
DeSocial: Blockchain-based Decentralized Social Networks

Jingyuan Huang
Rutgers University

Xi Zhu
Rutgers University

Minghao Guo
Rutgers University

Yongfeng Zhang*
Rutgers University

Abstract

Web 2.0 social platforms are inherently centralized, with user data and algorithmic decisions controlled by the platform. However, users can only passively receive social predictions without being able to choose the underlying algorithm, which limits personalization. Fortunately, with the emergence of blockchain, users are allowed to choose algorithms that are tailored to their local situation, improving prediction results in a personalized way. In a blockchain environment, each user possesses its own model to perform the social prediction, capturing different perspectives on social interactions. In our work, we propose DeSocial, a decentralized social network learning framework deployed on an Ethereum (ETH) local development chain that integrates distributed data storage, node-level consensus, and user-driven model selection through Ganache. In the first stage, each user leverages DeSocial to evaluate multiple backbone models on their local subgraph. DeSocial coordinates the execution and returns model-wise prediction results, enabling the user to select the most suitable backbone for personalized social prediction. Then, DeSocial uniformly selects several validation nodes that possess the algorithm specified by each user, and aggregates the prediction results by majority voting, to prevent errors caused by any single model’s misjudgment. Extensive experiments show that DeSocial has an evident improvement compared to the five classical centralized social network learning models, promoting user empowerment in blockchain-based decentralized social networks, showing the importance of multi-node validation and personalized algorithm selection based on blockchain. Our implementation is available at: <https://github.com/agiresearch/DeSocial>.

1 Introduction

Social network learning algorithms have become a central tool for modeling and predicting user behavior in social networks [26, 77, 69, 66, 83], enhancing content search, advertising and recommendation, and improve user experience across multiple platforms [91, 6, 68, 77, 64, 14, 59]. Despite impressive advances in graph-based recommendations [67, 16, 75, 78, 84, 72, 57, 58], the main applications are still deeply rooted in the Web 2.0 paradigm of centralized platforms with exclusive control over user data and model deployment. The platform itself selects the predictive algorithms, trains the models, and provides recommendations to users without transparency or user involvement [17, 12]. As a result, users are forced to accept predictions derived from a fixed model pipeline, regardless of whether the prediction fits their personal preferences or the structural context. This model limits personalization and does not reflect the diversity of the user’s local environments, especially in large graphs. Moreover, relying on a single model limits expressive power, even if that model is state-of-the-art. These challenges point to the need for frameworks that allow users to play a more active role in the algorithm selection and prediction process, especially in decentralized environments that effectively support model diversity and personalization.

* Author Emails: {chy.huang, xi.zhu, minghao.guo, yongfeng.zhang}@rutgers.edu

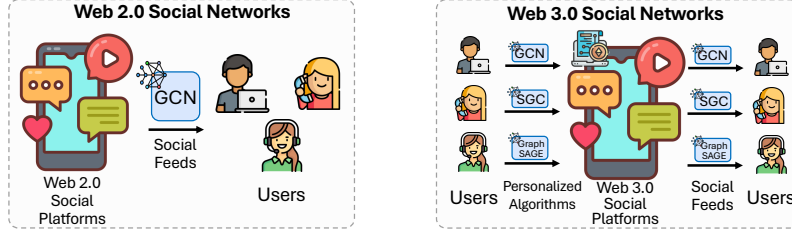


Figure 1: The differences between Web 2.0 social networks and Web 3.0 social networks. In Web 2.0, users passively receive social feeds. In Web 3.0, users receive feeds via personalized algorithms.

By contrast, blockchain as a Web 3.0 technology is a promising alternative [19, 42, 63, 76, 2]. With their emphasis on decentralization [40, 19], transparency [93, 29] and verifiable interactions [47, 30], blockchain-based systems offer a new paradigm for social, transactional and economic behavior that is inherently more user-centric and transparent [56, 62, 31, 41]. Despite their rapid development, blockchain systems are not yet fully integrated with modern artificial intelligence approaches, especially in social network applications, where such synergies could be most impactful [19, 56]. Therefore, there is a significant gap in applying graph learning algorithms to decentralized social prediction tasks, which provides new capabilities for building more personalized, user-driven, and decentralized social network systems.

To overcome this gap, we propose DeSocial framework that allows users to regain control over the forecasting process on their own behalf. In DeSocial, each user has their rights to select the most favorable prediction model from a library of graph learning networks (e.g., MLP [51], GCN [24], GAT [55], GraphSAGE [13], or SGC [64]). This selection is based on neighborhood sampling evaluation to ensure that the selected model best captures the user’s local environment. To ensure reliability, DeSocial applies a smart contract that is transparent to every nodes: a group of validators are selected to run their models independently, evaluate the link prediction query, and make the decision together by a majority vote. This approach not only increases the robustness of the predictions, but also removes single-point algorithmic control, aligning outcomes with both user intent and system integrity. Our contributions can be summarized as follows:

- **Problem Formulation.** We propose a novel task setting where link predictions emerge a decentralized consensus among validators rather than computed by a central model. Each validator runs their own graph learning backbone and a majority vote mechanism determines the final prediction. This formulation captures realistic constraints in blockchain environments, such as data locality, validator trust boundaries, and transparent observability.
- **Novel Framework.** We propose DeSocial, a novel framework that integrates blockchain infrastructure with graph learning for decentralized social network prediction. It enables personalized model selection, user-driven validator community formation, and majority-vote consensus, aligning with the logic of real-world blockchain protocols while improving social network predictions. DeSocial is deployed on an ETH local development chain environment.
- **Extensive Evaluations.** We conduct comprehensive experiments on four representative graph datasets spanning the domains of Web 3.0 transaction networks, email communication graphs, and interest-based social networks. Our results show that DeSocial outperforms all five classic centralized baselines in terms of link prediction accuracy, demonstrating the superiority of decentralized graph learning algorithms in the blockchain context.

2 Related Works

2.1 Graph Learning for Social Networks

Existing graph learning methods for social networks are advanced, including graph neural networks (GNN) [24, 55, 13, 64, 71, 48, 14, 26, 52, 43, 59], meta-learning frameworks [91, 6, 83, 77, 57], Transformer-based [66, 65, 82], and large language model (LLM)-based [34, 53, 4, 60, 80, 15, 85, 92], most of which are designed for centralized environments, where the full graph structure and node features are aggregated on a single server and optimized jointly. In contrast, our setting assumes a completely decentralized environment where each user node has full access to the local view of the

social network structure, but raw data cannot be shared across nodes [87, 45, 37, 46]. Furthermore, the storage and computational capabilities of each node are limited, making it infeasible to deploy large-scale models such as LLM-based encoders [28, 53, 20, 49] or even invoke external LLM APIs [73, 33, 36, 74, 89, 86, 22, 73, 23, 21]. This practical limitation leads us to limit each node to implementing a lightweight graph foundation. Therefore, we selected five representative and well-studied models: MLP [51], GCN [24], GAT [55], GraphSAGE [13], and SGC [64], as candidate backbones for our framework. These backbones achieve a practical balance between representational power and computational efficiency, which is sufficient to test the effectiveness of model selection and consensus in decentralized graph learning.

2.2 Blockchain Consensus Mechanism

Blockchain consensus mechanisms enable distributed agreement without central control [39, 42, 63]. The original Bitcoin blockchain relied on Proof-of-Work (PoW) [42] consensus, where the longest chain determines the accepted state. While PoW provides an open network, it is energy intensive and only provides probabilistic finiteness. More relevant to our context are the consensus protocols Proof-of-Stake (PoS) [63, 76, 2], Proof-of-Authority [1, 2], and Byzantine Fault Tolerance (BFT) [27, 35, 54], which achieve finality by voting among selected validators. However, these methods do not achieve complete decentralization [9, 18], or a large number of validators are selected to participate in the verification, which greatly reduces the operation speed of the network [81]. These concerns of scalability, centralization, and running efficiency bring challenges to blockchain-based decentralized social network algorithm design. Therefore, a limited set of validators are selected to participate in the consensus for each verification in DeSocial, which guarantees the efficiency, robustness, and complete decentralization of the network.

2.3 Ensemble Learning and Majority Voting

Ensemble learning that combines multiple models to improve predictions is well-recognized in machine learning [7, 3, 11, 38, 50]. In graph learning, ensemble methods help mitigate overfitting to specific topologies or noisy subgraphs, especially when models are trained on different views of the graph or initialized with different parameters [8, 90, 5]. Voting is essential in blockchain consensus process because it enables nodes to collectively agree on the validity of transactions [70]. Similarly, voting serves as a mechanism for aggregating decisions from multiple nodes in decentralized graph learning, ensuring that predictions are agreed upon through collective decision-making rather than centralized authority. Common voting strategies includes soft voting [61, 79], where models make probabilistic predictions and use averages, and hard voting [10, 61, 79], where each model casts a binary vote and the final outcome is the majority. DeSocial adopts hard voting since it aligns naturally with blockchain consensus mechanisms, where validations are made through majority approval. Voting-based decisions are also more robust and less vulnerable to manipulation.

3 Problem Definition

We formally define the decentralized temporal link prediction task on social graphs. We aim to fill the gap between centralized graph learning and blockchain-based decentralization. Every node has access to the complete structured information, since in blockchain, the formation of social relations corresponds to the successful validation of a transaction and its subsequent broadcasting to all nodes.

Definition 3.1 (Temporal Graph). A temporal graph can be formally introduced as $\mathcal{G}^t = (\mathcal{V}^t, \mathcal{E}^t)$ where $\mathcal{V}^t, \mathcal{E}^t$ denotes a set of N nodes and a set of directed edges at time t , respectively.

Definition 3.2 (Node-Specific Backbone). We denote the backbone used by node u , as $f_{\Theta_u} \in \mathcal{F}$ where u is a node and Θ_u is the parameters of its backbone, $\mathcal{F} = \{\mathcal{F}_1, \mathcal{F}_2, \dots\}$ is the backbone pool.

Definition 3.3 (Vote). A vote is the boolean decision made by a validator ϕ , denoted as $\text{Vote}(\phi, p, q, t) \in \{0, 1\}$, indicating whether ϕ agree with the link $(p, q) \in \mathcal{G}^t$.

Definition 3.4 (Verification). A verification by a group of validators $\Phi_{p,q,t}$ for validating the link (p, q) at time t , can be formally defined as:

$$\text{Ver}(\Phi_{p,q,t}, p, q, t) = \text{Majority}(\text{Vote}(\phi, p, q, t)), \phi \in \Phi_{p,q,t} \quad (1)$$

where $\text{Majority}(\cdot)$ is the majority of a list of decisions.

The goal of our task is to predict whether a future link (u, v) , $u, v \in \mathcal{V}$, will be formed in \mathcal{G}^{t+1} . \mathcal{V} denotes the full node set. Unlike classical settings where the entire graph and model are centrally managed, we study this task in a **decentralized, blockchain-based environment**. In this environment, the link prediction of (u, v) is regarded as a transaction verification requested by u , targeted at v .

