Neural Discovery of Balance-aware Polarized Communities

Francesco Gullo[1], Domenico Mandaglio[2], Andrea Tagarelli[2*]

[1]Dept. of Information Engineering, Computer Science, and Mathematics (DISIM), University of L'Aquila, Coppito (AQ), Italy.
[2]Dept. Computer Engineering, Modeling, Electronics, and Systems Engineering (DIMES), University of Calabria, Rende (CS), Italy.

*Corresponding author(s). E-mail(s): tagarelli@dimes.unical.it;
Contributing authors: gullof@acm.org; d.mandaglio@dimes.unical.it;

Abstract

Signed graphs are a model to depict friendly (*positive*) or antagonistic (*negative*) interactions (edges) among users (nodes). 2-Polarized-Communities (2pc) is a well-established combinatorial-optimization problem whose goal is to find two *polarized* communities from a signed graph, i.e., two subsets of nodes (disjoint, but not necessarily covering the entire node set) which exhibit a high number of both intra-community positive edges and negative inter-community edges. The state of the art in 2pc suffers from the limitations that (***i***) existing methods rely on a single (optimal) solution to a continuous relaxation of the problem in order to produce the ultimate discrete solution via rounding, and (***ii***) 2pc objective function comes with no control on size balance among communities.
In this paper, we provide advances to the 2pc problem by addressing both these limitations, with a twofold contribution. First, we devise a novel neural approach that allows for soundly and elegantly explore a variety of suboptimal solutions to the relaxed 2pc problem, so as to pick the one that leads to the best discrete solution after rounding. Second, we introduce a generalization of 2pc objective function – termed $\boldsymbol{\gamma}$-*polarity* – which fosters size balance among communities, and we incorporate it into the proposed machine-learning framework.
Extensive experiments attest high accuracy of our approach, its superiority over the state of the art, and capability of function $\boldsymbol{\gamma}$-polarity to discover high-quality size-balanced communities.

Keywords: polarization, signed graphs, neural networks

1 Introduction

The widespread use of modern social media has created a huge amount of online social interactions, fostered the formation of communities (e.g., [1–3]) and facilitating discussions about a variety of topics. Users establish *positive* relationships such as friendships, agreements, and supports, as well as *negative* relationships such as foes, disagreements, and distrusts. The existence of such mixed interactions has led to an ever-growing *polarization* phenomenon, i.e., a division of the set of users into groups with opposite view on controversial topics (e.g., politics, religion, sport).

In the past few years, we have witnessed a plethora of studies about polarization on social media [4–6]. Polarization is often distinguished in ideological and affective polarization [7]: the former refers to increased ideological divergence and reduced dialogue among individuals with differing views, whereas the latter focuses on affective attitude that individuals show toward others based on their opinions [5]. Nonetheless, other approaches to the study of polarization discard a particular qualification of the term polarization, while adopting a graph-theoretic setting where the goal is to discover *polarized communities* in *signed graphs* [8, 9]. In this work, we follow the latter line of studies. Remarkably, a key novelty in our work is the exploitation of machine learning, particularly *neural network* models, for discovering a polarization structure.

Polarization in signed graphs. Signed graphs are graphs whose edges are assigned either a *positive* or a *negative* label, denoting whether the interaction depicted by an edge is friendly or antagonistic, respectively [10]. Signed graphs are used to model a variety of data and study numerous (social) phenomena, such as emergence of polarized discussions in social media, or analysis of trust/distrust in review platforms [11–14]. Bonchi et al. [8] employ signed graphs to define the problem of 2-Polarized-Communities (for short, 2pc), which requires finding two subsets of nodes, generally referred to as *communities*, of the input signed graph such that there are (R1) mostly positive edges within each community and (R2) mostly negative edges between the two communities, and (R3) the subgraph induced by these two communities is as much dense as possible; for instance, assuming that positive (resp. negative) edges denote agreement (resp. disagreement) of social media users w.r.t. a given context of debate, identifying the two polarized communities correspond to detecting two groups of users, where users of the same group mostly agree with each other, while having divergent opinions with respect to the users of the other group. Also, the two communities are required to be *non-overlapping*, but they *do not necessarily need to cover the entire node set*. The rationale of the latter is to comply the most with real-world situations, where polarized communities are concealed within a body of other graph nodes which do not (yet) have a strongly formed opinion, and, as such, they are *neutral* in terms of polarization.

Motivation: limitations of the state of the art in 2pc. The above R1–R3 requirements for the 2pc problem are jointly pursued by maximizing a single objective function, termed *polarity*. Bonchi et al. [8] show that maximizing polarity is **NP**-hard, but also that a continuous relaxation of that problem is solvable in polynomial time. They exploit this finding to devise algorithms which consist in properly rounding (i.e., discretizing) the optimal solution of the relaxed problem.

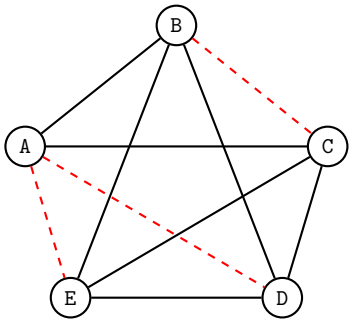


Fig. 1 Example signed graph, with solid lines and red dashed lines corresponding to positive and negative edges, respectively. The optimal solution of the relaxed 2PC problem yields, after rounding, one community containing node `A` only and the other community containing all the remaining nodes. However, the same rounding strategy applied to a suboptimal solution of the relaxed problem leads to node `A` and nodes `D-E` forming the two communities, while nodes `B-C` are detected as neutral: this is not only a more likely realistic configuration (due to the roles played by nodes `B-C`) than the communities derived from the optimal solution, but it also turns out to have *higher polarity* than the latter. For more details on this example, see Example 1 in Section 3.

Limitation 1. Despite Bonchi et al.'s algorithms are principled and rather effective, they suffer from the fact that deriving a solution to 2PC starting from the optimal solution of the relaxed problem may be limiting in terms of polarity. In fact, as illustrated in Figure 1, *suboptimal* solutions to the relaxed problem can lead to *better solutions* to 2PC after rounding.

Limitation 2. The polarity function does not require or foster the detection of *size-balanced* communities. Indeed, maximizing polarity can easily lead to degenerate solutions with a single sufficiently large community and another (almost) empty community, even if the input signed graph does contain "natural" polarized communities that are both non-empty and possibly of comparable size. Several types of social environments, from social media platforms to online forums, from political systems to scientific communities, can benefit from identifying and maintaining balanced polarized communities. By ensuring that communities are diverse while still containing balanced viewpoints, constructive debates can be facilitated, critical thinking can be encouraged, and the individuals' perspectives can be broadened, thus reducing echo chambers and mitigating the spread of misinformation. Also, companies involved in market research and product development can benefit from gathering insights from diverse consumer groups balanced in size, and ultimately better understand market preferences and anticipate consumer trends. Therefore, there is a need for methods that can detect *balanced polarized communities*. In this respect, turning back to example of Figure 1, the assignment of `A` to one community, `D-E` to the other community, and `B-C` as neutral, is also much more balanced than the one derived from the optimal relaxed-solution, where node `A` forms its own community, and all the other nodes are assigned to the other community.

The above example also highlights the need for detecting fine-grained polarization phenomena, i.e., polarized communities that may not be apparent in terms of node size or amount of connections involved therein. This is in fact essential to recognize minorities in polarization, which in turn might correspond to harmful situations like isolation, where a small group of individuals are marginalized or isolated by a larger, cohesive group; later in this paper, we will provide an example of such polarization setting.

Contributions. In this paper, we advance the state of the art in the 2PC problem by properly addressing the above limitations. Specifically, we provide a twofold contribution.

First, targeting Limitation 1, we devise a novel *machine-learning* approach that allows for soundly and effectively exploring a variety of suboptimal solutions to the relaxed problem, so as to ultimately select the one that leads to the best discrete solution to 2PC after rounding.

Second, to overcome Limitation 2, we devise a generalization of the polarity function, named γ-*polarity*. When optimizing standard polarity, in fact, 2PC solutions tend to produce strongly imbalanced polarized communities, especially when dealing with large graphs. Our proposed γ-*polarity* is designed to produce polarized communities that, depending on the setting of γ, can be either more balanced or larger than those yielded by standard polarity.

The proposed approach leverages a *neural-network*-based framework, whose core component is a *signed graph neural network* (GNN) model, to learn continuous vector representations of the input nodes, for the task of assigning each node a real-valued score between -1 and 1. Such a score is ultimately rounded onto $\{-1, 0, 1\}$, so as to determine whether the corresponding node is part of one of the two communities (-1 or 1), or is neutral (0). To this purpose, our neural framework is optimized via a loss corresponding to the relaxed polarization function, coupled with a suitable regularization term.

Rationale and benefits of our proposal are as follows:

(1) A neural approach is well-suited for 2PC due to the compatibility of continuous relaxation of 2PC with neural network differentiability. By setting the loss function to the relaxed 2PC objective and performing rounding after each learning step, we bridge the gap between discrete constraints of the underlying combinatorial-optimization problem and the inherently continuous mathematical framework of neural networks. Also, both input and output of (relaxed) 2PC are naturally handled by neural-network building blocks too: the input graph by a signed GNN, and the output $[-1, 1]$ score by a tanh activation function. Furthermore, external information associated with nodes can easily be integrated into our framework, since GNNs are designed to initialize the hidden node representations (embeddings) with any available node features.

(2) While simple, our approach is backed by solid machine-learning fundamentals, which make it principled and sound. In fact, training our neural framework via standard gradient descent provides an elegant solution to the aforementioned requirement of exploring a variety of suboptimal solutions to relaxed 2PC. Every epoch of training of our framework ends up with a rounding which produces a discrete 2PC solution,

where a proper loss regularization term is introduced to enforce the continuous scores to be closer to discrete $\{-1, 0, 1\}$ values.

(3) Our framework is lightweight, yet highly versatile and modular, facilitating easy maintenance and updates to keep pace with the latest GNN models and deep learning advancements. Future improvements, like enhanced signed GNNs, can be seamlessly integrated by modifying a single building block. Additionally, our framework allows for seamlessly incorporating additional requirements on the yielded solutions, such as fostering size balance, as we discuss next.

(4) We effectively address non-size-balanced communities by maximizing γ-polarity, a generalization of standard polarity, using it as a loss in our neural framework. This links our second main contribution (γ-polarity) with our first (neural approach to 2PC).

(5) Our proposal offers multiple benefits to the research community. It represents the first machine-learning approach to 2PC, which opens the door to further research and improvement. The same applies to γ-polarity, which warrants additional exploration from a combinatorial optimization perspective. Moreover, our work can serve as inspiration for other combinatorial optimization problems, which share the common trait with 2PC that suboptimal relaxed solutions may lead to improved rounded solutions.

Summary and roadmap. Our main contributions in this work can be summarized as follows:

- We tackle 2PC [8], i.e., the problem of discovering two polarized communities from an input signed graph (Section 2), and define a novel neural-network-based approach to address it (Section 3).
- We introduce a generalization of 2PC's objective function, termed γ-polarity, which favors size balance among communities, and show how to optimize it within the proposed neural framework (Section 4).
- We provide extensive experiments on a large variety of real-world and synthetic signed graphs (Section 5). Results (Section 6) attest high accuracy of our approach, its superiority over the state of the art, and the effectiveness of γ-polarity in detecting balanced communities.

Section 7 concludes the paper and discusses future work.

2 Preliminaries and Background

Let $G = (V, E^+, E^-)$ be an *undirected signed graph*, where V is a set of nodes, and $E^+, E^- \subseteq V \times V$, $E^+ \cap E^- = \emptyset$, are sets of *positive* and *negative* edges, respectively. We assume an arbitrary order over V, such that nodes are assigned a unique integer ID within $\{1, \ldots, |V|\}$. With a little abuse of notation, we interchangeably refer to $u \in V$ as both the node u itself and the u-th node in the order. This keeps vector/matrix notations simpler. $\mathbf{A} \in \{-1, 0, 1\}^{|V| \times |V|}$ is the *signed adjacency matrix* of G, defined as $\mathbf{A}[u, v] = 1$ if $(u, v) \in E^+$, $\mathbf{A}[u, v] = -1$ if $(u, v) \in E^-$, and $\mathbf{A}[u, v] = 0$ otherwise. Table 1 summarizes main notations used throughout this paper.

Table 1 Main notations used in this paper.

notation	description
$G = (V, E^+, E^-)$	Signed graph (V: node set; E^+: positive edge set; E^-: negative edge set)
$\mathbf{A}$	signed adjacency matrix of G ($A \in \{-1, 0, 1\}^{\lvert V\rvert \times \lvert V\rvert}$)
S_0, S_1, S_2	partition of V into polarized communities (S_1, S_2) and neutral nodes (S_0)
$\mathbf{x} \in \{-1, 0, 1\}^{\lvert V\rvert}$	node-to-community assignment
$\mathbf{z} \in [-1, 1]^{\lvert V\rvert}$	relaxed node-to-community assignment
$p(\mathbf{x}, \mathbf{A})$	polarity of $\mathbf{x}$ with respect to $\mathbf{A}$ (Def. 1)
$p_\gamma(\mathbf{x}, \mathbf{A})$	γ-polarity of $\mathbf{x}$ with respect to $\mathbf{A}$ (Def. 2)
$\mathbf{H}_0 \in \mathbb{R}^{\lvert V\rvert \times d_I}$	node feature matrix
$f_\theta(\cdot, \cdot)$	proposed neural-network model (Eq. (4))
$\mathcal{L}_{\text{2PC}}(\cdot, \cdot, \cdot)$	loss function used to train the proposed model (Eq. (5))
$\tau \in [0, 1]$	threshold for rounding a $[-1, 1]$ value onto $\{-1, 0, 1\}$
Z_i	set of threshold values derived from a continuous solution $\mathbf{z}$ by approximating each $z[u]$ at the i-th decimal digit
e_{max}	number of training epochs

2.1 Problem statement

We deal with the combinatorial-optimization problem of 2-Polarized-Communities (for short, 2PC), originally defined by Bonchi et al. [8]. Given a signed graph $G = (V, E^+, E^-)$, 2PC finds two disjoint subsets $S_1, S_2 \subseteq V$ of nodes such that (R1) there are as many positive edges and as few negative edges as possible within S_1 and within S_2; (R2) there are as many negative edges and as few positive edges as possible across S_1 and S_2; and (R3) the subgraph induced by $S_1 \cup S_2$ is as dense as possible, according to a density defined as the ratio between number of edges and number of nodes.

