycles, and in that case the complexity of the argumentation-based reasoning algorithm is higher than the one for the acyclic graphs we have considered in this work. Another future line of research is to explore how to combine the weights of the comments when defining the polarization measure to give more relevance to comments that have received more attention in the debate.

Acknowledgements The authors would like to thank the anonymous reviewers for providing helpful comments.

Author Contributions All authors have contributed equally to the work, and all authors have read and agreed to the submitted version of the manuscript.

Funding This work was partially funded by Spanish Project PID2019-111544GB-C22, and by the Universitat de Lleida Requalification (2021–2023) under the EU Next Generation.

Availability of Data and Materials To train and test our models, we use a dataset publicly available at https://greia-git.udl.cat/josep/OpenData/src/branch/master/data/2022-IJCIS/raw_data.tgz.

Declarations

Conflict of Interest The authors declare that they have no conflict of interest.

Ethics Approval and Consent to Participate Not applicable.

Consent for Publication Not applicable.

References

1. Aragon, P., Gomez, V., Garcia, D., Kaltenbrunner, A.: Generative models of online discussion threads: state of the art and research challenges. J. Intern. Serv. Appl. **8**(15), 1–17 (2017)
2. Barabási, A.-L., Albert, R.: Emergence of scaling in random networks. Science **286**(5439), 509–512 (1999). https://doi.org/10.1126/science.286.5439.509
3. Lusher, D., Koskinen, J., Robins, G. (eds.): Exponential Random Graph Models for Social Networks: Theory, Methods, and Applications. Structural Analysis in the Social Sciences. Cambridge University Press, Cambridge (2012). https://doi.org/10.1017/CBO9780511894701
4. Hamilton, W.L., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: Guyon, I., von Luxburg, U., Bengio, S., Wallach, H.M., Fergus, R., Vishwanathan, S.V.N., Garnett, R. (eds.) Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4–9, 2017, Long Beach, CA, USA, pp. 1024–1034 (2017). https://proceedings.neurips.cc/paper/2017/hash/5dd9db5e033da9c6fb5ba83c7a7ebea9-Abstract.html
5. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24–26, 2017, Conference Track Proceedings. OpenReview.net (2017). https://openreview.net/forum?id=SJU4ayYgl
6. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? In: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6–9,