At time t , each validator ϕ uses its backbone model f_{Θ_ϕ} with parameter Θ_ϕ , as well as its temporal graph data \mathcal{G}^t , to compute model output vectors of the initiator \mathbf{z}_p^T and the target \mathbf{z}_q . We also sample a negative target set $Neg(\phi, p, q, t)$ to indicate false links (p, q') where $q' \in Neg(\phi, p, q, t)$, to simulate multiple spendings in blockchain, for the comparison of prediction probability.

The decision can be interpreted as the comparisons among the cosine similarity of \mathbf{z}_p^T and \mathbf{z}_r , $r \in \{q\} \cup Neg(\phi, p, q, t)$. Therefore, the decision can be formally defined as:

$$Vote(\phi, p, q, t) = \mathbb{I}(\bigcap_{q' \in Neg(\phi, p, q, t)} \sigma(\frac{\mathbf{z}_p^T \mathbf{z}_q}{\|\mathbf{z}_p^T\| \|\mathbf{z}_q\|}; f_{\Theta_\phi}(\mathcal{D}^t)) > \sigma(\frac{\mathbf{z}_p^T \mathbf{z}_{q'}}{\|\mathbf{z}_p^T\| \|\mathbf{z}_{q'}\|}; f_{\Theta_\phi}(\mathcal{D}^t))) \quad (2)$$

where σ is the sigmoid function. When a decentralized network receives a verification request (u, v) at time t , the network selects a set of validators Φ . Each validator $\phi_i \in \Phi$ makes a decision $Vote(\phi_i, u, v, t)$ based on its backbone model f_{Θ_u} and temporal graph data $\mathcal{D}^t = \bigcup_{\tau=0}^t \mathcal{G}^\tau$.

Definition 3.5 (Decentralized Learning on Temporal Graphs). *Decentralized learning involves all validator nodes at current time period collectively participating in the prediction of the graph structure for the next time step, is formulated as two steps:*

$$\{\Theta_u^{t-1} | u \in \mathcal{V}_{val}^t\} \xrightarrow{\text{test}} \mathcal{G}^t \quad (3)$$

$$\min\{\text{Loss}(\hat{\mathcal{G}}^t, \mathcal{G}^t; \Theta_u^t)\} \text{ for each } u \in \mathcal{V}_{val}^t, \mathcal{V}_{val}^t = \bigcup_{(p,q) \in \mathcal{G}^t} \Phi_{p,q,t} \quad (4)$$

Instead of optimizing a centralized model Θ^t and test the next period of graph \mathcal{G}^t , decentralization utilizes model parameters of all nodes to test \mathcal{G}^t , and all validator nodes $u \in \mathcal{V}_{val}^t$ in current period t optimize their model parameters Θ_u^t independently. This problem has three unique challenges:

- **No global optimization:** There is no centralized end-to-end training because of the distributed storage of data and backbone models.
- **No shared parameters:** Validators may use different models with different inductive biases.
- **Consensus under variance:** Final decisions must tolerate noise, diversity among validators.

4 Our Framework

4.1 Framework Overview

Our framework DeSocial comprises two modules: (1) a personalized algorithm selection mechanism that allows users to select their algorithms for their prediction requests, and (2) a decentralized consensus voting scheme among validators of the chosen backbone to produce the final output. We also describe the architecture of DeSocial that orchestrates these modules on a blockchain platform.

Figure 2 illustrates the personalized algorithm selection. A user p requests the blockchain to that it wants to select an algorithm from the pool \mathcal{F} . In the meantime, p forms a historical neighbor validation exam, sampling a set of historical neighbors from $\mathcal{G}^0, \mathcal{G}^1, \dots, \mathcal{G}^t$. The blockchain then asks nodes $r_1, r_2, \dots, r_{|\mathcal{F}|}$ with each backbone in \mathcal{F} to take this exam, i.e., predicting the links between p . Then each r_i returns its exam result to p via blockchain. Finally, p selects the algorithm with the best exam result, as its personalized algorithm \mathcal{F}_p .

Figure 3 illustrates the decentralized consensus voting scheme. Each user $p_i, (p_i, q_i) \in \mathcal{G}^{t+1}$ first sends a request to the blockchain to validate q_i . After that, the blockchain uniformly samples n nodes, all of which applied \mathcal{F}_{p_i} , to build a validation community $\Phi_{p_i, q_i, t}$. Each $\phi_j \in \Phi_{p_i, q_i, t}$ constructs a prediction task by sampling different negative edges $(p_i, q'_i), q'_i \in Neg(\phi_j, p_i, q_i, t)$, using its model $f_{\Theta_{\phi_j}}$ to predict the edges, and voting the one with highest probability. (p_i, q_i) is predicted as true if more than half of the validators in $\Phi_{p_i, q_i, t}$ agrees it.

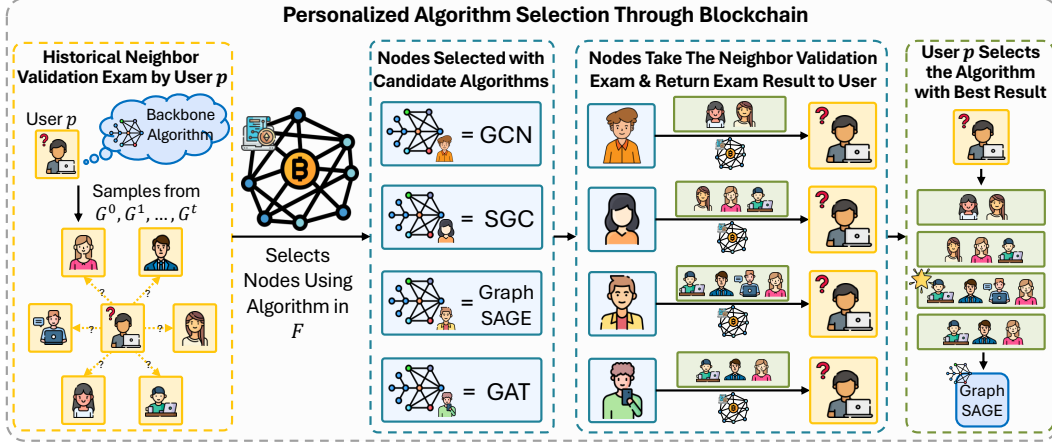


Figure 2: The personalized algorithm selection module allows user p to choose an algorithm from \mathcal{F} . In this case, $\mathcal{F} = \{\text{GCN}, \text{GAT}, \text{GraphSAGE}, \text{GAT}\}$.

4.2 Personalized Backbone Algorithm Selection

To illustrate that each user can independently select a social network algorithm, our framework enables users to make use of their local structural information into Algorithm 1. Given a set of backbone algorithm \mathcal{F} , DeSocial allows each node $u \in \mathcal{V}$ selects an algorithm $\mathcal{F}_u \in \mathcal{F}$ for each u .

Algorithm 1: Personalized Algorithm Selection

Input: Backbone pool set \mathcal{F} , neighborhood set size γ , adjust coefficient α , and graph data \mathcal{D}^t

Output: The personalized algorithm list $\{\mathcal{F}_u | (u, v) \in \mathcal{G}^{t+1}\}$

```

for  $u \in \{u | (u, v) \in \mathcal{G}^{t+1}\}$  do
  Calculate  $\Gamma$  by Eq. 5;
  for  $(v_p, v_n) \in \Gamma$  do
    Calculate  $\Pi_{u,v_p}$  given  $t_e$  and  $\alpha$ ;
  end
   $u$  requests the blockchain by the smart contract and finds  $r_1, r_2, \dots, r_{|\mathcal{F}|}$ ;
  for  $i \in \{1, 2, \dots, |\mathcal{F}|\}$  do
    for  $(v_p, v_n) \in \Gamma$  do
       $r_i$  calculates  $\mathbf{z}_u, \mathbf{z}_{v_p}, \mathbf{z}_{v_n}$  given  $f_{\Theta_{r_i}}(\mathcal{D}^t)$ ;
       $r_i$  calculates the probability of  $(u, v_p)$  and  $(u, v_n)$ ;
    end
     $r_i$  returns the weighted sum of the probability comparison via blockchain by the smart contract;
  end
   $u$  selects  $\mathcal{F}_u$  by Eq. 6;
end
return  $\{\mathcal{F}_u | (u, v) \in \mathcal{G}^{t+1}\}$ 

```

Specifically speaking, at time t , given a node u , we sample its positive and negative neighbor pairs

$$\Gamma = \{(v_p, v_n) | \left(\bigcup_{\tau=0}^t v_p \in \mathcal{N}^t(u) \right) \wedge \left(\bigcup_{\tau=0}^t v_n \notin \mathcal{N}^t(u) \right)\} \quad (5)$$

with a size of γ . We select \mathcal{F}_u through Eq. 6, where $\Pi_{u,v_p} = \exp(\alpha * (t - t_e))$, denoting the edge weights for algorithm selection. The edges that emerges later has greater weights. t_e is the emerge time of (u, v) , and α is the adjust coefficient. By leveraging the local subgraph and blockchain, users can choose models that best fit their individual network context. This selection improves validation success rates and enhances the overall robustness and performance of the social network prediction.

$$\mathcal{F}_u = \arg \max_{f \in \mathcal{F}} \sum_{(v_p, v_n) \in \Gamma} \mathbb{I} \left(\sigma \left(\frac{\mathbf{z}_u^T \mathbf{z}_{v_p}}{\|\mathbf{z}_u^T\| \|\mathbf{z}_{v_p}\|}; f(\mathcal{D}^t) \right) > \sigma \left(\frac{\mathbf{z}_u^T \mathbf{z}_{v_n}}{\|\mathbf{z}_u^T\| \|\mathbf{z}_{v_n}\|}; f(\mathcal{D}^t) \right) \right) \Pi_{u,v_p} \quad (6)$$

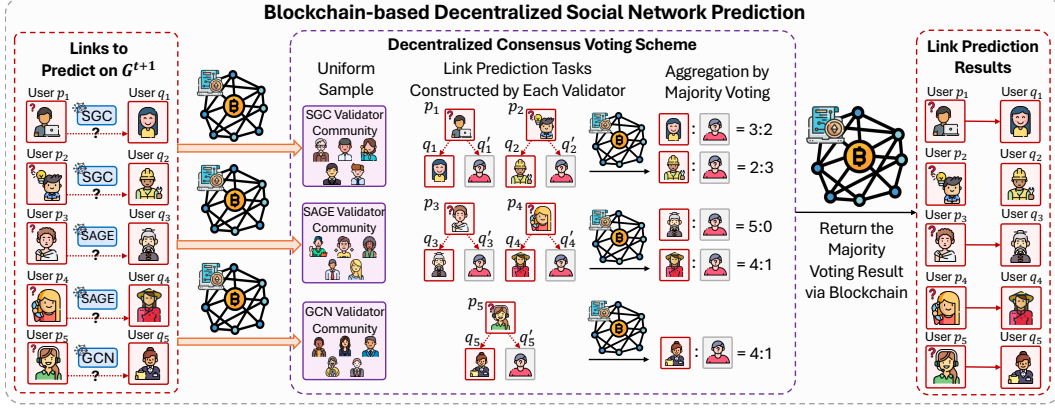


Figure 3: Decentralized consensus voting in DeSocial. To predict \mathcal{G}^{t+1} , users p_i request to predict links with q_i using personalized algorithms. The blockchain samples $n=5$ validators with matching algorithms to predict and finalizes the result via majority voting through a smart contract.