S_1 and S_2 are interpreted as *polarized communities*, i.e., groups of users (nodes) who are cohesive in terms of both intra-group positive relationships (edges) and inter-group negative relationships. Nodes included into neither S_1 nor S_2 – denoted $S_0 = V \setminus (S_1 \cup S_2)$ – form the set of *neutral* nodes. A partition $\{S_0, S_1, S_2\}$ of V can alternatively be represented by a (column) vector $\mathbf{x} \in \{-1, 0, 1\}^{\lvert V\rvert}$, whose u-th coordinate is $\mathbf{x}_u = 0$ if $u \in S_0$, $\mathbf{x}_u = 1$ if $u \in S_1$, and $\mathbf{x}_u = -1$ if $u \in S_2$.

The above R1–R3 requirements of 2PC are altogether encoded into a single function, termed *polarity*:

Definition 1 (Polarity [8])**.** *Given a vector* $\mathbf{x} \in \{-1, 0, 1\}^{\lvert V\rvert}$ *and a matrix* $\mathbf{A} \in \{-1, 0, 1\}^{\lvert V\rvert \times \lvert V\rvert}$*, the polarity* $p(\mathbf{x}, \mathbf{A})$ *of* $\mathbf{x}$ *with respect to* $\mathbf{A}$ *is defined as:*

$$p(\mathbf{x}, \mathbf{A}) = \frac{\mathbf{x}^\top \mathbf{A}\ \mathbf{x}}{\mathbf{x}^\top \mathbf{x}}. \tag{1}$$

The numerator of $p(\cdot, \cdot)$ accounts for R1 and R2, while numerator and denominator altogether model R3. In this regard, note that $\mathbf{x}^\top \mathbf{x} = |S_1 \cup S_2|$.

The 2PC problem is formulated as follows:

Problem 1 (2PC [8])**.** *Given a signed graph* $G = (V, E^+, E^-)$ *with signed adjacency matrix* $\mathbf{A}$*, find*

$$\mathbf{x}^* = \underset{\mathbf{x} \in \{-1,0,1\}^{\lvert V\rvert}}{\arg\max}\ p(\mathbf{x}, \mathbf{A}).$$

Relaxing node-to-community assignments to be continuous, i.e., $\in [-1, 1]$, leads to the following relaxed problem:

Problem 2 (2PC-RELAXED [8])**.** *Given a signed graph $G = (V, E^+, E^-)$ with signed adjacency matrix $\mathbf{A}$, find*

$$\mathbf{z}^* = \underset{\mathbf{z}\in[-1,1]^{|V|}}{\arg\max}\ p(\mathbf{z}, \mathbf{A}),$$

where polarity $p(\mathbf{z}, \mathbf{A}) = \mathbf{z}^\top \mathbf{A}\ \mathbf{z}/\mathbf{z}^\top \mathbf{z}$ of a real-valued vector $\mathbf{z} \in [-1, 1]^{|V|}$ is defined the same as Definition 1.

State of the art in 2pc. 2PC is shown to be **NP**-hard, while 2PC-RELAXED can be solved in polynomial time by finding the eigenvector of the signed adjacency matrix corresponding to the largest eigenvalue [8]. Bonchi et al. [8] exploit the latter to devise two approximation algorithms for 2PC. The first (deterministic) algorithm simply rounds the optimal solution $\mathbf{z}^*$ to 2PC-RELAXED as $\mathbf{x}^*_u = \text{sgn}(\mathbf{z}^*_u)$, for all $u \in V$, where $\text{sgn}(\cdot)$ is the sign function. The second (randomized) algorithm sets, for all $u \in V$, $\mathbf{x}^*_u = \text{sgn}(\mathbf{z}^*_u)$ if a Bernoulli experiment with success probability $|\mathbf{z}^*_u|$ succeeds, otherwise $\mathbf{x}^*_u = 0$.

Tzeng et al. [9] extend 2PC to a k-community setting, where the goal is to find $k \geq 2$ node subsets, each of which is positively connected internally, and negatively connected to the other subsets. Extending our approach to $k > 2$ communities is an interesting direction for future work.

2.2 Related works

Besides polarization in signed graphs, which is the focus of our study, it is useful to recall here methods that address related problems.

Representation learning for signed graphs. *Graph representation learning* is the problem of assigning elements of a graph (e.g., nodes, edges, subgraphs) to numerical vectors (*embeddings*) such that the similarity between those elements in the graph corresponds to the similarity between their embeddings. The literature on graph representation learning is vast, and includes learning approaches that are shallow, which optimize a certain criterion directly (e.g., d-hop reachability, random-walk co-occurrence) and deep, i.e., based on *graph neural networks* (GNNs) [15, 16]. Representation learning has been studied for signed graphs as well, including both undirected methods [17–21] and directed methods [22, 23].

In this work, we regard signed graph representation learning as a building block of the proposed framework. Note that our approach is versatile w.r.t. the choice of graph representation learning model; however, creating a custom model for our specific task is beyond the scope of this work.

Clustering signed graphs has also received attention in the literature [24–28]. However, those methods require every node to be part of an output cluster, hence they are not designed to detect neutral nodes and left them out of evaluation, unlike our approach. Also, signed graph clustering methods optimize criteria other than polarity. Nonetheless, given their relative popularity yet relatedness with our problem, we shall consider some of the most prominent methods in this category in our experimental evaluation (cf. Section 5).

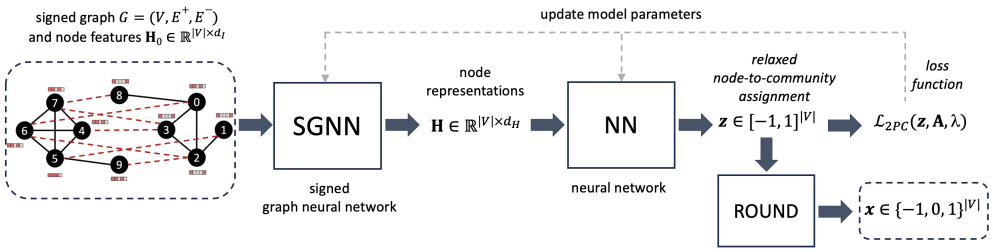


Fig. 2 Overview of the proposed Neural2PC approach.

Other patterns in signed graphs. A number of works focus on extracting subgraphs from signed graphs according to measures other than polarity. Ordozgoiti et al. [29] aim at extracting a maximum-size subgraph which exhibits *perfect balance* [10], i.e., it can be partitioned into two sets of nodes such that there are *only* positive intra-group edges and *only* negative inter-group edges. Despite the name, Ordozgoiti et al.'s notion of balance has nothing to do with *size balance* of the groups of the output subgraph. Also, the output groups may be arbitrarily sparse, as the goal is to maximize the size of the output subgraph, no matter how dense it is. Xiao et al. [30] extract subgraphs that maximize *signed bipartiteness ratio*. That measure mainly differs from polarity as it enforces separation from the identified subgraphs to the rest of the graph, rather than maximizing the density of the subgraphs. Also, Xiao et al. deal with a *local* setting, where subgraphs are built by expanding two given sets of seed nodes. Niu and Sariyüce [31] deal with *dichotomy*, a variant of polarity which is more oriented to cohesiveness. Chu et al. [32] optimize *oppositive cohesiveness*, a measure which considers cohesiveness in absolute terms, rather than in relation to the number of nodes (it is not based on any notion of density). *Please note that handling measures other than polarity is beyond our scope.*

Machine learning for combinatorial optimization leverages machine-learning techniques to solve combinatorial-optimization problems [33, 34]. This research area has focused on problems such as influence maximization [35], (graph) clustering [36, 37], community search and detection [38]. To the best of our knowledge, *we are the first to define a machine-learning approach to the* 2PC *problem.*

3 Proposed approach: Neural2PC

Overview. Unlike existing methods [8] which find the optimal solution $\mathbf{z}^*$ to 2PC-RELAXED (Problem 2) directly, we let a *neural-network* model f_θ – with parameters θ – produce a set $\{\mathbf{z}_e \mid e = 1, \ldots, e_{max}\}$ of feasible solutions to 2PC-RELAXED during multiple epochs $1, \ldots, e_{max}$ of training. All the various $\mathbf{z}_e$ are rounded in order to yield feasible discrete solutions $\mathbf{x}_e$ to 2PC. The best (in terms of polarity, Definition 1) of such $\mathbf{x}_e$ solutions is the definitive output.

The rationale of our neural approach is that it soundly allows for exploring a variety of suboptimal solutions to 2PC-RELAXED. As better shown in Example 1 at the end of this section, this favors obtaining ultimate discrete solutions (after rounding)

which exhibit higher polarity than the one derived by rounding the optimal solution to 2PC-RELAXED.

The ultimate objective in our approach is to find the model parameters θ that maximize the polarity of the (relaxed) solutions computed via f_θ (or, equivalently, minimize a loss defined based on the negative polarity). Note that, as parameter learning goes on, it is expected to get a deeper exploration of the space of relaxed solutions, and hence a higher likelihood of getting an effective discrete solution after rounding. This is confirmed by experimental evidence (Section 6).

The proposed neural approach is named Neural2PC. A graphical illustration of its main components is shown in Figure 2. Next, we delve into its technical details.

Neural model. Our f_θ model takes as input a signed graph $G = (V, E^+, E^-)$, and a matrix $\mathbf{H}_0 \in \mathbb{R}^{|V| \times d_I}$ containing a d_I-dimensional (real-valued) vector of features for every node. Should such features be not available, $\mathbf{H}_0$ can be initialized by considering structural information derived from G [17].

The first block of f_θ is a (m-layer) signed GNN [17–21] $\textsc{sgnn}(\cdot)$, with parameters $\theta_{\textsc{sgnn}}$. $\textsc{sgnn}(\cdot)$ properly processes G's topology and (possibly) node features $\mathbf{H}_0$, and outputs a matrix $\mathbf{H} \in \mathbb{R}^{|V| \times d_H} = [\mathbf{h}_u \in \mathbb{R}^{d_H}]_{u \in V}$ containing a hidden vector representation $\mathbf{h}_u$ of every node $u \in V$:

$$\mathbf{H} = \textsc{sgnn}(G, \mathbf{H}_0). \tag{2}$$

Then, vector representations produced by $\textsc{sgnn}(\cdot)$ feed into fully-connected neural-network linear layers $\textsc{nn}(\cdot)$, with parameters $\theta_{\textsc{nn}}$. Ultimately, a tanh activation function is used to cast the (node-to-community assignment) scores for every node to the desired $[-1, 1]$ range (cf. Problem 2):

$$\mathbf{z} = \tanh(\textsc{nn}(\mathbf{H})). \tag{3}$$

As a result, the overall f_θ model is as follows:

$$f_\theta(G, \mathbf{H}_0) = \tanh(\textsc{nn}(\textsc{sgnn}(G, \mathbf{H}_0))), \tag{4}$$

and its parameters are $\theta = \{\theta_{\textsc{sgnn}}, \theta_{\textsc{nn}}\}$.

Loss function. To optimize model parameters θ, we employ a loss function $\mathcal{L}_{\text{2PC}}$ defined as a combination of (the negative of) polarity $p(\cdot, \cdot)$ (Definition 1) and a proper regularization term. The role of the latter is to enforce the model produce continuous scores that are as close as possible to the ultimately desired discrete $\{-1, 0, 1\}$ scores. Specifically, we define the regularization term as the $||\cdot||_2$ L2-norm of a vector $\rho \in \mathbb{R}^{|V|}$, whose entries $\rho[u]$, for all $u \in V$, are set to the difference $\min\{|\mathbf{z}[u]|, 1 - |\mathbf{z}[u]|\}$ between $\mathbf{z}[u]$ and the closest valid discrete score. The intuition is that minimizing the norm of ρ (together with the other loss component) is expected to produce the desired effect of yielding output continuous $\mathbf{z}$ scores not too far from the valid discrete ones.

All in all, given $\mathbf{z} = f_\theta(G, \mathbf{H}_0)$, the signed adjacency matrix $\mathbf{A}$ of G, and a hyper-parameter $\lambda \in \mathbb{R}$ which properly weighs the importance of the regularization term,

Algorithm 1 Neural2PC

Input: Signed graph $G = (V, E^+, E^-)$ with signed adjacency matrix $\mathbf{A}$; node feature matrix $\mathbf{H}_0 \in \mathbb{R}^{|V| \times d_I}$; positive integer e_{max} (number of epochs); positive real number α (learning rate); real number λ (regularization hyperparameter)

Output: vector $\mathbf{x}_{best} \in \{-1, 0, 1\}^{|V|}$

$\mathbf{x}_{best} \leftarrow \mathbf{0}^{|V|}$, $p_{best} \leftarrow -\infty$, $\theta \leftarrow$ parameter initialization
for $e = 1, \ldots, e_{max}$ **do**
  $\mathbf{z} \leftarrow f_\theta(G, \mathbf{H}_0)$ {Eq. (4)}
  $\mathbf{x} \leftarrow \text{ROUND}(\mathbf{z})$ {Eq. (6)}
  if $p(\mathbf{x}, \mathbf{A}) > p_{best}$ **then**
    $\mathbf{x}_{best} \leftarrow \mathbf{x}$, $p_{best} \leftarrow p(\mathbf{x}, \mathbf{A})$
  end if
  $\theta \leftarrow$ gradient-descent step over θ, with loss $\mathcal{L}_{\text{2PC}}(\mathbf{z}, \mathbf{A}, \lambda)$ (Eq. (5)) and learning rate α
end for

the $\mathcal{L}_{\text{2PC}}$ loss function is defined as:

$$\mathcal{L}_{\text{2PC}}(\mathbf{z}, \mathbf{A}, \lambda) \;=\; -\underbrace{p(\mathbf{z}, \mathbf{A})}_{\text{polarity}} \;+\; \lambda \underbrace{||\rho||_2^2}_{\text{regularization}} \tag{5}$$

Rounding. To round a continuous solution $\mathbf{z} \in [-1, 1]^{|V|}$ onto a valid discrete $\mathbf{x} \in \{-1, 0, 1\}^{|V|}$ solution to 2PC, we borrow the procedure adopted by Bonchi et al. [8]. Specifically, given a threshold $\tau \in [0, 1]$, for all $u \in V$, $\mathbf{x}[u] = \text{sgn}(\mathbf{z}[u])$ if $|z[u]| \geq \tau$, $\mathbf{x}[u] = 0$ otherwise. In order to avoid sticking to a single τ, we follow [8], and try all the thresholds $\tau \in \{\lceil \mathbf{z}[u] \rfloor_i \mid u \in V\}$, where $\lceil \cdot \rfloor_i$ denotes approximating a real number at the i-th decimal digit (we use $i = 3$). Formally:

$$Z_i = \{\lceil \mathbf{z}[u] \rfloor_i \mid u \in V\}.$$

$$\forall u \in V: \; \mathbf{x}_\tau[u] = \begin{cases} \text{sgn}(\mathbf{z}[u]), \text{ if } |\mathbf{z}[u]| \geq \tau. \\ 0, \text{ otherwise.} \end{cases}$$

$$\text{ROUND}(\mathbf{z}) \;=\; \underset{\mathbf{x} \in \{\mathbf{x}_\tau \mid \tau \in Z_i\}}{\arg\max} \; p(\mathbf{x}, \mathbf{A}). \tag{6}$$

Algorithm. The algorithm we employ to produce a solution to 2PC simply consists in optimizing the $\theta = \{\theta_{\text{SGNN}}, \theta_{\text{NN}}\}$ parameters of the f_θ neural model end-to-end, via standard gradient descent, for a number e_{max} of training epochs. Specifically, the algorithm alternates a forward phase, which produces a continuous solution $\mathbf{z}$ given the current θ parameters, and a backward phase, where parameters θ are updated via gradient descent, using the $\mathcal{L}_{\text{2PC}}$ loss function, with a certain learning rate α. The continuous solution $\mathbf{z}$ yielded in every epoch is rounded according to the ROUND$(\cdot)$ procedure described above. The discrete rounded solution with the highest polarity score out of all the ones produced in the various epochs is ultimately output. The reason of performing rounding and evaluating the polarity of the discrete solution in

every epoch is that it is hard to know in advance the exact epoch leading to the best discrete solution (see Section 6).