2019. OpenReview.net (2019). https://openreview.net/forum?id=ryGs6iA5Km
7. Dunne, P.E.: The computational complexity of ideal semantics. Artif. Intell. **173**(18), 1559–1591 (2009)
8. Alsinet, T., Argelich, J., Béjar, R., Martínez, S.: An argumentation approach for agreement analysis in reddit debates. In: Artificial Intelligence Research and Development—Current Challenges, New Trends and Applications, CCIA 2018, 21st International Conference of the Catalan Association for Artificial Intelligence, Alt Empordà, Catalonia, Spain, 8–10th October 2018, pp. 217–226 (2018). https://doi.org/10.3233/978-1-61499-918-8-217
9. Alsinet, T., Argelich, J., Béjar, R., Martínez, S.: Measuring user relevance in online debates through an argumentative model. Pattern Recognit. Lett. **133**, 41–47 (2020). https://doi.org/10.1016/j.patrec.2020.02.008
10. Kuhlmann, I., Thimm, M.: Using graph convolutional networks for approximate reasoning with abstract argumentation frameworks: a feasibility study. In: Ben Amor, N., Quost, B., Theobald, M. (eds.) Scalable Uncertainty Management, pp. 24–37. Springer, Berlin (2019)
11. Craandijk, D., Bex, F.: Deep learning for abstract argumentation semantics. In: Bessiere, C. (ed.) Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20, pp. 1667—1673. International Joint Conferences on Artificial Intelligence Organization (2020). https://doi.org/10.24963/ijcai.2020/231. Main Track
12. Schmidt, R.M.: Recurrent neural networks (RNNs): a gentle Introduction and Overview. arXiv (2019). https://doi.org/10.48550/ARXIV.1912.05911
13. Choi, D., Han, J., Chung, T., Ahn, Y., Chun, B., Kwon, T.T.: Characterizing conversation patterns in reddit: from the perspectives of content properties and user participation behaviors. In: Proceedings of the 2015 ACM on Conference on Online Social Networks, COSN 2015, Palo Alto, California, USA, November 2–3, 2015, pp. 233–243 (2015)
14. Gómez, V., Kaltenbrunner, A., López, V.: Statistical analysis of the social network and discussion threads in Slashdot. In: Proceedings of the 17th International Conference on World Wide Web. WWW'08, pp. 645–654. Association for Computing Machinery, New York, NY, USA (2008). https://doi.org/10.1145/1367497.1367585
15. Waller, I., Anderson, A.: Quantifying social organization and political polarization in online platforms. Nature **600**, 264–268 (2021). https://doi.org/10.1038/s41586-021-04167-x
16. Bench-Capon, T.J.M.: Persuasion in practical argument using value-based argumentation frameworks. J. Log. Comput. **13**(3), 429–448 (2003)
17. Dung, P.M., Mancarella, P., Toni, F.: Computing ideal sceptical argumentation. Artif. Intell. **171**(10–15), 642–674 (2007)
18. Alsinet, T., Argelich, J., Béjar, R., Cemeli, J.: A distributed argumentation algorithm for mining consistent opinions in weighted twitter discussions. Soft. Comput. **23**(7), 2147–2166 (2019). https://doi.org/10.1007/s00500-018-3380-x
19. Malewicz, G., Austern, M.H., Bik, A.J.C., Dehnert, J.C., Horn, I., Leiser, N., Czajkowski, G.: Pregel: a system for large-scale graph processing. In: Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data. SIGMOD'10, pp. 135–146. Association for Computing Machinery, New York, NY, USA (2010). https://doi.org/10.1145/1807167.1807184
20. Errica, F., Podda, M., Bacciu, D., Micheli, A.: A fair comparison of graph neural networks for graph classification. In: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26–30, 2020. OpenReview.net (2020). https://openreview.net/forum?id=HygDF6NFPB
21. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Yu, P.S.: A comprehensive survey on graph neural networks. IEEE Trans. Neural Netw. Learn. Syst. **32**(1), 4–24 (2021). https://doi.org/10.1109/TNNLS.2020.2978386
22. Lim, J., Ryu, S., Park, K., Choe, Y.J., Ham, J., Kim, W.Y.: Predicting drug-target interaction using a novel graph neural network with 3D structure-embedded graph representation. J. Chem. Inf. Model. **59**(9), 3981–3988 (2019). https://doi.org/10.1021/acs.jcim.9b00387
23. Wen, J., Liu, Y., Shi, Y., et al.: A classification model for LNCRNA and MRNA based on *k*-MERS and a convolutional neural network. BMC Bioinform. (2019). https://doi.org/10.1186/s12859-019-3039-3
24. Sanchez-Gonzalez, A., Godwin, J., Pfaff, T., Ying, R., Leskovec, J., Battaglia, P.: Learning to simulate complex physics with graph networks. In: III, H.D., Singh, A. (eds.) Proceedings of the 37th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 119, pp. 8459–8468. PMLR (2020). https://proceedings.mlr.press/v119/sanchez-gonzalez20a.html
25. Weisfeiler, B.Y., Leman, A.A.: A reduction of a graph to a canonical form and an algebra arising during this reduction. Nauchno Tech. Inf. **2**(9), 12–16 (1968)
26. Ba, L.J., Kiros, J.R., Hinton, G.E.: Layer normalization. CoRR (2016) arXiv:1607.06450
27. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Improving neural networks by preventing co-adaptation of feature detectors. CoRR (2012) arXiv:1207.0580
28. Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S.J., McClosky, D.: The Stanford CoreNLP natural language processing toolkit. In: Association for Computational Linguistics (ACL) System Demonstrations, pp. 55–60 (2014). http://www.aclweb.org/anthology/P/P14/P14-5010
29. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. CoRR (2017) arXiv:1412.6980