4.3 Decentralized Consensus Voting Scheme

In decentralized prediction, the prediction of a single validator may be affected by local noise or model variance. With multiple validators, some errors made by a single validator can be tolerated through the majority voting mechanism. DeSocial conducts a consensus process between validators in the communities by Algorithm 2, incorporating with a smart contract deployed in the blockchain.

Algorithm 2: Blockchain-based Decentralized Social Network Learning

Input: Temporal graph data \mathcal{D}^t , validator set size n , randomly initialized parameters of each node Θ_u , backbone pool \mathcal{F}

Output: Model parameters $\{\Theta_u^* | u \in \mathcal{V}\}$

for time period $t \in \{0, 1, \dots, T-1\}$ **do**

 # Validation committee sampling

 Calculate $\Phi_{p,q,t+1}, \forall (p, q) \in \mathcal{G}^{t+1}$ by Eq. 7 by requesting the blockchain by the smart contract;

 Calculate \mathcal{V}_{val}^t by union all the $\Phi_{p,q,t+1}$;

 # Local inference

while $\exists u \in \mathcal{V}_{val}^t, \Theta_u$ does not converge **do**

for $u \in \mathcal{V}_{val}^t$ **do**

 Predict the future graph $\hat{\mathcal{G}}^{t+1}$ given Θ_u and \mathcal{D}^t ;

 Calculate the prediction loss $\mathcal{L}(\hat{\mathcal{G}}^{t+1}, \mathcal{G}^{t+1})$;

 Optimize Θ_u given $\mathcal{L}(\hat{\mathcal{G}}^{t+1}, \mathcal{G}^{t+1})$;

end

end

 # Aggregation decisions

for $(p, q) \in \mathcal{G}^{t+1}$ **do**

for $u \in \Phi_{p,q,t+1}$ **do**

u sends $Vote(\Phi_{p,q,t+1}, p, q, t+1)$ to blockchain by the smart contract;

end

 The smart contract calculates $Ver(\Phi_{p,q,t+1}, p, q, t+1)$ by Eq. 1 ;

end

end

$\forall u \in \mathcal{V}, \Theta_u^* \leftarrow \Theta_u$;

return $\{\Theta_u^* | u \in \mathcal{V}\}$;

Validation Committee Sampling: To predict a social connection (p_i, q_i) at time $t+1$, the blockchain selects n nodes using \mathcal{F}_{p_i} and form a validation committee $\Phi_{p_i, q_i, t+1}$, as defined in Eq. 7 by the smart contract, where $\mathcal{V}_{\mathcal{F}_{p_i}}$ denotes the nodes using \mathcal{F}_{p_i} . $\Phi_{p_i, q_i, t+1}$ is fixed at t and specified by \mathcal{F}_{p_i} .

$$\Phi_{p_i, q_i, t+1} \sim \text{UniformSample}(\mathcal{V}_{\mathcal{F}_{p_i}}, n) \quad (7)$$

Local Inference: Each selected validator $\phi_j \in \Phi_{p_i, q_i, t+1}$ copies \mathcal{D}^t to its local memory to run \mathcal{F}_{ϕ_j} locally and independently. Each ϕ_j predicts the edges (p_i, q_i) and $(p_i, q'_i), q'_i \in Neg(\phi, p_i, q_i, t)$, making a binary decision $Vote(\phi_j, p_i, q_i, t+1)$ through Eq. 2. Each ϕ_j then sends its vote to the blockchain by the smart contract.

Aggregation: The smart contract initiates a roll call procedure that collects the individual validation outcomes from the selected committee members $\phi_1, \phi_2, \dots, \phi_n$. Then, it makes a summation of $Vote(\phi_j, p_i, q_i, t+1)$. As shown in Eq. 1, the smart contract returns a positive decision if more than half of the committee agrees, i.e., $\sum_{i=1}^n Vote(\phi_j, p_i, q_i, t+1) > \lfloor \frac{n}{2} \rfloor$.

5 Experiments

5.1 Experimental Setups

We use four real-world temporal graph datasets in the scope of Web 2.0 (**UCI** [44], **Enron** [88], and **GDELT** [88]) and Web 3.0 (**Memo-Tx** [94]). In order to evaluate the performance of DeSocial, we used the five most classic centralized models (**MLP** [51], **GCN** [24], **GAT** [55], **GraphSAGE** [13], and **SGC** [64]) in the field of social network learning as baselines for comparison. Detailed information for the datasets and baselines is described in Appendix A.1 and Appendix A.2, respectively.

5.2 Evaluation Metrics

Given the decentralized nature of our problem formulation, we employ evaluation metrics tailored to assess the effectiveness of DeSocial in decentralized environments, enlightened by the validation of the double spending problem on blockchain [32, 25]. That is, they lack access to the underlying intent of a transaction and can only judge based on structural context. This motivates our use of **Acc@K**, a set of relative metrics that evaluates whether the predicted positive link ranks higher than its sampled alternatives.

Given a link prediction task that each test case consists of one positive edge and $K - 1$ negative edges, **Acc@K** indicates the probability that the model assigns the highest score to the positive edge among the K candidates. We adopt $K \in \{2, 3, 5\}$ to evaluate the performance of decentralized graph learning. The greater K is, the harder evaluation tasks will be. We follow [88] for the randomized negative sampling method. Appendix A.3 provides our experiment setups and implementation details.

5.3 Impacts of the Personalized Algorithms

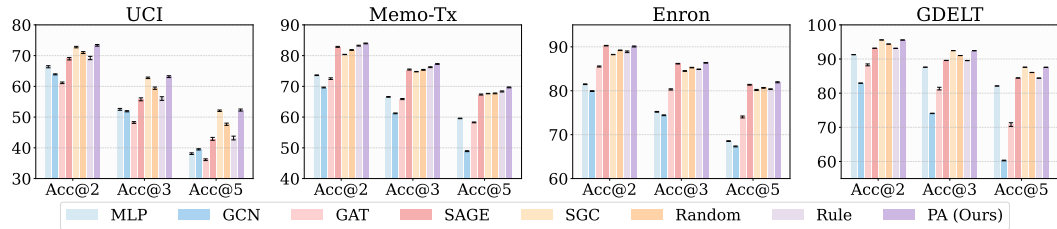


Figure 4: Comparison of the performance among different centralized methods, random selection, simple rule-based selection, and DeSocial PA on Acc@2, Acc@3, Acc@5 for each dataset.

Figure 4 reports the performance of five centralized baselines, random selection, simple rule-based selection, and our personalized algorithm selection method DeSocial PA across multiple datasets and evaluation metrics. In hybrid settings, each validator uses a distinct backbone. Random selection represents that every node selects an algorithm in \mathcal{F} . Simple rule-based selection represents that every node selects an algorithm in \mathcal{F} based on its two features (degree and clustering) of the local structure. Table 1 and Table 2 shows the detailed statistics of the centralized baselines and DeSocial PA . Appendix A.5 shows the simple rule of algorithm selection.

The results highlight the importance of allowing users to select personalized models in decentralized settings. Compared to two simpler hybrid baselines, random selection and rule-based selection, DeSocial PA consistently delivers higher performance across most datasets and evaluation metrics. Specifically, DeSocial PA outperforms all centralized baselines in all three metrics for UCI and

Memo-Tx, and shows gains in Enron for Acc@3 and Acc@5 (average gain 1.18% against the strongest centralized baseline). Even in GDELT and Acc@2 on Enron, where performance is already saturated, DeSocial_{PA} achieves comparable results without degradation (within 0.25%), demonstrating its robustness. These results realize the motivation behind DeSocial_{PA}: allowing each user to select the most suitable model based on their local context is more effective than relying on random or hand-crafted strategies and thus enhances the overall prediction.

Table 1: Mean and standard deviation (%) of Acc@2 and Acc@3 in the centralized and decentralized settings. The purple boxes, DeSocial_{PA}, and DeSocial_{Full} represents multi-validator consensus, personalized algorithm selection, and both, respectively. The best centralized scores are underlined; improvements from decentralized methods are in bold. Gain measures the performance gap between the best decentralized and best centralized methods.

Model	Metric	UCI	Memo-Tx	Enron	GDELT
MLP	Acc@2	66.38 ± 0.34	73.61 ± 0.11	81.48 ± 0.08	91.28 ± 0.02
	Acc@3	52.52 ± 0.30	66.57 ± 0.11	75.20 ± 0.09	87.61 ± 0.02
DeSocial _{MLP}	Acc@2	71.03 ± 0.67	74.83 ± 0.14	83.18 ± 0.10	94.04 ± 0.03
	Acc@3	53.94 ± 0.58	67.11 ± 0.14	75.97 ± 0.10	91.54 ± 0.04
GCN	Acc@2	63.90 ± 0.17	69.62 ± 0.13	79.92 ± 0.09	82.94 ± 0.04
	Acc@3	51.90 ± 0.19	61.21 ± 0.13	74.41 ± 0.09	74.08 ± 0.06
DeSocial _{GCN}	Acc@2	66.89 ± 0.47	75.03 ± 0.18	81.95 ± 0.18	90.57 ± 0.03
	Acc@3	53.54 ± 0.43	69.17 ± 0.15	78.57 ± 0.15	84.96 ± 0.07
GAT	Acc@2	61.15 ± 0.26	72.51 ± 0.26	85.52 ± 0.12	88.29 ± 0.28
	Acc@3	48.24 ± 0.28	65.86 ± 0.18	80.30 ± 0.14	81.34 ± 0.39
DeSocial _{GAT}	Acc@2	63.79 ± 0.46	73.42 ± 0.32	87.51 ± 0.13	94.09 ± 0.20
	Acc@3	48.01 ± 0.52	68.28 ± 0.21	84.29 ± 0.12	89.52 ± 0.21
SAGE	Acc@2	69.00 ± 0.40	82.85 ± 0.15	90.27 ± 0.06	93.16 ± 0.02
	Acc@3	55.78 ± 0.48	75.47 ± 0.16	86.16 ± 0.07	89.59 ± 0.03
DeSocial _{SAGE}	Acc@2	73.31 ± 0.65	85.84 ± 0.21	92.18 ± 0.08	95.68 ± 0.03
	Acc@3	56.22 ± 0.72	76.91 ± 0.21	88.17 ± 0.13	93.14 ± 0.04
SGC	Acc@2	72.77 ± 0.24	80.37 ± 0.05	88.24 ± 0.04	95.59 ± 0.02
	Acc@3	62.77 ± 0.24	74.78 ± 0.07	84.50 ± 0.08	92.46 ± 0.02
DeSocial _{SGC}	Acc@2	76.37 ± 0.44	83.16 ± 0.07	89.62 ± 0.06	98.12 ± 0.02
	Acc@3	65.62 ± 0.39	79.19 ± 0.09	86.13 ± 0.09	96.11 ± 0.03
DeSocial _{PA}	Acc@2	73.35 ± 0.27	83.96 ± 0.13	90.08 ± 0.11	95.57 ± 0.02
	Acc@3	63.16 ± 0.30	77.28 ± 0.12	86.36 ± 0.08	92.44 ± 0.02
DeSocial _{Full}	Acc@2	77.63 ± 0.36	87.25 ± 0.17	92.11 ± 0.13	98.13 ± 0.03
	Acc@3	66.01 ± 0.44	79.65 ± 0.16	88.39 ± 0.13	96.09 ± 0.04
Gain(%)	Acc@2	6.68	5.31	2.12	2.66
	Acc@3	5.16	5.54	2.59	3.95

5.4 Impacts of the Multiple Validators

We first examine Acc@2 and Acc@3 to evaluate the effect of decentralized consensus voting. As shown in Table 1, introducing a 5-validator committee under a single backbone consistently improves performance across most datasets, with average gains of 3.36% in Acc@2 and 3.18% in Acc@3. The only notable drop occurs for GAT on UCI on Acc@3, where the baseline performs below random (i.e., below 0.5), making aggregation more likely to amplify errors. In the more difficult Acc@5 task shown in Table 2, although most backbones still benefit from ensemble voting, four centralized models on UCI fall below 0.5 and further degrade after aggregation, due to the amplification of error. Despite this, consistent improvements on other datasets highlight the utility of decentralized voting in reducing variance and correcting individual prediction noise.