The proposed algorithm is outlined as Algorithm 1.

Motivating example. The following example shows the relevance of considering suboptimal solutions to 2PC-RELAXED, and validates the main motivation of our neural approach.

Example 1. *Consider again the toy signed graph in Figure 1. For the sake of vector notation, let the order over node set* $\{\texttt{A},\texttt{B},\texttt{C},\texttt{D},\texttt{E}\}$ *correspond to the lexicographic node order, i.e.,* `A` *corresponds to entry 1 in the vectors,* `B` *corresponds to entry 2, and so on.* $\mathbf{z}^* = [0.282, -0.282, -0.282, -0.616, -0.616]$ *is the optimal solution to* 2PC-RELAXED *on that example graph, and* $\mathbf{z} = [0.213, -0.144, -0.144, -0.378, -0.378]$ *is a suboptimal solution (yielded by our neural approach). The polarity (Definition 1) of* $\mathbf{z}^*$ *and* $\mathbf{z}$ *is 2.372 and 2.363, respectively. Adopting the aforementioned rounding (Equation (6)),* $\mathbf{z}^*$ *leads to a discrete solution* $\mathbf{x}_1 = [1, -1, -1, -1, -1]$ *(with threshold* $\tau = 0.282$*). Instead,* $\mathbf{z}$ *yields* $\mathbf{x}_2 = [1, 0, 0, -1, -1]$ *(with* $\tau = 0.213$*). The polarity of* $\mathbf{x}_2$ *is 2, which is higher than the polarity 1.6 of* $\mathbf{x}_1$*. Note also that* $\mathbf{x}_2$ *corresponds to the optimal solution to* 2PC *on the example graph at hand.*

Computational complexity. We discuss computational complexity aspects of our proposed approach.

The temporal cost associated to the SGNN block depends on the particular neural-network architecture. Nonetheless, assuming a sparse input signed graph with edge set $E = E^+ \cup E^-$, the overall time complexity of a SGNN model with m layers can be regarded as bounded by $\mathcal{O}(m(|V|d^2 + |E|d))$, where $d = max(d_I, d_H)$ (e.g., 64), which reduces to $\mathcal{O}(|V|d^2 + |E|d)$, since m typically corresponds to few units. The role of the NN module is to transform the vector representations generated by the SGNN module into node-to-community assignment scores. This transformation can straightforwardly be achieved through a simple multi-layer perceptron neural network, whose computational cost is dominated by that of the SGNN module. The time complexity of the rounding module is $\mathcal{O}(|Z_i|(|V| + |E|))$, as for each potential threshold in Z_i, the polarity of the rounded solution is computed in $\mathcal{O}(|V| + |E|)$ time to select the optimal discrete solution. It is worth emphasizing that the cost of the rounding module constitutes the sole overhead introduced by us compared to the cost of existing SGNN and NN modules embedded into our approach. By combining the two aforementioned costs for e_{max} epochs, the overall time complexity of Neural2PC is $\mathcal{O}(e_{max}((|Z_i| + d^2)|V| + (|Z_i| + d)|E|))$.

Regarding the space complexity of our method, Neural2PC requires $\mathcal{O}(|V|)$ space to store intermediate variables $\mathbf{x}$ and $\mathbf{z}$ (lines 3-4 in Algorithm 1). This cost is to be considered as negligible compared to the space required to store intermediate node representations for all nodes, which is $\mathcal{O}(|V|d_H)$ since we compute and store a vector of d_H components (cf. Equation (2)) for each node $v \in V$. Moreover, the space complexity of our method also depends on the space complexity of the specific SGNN and NN modules, which in turn depends on the particular adopted models for such modules. However, the number of model parameters associated with SGNN and NN is known to be not directly affected by the size of the input graph, and for large graphs and typical network settings, the dominant term indeed corresponds to the cost of storing

the node representations, i.e., $\mathcal{O}(|V|d_H)$. Consequently, the overall space complexity of Neural2PC can be reasonably assumed to be $\mathcal{O}(|V|d_H)$, attributed to the space required to store the node representations.

4 Balancing the size of the communities

A well-known issue of the polarity measure (Definition 1) is that it favors solutions with size-imbalanced output communities. It is not unlikely that this may even degenerate to solutions with one of the two communities overwhelming any other polarized formation, with the result of having the second community empty [8]. Motivated by this, here we devise a generalization of the polarity measure, dubbed γ-*polarity*, which, by properly playing with its parameter γ, yields polarized communities more balanced in size.

We define γ-polarity by properly modifying the denominator of the polarity measure, while keeping the numerator the same. Given a node-to-community assignment vector $\mathbf{x} \in \{-1, 0, 1\}^{|V|}$, let $s_1 = \sum_{u \in V, \mathbf{x}[u]<0} |\mathbf{x}[u]|$ and $s_2 = \sum_{u \in V, \mathbf{x}[u]>0} \mathbf{x}[u]$ be the size of the two communities, with $s_{max} = \max\{s_1, s_2\}$, $s_{min} = \min\{s_1, s_2\}$. The denominator of the polarity measure is equal to the sum of the sizes of the two communities, i.e., to $\mathbf{x}^\top\mathbf{x} = s_{max} + s_{min}$. Noticing that $\mathbf{x}^\top\mathbf{x} = (s_{max} - s_{min}) + 2\,s_{min}$, the main intuition behind γ-polarity is to break $\mathbf{x}^\top\mathbf{x}$ into such two terms $(s_{max} - s_{min})$ and $2\,s_{min}$, and weigh differently – i.e., by $\gamma > 0$ – the $(s_{max} - s_{min})$ term corresponding to the difference in size between the two communities. This leads to the following formal definition of γ-polarity:

Definition 2 (γ-polarity)**.** *Given a vector* $\mathbf{x} \in \{-1, 0, 1\}^{|V|}$*, a matrix* $\mathbf{A} \in \{-1, 0, 1\}^{|V|\times|V|}$*, and a real number* $\gamma > 0$*, the* γ*-polarity* $p_\gamma(\mathbf{x}, A)$ *of* $\mathbf{x}$ *with respect to* $\mathbf{A}$ *is defined as:*

$$p_\gamma(\mathbf{x}, \mathbf{A}) = \frac{\mathbf{x}^\top\mathbf{A}\ \mathbf{x}}{(s_{max} - s_{min})\,\gamma + 2\,s_{min}}. \tag{7}$$

Note that, if $\gamma > 1$, the size-difference $(s_{max} - s_{min})$ term is amplified: thus, maximizing p_γ enforces such a term to be small, which corresponds to favoring size balance among communities. The opposite happens if $\gamma \in (0, 1)$. Instead, $\gamma = 1$ makes γ-polarity boil down to standard polarity.

The relaxed counterpart of γ-polarity, for a given continuous vector $\mathbf{z} \in [-1, 1]^{|V|}$ is defined by simply replacing $\mathbf{x}$ with $\mathbf{z}$ in Equation (7) (including in the computation of s_{max} and s_{min}). Relaxed γ-polarity can be incorporated in the proposed Neural2PC approach by simply replacing the polarity term $p(\mathbf{z}, \mathbf{A})$ with a relaxed γ-polarity term $p_\gamma(\mathbf{z}, \mathbf{A})$ in the $\mathcal{L}_{\text{2PC}}$ loss function (Equation (5)).

Example 2. *Consider the example graph in Figure 3, where positive edges are depicted by solid lines, while negative edges correspond to red dashed lines. Let* $P_1 = \{\{\texttt{A,B,C,D}\}, \{\texttt{E,F,G,H}\}\}$*,* $P_2 = \{\{\texttt{A,B,C,D}\}, \{\texttt{E,F,G,H,I,J,K,L}\}\}$*, and* $P_3 = \{\emptyset, \{\texttt{E,F,G,H,I,J,K,L}\}\}$ *be three possible pairs of polarized communities.*

The basic polarity (Definition 1) of P_1*,* P_2*, and* P_3 *is* $(15 \times 2)/8 = 3.75$*,* $(22 \times 2)/12 = 3.67$*, and* $(15 \times 2)/8 = 3.75$*, respectively. Despite,* P_1 *and* P_3 *both exhibit the highest polarity,* P_1 *is much more size-balanced, thus intuitively preferable.*

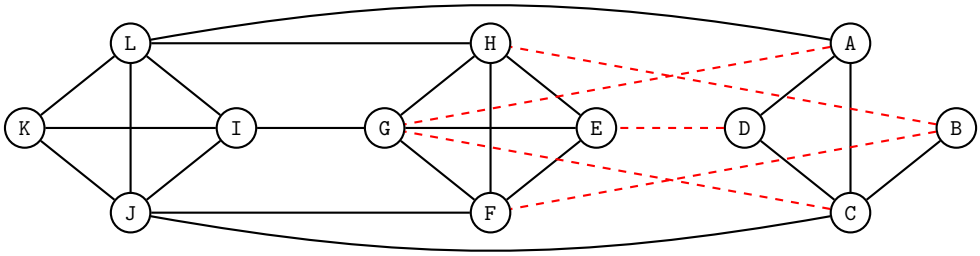


Fig. 3 Example signed graph where, unlike basic polarity, γ-polarity detects effective polarized communities that are also size-balanced (Example 2).

It is easy to see that using γ-polarity, with any $\gamma > 1$, instead of basic polarity allows for recognizing P_1 as better than P_3 (as P_1 and P_3 have the same γ-polarity numerator, while the γ-polarity denominator for P_3 would be higher than the one for P_1 for any $\gamma > 1$).

Detecting fine-grained polarization phenomena within signed networks is crucial, being essential to recognize minorities in polarization which might correspond to harmful situations like isolation, as illustrated in the following running example.

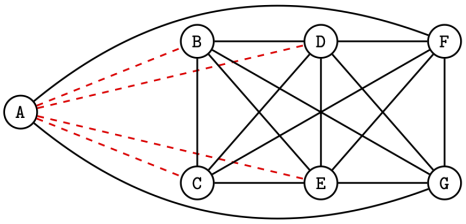
Example 3. *Consider the example graph in Figure 4-(a), where positive edges are depicted by solid lines, and negative edges correspond to dashed lines. The scenario illustrated in the figure might be regarded as a case of isolation, wherein one node (`A`) has a majority of conflicting relations with a clique (i.e., `B, C, D, E`), and only few links with a lower number of nodes (i.e., `F,G`). Therefore, it is desirable to capture a polarization solution wherein the clique is regarded as one group while node `A` forms the other yet isolated group.*

Existing methods fail to identify such a desired solution: indeed, given the inherent bias of the polarity objective towards outputs favoring communities with imbalanced sizes, both Eigen *and* R-Eigen *yield $\{S_1, \emptyset\} = \{\{$`B, C, D, E, F, G`$\}, \emptyset\}$. This also corresponds to an intermediate solution provided by our proposed method equipped with the $\gamma-$polarity objective (e.g., $\gamma = 2.1$) (cf. Figure 4-(b)), before finally producing the desired output $\{S_1, S_2\} = \{\{$`B, C, D, E`$\}, \{$`A`$\}\}$ at convergence (cf. Figure 4-(c)).*

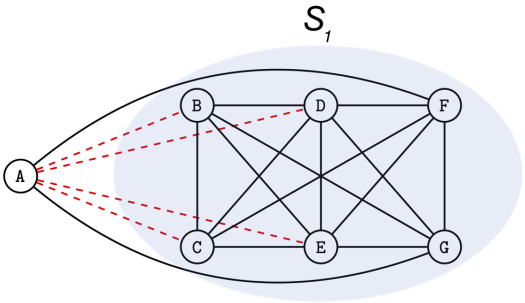
5 Experimental Methodology

Evaluation goals. We assessed accuracy of the proposed Neural2PC (Algorithm 1) and competitors/baselines on (1) real datasets, and (2) synthetic datasets; (3) impact of different signed GNNs when used as a module of Neural2PC; (4) runtimes of the considered methods; (5) effectiveness of the individual components of Neural2PC through an ablation study; (6) effectiveness of the γ-polarity measure in yielding communities that are both size-balanced and high-quality.

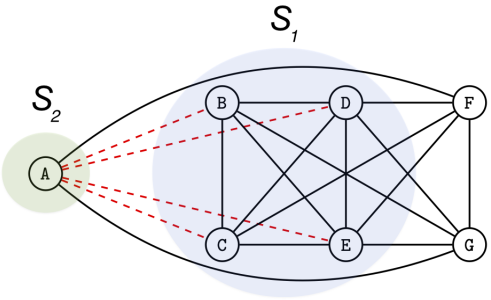
Real datasets. We selected publicly-available real-world signed graphs, whose characteristics are summarized in Table 2. *Bitcoin* and *Epinions* are the trust-distrust network of users of the Bitcoin OTC trading platform and the Epinions social network, respectively. *Cloister* contains the esteem/disesteem relations of monks living



(a) Input signed graph



(b) Solution yielded by Neural2PC after the 1st training epoch



(c) Solution yielded by Neural2PC at convergence (11-th training epoch)

Fig. 4 Running example illustrating a scenario of node-isolation detection (Example 3).

in a cloister in New England (USA). *Congress* reports (un/)favorable mentions of politicians speaking in the US Congress. *HTribes* describes the alliances/enemies relationships of a tribe in New Guinea. *Slashdot* is a friend-foe network collected from the Slashdot technology news site. *TwitterRef* collects the tweets about the Italian constitutional referendum in 2016, and edge signs express whether two users have the same stance or not. *WikiCon* contains positive/negative edit conflicts between the users of the English Wikipedia. *WikiEle* collects the positive/negative votes for electing admins in the English Wikipedia. *WikiPol* describes the signed interactions of users who have edited the English Wikipedia pages about politics.