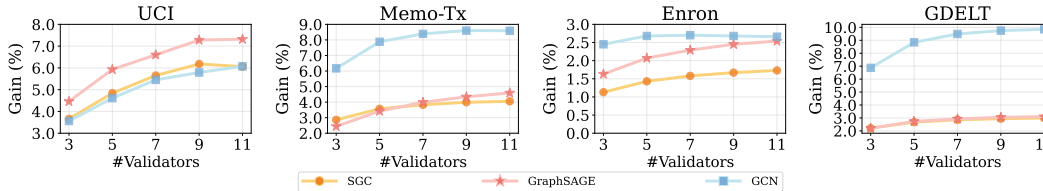


Figure 5: Gains versus number of validators. We vary the validator committee size $n \in \{3, 5, 7, 9, 11\}$ and report the corresponding gain in prediction accuracy. Gains converges as n increases to 9.

We further analyze how the performance gain from multiple node consensus over a single centralized backbone varies with the number of validators, from 3 to 11, as shown in Figure 5. As the committee size increases, the marginal improvement diminishes, and the gain converges around nine validators.

DeSocial_{Full} integrates both modules, and further gained performance on most of the tasks. This suggests that personalized algorithm selection can effectively complement decentralized voting in improving prediction accuracy. On the dataset GDELT, however, the benefit is less pronounced due to the high overall accuracy of individual models and the relatively homogeneous nature of the graph.

Table 2: Mean and standard deviation (%) of Acc@5 in the centralized and decentralized settings.

Model	Metric	UCI	Memo-Tx	Enron	GDELT
MLP	Acc@5	38.10 \pm 0.27	59.57 \pm 0.09	68.53 \pm 0.09	82.11 \pm 0.03
DeSocial _{MLP}	Acc@5	37.03 \pm 0.46	60.37 \pm 0.10	69.96 \pm 0.10	86.81 \pm 0.05
GCN	Acc@5	39.51 \pm 0.20	48.91 \pm 0.15	67.32 \pm 0.10	60.29 \pm 0.10
DeSocial _{GCN}	Acc@5	39.47 \pm 0.50	54.73 \pm 0.21	74.10 \pm 0.11	68.25 \pm 0.15
GAT	Acc@5	36.13 \pm 0.28	58.27 \pm 0.13	74.03 \pm 0.22	70.81 \pm 0.54
DeSocial _{GAT}	Acc@5	34.34 \pm 0.43	62.39 \pm 0.15	80.99 \pm 0.20	79.69 \pm 0.36
SAGE	Acc@5	42.92 \pm 0.47	67.34 \pm 0.16	81.36 \pm 0.09	84.45 \pm 0.03
DeSocial _{SAGE}	Acc@5	41.36 \pm 0.59	67.58 \pm 0.21	83.38 \pm 0.15	88.76 \pm 0.05
SGC	Acc@5	52.06 \pm 0.24	67.65 \pm 0.07	80.18 \pm 0.12	87.60 \pm 0.03
DeSocial _{SGC}	Acc@5	53.91 \pm 0.40	72.92 \pm 0.12	82.14 \pm 0.10	92.15 \pm 0.04
DeSocial _{PA}	Acc@5	52.29 \pm 0.33	69.66 \pm 0.14	81.93 \pm 0.12	87.58 \pm 0.04
DeSocial _{Full}	Acc@5	54.14 \pm 0.49	72.72 \pm 0.14	84.28 \pm 0.15	92.12 \pm 0.05
Gain(%)	Acc@5	3.80	7.79	3.59	5.19

5.5 Efficiency Analysis

Table 3: Efficiency analysis of the run time (s) of the whole process at one test period (100 epochs) at UCI and Memo-Tx, comparing decentralized methods with centralized ones. Rounded for 4 sig. figs.

Dataset	Centralized					Decentralized (Amortized)		
	MLP	GCN	GAT	SAGE	SGC	DeSocial _{PA}	DeSocial _{Vote}	DeSocial _{Full}
UCI	44.08	54.48	140.5	51.79	46.43	54.89	47.19	55.28
Memo-Tx	4257	2402	20750	2011	439.8	2016	2021	2021

To evaluate the runtime efficiency of DeSocial, we compare five centralized models with three decentralized variants: personalized algorithm selection alone (DeSocial_{PA}), multiple-validator voting alone (DeSocial_{Vote}, referring to the DeSocial_F with highest metrics. i.e., DeSocial_{SGC} for UCI, and DeSocial_{SAGE} for Memo-Tx), and both configuration (DeSocial_{Full}). All decentralized variants account for the additional computational overhead introduced by the ETH Ganache infrastructure, which serves as a local blockchain emulator to simulate on-chain operations such as user requests, validator selections, validator votings, and decision aggregations. All graph learning processes are executed on an off-chain local server to simulate DeSocial in a single-machine environment, with each graph model stored independently.

We use Ganache to capture blockchain-side latency and execution cost in decentralized social prediction processes, avoiding the complexity of deploying to a full public ETH testnet. For decentralized methods, runtime is measured from a single user’s perspective. That is, starting from a social link request and ending to the decision aggregated. Appendix B shows the details of calculating the run time of each decentralized method.

As shown in Table 3, the amortized runtime of DeSocial is not significantly impacted by the integration of the ETH local development chain. Our experiments are conducted in a single machine simulation where both graph training and blockchain interactions are executed serially. Appendix A.3 shows the details of the computation device. In a real ETH network deployment with multiple participating users, the system would better exploit parallelism and further improve the efficiency.

6 Conclusions

We extended the prediction of social networks to a decentralized setting and implemented our DeSocial framework with the Web 3.0 blockchain technology. DeSocial leverages the decentralized nature of blockchain to enable user-driven algorithm selection and validator-level majority voting for social prediction. By allowing each user to select the most suitable backbone model based on their local subgraph, and aggregating predictions from multiple independently hosted validators, DeSocial overcomes the limitations of centralized frameworks and consistently improves link prediction performance. Our results highlight the value of personalized model choice and decentralized consensus in building personalized and robust social network algorithm on blockchain infrastructures.

Limitations: Although our framework demonstrates clear improvements in social link prediction, it also highlights several open challenges that present opportunities for future research. First, blockchain testnets are easy to deploy but inefficient, while real chains like ETH offer higher throughput yet require multi-machine coordination, making them difficult to simulate on a single machine. Secondly, future work can apply stronger graph learning backbones and more sophisticated validation methods to further improve social network predictions.

References

- [1] Binance. Proof of authority explained, 2018.
- [2] Binance US. Binance us, 2025.
- [3] Leo Breiman. Bagging predictors. *Machine learning*, 24:123–140, 1996.
- [4] Runjin Chen, Tong Zhao, Ajay Kumar Jaiswal, Neil Shah, and Zhangyang Wang. Llaga: Large language and graph assistant. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024.
- [5] Yen-Liang Chen, Chen-Hsin Hsiao, and Chia-Chi Wu. An ensemble model for link prediction based on graph embedding. *Decision Support Systems*, 157:113753, 2022.
- [6] Sihao Ding, Fuli Feng, Xiangnan He, Yong Liao, Jun Shi, and Yongdong Zhang. Causal incremental graph convolution for recommender system retraining. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.
- [7] Xibin Dong, Zhiwen Yu, Wenming Cao, Yifan Shi, and Qianli Ma. A survey on ensemble learning. *Frontiers of Computer Science*, 14:241–258, 2020.
- [8] Rui Duan, Chungang Yan, Junli Wang, and Changjun Jiang. Graph ensemble neural network. *Information Fusion*, 110:102461, 2024.
- [9] Shahriar Fahim, S Katibur Rahman, and Sharfuddin Mahmood. Blockchain: A comparative study of consensus algorithms pow, pos, poa, pov. *Int. J. Math. Sci. Comput*, 3(1):46–57, 2023.
- [10] Michael J Franklin, Donald Kossmann, Tim Kraska, Sukriti Ramesh, and Reynold Xin. Crowddb: answering queries with crowdsourcing. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, pages 61–72, 2011.
- [11] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.
- [12] Yingqiang Ge, Shuchang Liu, Zuohui Fu, Juntao Tan, Zelong Li, Shuyuan Xu, Yunqi Li, Yikun Xian, and Yongfeng Zhang. A survey on trustworthy recommender systems. *ACM Transactions on Recommender Systems*, 3(2):1–68, 2024.
- [13] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS’17*, page 1025–1035, 2017.
- [14] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 639–648, 2020.
- [15] Xiaoxin He, Xavier Bresson, Thomas Laurent, Adam Perold, Yann LeCun, and Bryan Hooi. Harnessing explanations: Llm-to-lm interpreter for enhanced text-attributed graph representation learning. *arXiv preprint arXiv:2305.19523*, 2023.
- [16] Liwei Huang, Yutao Ma, Yanbo Liu, Bohong Danny Du, Shuliang Wang, and Deyi Li. Position-enhanced and time-aware graph convolutional network for sequential recommendations. *ACM Transactions on Information Systems*, 41(1):1–32, 2023.
- [17] Dietmar Jannach, Sidra Naveed, and Michael Jugovac. User control in recommender systems: Overview and interaction challenges. In Derek Bridge and Heiner Stuckenschmidt, editors, *E-Commerce and Web Technologies*, pages 21–33, Cham, 2017. Springer International Publishing.
- [18] Seungwon Jeong. Centralized decentralization simple economics of the dpos blockchain governance. *Applied Economics Letters*, pages 1–6, 2024.
- [19] Ujun Jeong, Lynnette Hui Xian Ng, Kathleen M Carley, and Huan Liu. Navigating decentralized online social networks: An overview of technical and societal challenges in architectural choices. *arXiv preprint arXiv:2504.00071*, 2025.

- [20] Bowen Jin, Gang Liu, Chi Han, Meng Jiang, Heng Ji, and Jiawei Han. Large language models on graphs: A comprehensive survey. *IEEE Transactions on Knowledge and Data Engineering*, 36(12):8622–8642, 2024.
- [21] Mingyu Jin, Weidi Luo, Sitao Cheng, Xinyi Wang, Wenyue Hua, Ruixiang Tang, William Yang Wang, and Yongfeng Zhang. Disentangling memory and reasoning ability in large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, 2025.
- [22] Mingyu Jin, Kai Mei, Wujiang Xu, Mingjie Sun, Ruixiang Tang, Mengnan Du, Zirui Liu, and Yongfeng Zhang. Massive values in self-attention modules are the key to contextual knowledge understanding. In *Forty-second International Conference on Machine Learning*, 2025.
- [23] Mingyu Jin, Qinkai Yu, Jingyuan Huang, Qingcheng Zeng, Zhenting Wang, Wenyue Hua, Haiyan Zhao, Kai Mei, Yanda Meng, Kaize Ding, Fan Yang, Mengnan Du, and Yongfeng Zhang. Exploring concept depth: How large language models acquire knowledge and concept at different layers? In Owen Rambow, Leo Wanner, Marianna Apidianaki, Hend Al-Khalifa, Barbara Di Eugenio, and Steven Schockaert, editors, *Proceedings of the 31st International Conference on Computational Linguistics*, pages 558–573, Abu Dhabi, UAE, January 2025. Association for Computational Linguistics.
- [24] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.
- [25] Abhishek Kumar, Bashant Kumar Sah, Tushar Mehrotra, and Gaurav Kumar Rajput. A review on double spending problem in blockchain. In *2023 International Conference on Computational Intelligence and Sustainable Engineering Solutions (CISES)*, pages 881–889. IEEE, 2023.
- [26] Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In *KDD '19*, 2019.
- [27] Aptos Labs. The aptos blockchain: Safe, scalable, and upgradeable web3 infrastructure. https://aptosfoundation.org/whitepaper/aptos-whitepaper_en.pdf, 2022.
- [28] Chaoliu Li, Lianghao Xia, Xubin Ren, Yaowen Ye, Yong Xu, and Chao Huang. Graph transformer for recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '23*, page 1680–1689, New York, NY, USA, 2023. Association for Computing Machinery.
- [29] Zihao Li, Jianfeng Li, Zheyuan He, Xiapu Luo, Ting Wang, Xiaoze Ni, Wenwu Yang, Xi Chen, and Ting Chen. Demystifying defi mev activities in flashbots bundle. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 165–179, 2023.
- [30] Jing Liu and Zhentian Liu. A survey on security verification of blockchain smart contracts. *IEEE access*, 7:77894–77904, 2019.
- [31] Kai Liu, Minghao Yu, Yang Jin, Yue Wang, Jiaqi Yan, and Xiao Fan Liu. Tokenomic model of friend. tech social platform: A data-driven analysis. In *2023 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 656–662. IEEE, 2023.
- [32] Weikang Liu, Bin Cao, and Mugen Peng. Two-tier multi-zone consensus: Enable intelligence sharing for aiot with enhanced security. In *2024 IEEE Annual Congress on Artificial Intelligence of Things (AIoT)*, pages 232–238. IEEE, 2024.
- [33] Xinyi Liu, Ruijie Wang, Dachun Sun, Dilek Hakkani Tur, and Tarek Abdelzaher. Uncovering cross-domain recommendation ability of large language models. In *Companion Proceedings of the ACM on Web Conference 2025, WWW '25*, page 2736–2743, New York, NY, USA, 2025. Association for Computing Machinery.
- [34] Zheyuan Liu, Xiaoxin He, Yijun Tian, and Nitesh V. Chawla. Can we soft prompt llms for graph learning tasks? In Tat-Seng Chua, Chong-Wah Ngo, Roy Ka-Wei Lee, Ravi Kumar, and Hady W. Lauw, editors, *Companion Proceedings of the ACM on Web Conference 2024, WWW 2024, Singapore, Singapore, May 13-17, 2024*, pages 481–484. ACM, 2024.

- [35] David Mazieres. The stellar consensus protocol: A federated model for internet-level consensus. *Stellar Development Foundation*, 2015.
- [36] Kai Mei, Xi Zhu, Wujiang Xu, Wenyue Hua, Mingyu Jin, Zelong Li, Shuyuan Xu, Ruosong Ye, Yingqiang Ge, and Yongfeng Zhang. Aios: Llm agent operating system. *arXiv:2403.16971*, 2024.
- [37] Chuizheng Meng, Sirisha Rambhatla, and Yan Liu. Cross-node federated graph neural network for spatio-temporal data modeling. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 1202–1211, 2021.
- [38] Ibomoiye Domor Mienye and Yanxia Sun. A survey of ensemble learning: Concepts, algorithms, applications, and prospects. *Ieee Access*, 10:99129–99149, 2022.
- [39] Du Mingxiao, Ma Xiaofeng, Zhang Zhe, Wang Xiangwei, and Chen Qijun. A review on consensus algorithm of blockchain. In *2017 IEEE international conference on systems, man, and cybernetics (SMC)*, pages 2567–2572. IEEE, 2017.
- [40] Fatma Mlika, Wafa Karoui, and Lotfi Ben Romdhane. Blockchain solutions for trustworthy decentralization in social networks. *Computer Networks*, page 110336, 2024.
- [41] MyShell AI. Myshell documentation, 2025.
- [42] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Satoshi Nakamoto*, 2008.
- [43] Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao Schardl, and Charles Leiserson. Evolvegn: Evolving graph convolutional networks for dynamic graphs. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 5363–5370, 2020.
- [44] Farimah Poursafaei, Shenyang Huang, Kellin Pelrine, and Reihaneh Rabbany. Towards better evaluation for dynamic link prediction. *Advances in Neural Information Processing Systems*, 35:32928–32941, 2022.
- [45] Tao Qi, Fangzhao Wu, Chuhan Wu, Lingjuan Lyu, Tong Xu, Hao Liao, Zhongliang Yang, Yongfeng Huang, and Xing Xie. Fairvfl: A fair vertical federated learning framework with contrastive adversarial learning. *Advances in neural information processing systems*, 35:7852–7865, 2022.
- [46] Zhen Qin, Xueqiang Yan, Mengchu Zhou, and Shuiguang Deng. Blockdfl: A blockchain-based fully decentralized peer-to-peer federated learning framework. In *Proceedings of the ACM on Web Conference 2024*, pages 2914–2925, 2024.
- [47] Longfei Qiu, Yoonseung Kim, Ji-Yong Shin, Jieung Kim, Wolf Honoré, and Zhong Shao. Lido: Linearizable byzantine distributed objects with refinement-based liveness proofs. *Proceedings of the ACM on Programming Languages*, 8(PLDI):1140–1164, 2024.
- [48] Ladislav Rampášek, Michael Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. *Advances in Neural Information Processing Systems*, 35:14501–14515, 2022.
- [49] Xubin Ren, Jiabin Tang, Dawei Yin, Nitesh Chawla, and Chao Huang. A survey of large language models for graphs. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD '24*, page 6616–6626, New York, NY, USA, 2024. Association for Computing Machinery.
- [50] Yuji Roh, Qingyun Liu, Huan Gui, Zhe Yuan, Yujin Tang, Steven Euijong Whang, Liang Liu, Shuchao Bi, Lichan Hong, Ed H Chi, et al. Levi: generalizable fine-tuning via layer-wise ensemble of different views. *arXiv preprint arXiv:2402.04644*, 2024.
- [51] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

- [52] Aravind Sankar, Yanhong Wu, Liang Gou, Wei Zhang, and Hao Yang. Dysat: Deep neural representation learning on dynamic graphs via self-attention networks. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 519–527, 2020.
- [53] Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin, and Chao Huang. Graphgpt: Graph instruction tuning for large language models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 491–500, New York, NY, USA, 2024. Association for Computing Machinery.
- [54] The MystenLabs Team. The sui smart contracts platform. <https://github.com/MystenLabs/sui/blob/main/doc/paper/sui.pdf>, 2023.
- [55] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.
- [56] Shicheng Wan, Hong Lin, Wensheng Gan, Jiahui Chen, and Philip S Yu. Web3: The next internet revolution. *IEEE Internet of Things Journal*, 11(21):34811–34825, 2024.
- [57] Ruijie Wang, Jingyuan Huang, Yutong Zhang, Jinyang Li, Yufeng Wang, Wanyu Zhao, Shengzhong Liu, Charith Mendis, and Tarek Abdelzaher. Tgonline: Enhancing temporal graph learning with adaptive online meta-learning. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1659–1669, 2024.
- [58] Ruijie Wang, Zheng Li, Danqing Zhang, Qingyu Yin, Tong Zhao, Bing Yin, and Tarek Abdelzaher. Rete: Retrieval-enhanced temporal event forecasting on unified query product evolutionary graph. In *Proceedings of the ACM Web Conference 2022*, pages 462–472, 2022.
- [59] Yanbang Wang, Yen-Yu Chang, Yunyu Liu, Jure Leskovec, and Pan Li. Inductive representation learning in temporal networks via causal anonymous walks. *arXiv preprint arXiv:2101.05974*, 2021.
- [60] Wei Wei, Xubin Ren, Jiabin Tang, Qinyong Wang, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. Llmrec: Large language models with graph augmentation for recommendation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, WSDM '24, page 806–815, New York, NY, USA, 2024. Association for Computing Machinery.
- [61] Liangliang Wen, Jiye Liang, Kaixuan Yao, and Zhiqiang Wang. Black-box adversarial attack on graph neural networks with node voting mechanism. *IEEE Transactions on Knowledge and Data Engineering*, 2024.
- [62] Brian Wohlhieter. Bitclout: Decentralized social media or nfts for celebrities?, 2021.
- [63] Gavin Wood et al. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, 151(2014):1–32, 2014.
- [64] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *International conference on machine learning*, pages 6861–6871. Pmlr, 2019.
- [65] Qitian Wu, Wentao Zhao, Zenan Li, David P. Wipf, and Junchi Yan. Nodeformer: A scalable graph structure learning transformer for node classification. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.
- [66] Qitian Wu, Wentao Zhao, Chenxiao Yang, Hengrui Zhang, Fan Nie, Haitian Jiang, Yatao Bian, and Junchi Yan. Simplifying and empowering transformers for large-graph representations. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