Table 2 Main characteristics of real data used in our evaluation. $E = E^+ \cup E^-$: overall edge set; density: $|E|/(|V|(|V|-1)/2)$; $\deg^+$, $\deg^-$: avg of positive and negative node degrees, resp.; cc^+, cc^-: #connected components in the subgraph induced by E^+ and E^-, resp.; cc: overall #connected components.

dataset	$\|V\|$	$\|E\|$	$\|E^-\|/\|E\|$	density	$\deg^+$	$\deg^-$	cc^+	cc^-	cc
Bitcoin [39]	5 881	21 492	0.152	0.00124	6.2	1.11	355	4 344	4
Cloister [40]	18	125	0.552	0.81699	6.22	7.67	1	1	1
Congress [40]	219	521	0.205	0.02183	3.78	0.98	8	127	1
Epinions [39]	131 580	711 210	0.171	8e-05	8.97	1.84	23 366	90 846	5 568
HTribes [40]	16	58	0.5	0.48333	3.62	3.62	2	2	1
Slashdot [39]	82 140	500 481	0.239	0.00015	9.28	2.91	7427	46 991	1
TwitterRef [41]	10 884	251 406	0.051	0.00424	43.85	2.35	69	6 801	11
WikiCon [39]	116 717	2 026 646	0.628	0.0003	12.9	21.83	70 284	1 788	1 785
WikiEle [40]	7 115	100 693	0.221	0.00398	22.05	6.26	892	2 926	24
WikiPol [41]	138 587	715 883	0.123	7e-05	9.06	1.27	14 458	97 459	305

Table 3 Main characteristics of M-SSBM-generated network data used in our evaluation. $E = E^+ \cup E^-$: overall edge set; density: $|E|/(|V|(|V|-1)/2)$; $\deg^+$, $\deg^-$: avg of positive and negative node degrees, respectively.

dataset	$\|V\|$	$\|E\|$	$\|E^-\|/\|E\|$	density	$\deg^+$	$\deg^-$
syn-η=0.0-n=250-n_c=25	250	1225	0.51	0.03936	4.8 ± 9.6	5.0 ± 10.0
syn-η=0.1-n=250-n_c=25	250	11503	0.502	0.36957	45.82 ± 8.18	46.21 ± 8.56
syn-η=0.2-n=250-n_c=25	250	17803	0.498	0.57198	71.49 ± 8.12	70.94 ± 7.9
syn-η=0.3-n=250-n_c=25	250	20688	0.5	0.66467	82.75 ± 7.73	82.75 ± 7.23
syn-η=0.4-n=250-n_c=25	250	20151	0.505	0.64742	79.73 ± 7.32	81.48 ± 7.55
syn-η=0.5-n=250-n_c=25	250	15994	0.494	0.51386	64.73 ± 7.5	63.22 ± 7.04
syn-η=0.6-n=250-n_c=25	250	15801	0.502	0.50766	62.91 ± 6.96	63.5 ± 7.14
syn-η=0.0-n=500-n_c=50	500	4950	0.505	0.03968	9.8 ± 19.6	10.0 ± 20.0
syn-η=0.1-n=500-n_c=50	500	45557	0.498	0.36519	91.49 ± 14.51	90.74 ± 15.15
syn-η=0.2-n=500-n_c=50	500	71688	0.5	0.57465	143.34 ± 11.77	143.42 ± 11.5
syn-η=0.3-n=500-n_c=50	500	83360	0.5	0.66822	166.68 ± 11.18	166.76 ± 11.1
syn-η=0.4-n=500-n_c=50	500	80755	0.502	0.64733	160.94 ± 10.99	162.08 ± 11.16
syn-η=0.5-n=500-n_c=50	500	63682	0.498	0.51048	127.95 ± 10.57	126.78 ± 11.05
syn-η=0.6-n=500-n_c=50	500	63194	0.503	0.50657	125.55 ± 10.03	127.22 ± 10.79
syn-η=0.0-n=1K-n_c=100	1000	19900	0.503	0.03984	19.8 ± 39.6	20.0 ± 40.0
syn-η=0.1-n=1K-n_c=100	1000	182117	0.501	0.3646	181.66 ± 26.47	182.58 ± 26.93
syn-η=0.2-n=1K-n_c=100	1000	286637	0.5	0.57385	286.65 ± 19.19	286.62 ± 19.47
syn-η=0.3-n=1K-n_c=100	1000	332555	0.499	0.66578	333.18 ± 16.37	331.93 ± 16.94
syn-η=0.4-n=1K-n_c=100	1000	322424	0.501	0.64549	321.8 ± 15.9	323.05 ± 15.98
syn-η=0.5-n=1K-n_c=100	1000	254789	0.5	0.51009	254.99 ± 17.44	254.59 ± 16.41
syn-η=0.6-n=1K-n_c=100	1000	253394	0.499	0.5073	253.65 ± 16.05	253.14 ± 15.8
syn-η=0.0-n=2K-n_c=200	2000	79800	0.501	0.03992	39.8 ± 79.6	40.0 ± 80.0
syn-η=0.1-n=2K-n_c=200	2000	728584	0.5	0.36447	364.39 ± 51.16	364.19 ± 51.88
syn-η=0.2-n=2K-n_c=200	2000	1147517	0.5	0.57405	573.67 ± 33.78	573.85 ± 34.57
syn-η=0.3-n=2K-n_c=200	2000	1334724	0.5	0.6677	667.41 ± 25.78	667.31 ± 25.61
syn-η=0.4-n=2K-n_c=200	2000	1291962	0.5	0.6463	646.21 ± 24.06	645.75 ± 24.32
syn-η=0.5-n=2K-n_c=200	2000	1019626	0.499	0.51007	510.51 ± 27.66	509.12 ± 28.59
syn-η=0.6-n=2K-n_c=200	2000	1014422	0.5	0.50746	507.56 ± 24.93	506.87 ± 25.52

Synthetic datasets. We also employed synthetic signed graphs in order to test the methods in recovering ground-truth polarized communities. We used *modified signed stochastic block model* (M-SSBM) [9] as a generator. This model has three parameters, namely the total number n of nodes, the size $n_c = |S_1| = |S_2|$ of a planted polarized community (all have the same size) and a parameter $\eta \in [0, 1]$ to control edge probabilities: (i) an edge in the same group (resp. between two polarized groups) is drawn as positive (resp. as negative), with probability $1 - \eta$, as negative (resp. as positive), with probability $\eta/2$, and is not drawn with probability $\eta/2$; (ii) all other edges have equal probability of $\min(\eta, 1/2)$ of being positive or negative. Note that the smaller η, the lower the noise level. The case with no noise ($\eta = 0$) corresponds to the "perfect" structure (i.e., all nodes are disconnected except those linked within or across polarized communities), while the polarized communities only emerge when $\eta \leq 2/3$, since for $\eta > 2/3$ the generated graph has more negative edges in the groups and more positive edges between the groups.

We considered different synthetic graphs by varying number of nodes (n), community size n_c, and $\eta \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6\}$. For each configuration, we generated 10 different graphs. Table 3 summarizes main characteristics (average statistics) of the generated network data.

Competing methods. We compare our Neural2PC (Algorithm 1) to the state-of-the-art methods for discovering polarized communities, as well as against non-trivial baselines inspired by methods devised for different yet related problems.

We consider both the methods originally conceived by Bonchi et al. [8], namely EIGEN and its randomized R-EIGEN counterpart, as our direct competing methods, since they target the same optimization problem (2PC) we tackle in this work.

Like [8], we include PIVOT too, a baseline inspired by a correlation clustering algorithm [42]. For each node $u \in V$, PIVOT identifies u and the nodes sharing a positive edge with u as one cluster, and the nodes sharing a negative edge as the other cluster. From the $|V|$ possible solutions, it returns the one maximizing polarity. We also consider GREEDY [8], a method inspired by a 2-approximation algorithm for densest subgraph [43]. It iteratively removes nodes to maximize the difference between positive and negative adjacent edges until the graph is empty. At the end, it returns the subgraph having the highest polarity among all produced subgraphs. Node-to-cluster assignment is guided by the sign of the components of the eigenvector corresponding to the largest eigenvalue of $\mathbf{A}$.

Furthermore, we consider signed graph clustering algorithms BNC [24], SPONGE [25] and SSSNET [26]. Since they all require the number k of output clusters (communities), we denote them with BNC(K), SPONGE(K) and SSSNET(K), respectively. As previously done in [9], we consider two variants of these competitors. The first one consists in setting $k = 2$ and return the 2 detected clusters as polarized communities. The second variant sets $k = 3$, and it treats the largest of the 3 detected clusters as the group of neutral nodes, and return the 2 smallest clusters as polarized communities.

Experimental setting. We instantiate the SGNN($\cdot$) block of our Neural2PC framework with well-established signed GNNs, namely SGCN [17], SNEA [19], SGDNET [18]. As for SGCN, we consider different variants by varying the neighbor

aggregation operator in $\{\text{mean}, \text{sum}, \text{attention}\}$. We denote such variants $\text{SGCN}_{\text{mean}}$, SGCN_{sum} and SGCN_{att}, respectively. Also, for any considered signed GNN, we add the "-DR" suffix if the proposed loss regularization term is used (Equation (5)). The absence of such a suffix means that λ was set to 0.

All signed GNN models, implemented using PyTorch Geometric and trained on CPUs, share uniform settings with a node representation dimensionality d_H of 64, number of layers m as 2, and default values for other parameters. Concerning SGDNET, we set the number of latent groups to 3. Since no initial node-features are available for the selected signed graphs, following previous works [17] we used the final embedding of a signed spectral embedding model [27] as the input feature matrix, with $d_I = 64$. For each configuration of our method and SSSNET, model training was carried out by the Adam optimizer, for $e_{max} = 300$ epochs and by grid searching the α learning rate from $\{0.01, 0.005, 0.001\}$ and the λ regularization factor from $\{0.1, 0.01, 0.001\}$. All reported measurements correspond to averages over 30 runs. Details about the execution environment can be found in the ***Appendix***.

6 Results

Neural2PC vs. competitors, real datasets. Table 4 reports the values of polarity (Definition 1) achieved by all compared algorithms on all datasets, and the size of the produced two communities. Concerning our Neural2PC, we only report the results obtained by the best-performing (in terms of polarity) graph representation learning method. As for the graph clustering methods (i.e., SPONGE(K), BNC(K) and SSSNET(K)), we also report the number of desired communities $k \in \{2, 3\}$ which led to the best results in terms of polarity.

As a first remark, our Neural2PC generally reveals to be the most competitive method in terms of polarity. Note that the exceptions of Slashdot, WikiEle and WikiPol datasets (where GREEDY precedes Neural2PC) correspond, however, to a very dense subgraph returned by GREEDY as one of the two polarized communities, leaving the second community totally empty, which is clearly undesired in practice. Conversely, our method is able to return both non-empty communities with high polarity in WikiEle and WikiPol.

Among our competitors, EIGEN and R-EIGEN, who also address the 2PC problem, both achieve strong polarity results. EIGEN outperforms R-EIGEN, and on the smallest datasets, Congress and HTribes, our method matches EIGEN's solutions regardless of the adopted GNN model. In general, however, our method performs better than EIGEN and R-EIGEN.

PIVOT overall performs poorly in terms of polarity. This can be explained since, in identifying polarized groups by exploring local neighborhoods, its search space strongly depends on the neighborhood structure of the nodes. Also, its detected communities tend to be located around high-degree nodes but, in general, the pair of communities with the highest polarity do not necessarily lie around the high-degree nodes.

Concerning the BNC and SPONGE methods, when $k = 2$, as expected, they perform poorly since all neutral nodes are put in one of the two detected communities, resulting in solutions with low polarity. In fact, the two methods often yield solutions

Table 4 Polarity (Def. 1) and solution size ($|S_1|;|S_2|$) of the proposed Neural2PC method vs. competing methods on real datasets. Best results in bold, second-best underlined.

method	criteria	*Bitcoin*	*Cloister*	*Congress*	*Epinions*	*HTribes*	*Slashdot*	*TwitterRef*	*WikiCon*	*WikiEle*	*WikiPol*
EIGEN	pol.	<u>29.52</u>	**7.45**	<u>6.58</u>	128.72	**6.18**	79.7	<u>174.1</u>	<u>175.65</u>	71.73	88.44
	size	136;2	8;3	28;24	999;18	7;4	233;1	669;4	1993;449	745;3	646;2
R-EIGEN	pol.	14.12	6.23	5.38	71.36	5.82	29.21	118.81	99.93	55.91	35.72
	size	725;103	14;3	50;50	4057;407	7;4	2827;125	1487;20	9778;3109	1054;67	6838;579
PIVOT	pol.	21.65	4.17	3.1	156.38	3.5	61.0	116.25	129.33	37.59	46.52
	size	21;19	9;3	6;5	248;5	5;3	283;6	1142;16	368;134	407;7	598;11
GREEDY	pol.	29.01	6.11	5.77	<u>170.3</u>	5.5	**82.72**	173.94	127.96	**72.67**	**90.02**
	size	140;0	15;3	36;33	269;0	12;4	200;0	685;0	1151;0	730;0	543;0
SPONGE(K)	pol.	8.36	6.11	4.43	7.12	5.5	6.36	28.03	-8.92	15.79	7.79
	size	26;2	15;3	115;104	131578;2	12;4	82138;2	8274;2610	116712;5	7113;2	138585;2
	k	3	2	2	2	2	2	2	3	2	2
BNC(K)	pol.	5.27	1.0	2.75	7.12	-0.5	6.36	41.49	8.92	15.79	7.79
	size	5834;47	17;1	216;3	131225;355	10;6	82138;2	10882;2	115730;987	7102;13	138556;31
	k	2	2	2	2	2	2	2	2	2	2
SSSNET(K)	pol.	9.06	6.93	4.43	73.19	5.0	7.26	41.49	28.56	17.09	7.82
	size	713;8	6;3	115;104	953;0	11;5	72915;9225	10864;20	50835;5401	6402;713	132566;6021
	k	3	3	2	3	2	2	2	3	2	2
Neural2PC	pol.	**30.28**	**7.45**	**6.64**	**171.1**	**6.18**	<u>82.25</u>	**174.35**	**187.29**	<u>72.17</u>	<u>88.89</u>
	size	158;32	8;3	29;24	268;1	7;4	203;0	677;4	1788;559	742;2	618;2
	model	SGCN_{sum}	All	SGDNET	$\text{SGCN}_{\text{mean}}$	All	$\text{SGCN-DR}_{\text{mean}}$	SGCN_{sum}	SGCN_{sum}	$\text{SGCN-DR}_{\text{sum}}$	SGCN_{sum}

Table 5 Edge-agreement ratio of the proposed Neural2PC method vs. competing methods on real datasets.

dataset	EIGEN	R-EIGEN	PIVOT	GREEDY	SPONGE	BNC	SSSNET	Neural2PC
Bitcoin	0.95	0.95	0.99	0.96	1.0	0.86	0.98	0.95
Cloister	0.94	0.74	0.72	0.72	0.72	0.54	0.97	0.94
Congress	0.98	0.98	1.0	0.96	0.96	0.79	0.96	0.98
Epinions	0.95	0.94	1.0	1.0	0.83	0.83	0.99	1.0
HTribes	1.0	1.0	1.0	0.88	0.88	0.47	0.84	1.0
Slashdot	0.99	0.93	0.98	0.99	0.76	0.76	0.8	0.99
TwitterRef	0.99	0.99	0.99	1.0	1.0	0.8	0.95	0.99
WikiCon	0.95	0.94	0.94	0.966	1.0	0.37	0.84	0.95
WikiEle	0.93	0.91	0.88	0.93	0.78	0.78	0.8	0.92
WikiPol	0.96	0.95	0.97	0.97	0.88	0.88	0.89	0.96
avg	0.96	0.93	0.95	0.94	0.88	0.71	0.90	**0.97**

Table 6 Performance of the proposed Neural2PC vs. competing methods on synthetic datasets, in terms of F_1-score and polarity (Def. 1) as a function of the noise parameter η.

method	criteria	η = 0	0.1	0.2	0.3	0.4	0.5	0.6
EIGEN	F_1	**1.0**	.998	.998	.998	**.995**	.972	.307
	pol.	**199**	168.04	140.31	110.5	**81.44**	50.02	35.52
R-EIGEN	F_1	**1.0**	.911	.861	.829	.755	.678	.309
	pol.	**199**	144.23	112.96	87.55	61.98	39.4	30.67
PIVOT	F_1	.997	.584	.426	.338	.305	.28	.236
	pol.	198	61.66	32.56	17.06	8.5	4.06	3.1
GREEDY	F_1	.667	.644	.621	.63	.605	.334	.264
	pol.	99	79.79	62.64	50.76	38.77	30.31	28.86
SPONGE(K)	F_1	**1.0**	**1.0**	.714	.714	.714	.551	.247
	pol.	**199**	**168.62**	28.01	23.96	15.81	23.36	23.85
	k	3	3	2	2	2	3	3
BNC(K)	F_1	.5	.5	.5	.167	.167	.476	**.374**
	pol.	-0.2	-0.26	-0.75	1.86	-0.98	1.01	1.09
	k	2	2	2	2	2	3	2
SSSNET(K)	F_1	**1.0**	**1.0**	.99	.99	.976	.365	.267
	pol.	**199**	**168.62**	140.2	109.97	74.99	34.83	26.89
	k	3	3	3	3	3	2	3
Neural2PC	F_1	**1.0**	**1.0**	**1.0**	**1.0**	**.995**	**.997**	.341
	pol.	**199**	**168.62**	**140.65**	**110.69**	**81.44**	**50.27**	**36.16**

where one community is a large group likely including all neutral nodes, and the other community consists of few nodes. Conversely, when $k = 3$, even if the methods can use the spare cluster to put the neutral nodes, they both perform worse than the case $k = 2$ yielding solutions that are of extremely small size. The same consideration holds for the SSSNET method, although to a less extent, as it performs better then SPONGE and BNC. In any case, the three selected clustering methods are consistently outperformed by our method.

We complement our evaluation by analyzing the *edge-agreement ratio* which measures the portion of edges in the solution that comply with the polarized structure, i.e., the number of intra-community positive edges inside S_1 and S_2 plus the number of inter-community negative edges, divided by all the edges in the subgraph induced by

Table 7 Performance of the proposed Neural2PC on synthetic datasets with varying number of nodes n and community sizes n_c, in terms of F_1-score, polarity (Def. 1) and running time (in seconds) as a function of the noise parameter η.

n	n_c	criteria	η 0	0.1	0.2	0.3	0.4	0.5	0.6
250	25	F_1	1.0	1.0	1.0	1.0	0.658	0.361	0.196
		pol.	49.0	41.52	36.28	29.882	22.365	18.688	18.022
		time	10.7	35.3	52.6	49.6	64.6	67.5	69.9
500	50	F_1	1.0	1.0	1.0	1.0	1.0	0.723	0.329
		pol.	99.0	84.02	69.54	53.92	39.725	25.775	24.641
		time	26.4	84.4	92.7	114.6	144.4	154.3	176.3
1000	100	F_1	1.0	1.0	1.0	1.0	.995	.997	.341
		pol.	199	168.62	140.65	110.69	81.44	50.27	36.16
		time	58.6	212.6	285.9	351.6	400.0	407.8	557.8
2000	200	F_1	1.0	1.0	1.0	1.0	1.0	1.0	0.889
		pol.	399.0	337.845	278.76	220.485	160.105	98.566	39.589
		time	123.7	555.7	621.8	686.5	783.6	835.4	963.6

$S_1 \cup S_2$. Table 5 reports the edge-agreement ratio corresponding to the best-polarity solutions (cf. Table 4). Overall, the edge-agreement ratio is consistently close or equal to 1, especially for Neural2PC and EIGEN, with an average edge-agreement ratio across all datasets of 0.97 and 0.96, respectively. The remaining methods follow a similar, although weaker, trend: this means that the solutions have a coherent polarized structure.

Neural2PC vs. competitors, synthetic datasets. Table 6 shows the F_1-scores and corresponding polarity scores averaged over the 10 synthetic graphs that were generated for each configuration, by varying the noise parameter η while keeping fixed the network size ($n = 1000$) and the number of communities ($n_c = 100$). For our method, we only report the results corresponding to the best-performing (in terms of F_1-score) signed GNN model, which was SGDNET for all the datasets and noise levels.

From the table, it is evident that our Neural2PC is very robust in handling an increasing noise level, as it mostly outperforms all competitors both in terms of F_1-score and polarity. The only exception arises for $\eta = 0.6$, where BNC(2) yields a slightly higher F_1-score than Neural2PC, but the polarity of the solution yielded by BNC(2) is 36× times lower, thus resulting in a low-quality solution – recall that the M-SSBM model generates very noisy graphs for mid-high values of η, where the ground-truth communities are "hidden" by the graph topology since there are many intra-community negative edges and many inter-community positive edges. In such cases, a higher polarity is a more reliable indicator for the quality of the discovered solutions: indeed, it holds that the polarity score is consistent with the F_1-score, i.e., high values for the polarity mostly correspond to high F_1-scores.

In addition, as shown in Table 7, our Neural2PC keeps a similar behavior and is still robust w.r.t. η for varying numbers of nodes and community sizes (cf. Table 3); again, results correspond to averages over 10 synthetic networks, and the group of results in the second last row coincide with those for Neural2PC reported in Table 6. Note that, according to the definition of M-SSBM and its edge-formation rules based on η, it should not come to our surprise how the impact of η in terms of polarity decrease-rate

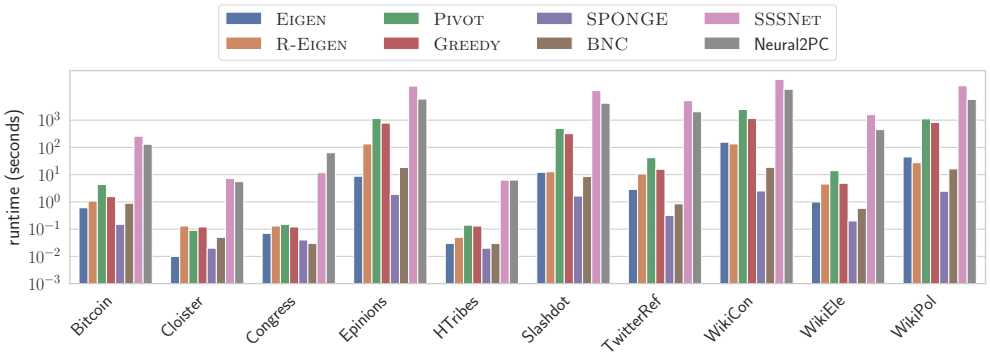


Fig. 5 Execution times (in seconds) of the proposed Neural2PC method vs. competing methods on real-world network datasets.

tends to be higher on larger M-SSBM synthetic networks, given the increased amount of noise introduced by η.

Impact of different signed GNNs on Neural2PC. We analyzed polarity and community size values corresponding to all the variants of our Neural2PC method based on different signed GNN models (results are shown in the ***Appendix***).

Our experiments revealed that the polarity of the solutions provided by Neural2PC does not significantly change across the various GNNs, which indicates robustness of our approach in terms of one of its main components.

Execution times. Figure 5 shows the average time performance of the methods, over the various runs; in the ***Appendix***, Table B7 reports detailed information, particularly for Neural2PC we show details about the total running time and, in brackets, the time discarding the rounding steps (i.e., by discarding the cumulative time spent in executing the step at Line 4 of Algorithm 1 over all the epochs).

As expected, the learning-based methods, i.e., SSSNET and our Neural2PC, exhibit the highest running times, which is clearly affected by their number of epochs ($e_{max} = 300$). Importantly, we remark that the reported running times of the Neural2PC method refer to its execution on a single CPU, without any parallelization. However, Neural2PC, like any neural-network method, can easily benefit from dedicated hardware (e.g., GPUs) and parallelization, which can drastically improve its execution time. Also, note that the time per epoch of Neural2PC is comparable to the runtime of the fastest method(s).

Among the other methods, SPONGE performs the best, followed by BNC and EIGEN. R-EIGEN is slightly worse than EIGEN due to its randomized nature involving sampling. Among the non-learning-based methods, PIVOT and GREEDY are inefficient since they extract and evaluate $|V|$ solutions.

In addition, remarks about scalability of Neural2PC can be drawn by looking at the results on synthetic networks shown in Table 7: for any given value of η, the runtime of Neural2PC increases linearly with the size of the network, which is in accord with our previously discussed analysis of computation aspects.

Table 8 Ablation study results for the proposed Neural2PC method.

dataset	DIRECT			NN			Neural2PC		
	pol.	size	time	pol.	size	time	pol.	size	time
Bitcoin	29.79	152;10	391	29.68	141;1	43.4	**30.28**	158;32	130
Cloister	7.43	9;3	2.3	**7.45**	8;3	5	**7.45**	8;3	5.5
Congress	6.59	29;24	30	6.62	30;23	25	**6.64**	29;24	64
Epinions	139.43	715;223	135	167.92	266;1	1052	**171.1**	268;1	5956
HTribes	**6.18**	7;4	3.2	**6.18**	7;4	5.1	**6.18**	7;4	6.3
Slashdot	79.88	224;1	7232	81.48	203;0	590	**82.25**	203;0	4178
Twit.Ref	**174.43**	675;4	3544	173.39	684;4	232	174.35	677;4	2033
WikiCon	185.48	1937;551	21322	167.02	1884;404	970	**187.29**	1788;559	13602
WikiEle	72.06	704;2	647	71.52	708;2	44	**72.17**	742;2	448
WikiPol	**89.05**	563;2	9876	84.45	462;0	762	88.89	618;2	5768

Ablation study. To assess the effect of the main components of our Neural2PC framework, we focused an ablation study on the following simplifications of Neural2PC: (i) NN, which discards the SGNN$(\cdot)$ block, hence it is composed of the NN$(\cdot)$ block only, and (ii) DIRECT, which performs a direct optimization of the continuous $\mathbf{z}$ node-to-community assignments by minimizing the $\mathcal{L}_{2\text{PC}}$ loss function (Equation 5) via projected gradient descent, i.e., at every step, gradient descent along with projection of the assignment variables onto the range $[-1, 1]$.

Results of the ablation study are shown in Table 8, in terms of polarity, size of the solution and total execution time. The full Neural2PC is confirmed to be necessary to achieve the best polarity on all datasets, or at least comparable with DIRECT (TwitterRef and WikiEle); the latter, however, is less efficient than Neural2PC, especially on larger datasets. This can be explained by the fact that the SGNN$(\cdot)$ module in Neural2PC helps produce more similar continuous scores for nodes that are assigned to the same community after rounding. That is, the rounding step of Neural2PC has to test less thresholds than DIRECT, hence it takes less time. Also, concerning the size of the communities, both Neural2PC and DIRECT yield solutions that involve more nodes than NN.

The above results are complemented by an analysis of the trends of polarity achieved by Neural2PC and its simplifications, by varying the number of training epochs (up to $e_{max} = 300$); results are shown in the ***Appendix***. This analysis revealed that the DIRECT variant often requires more epochs (on average, at least double) than Neural2PC and NN to reach the maximum polarity.

Overall, the outcomes of this ablation study justify the need for all components of the proposed Neural2PC framework.

γ-polarity results. In this experimental stage, we aim to delve into the impact of the value of γ to the size as well as the quality of the solutions provided by Neural2PC equipped with the γ-polarity loss function. Given γ, let $\mathbf{x}_\gamma$ be the solution obtained by optimizing the γ-polarity, e.g., $\mathbf{x}_1$ corresponds to the solutions obtained by optimizing the $\mathcal{L}_{2\text{PC}}$ loss. We are interested in evaluating the quality of the $\mathbf{x}_\gamma$ solutions for different values of γ both in terms of γ-polarity (i.e., the same value for γ used for the training process) as well as in terms of the standard polarity, i.e., 1-polarity. Also, we aim to analyze the γ-polarity of the $\mathbf{x}_1$ solutions as well as the size of the

Table 9 γ-polarity scores of the proposed Neural2PC method vs. its best-performing competing method, on Congress dataset.