- [67] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.
- [68] Lianghao Xia, Chao Huang, Chunzhen Huang, Kangyi Lin, Tao Yu, and Ben Kao. Automated self-supervised learning for recommendation. In *Proceedings of the ACM web conference 2023*, pages 992–1002, 2023.
- [69] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Inductive representation learning on temporal graphs. *arXiv preprint arXiv:2002.07962*, 2020.
- [70] Dongliang Xu, Wei Shi, Wensheng Zhai, and Zhihong Tian. Multi-candidate voting model based on blockchain. *IEEE/CAA Journal of Automatica Sinica*, 8(12):1891–1900, 2021.
- [71] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.
- [72] Wujiang Xu, Shaoshuai Li, Mingming Ha, Xiaobo Guo, Qiongxi Ma, Xiaolei Liu, Linxun Chen, and Zhenfeng Zhu. Neural node matching for multi-target cross domain recommendation. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, pages 2154–2166. IEEE, 2023.
- [73] Wujiang Xu, Zujie Liang, Jiaojiao Han, Xuying Ning, Wenfang Lin, Linxun Chen, Feng Wei, and Yongfeng Zhang. Slmrec: empowering small language models for sequential recommendation. *arXiv e-prints*, pages arXiv–2405, 2024.
- [74] Wujiang Xu, Kai Mei, Hang Gao, Juntao Tan, Zujie Liang, and Yongfeng Zhang. A-mem: Agentic memory for llm agents. *arXiv preprint arXiv:2502.12110*, 2025.
- [75] Wujiang Xu, Qitian Wu, Runzhong Wang, Mingming Ha, Qiongxi Ma, Linxun Chen, Bing Han, and Junchi Yan. Rethinking cross-domain sequential recommendation under open-world assumptions. In *Proceedings of the ACM Web Conference 2024*, pages 3173–3184, 2024.
- [76] Anatoly Yakovenko. Solana: A new architecture for a high performance blockchain v0. 8.13. *Whitepaper*, 2018.
- [77] Cheng Yang, Chunchen Wang, Yuanfu Lu, Xumeng Gong, Chuan Shi, Wei Wang, and Xu Zhang. Few-shot link prediction in dynamic networks. In *WSDM ’22*, 2022.
- [78] Liangwei Yang, Shengjie Wang, Yunzhe Tao, Jiankai Sun, Xiaolong Liu, Philip S Yu, and Taiqing Wang. Dgrec: Graph neural network for recommendation with diversified embedding generation. In *Proceedings of the sixteenth ACM international conference on web search and data mining*, pages 661–669, 2023.
- [79] Ruimeng Ye, Yang Xiao, and Bo Hui. A pilot study of weak-to-strong generalization in safety, toxicity, and legal reasoning. In *ICLR 2025 Workshop on Bidirectional Human-AI Alignment*, 2025.
- [80] Ruosong Ye, Caiqi Zhang, Runhui Wang, Shuyuan Xu, and Yongfeng Zhang. Language is all a graph needs. *arXiv preprint arXiv:2308.07134*, 2023.
- [81] Maofan Yin, Dahlia Malkhi, Michael K Reiter, Guy Golan Gueta, and Ittai Abraham. Hotstuff: Bft consensus with linearity and responsiveness. In *Proceedings of the 2019 ACM symposium on principles of distributed computing*, pages 347–356, 2019.
- [82] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? *Advances in neural information processing systems*, 34:28877–28888, 2021.
- [83] Jiaxuan You, Tianyu Du, and Jure Leskovec. Roland: graph learning framework for dynamic graphs. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pages 2358–2366, 2022.

- [84] Le Yu, Leilei Sun, Bowen Du, and Weifeng Lv. Towards better dynamic graph learning: New architecture and unified library. *Advances in Neural Information Processing Systems*, 36:67686–67700, 2023.
- [85] Shuo Yu, Yingbo Wang, Ruolin Li, Guchun Liu, Yanming Shen, Shaoxiong Ji, Bowen Li, Fengling Han, Xiuzhen Zhang, and Feng Xia. Graph2text or graph2token: A perspective of large language models for graph learning. *arXiv preprint arXiv:2501.01124*, 2025.
- [86] Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua B Tenenbaum, Tianmin Shu, and Chuang Gan. Building cooperative embodied agents modularly with large language models. *arXiv preprint arXiv:2307.02485*, 2023.
- [87] Huanding Zhang, Tao Shen, Fei Wu, Mingyang Yin, Hongxia Yang, and Chao Wu. Federated graph learning—a position paper. *arXiv preprint arXiv:2105.11099*, 2021.
- [88] Jiasheng Zhang, Jialin Chen, Menglin Yang, Aosong Feng, Shuang Liang, Jie Shao, and Rex Ying. Dtgb: A comprehensive benchmark for dynamic text-attributed graphs. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 91405–91429. Curran Associates, Inc., 2024.
- [89] Wei Zhang, Hongcheng Guo, Jian Yang, Yi Zhang, Chaoran Yan, Zhoujin Tian, Hangyuan Ji, Zhoujun Li, Tongliang Li, Tieqiao Zheng, et al. mabc: multi-agent blockchain-inspired collaboration for root cause analysis in micro-services architecture. *arXiv preprint arXiv:2404.12135*, 2024.
- [90] Xin Zhang, Daochen Zha, and Qiaoyu Tan. E2gnn: Efficient graph neural network ensembles for semi-supervised classification. *arXiv preprint arXiv:2405.03401*, 2024.
- [91] Yang Zhang, Fuli Feng, Chenxu Wang, Xiangnan He, Meng Wang, Yan Li, and Yongdong Zhang. How to retrain recommender system? a sequential meta-learning method. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’20, page 1479–1488, 2020.
- [92] Xi Zhu, Haochen Xue, Ziwei Zhao, Wujiang Xu, Jingyuan Huang, Minghao Guo, Qifan Wang, Kaixiong Zhou, and Yongfeng Zhang. Llm as gnn: Graph vocabulary learning for text-attributed graph foundation models. *arXiv preprint arXiv:2503.03313*, 2025.
- [93] Weiqin Zou, David Lo, Pavneet Singh Kochhar, Xuan-Bach Dinh Le, Xin Xia, Yang Feng, Zhenyu Chen, and Baowen Xu. Smart contract development: Challenges and opportunities. *IEEE transactions on software engineering*, 47(10):2084–2106, 2019.
- [94] Wenrui Zuo, Aravindh Raman, Raul J Mondragón, and Gareth Tyson. Set in stone: Analysis of an immutable web3 social media platform. In *Proceedings of the ACM Web Conference 2023*, pages 1865–1874, 2023.

Contents

1	Introduction	1
2	Related Works	2
2.1	Graph Learning for Social Networks	2
2.2	Blockchain Consensus Mechanism	3
2.3	Ensemble Learning and Majority Voting	3
3	Problem Definition	3
4	Our Framework	4
4.1	Framework Overview	4
4.2	Personalized Backbone Algorithm Selection	5
4.3	Decentralized Consensus Voting Scheme	6
5	Experiments	7
5.1	Experimental Setups	7
5.2	Evaluation Metrics	7
5.3	Impacts of the Personalized Algorithms	7
5.4	Impacts of the Multiple Validators	8
5.5	Efficiency Analysis	9
6	Conclusions	9
A	Details for the Experiments	18
A.1	Datasets	18
A.2	Baselines	18
A.3	Implementation Details	19
A.4	Framework Training	19
A.5	Simple Rule-Based Selection	20
A.6	Ganache Usage Introduction	20
A.7	Table of Notation	21
B	Operation Time Analysis of DeSocial	21
B.1	DeSocial Operation Time Breakdown	21
B.2	Run Time Analysis of Decentralized Methods	22
B.2.1	Both Modules Enabled	22
B.2.2	Personalized Algorithm Module Enabled Only	23
B.2.3	Decentralized Consensus Module Enabled Only	23
C	Additional Studies	24
C.1	Sensitivity to Hyperparameters in Personalized Algorithm Module	24

C.2	Enhancements of Decentralized Consensus	27
C.3	Consensus Analysis of Different Node Groups	28
D	Broader Impacts	28
E	Ethics Statement	29

A Details for the Experiments

A.1 Datasets

Table 4 shows the number of nodes and edges, network density and network types, in each dataset. For each dataset, we divide the temporal graph into a sequence of discrete time slices by uniformly partitioning all edges according to their timestamps, ensuring that each time slice contains approximately the same number of interactions while preserving the overall temporal order of events. We divide the 40 temporal slices into training, validation, and testing periods following a ratio of 25:5:10.

Table 4: Dataset Statistics

Dataset	#Users	#Interactions	Density	Network
UCI	1,899	59,835	0.016592	Web 2.0
Enron	42,711	797,907	0.000437	Web 2.0
GDELT	6,786	1,339,245	0.029083	Web 2.0
Memo-Tx	10,907	994,131	0.008357	Web 3.0

Memo-Tx [94] is a decentralized transaction network with timestamps on memo.cash, which is a Web3.0 social networking platform built directly on the Bitcoin Cash blockchain. Every user action on the platform, such as posting content, replying to others, liking posts, following users, trading cryptocurrencies, or updating profile information, is implemented as an on-chain transaction. The transaction network of memo.cash is constructed using timestamped transaction data from April 6, 2018 to November 30, 2021. Transactions are organized into blocks, each block containing multiple inputs and outputs. For transactions involving the same cryptocurrency within a block, edges are established from each input node to each output node, indicating the currency flows between nodes.

UCI [44] is a spatio-temporal network map of student interactions within the University of California, Irvine, showing communication relationships between students. These interactions are timestamped.

Enron [88] is a temporal graph dataset constructed from email communications between employees of the Enron corporation from 1999 to 2002. Each edge corresponds to a timestamped email between the Enron employees. The edges are arranged in chronological order according to the sending time.

GDELT [88] is a temporal graph dataset that originates from Web2.0 media sources and captures global political and social interactions over time. Each node represents an entity (e.g., a person, country, or organization). Edges arranged in chronological order indicate interactions between entities at specific timestamps, reflecting when these entities were mentioned together or engaged in an event.

A.2 Baselines

MLP [51]: The multilayer perceptron only uses node attributes, ignoring the topology of the graph. It is a powerful feature-based basis that is computationally efficient as it does not rely on the structure.

GCN [24]: The graph convolutional network performs graph spectral convolution by aggregating information about the immediate neighbors’ features. It efficiently captures local homophilic patterns.

GAT [55]: The graph attention network introduces attention mechanism to assign adaptive weights to neighbors during aggregation. This is particularly effective on heterogeneous or noisy connected graphs, but has a high computational cost and time-consuming because of the attention computation.

GraphSAGE [13]: GraphSAGE is a framework to learn aggregation functions in sampled neighbors, supporting the generalization to unseen nodes, suitable for large and dynamic graphs, balancing performance and scalability. It computes faster by reducing the number of neighbors for aggregation.

SGC [64]: The simplified graph convolution network removes nonlinearity and collapses multiple layers of graph convolution network into a single linear transformation with pre-computed propagation. It improves speed significantly while maintaining fairly high performances on homogeneous graphs.

A.3 Implementation Details

Hyper-Parameters in Backbone Selection. We run the heuristic backbone selection by setting the hyperparameters as follows:

- Time decay coefficient $\alpha \in \{0, -0.01, -0.1, -1\}$ explores increasing levels of decay. $\alpha = 0$ means no time decay is applied, all past neighbor interactions are treated equally. $\alpha < 0$ stands for an exponential or linear time discounting, where older interactions become less influential. The more negative α is, the faster the decay.
- Number of sampled neighborhood pairs $\gamma \in \{250, 500, 750, 1000, 1250\}$ determines the number of positive-negative neighbor edge pairs sampled for each node u during the heuristic backbone evaluation. Larger γ can let the backbone model test multiple negative samples from the same neighbor to make the selection more reliable.
- Backbone pool set $\mathcal{F} \in 2^{\mathcal{F}_{Full}}$ is a chosen subset of the full backbone pool \mathcal{F}_{Full} (i.e., MLP, GCN, GAT, GraphSAGE, and SGC). In real-life software use, users tend not to pay attention to all the backbone, and removing weaker or redundant models might improve selection precision. We can therefore study the variance of the performance influenced by different \mathcal{F} .