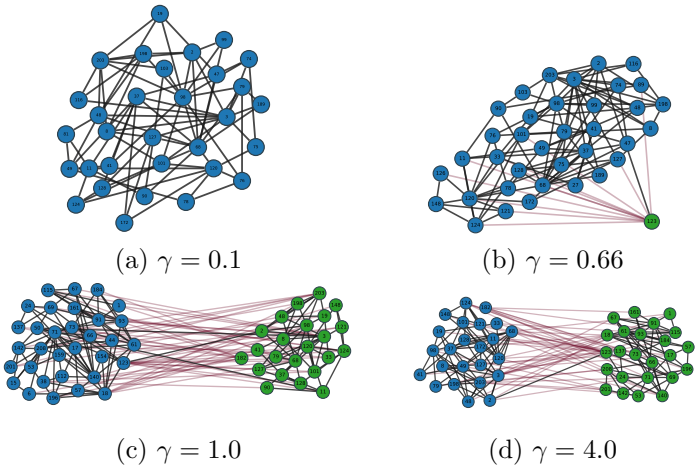
γ	Neural2PC					competing		
	$p_\gamma(\mathbf{x}_\gamma)$	$p_\gamma(\mathbf{x}_1)$	$p_1(\mathbf{x}_\gamma)$	size	model	p_γ	size	method
0.05	**108.75**	7.4	5.44	32;0	SGDNET	36.16	216.0;3.0	BNC(2)
0.1	**54.38**	7.35	5.44	32;0	SGDNET	22.05	216;3	BNC(2)
0.25	**21.75**	7.22	5.44	32;0	SGDNET	10.16	216;3	BNC(2)
0.5	**10.88**	7.16	5.44	32;0	SGCN_{att}	6.84	28;24	EIGEN
0.66	**8.21**	6.89	5.42	31;0	SGDNET	6.75	28;24	EIGEN
0.769	**7.18**	6.86	5.69	37;2	SGCN_{att}	6.70	28;24	EIGEN
1.0	**6.64**	6.64	6.64	27;23	SGDNET	6.58	28;24	EIGEN
1.3	**6.62**	6.44	6.62	26;26	SGDNET	6.43	28;24	EIGEN
1.5	**6.62**	6.31	6.62	26;26	SGDNET	6.33	28;24	EIGEN
2.0	**6.65**	5.77	6.65	23;23	SGCN_{att}	6.11	28;24	EIGEN
4.0	**6.67**	4.44	6.67	24;24	$\text{SGCN}_{\text{mean}}$	5.38	28;24	EIGEN
10	**6.62**	4.09	6.62	32;32	SGCN_{sum}	5.38	28;24	EIGEN
20	**6.56**	2.87	6.56	27;27	SGCN_{sum}	5.38	28;24	EIGEN

Table 10 γ-polarity scores of the proposed Neural2PC method vs. its best-performing competing method, on Slashdot dataset.

γ	Neural2PC					competing		
	$p_\gamma(\mathbf{x}_\gamma)$	$p_\gamma(\mathbf{x}_1)$	$p_1(\mathbf{x}_\gamma)$	size	model	p_γ	size	method
0.05	1644.4	1645.69	82.22	209;0	$\text{SGCN}_{\text{mean}}$	**1654.4**	200;0	GREEDY
0.1	822.36	822.84	82.24	195;0	$\text{SGCN}_{\text{mean}}$	**827.2**	200;0	GREEDY
0.25	328.8	329.14	82.2	200;0	$\text{SGCN}_{\text{mean}}$	**330.88**	200;0	GREEDY
0.5	164.51	164.57	82.25	204;0	$\text{SGCN}_{\text{mean}}$	**165.44**	200;0	GREEDY
0.66	124.61	124.67	82.24	207;0	$\text{SGCN}_{\text{mean}}$	**125.33**	200;0	GREEDY
0.769	107.0	107.0	82.28	205;0	$\text{SGCN}_{\text{mean}}$	**107.57**	200;0	GREEDY
1.0	82.28	82.28	82.28	204;0	$\text{SGCN}_{\text{mean}}$	**82.72**	200;0	GREEDY
1.3	63.26	63.3	82.24	200;0	$\text{SGCN}_{\text{mean}}$	**63.63**	200;0	GREEDY
1.5	**57.07**	54.66	60.6	191;149	SGCN_{att}	55.15	200;0	GREEDY
2.0	**57.41**	40.84	57.41	172;172	SGCN_{sum}	41.36	200;0	GREEDY
4.0	**57.78**	20.42	57.78	171;171	SGCN_{sum}	20.68	200;0	GREEDY
10.0	**40.69**	8.17	42.45	416;412	SGCN_{sum}	8.27	200;0	GREEDY
20.0	**39.06**	4.08	39.06	268;268	SGCN_{sum}	4.14	200;0	GREEDY

discovered polarized communities by varying γ. We selected several values of γ spanning a relatively large interval above 1 (up to 20); for each of such values, say n, we also considered the reciprocal of n, in order to explore a comparable range of values below 1; for instance, the dual of $\gamma = 1.5$ is $\gamma = 0.66$. We conducted this experiment on 5 datasets but, for the sake of simplicity, we only show a subset of these results for Congress (cf. Table 9) and Slashdot (cf. Table 10): details about these experiments and on other datasets can be found in ***Appendix***. Concerning our Neural2PC and the competing methods, we report the results obtained by the best-performing (in terms of γ-polarity) method.

Looking at the tables, several remarks stand out. First, Neural2PC results to be the most competitive method in terms of γ-polarity on Congress for any γ; this holds only after a certain value for γ (i.e., 1.5) on Slashdot. Second, as expected, the γ-polarity of the solutions produced by the competing methods (as well as by Neural2PC trained



(a) $\gamma = 0.1$ (b) $\gamma = 0.66$

(c) $\gamma = 1.0$ (d) $\gamma = 4.0$

Fig. 6 Solutions yielded by Neural2PC by optimizing γ-polarity (Def. 2), for different values of γ, on the Congress dataset.

with the 1-polarity, i.e., $p_\gamma(\mathbf{x}_1)$) monotonically decreases as γ increases. This is due since, as γ increases, the denominator of γ-polarity increases. On the other hand, the values for $p_\gamma(\mathbf{x}_\gamma)$ have a decreasing trend as γ increases, but not monotonic.

As for solution size, the two discovered polarized communities yielded by Neural2PC tend to become more balanced (resp. unbalanced) for higher (resp. lower) values of γ until, after a certain value threshold value, the two communities have the same size (resp., one community is empty). On the contrary, the solutions provided by the competing methods are highly unbalanced in size, especially on Slashdot where one of the two detected community is empty. More interestingly, controlling the balance between the size of the discovered communities is also beneficial in terms of 1-polarity. In particular, this happens on Congress, where favoring polarized communities more balanced in size (i.e., by setting $\gamma > 1$) leads to the best solutions in terms of 1-polarity, i.e., $p_1(\mathbf{x}_{4.0}) = 6.67 > 6.64 = p_1(\mathbf{x}_1)$ (cf. Table 9).

Figure 6 complements the above results by providing insights into the structural properties of the detected polarized communities by Neural2PC (equipped with the $\text{SGCN}_{\text{mean}}$ GNN) on the Congress dataset. Specifically, for $\gamma = 0.1$, one community is empty and the other has 31 nodes with 81 inter-community positive edges and no negative edges. With $\gamma = 0.66$, the empty community becomes a singleton (i.e., one node), and the second has 36 nodes with 93 intra-community positive edges and 11 inter-community negative edges (1 positive edge). For $\gamma = 1$, there are two communities with 32 and 23 nodes, with 83 (resp. 52) internal positive edges, and they are connected by 50 negative and 3 positive edges. Optimizing for 4-polarity results in two communities with 24 nodes each, featuring 57 and 58 intra-community positive edges, connected by 46 negative and 1 positive edge.

Overall, our experiments revealed the practical benefit of γ-polarity since we can inspect the communities obtained for different γ values and keep the ones that best suit our purposes.

7 Conclusion

Summary. In this paper, we advanced the state of the art in 2PC, a well-established combinatorial-optimization problem which aims to discover two polarized communities from a signed graph, through maximization of the so-called polarity function. We provided a twofold contribution: (*i*) a novel neural-network-based approach to 2PC, and (*ii*) a generalization of the polarity function, γ-polarity, which we suitably incorporate into the proposed neural framework. Notably, we addressed two key limitations in the state of the art in 2PC: relying on a single relaxed solution for producing the ultimate discrete solution, and mitigating size-unbalanced output communities.

Our experimental evaluation, which was conducted on 10 real-world signed networks and synthetically generated signed networks has provided empirical evidence of the meaningfulness and relevance of our Neural2PC versus all competing methods for discovering polarized groups. In particular, (*i*) Neural2PC outperforms all competitors in terms of polarity value as well as coherent polarized structure and ability to produce non-empty groups; (*ii*) using the γ-polarity loss enables flexibility of our Neural2PC, which is useful to effectively control the balance of the sizes of the discovered polarized groups; (*iii*) all components of Neural2PC are justified to lead it to achieve the best polarity performance, and (*iv*) this appears to be relatively robust w.r.t. different signed GNN models constituting its SGNN component; (*v*) the runtime of Neural2PC scales linearly with the size of the network.

Future works. We believe that a number of further developments are worthy of investigation. First, we aim to address two main limitations of our framework, namely the number of polarized groups that can be detected, and the separation of the rounding block from the neural-network components. Addressing the former would be key-enabling for extended polarization scenarios, such as on multi-party political networks. To this purpose, major interventions would be to revise our approach to integrate the extension of the 2PC task to a $k > 2$ polarization setting [9], and to refine the (γ-)polarity loss and the computation of the continuous node-to-community assignment solutions. The latter limitation would be overcome by integrating the rounding block into the neural architecture: this way, by learning the rounding of the continuous solution, the repeated trials of rounding thresholds τ could be avoided, thus also overcoming a limitation of the early approximation algorithms for 2PC. Further interesting research paths include to devise a custom GNN model for the 2PC task, and to delve into the γ-polarity function from a combinatorial-optimization perspective.

Funding. This work is partly funded by the PNRR Future AI Research (FAIR) project, spoke 9, H23C22000860006, M4C21.3.

Conflicts of interest/Competing interests. The authors have no competing interests as defined by Springer, or other interests that might be perceived to influence the results and/or discussion reported in this paper.

Ethics approval. We declare that this research did not require Ethics approval.

Consent to participate. Not applicable.

Consent for publication. All the authors of this manuscript consent to its publication.

Availability of data and material. Not applicable.

Code availability. We make source code and evaluation data available at https://github.com/Ralyhu/neural2pc.

Authors' contributions. All the authors contributed to the conceptualization, methodology and writing of the manuscript. DM performed numerical experiments. All authors reviewed the manuscript.

References

[1] Ghoshal, A.K., Das, N., Das, S.: Disjoint and overlapping community detection in small-world networks leveraging mean path length. TCSS **9**(2), 406–418 (2021)

[2] Ghoshal, A.K., Das, N.: On diameter based community structure identification in networks. In: Proceedings of the 18th International Conference on Distributed Computing and Networking, pp. 1–6 (2017)

[3] Ghoshal, A.K., Das, N., Bhattacharjee, S., Chakraborty, G.: A fast parallel genetic algorithm based approach for community detection in large networks. In: 2019 11th International Conference on Communication Systems & Networks (COMSNETS), pp. 95–101 (2019)

[4] Conover, M., Ratkiewicz, J., Francisco, M., Goncalves, B., Menczer, F., Flammini, A.: Political polarization on Twitter. In: Proc. ICWSM Conf., pp. 89–96 (2011)

[5] Hohmann, M., Devriendt, K., Coscia, M.: Quantifying ideological polarization on a network using generalized Euclidean distance. Science Advances **9**(9) (2023)

[6] Garimella, K., De Francisci Morales, G., Gionis, A., Mathioudakis, M.: Reducing controversy by connecting opposing views. In: Proc. WSDM Conf., pp. 81–90 (2017)

[7] Lelkes, Y.: Mass polarization: Manifestations and measurements. Public Opinion Quarterly **80**(S1), 392–410 (2016)

[8] Bonchi, F., Galimberti, E., Gionis, A., Ordozgoiti, B., Ruffo, G.: Discovering polarized communities in signed networks. In: Proc. CIKM Conf., pp. 961–970 (2019)

[9] Tzeng, R.-C., Ordozgoiti, B., Gionis, A.: Discovering conflicting groups in signed networks. Proc. NIPS Conf. **33**, 10974–10985 (2020)

[10] Harary, F.: On the notion of balance of a signed graph. Michigan Mathematical Journal **2**, 143–146 (1953)

[11] Beigi, G., Tang, J., Liu, H.: Signed link analysis in social media networks. In: Proc. ICWSM Conf., pp. 539–542 (2016)

[12] Derr, T., Aggarwal, C.C., Tang, J.: Signed network modeling based on structural balance theory. In: Proc. CIKM Conf., pp. 557–566 (2018)

[13] Yao, K., Chang, L., Qin, L.: Computing maximum structural balanced cliques in signed graphs. In: Proc. ICDE Conf., pp. 1004–1016 (2022)

[14] Zheng, X., Zeng, D.D., Wang, F.: Social balance in signed networks. Inf. Syst. Frontiers **17**(5), 1077–1095 (2015)

[15] Cai, H., Zheng, V.W., Chang, K.C.: A comprehensive survey of graph embedding: Problems, techniques, and applications. TKDE **30**(9), 1616–1637 (2018)

[16] Wu, L., Cui, P., Pei, J., Zhao, L.: Graph Neural Networks: Foundations, Frontiers, and Applications, p. 725. Springer, Singapore (2022)

[17] Derr, T., Ma, Y., Tang, J.: Signed graph convolutional networks. In: Proc. ICDM Conf., pp. 929–934 (2018)

[18] Liu, H., Zhang, Z., Cui, P., Zhang, Y., Cui, Q., Liu, J., Zhu, W.: Signed graph neural network with latent groups. In: Proc. KDD Conf., pp. 1066–1075 (2021)

[19] Li, Y., Tian, Y., Zhang, J., Chang, Y.: Learning signed network embedding via graph attention. In: Proc. AAAI Conf., pp. 4772–4779 (2020)

[20] Wang, S., Tang, J., Aggarwal, C., Chang, Y., Liu, H.: Signed network embedding in social media. In: Proc. SDM Conf., pp. 327–335 (2017)

[21] Kim, J., Park, H., Lee, J.-E., Kang, U.: Side: representation learning in signed directed networks. In: Proc. WWW Conf., pp. 509–518 (2018)

[22] Huang, J., Shen, H., Hou, L., Cheng, X.: SDGNN: learning node representation for signed directed networks. In: Proc. AAAI Conf., pp. 196–203 (2021)

[23] Huang, J., Shen, H., Hou, L., Cheng, X.: Signed graph attention networks. In: Proc. ICANN Work., pp. 566–577 (2019)

[24] Chiang, K.-Y., Whang, J.J., Dhillon, I.S.: Scalable clustering of signed networks using balance normalized cut. In: Proc. CIKM Conf., pp. 615–624 (2012)

[25] Cucuringu, M., Davies, P., Glielmo, A., Tyagi, H.: SPONGE: A generalized eigenproblem for clustering signed networks. In: Proc. AISTATS Conf., pp. 1088–1098 (2019)

[26] He, Y., Reinert, G., Wang, S., Cucuringu, M.: SSSNET: semi-supervised signed network clustering. In: Proc. SDM Conf., pp. 244–252 (2022)

[27] Kunegis, J., Schmidt, S., Lommatzsch, A., Lerner, J., Luca, E.W.D., Albayrak, S.: Spectral analysis of signed graphs for clustering, prediction and visualization. In: Proc. SDM Conf., pp. 559–570 (2010)

[28] Mercado, P., Tudisco, F., Hein, M.: Clustering signed networks with the geometric mean of laplacians. In: Proc. NIPS Conf., pp. 4421–4429 (2016)

[29] Ordozgoiti, B., Matakos, A., Gionis, A.: Finding large balanced subgraphs in signed networks. In: Proc. WWW Conf., pp. 1378–1388 (2020)

[30] Xiao, H., Ordozgoiti, B., Gionis, A.: Searching for polarization in signed graphs: a local spectral approach. In: Proc. WWW Conf., pp. 362–372 (2020)

[31] Niu, J., Sariyüce, A.E.: On cohesively polarized communities in signed networks. In: Proc. WWW Conf., pp. 1339–1347 (2023)

[32] Chu, L., Wang, Z., Pei, J., Wang, J., Zhao, Z., Chen, E.: Finding gangs in war from signed networks. In: Proc. KDD Conf., pp. 1505–1514 (2016)

[33] Cappart, Q., Chételat, D., Khalil, E.B., Lodi, A., Morris, C., Velickovic, P.: Combinatorial optimization and reasoning with graph neural networks. JMLR **24**, 130–113061 (2023)

[34] Peng, Y., Choi, B., Xu, J.: Graph learning for combinatorial optimization: A survey of state-of-the-art. Data Sci. Eng. **6**(2), 119–141 (2021)

[35] Li, H., Xu, M., Bhowmick, S.S., Joty, S.R., Sun, C., Cui, J.: PIANO: influence maximization meets deep reinforcement learning. TCSS **10**(3), 1288–1300 (2023)

[36] Jung, S., Keuper, M.: Learning to solve minimum cost multicuts efficiently using edge-weighted graph convolutional neural networks. In: Proc. ECML PKDD Conf., pp. 485–501 (2022)

[37] Tsitsulin, A., Palowitch, J., Perozzi, B., Müller, E.: Graph clustering with graph neural networks. JMLR **24**, 1–21 (2023)

[38] Su, X., Xue, S., Liu, F., Wu, J., Yang, J., Zhou, C., Hu, W., Paris, C., Nepal, S., Jin, D., Sheng, Q.Z., Yu, P.S.: A comprehensive survey on community detection with deep learning. TNNLS, 1–21 (2022)

[39] Leskovec, J., Krevl, A.: SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data (2014)

[40] Kunegis, J.: KONECT – The Koblenz Network Collection. In: Proc. WWW Conf., pp. 1343–1350 (2013). http://konect.cc

[41] Lai, M., Patti, V., Ruffo, G., Rosso, P.: Stance evolution and twitter interactions in an Italian political debate. In: Proc. NLDB Conf., pp. 15–27 (2018)

[42] Bansal, N., Blum, A., Chawla, S.: Correlation clustering. Machine Learning **56**(1), 89–113 (2004)

[43] Charikar, M.: Greedy approximation algorithms for finding dense components in a graph. In: Proc. APPROX Work., pp. 84–95 (2003)

Appendix A Software and Hardware Configurations

All the experiments were carried out on the Cresco6 cluster[1], a high-performance computing system running Linux Centos 7.4, and consisting of 434 nodes, where each one is equipped with two Intel(R) Xeon(R) Platinum 8160 CPU @2.10GHz x24 processor and 192GB ram.

Appendix B Additional Results

Impact of different signed GNNs on Neural2PC. Table B1 reports polarity and community size values corresponding to all the variants of our Neural2PC method based on different signed GNN models. It can be noticed that the polarity of the solutions provided by Neural2PC does not significantly change across the various GNNs, which indicates robustness of our approach in terms of one of its main components. The only apparent exception is WikiCon, where the polarity change difference between best-performing and worst-performing GNN model is about 13%. In all other cases, this difference is $\leq 4\%$. Overall, SGCN_{sum} achieves (slightly) best performance in terms of polarity in most cases, followed by its variant $\text{SGCN}_{\text{mean}}$. The better performance of SGCN_{sum} can be explained since the sum aggregator was demonstrated to achieve better expressiveness in graph representation learning for signed graphs [18]. Concerning the impact of the regularization term, all the signed GNNs take benefit from it in most cases. Also as concerns the size of communities, no evident difference is observed across the various models.

Ablation study. Figures B1–B2 complement the ablation results discussed in the main paper by providing insights into the trends of polarity achieved by Neural2PC and its simplifications, as the number of training epochs increases. At first glance, it can be noticed that the Direct variant generally converges much slower than Neural2PC and NN, as the former requires more epochs to reach the maximum polarity. By contrast, the polarity trend of full Neural2PC tends to have sharp increase just after few epochs; this is generally followed by a plateau containing the polarity peak (or, like, e.g., in Congress, a further increase at late epochs). Another interesting remark arises from the number of epochs corresponding to the peak of the rounded solution: while it is usually set around mid-high epochs for NN and Direct, it can be at low-mid epochs for Neural2PC that, which would hint at much less requirements in terms of training epochs when the Neural2PC framework is considered in its entirety.

[1] https://www.eneagrid.enea.it/CRESCOportal/

Table B1 Polarity and solution size ($|S_1|;|S_2|$) of the proposed Neural2PC method when equipped with different signed GNNs. Best results in bold, second-best underlined.

method	criteria	*Bitcoin*	*Cloister*	*Congress*	*Epinions*	*HTribes*	*Slashdot*	*TwitterRef*	*WikiCon*	*WikiEle*	*WikiPol*
$SGCN_{mean}$	pol.	29.84	**7.455**	6.62	**171.1**	**6.18**	82.24	174.24	178.37	72.13	88.62
	size	152;11	8;3	29;24	268;1	7;4	203;0	678;4	1951;602	717;3	596;0
$SGCN_{mean}$-DR	pol.	29.86	**7.455**	6.61	**171.1**	**6.18**	**82.25**	174.22	178.48	72.11	88.63
	size	149;11	8;3	29;23	267;1	7;4	207;0	678;4	1992;580	723;2	558;0
$SGCN_{sum}$	pol.	**30.28**	**7.455**	6.62	168.55	**6.18**	81.57	**174.35**	**187.29**	72.14	**88.89**
	size	158;32	8;3	31;24	255;1	7;4	195;0	677;4	1788;559	742;2	618;2
$SGCN_{sum}$-DR	pol.	30.25	**7.455**	6.62	170.31	**6.18**	81.68	174.32	187.16	**72.17**	88.84
	size	155;33	8;3	29;24	273;0	7;4	203;1	669;4	1734;566	750;2	617;1
$SGCN_{att}$	pol.	29.75	**7.455**	6.63	170.96	**6.18**	81.92	174.31	179.24	71.91	88.66
	size	143;12	8;3	29;24	268;1	7;4	200;0	674;4	1999;698	692;2	539;1
$SGCN_{att}$-DR	pol.	29.78	**7.455**	6.63	**171.1**	**6.18**	81.69	174.32	179.53	71.96	88.68
	size	139;11	8;3	30;23	270;1	7;4	215;0	682;4	2041;618	692;2	558;2
SNEA	pol.	29.36	**7.455**	6.63	168.24	**6.18**	82	166.19	165.87	70.08	82.16
	size	144;10	8;3	29;24	265;1	7;4	197;0	638;2	2009;647	640;0	468;0
SNEA-DR	pol.	29.41	**7.455**	**6.64**	168.3	**6.18**	81.92	166.76	166.05	70.25	82.47
	size	135;2	8;3	29;23	277;0	7;4	194;0	623;2	1920;589	635;0	461;0
SGDNET	pol.	29.68	**7.455**	**6.64**	170.19	**6.18**	81.51	174	177.57	71.84	88.19
	size	137;1	8;3	29;24	270;1	7;4	209;1	668;4	1904;571	685;2	578;3
SGDNET-DR	pol.	29.67	**7.455**	6.61	170.06	**6.18**	81.66	173.94	178.42	71.85	88.22
	size	135;1	8;3	31;25	266;1	7;4	201;0	676;4	1893;570	729;2	578;3

γ-polarity results. Tables B2-B6 show the impact of the value of γ to the size as well as the quality of the solutions provided by Neural2PC equipped with the γ-polarity loss function. Looking at the tables, the Neural2PC method stands out as the most competitive approach in terms of γ-polarity on Congress and WikiCon for any γ; this holds only after a certain value for γ on the remaining datasets, i.e., 0.66 on Epinions (cf. Table-B3), 1.5 on Slashdot (cf. Table-B4), and 0.83 on TwitterRef (cf. Table-B5).

As in the case of Congress dataset, discussed in the main paper, controlling the balance between the size of the discovered communities is beneficial in terms of 1-polarity also on the WikiCon dataset. However, this happens in such two cases with different characteristics: (i) on Congress, where favoring polarized communities more balanced in size (i.e., by setting $\gamma > 1$) leads to the best solutions in terms of 1-polarity, i.e., $p_1(\mathbf{x}_{4.0}) = 6.67 > 6.64 = p_1(\mathbf{x}_1)$ (cf. Table B2); (ii) on WikiCon, where favoring size unbalance between communities (i.e., by setting $\gamma < 1$) has also a positive effect in terms of 1-polarity, i.e., $p_1(\mathbf{x}_{0.909}) = 187.52 > 184.89 = p_1(\mathbf{x}_1)$ (cf. Table B6).

Details on execution times. Table B7 shows details on the execution times by Neural2PC and competing methods as shown in Fig. 5.

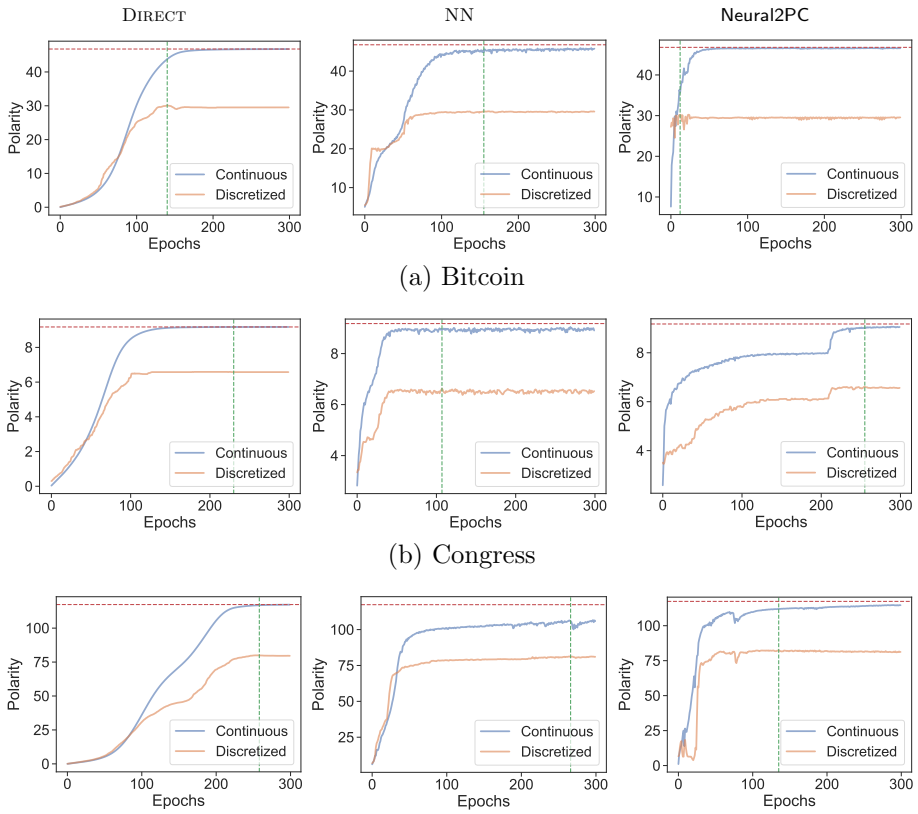
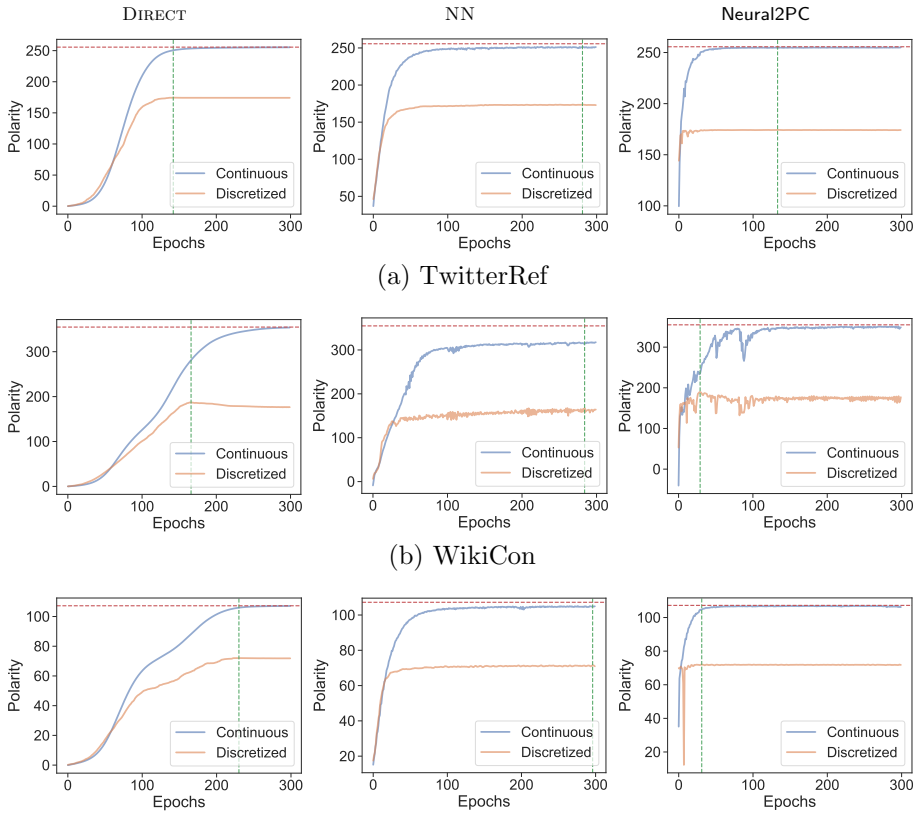


Fig. B1 Polarity scores by varying the number of epochs, for Neural2PC and its simplifications relevant to the ablation study (i.e., DIRECT and NN). Each plot distinguishes between polarity of the continuous (i.e., relaxed) solution and polarity of the discretized (i.e., rounded) solution. Green vertical lines correspond to peaks of the rounded solutions' polarity, while top red horizontal lines set the upper-bound of polarity (i.e., polarity of the optimal (relaxed) solution to the 2PC-RELAXED instance on a particular graph).