Hyper-Parameters in Consensus Mechanism. To study the effectiveness of decentralized modeling, we compare with MLP, GCN, GAT, GraphSAGE, and SGC with number of validators ranged in $\{3, 5, 7\}$. Each expert is trained independently on the same training data but with different random seeds to encourage diversity, they also pick different negative samples by setting different random seeds. Besides, we perform hyperparameter tuning over the following ranges: learning rate in $\{1e-1, 5e-2, 1e-2, 5e-3, 1e-3, 5e-4, 1e-4, 5e-5, 1e-5, 5e-6, 1e-6\}$ and dropout in $\{0.3, 0.5, 0.7\}$. For each model on each dataset, we report the performance of all metrics on the test period using the hyperparameter setting that achieves the highest value of each metric on the validation period.

Baseline Implementation. For GCN, GAT, and GraphSAGE, we implemented the models by applying two GCNConv, GATv2Conv and SAGEConv in PyG² respectively, and we used dot products for decoding. In GAT, we used 4 head attention instead of 8 to save computation memory and time, and added BatchNorm. For implementing SGC, we used SGConv for encoding and multiperception layers for decoding. The graph training algorithms are implemented based on the open-source DTGB benchmark [88].

Execution Environment. Our experiments are conducted on a server equipped with an Intel Xeon Gold 6226R CPU and eight NVIDIA A100 GPU. We employ a local ETH development environment powered by Ganache v7.9.2, with @ganache/cli and @ganache/core in version 0.10.2, to simulate blockchain behavior and smart contract interactions. The smart contract is implemented in Solidity and compiled by the truffle suite.

Code Availability. We have included the implementation code in our supplementary zip file. Although we originally planned to provide an anonymous Github repository, we decided that the supplementary materials already suffice for ensuring reproducibility.

A.4 Framework Training

Training Strategies. At each time step t , the model is trained by fully retrained strategy, taking $\bigcup_{\tau=0}^{t-1} \mathcal{G}^\tau$ as the training dataset. The model is validated on \mathcal{G}^t , and tested on \mathcal{G}^{t+1} , simulating an inductive setting where future edges must be predicted without directly observing them during training. We trained the dataset by 100 epochs at most, and also apply early stopping strategy, with 20 epochs as patience.

Model Updates. We restrict model updates to only the nodes selected as validators to test \mathcal{G}^{t+1} . Originally, DeSocial is designed to update the representations of all nodes using \mathcal{G}^t . However, this process is computationally intensive for single server simulation, and in practice, only a small subset of nodes are involved in validation. Therefore, to improve efficiency, In a practical blockchain setting, we envision that DeSocial would be executed in a decentralized manner, where multiple

²<https://pytorch-geometric.readthedocs.io/en/latest/index.html>

machines perform prediction tasks independently and concurrently, thereby substantially reducing computational latency.

A.5 Simple Rule-Based Selection

To demonstrate the effectiveness of DeSocial’s personalized algorithm selection, we conducted an additional ablation study. In this experiment, each user simply selects a personalized backbone algorithm based solely on two trivial local subgraph features, specifically:

- Node Degree ($Deg(u)$): Defined as the number of 1-hop neighbors of node u in the graph.
- Clustering Coefficient (c_u): Measures the density of the ego-network of node u , computed as the ratio between the number of edges among its neighbors and the maximum possible number of such edges.

Formally, given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, let $\mathcal{N}(u)$ be the 1-hop neighbor set of node u , and $|\mathcal{N}(u)| = Deg(u)$. The clustering coefficient c_u is calculated as:

$$c_u = \begin{cases} 0, & \text{if } Deg(u) \leq 1 \\ \frac{2 \cdot |\{(v,w) \in \mathcal{E} | v,w \in \mathcal{N}(u)\}|}{Deg(u) \cdot (Deg(u) - 1)}, & \text{otherwise} \end{cases} \quad (8)$$

The simple rule-based backbone selection rules are illustrated in the Algorithm 3.

Algorithm 3: : Rule-based Backbone Selection

Input: Node degree $Deg(u)$, clustering coefficient c_u , backbone pool \mathcal{F}

Output: Selected model \mathcal{F}_u for node u

```

if  $Deg(u) \geq 6$  and  $SGC \in \mathcal{F}$  then
  |  $\mathcal{F}_u \leftarrow SGC$ ;
end
else if  $c_u < 0.2$  and  $Deg(u) \geq 4$  and  $SAGE \in \mathcal{F}$  then
  |  $\mathcal{F}_u \leftarrow SAGE$ ;
end
else if  $Deg(u) \leq 2$  and  $MLP \in \mathcal{F}$  then
  |  $\mathcal{F}_u \leftarrow MLP$ ;
end
else if  $c_u \geq 0.4$  and  $GCN \in \mathcal{F}$  then
  |  $\mathcal{F}_u \leftarrow GCN$ ;
end
else
  |  $\mathcal{F}_u \leftarrow$  the last model in  $\mathcal{F}$ ;
end
return  $\mathcal{F}_u$ 

```

A.6 Ganache Usage Introduction

In our framework, decentralized social prediction is based on the deployment of smart contracts and validator interactions running on the blockchain infrastructure. In this paper, we focus on the design and algorithmic implications of decentralization, and here we provide additional background on the use of Ganache, the local Ethereum (ETH) development environment we used in our experiments.

Ganache is a widely-adopted in-memory ETH simulator designed for testing and development that allows developers to run a private blockchain instance with controllable parameters, including account creation, transaction latency, and miner behavior. Unlike ETH public testnet or main network, which can cause real-world network latency and gas costs, Ganache executes smart contracts quickly and deterministically without causing network instability.

In our deployment, Ganache is used to simulate validator selections, user-initiated prediction requests, and smart contract-based voting under real blockchain interfaces. This option allows for a comprehensive evaluation on DeSocial of decentralized workflows, including contract deployment, transaction

submission, and vote collection, without having to pay the high implementation and economical costs required for deployment to a public ETH network.

A.7 Table of Notation

Table 5: List of main notations used in this paper.

Symbol	Description
$\mathcal{G}^t = (\mathcal{V}^t, \mathcal{E}^t)$	Temporal graph at time t , with node set \mathcal{V}^t and edge set \mathcal{E}^t
\mathcal{F}	Pool of backbone models.
\mathcal{F}_{Full}	The set of backbones used in this paper, i.e., {MLP, GCN, GAT, SAGE, SGC}
f_{Θ_u}	Graph learning model used by node u with parameters Θ_u
$\Phi_{p,q,t}$	Validator committee for verifying link (p, q) at time t
$Vote(\phi, p, q, t)$	Binary decision by validator ϕ on link (p, q) at time t
$Ver(\Phi_{p,q,t}, p, q, t)$	Aggregated verification result by majority voting over $\Phi_{p,q,t}$
\mathbf{z}_u	Embedding vector of node u
$\sigma(\cdot)$	Sigmoid activation function
$Neg(\phi, p, q, t)$	Set of negative samples selected by validator $\phi \in \Phi_{p,q,t}$
Γ	Set of historical neighbor pairs (v_p, v_n) for model selection
$\Pi_{u,v}$	Temporal weight of edge (u, v) based on its emergence time
\mathcal{D}^t	All graph data up to time t , i.e., $\cup_{\tau=0}^t \mathcal{G}^\tau$
$\mathcal{N}^t(u)$	1-hop neighbor set of u in \mathcal{G}^t

B Operation Time Analysis of DeSocial

B.1 DeSocial Operation Time Breakdown

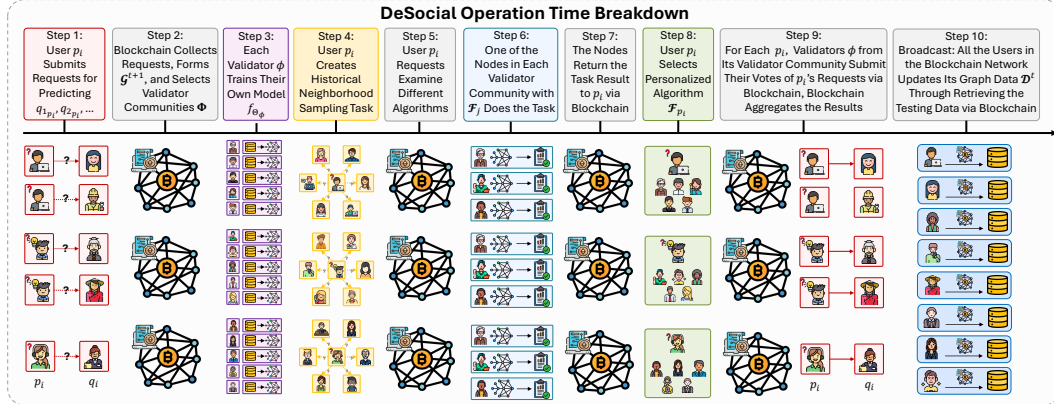


Figure 6: Illustration of the DeSocial operation pipeline for a single prediction period. Each row represents one user’s end-to-end process, from request submission to obtain decision. Different roles of the users are depicted using distinct person icons.

In this section, we provide a detailed breakdown of the runtime operations of DeSocial, deployed on Ganache, a local development chain of ETH, to better understand its step-by-step execution process. As shown in Figure 6, the full pipeline of DeSocial_{Full} is analyzed from the perspective of an individual user. Below, we enumerate each operational step to clarify the sequence of interactions between the requesting users, validators, and the blockchain infrastructure:

- **Step 1:** User p_i submits requests to predict social links with target nodes $q_{1p_i}, q_{2p_i}, \dots$
- **Step 2:** The blockchain collects all user requests, constructs \mathcal{G}^{t+1} , and assigns a validator community Φ according to each backbone model $\mathcal{F}_i \in \mathcal{F}$ through the smart contract.

- **Step 3:** Each validator $\phi \in \mathcal{V}_{val}^t$ independently trains their own graph learning model f_{θ_ϕ} based on the data \mathcal{D}^t stored in their own local memory. \mathcal{D}^t describes the union of the historical snapshots $\mathcal{G}^0, \mathcal{G}^1, \dots, \mathcal{G}^t$, and each node stored one copy of \mathcal{D}^t .
- **Step 4:** User p_i creates a personalized neighborhood sampling task based on local graph structure.
- **Step 5:** Validator nodes retrieve p_i 's request through the blockchain smart contract, and evaluate it using different available algorithms \mathcal{F}_j .
- **Step 6:** One selected validator in each community executes the sampling task using algorithm \mathcal{F}_j and returns results to the blockchain through the smart contract.
- **Step 7:** The result of each algorithm trial is returned to p_i through the blockchain for evaluation.
- **Step 8:** User p_i selects a preferred model \mathcal{F}_{p_i} based on the returned results.
- **Step 9:** Validators in Φ run \mathcal{F}_{p_i} on p_i 's request and submit their binary votes to the blockchain. The blockchain aggregates the votes to form the final prediction \mathcal{G}_{pred}^{t+1} . Both the voting and aggregating operations are defined by the smart contract.
- **Step 10:** The period ends, all the nodes in the network copy the social network data \mathcal{G}^{t+1} and merge it to their graph database via the blockchain by the smart contract.