(a) TwitterRef

(b) WikiCon

(c) WikiEle

Fig. B2 *(Cont.)* Polarity scores by varying the number of epochs, for Neural2PC and its simplifications relevant to the ablation study (i.e., DIRECT and NN). Each plot distinguishes between polarity of the continuous (i.e., relaxed) solution and polarity of the discretized (i.e., rounded) solution. Green vertical lines correspond to peaks of the rounded solutions' polarity, while top red horizontal lines set the upper-bound of polarity (i.e., polarity of the optimal (relaxed) solution to the 2PC-RELAXED instance on a particular graph).

Table B2 γ-polarity scores of the proposed Neural2PC method vs. its best-performing competing method, on Congress dataset. Best results in bold.

γ	Neural2PC					competing		
	$p_\gamma(\mathbf{x}_\gamma)$	$p_\gamma(\mathbf{x}_1)$	$p_1(\mathbf{x}_\gamma)$	size	model	p_γ	size	method
0.05	**108.75**	7.4	5.44	32;0	SGDNET	36.16	216;3	BNC(2)
0.1	**54.38**	7.35	5.44	32;0	SGDNET	22.05	216;3	BNC(2)
0.25	**21.75**	7.22	5.44	32;0	SGDNET	10.16	216;3	BNC(2)
0.5	**10.88**	7.16	5.44	32;0	SGCN_{att}	6.84	28;24	EIGEN
0.66	**8.21**	6.89	5.42	31;0	SGDNET	6.75	28;24	EIGEN
0.714	**7.64**	6.92	5.68	36;2	SGCN_{att}	6.72	28;24	EIGEN
0.769	**7.18**	6.86	5.69	37;2	SGCN_{att}	6.70	28;24	EIGEN
0.83	**6.86**	6.77	6.3	34;12	SGDNET	6.66	28;24	EIGEN
0.909	**6.73**	6.71	6.6	32;21	SGDNET	6.62	28;24	EIGEN
1.0	**6.64**	6.64	6.64	27;23	SGDNET	6.58	28;24	EIGEN
1.1	**6.61**	6.57	6.61	28;28	SGDNET	6.53	28;24	EIGEN
1.2	**6.62**	6.5	6.64	27;26	SGDNET	6.48	28;24	EIGEN
1.3	**6.62**	6.44	6.62	26;26	SGDNET	6.43	28;24	EIGEN
1.4	**6.61**	6.44	6.61	28;28	SGCN_{sum}	6.38	28;24	EIGEN
1.5	**6.62**	6.31	6.62	26;26	SGDNET	6.33	28;24	EIGEN
2.0	**6.65**	5.77	6.65	23;23	SGCN_{att}	6.11	28;24	EIGEN
4.0	**6.67**	4.44	6.67	24;24	$\text{SGCN}_{\text{mean}}$	5.38	28;24	EIGEN
10	**6.62**	4.09	6.62	32;32	SGCN_{sum}	5.38	28;24	EIGEN
20	**6.56**	2.87	6.56	27;27	SGCN_{sum}	5.38	28;24	EIGEN

Table B3 γ-polarity scores of the proposed Neural2PC method vs. its best-performing competing method, on Epinions dataset. Best results in bold.

γ	Neural2PC					competing		
	$p_\gamma(\mathbf{x}_\gamma)$	$p_\gamma(\mathbf{x}_1)$	$p_1(\mathbf{x}_\gamma)$	size	model	p_γ	size	method
0.05	3405.82	3378.87	170.29	268;0	SGCN_{sum}	**3405.95**	269;0	GREEDY
0.1	1702.91	1689.44	170.29	268;0	SGCN_{sum}	**1702.97**	269;0	GREEDY
0.25	681.16	675.77	170.29	268;0	SGCN_{sum}	**681.19**	269;0	GREEDY
0.5	340.58	337.89	170.29	268;0	SGCN_{sum}	**340.59**	269;0	GREEDY
0.66	**258.36**	258.29	171.17	269;1	$\text{SGCN}_{\text{mean}}$	258.03	269;0	GREEDY
0.714	**239.03**	238.96	171.17	269;1	$\text{SGCN}_{\text{mean}}$	238.51	269;0	GREEDY
0.769	**222.08**	222.03	171.16	270;1	$\text{SGCN}_{\text{mean}}$	221.45	269;0	GREEDY
0.83	**205.87**	205.86	171.13	268;1	$\text{SGCN}_{\text{mean}}$	205.18	269;0	GREEDY
0.909	**188.15**	188.12	171.15	268;1	$\text{SGCN}_{\text{mean}}$	187.35	269;0	GREEDY
1.0	**171.13**	171.13	171.13	267;1	$\text{SGCN}_{\text{mean}}$	170.3	269;0	GREEDY
1.1	**155.71**	155.68	171.16	268;1	$\text{SGCN}_{\text{mean}}$	154.82	269;0	GREEDY
1.2	**142.74**	142.78	171.07	263;1	$\text{SGCN}_{\text{mean}}$	141.91	269;0	GREEDY
1.3	**138.32**	129.96	138.32	317;317	SGCN_{sum}	131.0	269;0	GREEDY
1.4	**138.23**	120.67	138.23	307;307	SGCN_{sum}	121.64	269;0	GREEDY
1.5	**139.24**	112.63	139.24	338;338	SGCN_{sum}	113.53	269;0	GREEDY
2.0	**137.65**	84.47	137.65	332;332	SGCN_{sum}	85.15	269;0	GREEDY
4.0	**128.14**	42.77	128.14	343;343	SGDNET	42.57	269;0	GREEDY
10.0	**84.39**	16.89	84.39	867;867	SGCN_{sum}	17.03	269;0	GREEDY
20.0	**94.61**	8.45	100.51	918;912	SGCN_{sum}	8.51	269;0	GREEDY

Table B4 γ-polarity scores of the proposed Neural2PC method vs. its best-performing competing method, on Slashdot dataset. Best results in bold.

γ	Neural2PC					competing		
	$p_\gamma(\mathbf{x}_\gamma)$	$p_\gamma(\mathbf{x}_1)$	$p_1(\mathbf{x}_\gamma)$	size	model	p_γ	size	method
0.05	1644.4	1645.69	82.22	209;0	$\text{SGCN}_{\text{mean}}$	**1654.4**	200;0	Greedy
0.1	822.36	822.84	82.24	195;0	$\text{SGCN}_{\text{mean}}$	**827.2**	200;0	Greedy
0.25	328.8	329.14	82.2	200;0	$\text{SGCN}_{\text{mean}}$	**330.88**	200;0	Greedy
0.5	164.51	164.57	82.25	204;0	$\text{SGCN}_{\text{mean}}$	**165.44**	200;0	Greedy
0.66	124.61	124.67	82.24	207;0	$\text{SGCN}_{\text{mean}}$	**125.33**	200;0	Greedy
0.714	115.3	115.24	82.32	210;0	$\text{SGCN}_{\text{mean}}$	**115.85**	200;0	Greedy
0.769	107.0	107.0	82.28	205;0	$\text{SGCN}_{\text{mean}}$	**107.57**	200;0	Greedy
0.83	99.05	99.14	82.21	212;0	$\text{SGCN}_{\text{mean}}$	**99.66**	200;0	Greedy
0.909	90.33	90.52	82.11	201;0	$\text{SGCN}_{\text{mean}}$	**91.0**	200;0	Greedy
1.0	82.28	82.28	82.28	204;0	$\text{SGCN}_{\text{mean}}$	**82.72**	200;0	Greedy
1.1	74.68	74.8	82.15	212;0	$\text{SGCN}_{\text{mean}}$	**75.2**	200;0	Greedy
1.2	68.41	68.57	82.09	199;0	$\text{SGCN}_{\text{mean}}$	**68.93**	200;0	Greedy
1.3	63.26	63.3	82.24	200;0	$\text{SGCN}_{\text{mean}}$	**63.63**	200;0	Greedy
1.4	58.77	58.77	81.83	204;2	$\text{SGCN}_{\text{mean}}$	**59.09**	200;0	Greedy
1.5	**57.07**	54.66	60.6	191;149	SGCN_{att}	55.15	200;0	Greedy
2.0	**57.41**	40.84	57.41	172;172	SGCN_{sum}	41.36	200;0	Greedy
4.0	**57.78**	20.42	57.78	171;171	SGCN_{sum}	20.68	200;0	Greedy
10.0	**40.69**	8.17	42.45	416;412	SGCN_{sum}	8.27	200;0	Greedy
20.0	**39.06**	4.08	39.06	268;268	SGCN_{sum}	4.14	200;0	Greedy

Table B5 γ-polarity scores of the proposed Neural2PC method vs. its best-performing competing method, on TwitterRef dataset. Best results in bold.

γ	Neural2PC					competing		
	$p_\gamma(\mathbf{x}_\gamma)$	$p_\gamma(\mathbf{x}_1)$	$p_1(\mathbf{x}_\gamma)$	size	model	p_γ	size	method
0.05	3471.67	2718.61	173.58	687;0	SGCN_{sum}	**3478.77**	685;0	Greedy
0.1	1736.32	1576.08	173.63	695;0	$\text{SGCN}_{\text{mean}}$	**1739.39**	685;0	Greedy
0.25	694.67	667.65	173.67	692;0	SGCN_{sum}	**695.75**	685;0	Greedy
0.5	347.44	343.61	173.72	680;0	SGCN_{sum}	**347.88**	685;0	Greedy
0.66	263.21	262.18	173.99	670;1	SGCN_{sum}	**263.54**	685;0	Greedy
0.714	243.34	242.76	173.95	661;1	SGCN_{sum}	**243.61**	685;0	Greedy
0.769	226.18	225.73	174.09	672;1	SGCN_{sum}	**226.19**	685;0	Greedy
0.83	**209.66**	209.44	174.13	669;1	SGCN_{sum}	209.56	685;0	Greedy
0.909	**191.57**	191.53	174.29	677;3	SGCN_{sum}	191.35	685;0	Greedy
1.0	**174.36**	174.36	174.36	667;5	SGCN_{sum}	174.08	669;4	Eigen
1.1	**158.67**	158.67	174.26	681;6	SGCN_{att}	158.43	669;4	Eigen
1.2	**145.64**	145.66	174.26	679;6	SGCN_{sum}	145.36	669;4	Eigen
1.3	**134.65**	134.59	174.21	677;7	SGCN_{sum}	134.28	669;4	Eigen
1.4	**125.36**	125.08	144.3	631;285	SGCN_{sum}	124.77	669;4	Eigen
1.5	**122.19**	116.82	133.85	577;392	SGCN_{sum}	116.52	669;4	Eigen
2.0	**121.05**	87.84	121.05	535;535	SGCN_{sum}	87.56	669;4	Eigen
4.0	**120.69**	44.08	120.69	511;511	SGCN_{sum}	43.91	669;4	Eigen
10.0	**120.21**	17.67	120.21	525;525	SGCN_{sum}	17.6	669;4	Eigen
20.0	**119.44**	8.84	119.44	472;472	SGCN_{sum}	8.8	669;4	Eigen

Table B6 γ-polarity scores of the proposed Neural2PC method vs. its best-performing competing method, on WikiCon dataset. Best results in bold.

γ	Neural2PC					competing		
	$p_\gamma(\mathbf{x}_\gamma)$	$p_\gamma(\mathbf{x}_1)$	$p_1(\mathbf{x}_\gamma)$	size	model	p_γ	size	method
0.05	**2786.89**	394.05	139.34	1346;0	SGCN_{sum}	2559.3	1151;0	Greedy
0.1	**1392.95**	371.91	139.29	1344;0	SGCN_{sum}	1279.65	1151;0	Greedy
0.25	**556.73**	318.26	139.18	1373;0	SGCN_{sum}	511.86	1151;0	Greedy
0.5	**282.72**	256.57	153.32	1562;69	SGCN_{sum}	256.85	1993;449	Eigen
0.66	**235.28**	228.25	174.4	2372;322	SGCN_{sum}	223.75	1993;449	Eigen
0.714	**222.61**	220.06	183.16	1873;440	SGCN_{sum}	214.43	1993;449	Eigen
0.769	**213.78**	212.29	177.3	1311;197	SGCN_{sum}	205.7	1993;449	Eigen
0.83	**204.45**	204.3	182.58	2027;461	SGCN_{sum}	196.81	1993;449	Eigen
0.909	**197.99**	194.8	187.52	1725;457	SGCN_{sum}	186.38	1993;449	Eigen
1.0	**184.89**	184.89	184.89	1911;541	SGCN_{sum}	175.65	1993;449	Eigen
1.1	**176.76**	175.11	182.87	1766;859	SGCN_{sum}	165.21	1993;449	Eigen
1.2	**173.08**	166.31	184.58	1636;820	SGCN_{sum}	155.94	1993;449	Eigen
1.3	**167.48**	158.35	181.0	1608;926	SGCN_{sum}	147.65	1993;449	Eigen
1.4	**164.55**	151.12	181.17	1602;956	SGCN_{sum}	140.2	1993;449	Eigen
1.5	**163.01**	144.52	170.88	1464;1206	SGCN_{sum}	133.46	1993;449	Eigen
2.0	**162.57**	118.62	162.78	1538;1534	SGCN_{sum}	107.61	1993;449	Eigen
4.0	**163.52**	69.09	163.67	1611;1610	SGCN_{sum}	60.64	1993;449	Eigen
10.0	**158.57**	30.67	158.57	1808;1808	SGCN_{sum}	26.25	1993;449	Eigen
20.0	**154.4**	15.92	155.11	2054;2053	SGCN_{sum}	13.5	1993;449	Eigen

Table B7 Execution times (in seconds) of the proposed Neural2PC method vs. competing methods on real datasets.

dataset	Eigen	R-Eigen	Pivot	Greedy	SPONGE	BNC	SSSNet	Neural2PC
Bitcoin	0.61	1.07	4.33	1.56	0.15	0.89	258	130.4 (107.8)
Cloister	0.01	0.13	0.09	0.12	0.02	0.05	7.3	5.53 (5.5)
Congress	0.07	0.13	0.15	0.12	0.04	0.03	11.8	63.6 (62.4)
Epinions	8.75	133.66	1150.41	776.73	1.89	18.23	17799.6	5955.8 (5364)
HTribes	0.03	0.05	0.14	0.13	0.02	0.03	6.3	6.32 (6.3)
Slashdot	12.2	12.74	492.43	319.06	1.64	8.52	12121.1	4177.8 (3846.3)
TwitterRef	2.88	10.59	41.92	15.75	0.32	0.85	5214.2	2032.8 (1954.3)
WikiCon	155.21	133.84	2449.51	1145.88	2.52	18.61	31082.6	13601.8 (12966.9)
WikiEle	0.99	4.52	13.94	4.81	0.2	0.58	1591.8	447.8 (413.9)
WikiPol	44.56	27.39	1110.98	819.95	2.46	16.27	18259.8	5768.4 (5214.5)