We observe that as the number of requests to the ETH Ganache local blockchain increases, the overall runtime becomes slower. To better understand the computational overhead introduced by blockchain environment, we analyze the on-chain runtime under three configurations: (1) enabling both modules (Figure 6), (2) enabling only the personalized algorithm selection module (Figure 7), and (3) enabling only the multi-validator decentralized consensus module (Figure 8). We are going to give an analysis of the run time on the UCI and Memo-Tx dataset, representing the small graph and the large graph.

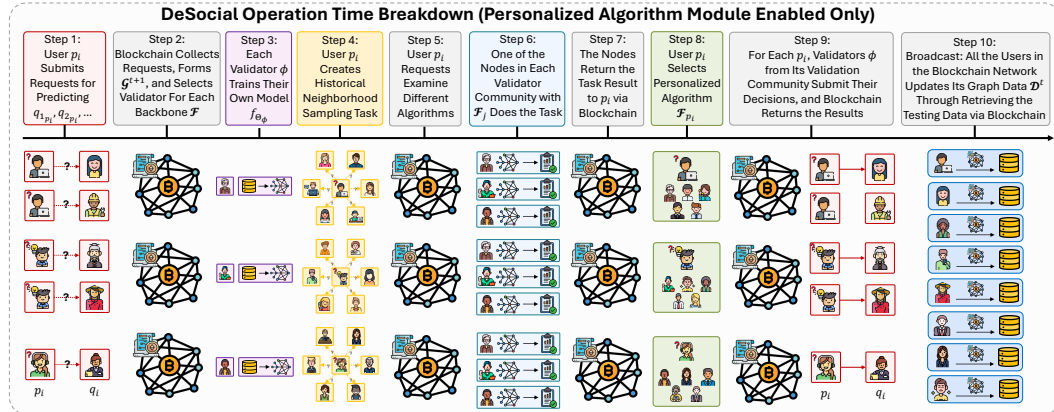


Figure 7: DeSocial pipeline with only the personalized algorithm selection module enabled. The blockchain assigns validators to evaluate candidate algorithms, and users select the best model without executing consensus voting.

B.2 Run Time Analysis of Decentralized Methods

B.2.1 Both Modules Enabled

This configuration is shown in Figure 6. Among the pipeline steps, Steps 1 and 8 cost negligible run time, as they involve only local Python instructions executed at a scale proportional to $|\mathcal{E}^t| \sim 10^4$ and $|\mathcal{V}^t| \sim 10^3$, respectively. We now analyze the run time costs for these steps shown in Table 6. **The runtime of all steps is reported as an amortized average of all users sending link prediction requests at t .** Steps 4, 5, 6, 7 and 10 have lower latency due to operations on user-neighborhood structures, which scale with $|\mathcal{V}^t| \sim 10^3$. Although Step 10 yields calls to the smart contract via the blockchain with the number of time scaling with $|\mathcal{V}| \sim 10^4$, bringing less overhead because they don't add blocks. The primary contributions to the data merging is the data copy in Python. The primary contributors to runtime overhead are Steps 2, 3, and 9. Step 3, the graph model training step, dominates the computation time, as training over \mathcal{D}^t is inherently sequential within each model and

cannot be parallelized. Steps 2 and 9 involve frequent blockchain interactions proportional to the number of prediction requests, again scaling with $|\mathcal{E}^t|$.

Table 6: Run time (s) for each major step in DeSocial $_{Full}$ under ETH Ganache simulation.

Step	Operation	Scale	Major Resource	Runtime (UCI)	Runtime (Memo-Tx)
2	Validator Community Formation	$ \mathcal{E}^t $	Smart Contract Execution	0.1193	1.414
3	Graph Model Training	\mathcal{D}^t	Model Training (GPU)	54.48	2011
4	Create Backbone Evaluation Task	$ \mathcal{V}^t $	Python Random Function	0.0022	0.0005
5	Request on Backbone Evaluation	$ \mathcal{V}^t $	Smart Contract Execution	0.0160	0.0177
6	Evaluation Execution	$ \mathcal{V}^t $	Model Inference	0.0002	0.0002
7	Return Evaluation Results	$ \mathcal{V}^t $	Smart Contract Execution	0.0161	0.0217
9	Vote and Aggregation	$ \mathcal{E}^t $	Smart Contract Execution	0.6233	9.081
10	Merge Graph Data	$ \mathcal{V} $	Python Data Copy	0.0249	0.2429

B.2.2 Personalized Algorithm Module Enabled Only

As shown in Figure 7, when only the personalized algorithm module is enabled, DeSocial $_{PA}$ still involves multiple rounds of blockchain interaction, primarily for coordination and information exchange between users and validators. Most of the pipeline remain unchanged, except the three aspects described as follows:

- In Step 2, for each validation community, the blockchain only selects one validator.
- In Step 3, for each validation community, only one validator is training the graph data \mathcal{D}^t .
- In Step 9, as there is only one validator, it can make the decision directly without consensus.

Table 7: Run time (s) for each major step in DeSocial $_{PA}$ under ETH Ganache simulation.

Step	Operation	Scale	Major Resource	Runtime (UCI)	Runtime (Memo-Tx)
2	Validator Community Formation	$ \mathcal{E}^t $	Smart Contract Execution	0.1419	1.360
3	Graph Model Training	\mathcal{D}^t	Model Training (GPU)	54.48	2011
4	Create Backbone Evaluation Task	$ \mathcal{V}^t $	Python Random Function	0.0017	0.0005
5	Request on Backbone Evaluation	$ \mathcal{V}^t $	Smart Contract Execution	0.0142	0.0169
6	Evaluation Execution	$ \mathcal{V}^t $	Model Inference	0.0002	0.0002
7	Return Evaluation Results	$ \mathcal{V}^t $	Smart Contract Execution	0.0182	0.0209
9	Submit Decisions	$ \mathcal{E}^t $	Smart Contract Execution	0.2396	3.220
10	Merge Graph Data	$ \mathcal{V} $	Python Data Copy	0.0249	0.2429

After removing the operation of forming a five-validator committee, the number of blockchain interactions in Step 9 decreases, as each link prediction now requires only a single vote submission instead of five. However, the run-time reduction is not strictly linear, since in our implementation, the smart contract still invokes the aggregation function to compare the number of True and False votes. Even with only one vote, this consensus logic introduces a small but non-negligible overhead.

It is important to note that when a user selects a slower algorithm that yields better performance, the runtime of Step 3 should be determined by the slowest backbone in \mathcal{F} , because each validator independently trains their own model in parallel, and the overall execution must wait for the slowest backbone to complete. Here, GCN and GraphSAGE are the slowest backbone for DeSocial $_{PA}$ on UCI and Memo-Tx, respectively.

B.2.3 Decentralized Consensus Module Enabled Only

Table 8: Run time (s) for each major step in DeSocial $_{Vote}$ under ETH Ganache simulation.

Step	Operation	Scale	Major Resource	Runtime (UCI)	Runtime (Memo-Tx)
2	Validator Selection	$ \mathcal{E}^t $	Smart Contract Execution	0.0855	1.130
3	Graph Model Training	\mathcal{D}^t	Model Training (GPU)	46.43	2011
4	Vote and Aggregation	$ \mathcal{E}^t $	Smart Contract Execution	0.4538	7.460
5	Merge Graph Data	$ \mathcal{V} $	Python Data Copy	0.0249	0.2429

When only the decentralized consensus module is enabled, the DeSocial $_{Vote}$ framework skips the personalized algorithm selection phase and instead applies a fixed, prespecified backbone for all users, as depicted in Figure 8. In this setup, validators train their own models, but unlike DeSocial $_{Full}$, no personalized selection process takes place. The models trained are simply evaluated for their effectiveness in the decentralized setting. After training, the validators submit their votes to the blockchain, where consensus is reached through majority voting.

We analyze the run-time of Steps 2 to 5 in Table 8. After reducing the overhead of blockchain requests of personalized algorithm selection, i.e., forming multiple validation communities and the backbone evaluation tasks, the run time of selecting validators (Step 2), voting and aggregating the decisions (Step 4), and broadcasting graph data (Step 5) is reduced.

Notably, the error of the blockchain operations in all three configurations is mainly due to CPU occupancy. Step 3 remained unchanged as centralized backbone because we don't need to run other backbones.

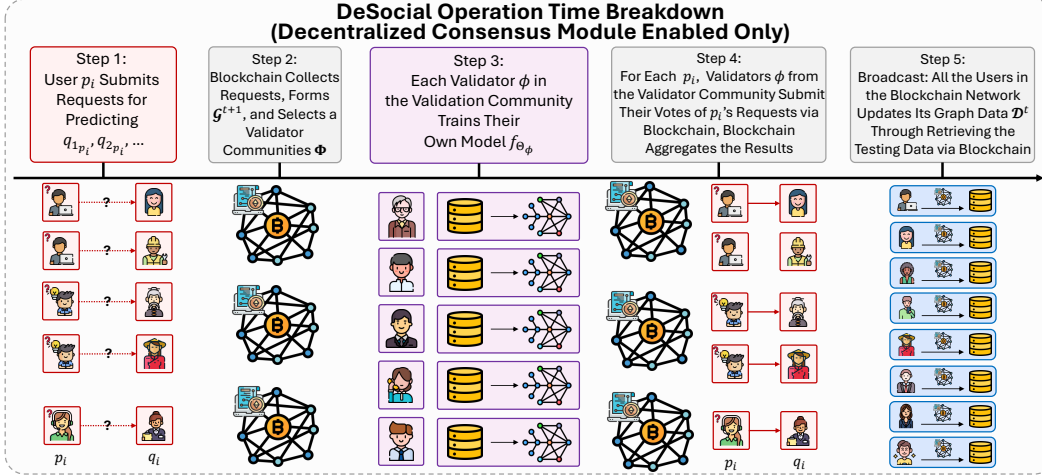


Figure 8: DeSocial pipeline with only the decentralized consensus module enabled. Users do not select personalized algorithms, and validators independently train models and the blockchain finalizes predictions via majority voting.

C Additional Studies

C.1 Sensitivity to Hyperparameters in Personalized Algorithm Module

We conduct a sensitivity analysis on the personalized algorithm module by varying three key parameters: the number of the time decay coefficient α , sampled neighborhood pairs γ , and the backbone selection pool \mathcal{F} . The description of these hyperparameters are in Appendix A.3.

Table 9: Impact of time decay coefficient α on accuracy across datasets. Each block reports the means and the standard deviations (%). Bold denotes the highest value per dataset at three evaluation metrics. All other parameters are fixed to their optimal values.

Dataset	α	Acc@2	Acc@3	Acc@5
UCI	0	72.84 \pm 0.27	62.38 \pm 0.32	51.00 \pm 0.33
	-0.01	73.02 \pm 0.28	62.74 \pm 0.32	51.37 \pm 0.38
	-0.1	73.35 \pm 0.28	62.80 \pm 0.30	51.34 \pm 0.32
	-1	72.35 \pm 0.26	61.60 \pm 0.31	49.94 \pm 0.31
Memo-Tx	0	83.96 \pm 0.13	77.18 \pm 0.14	69.31 \pm 0.15
	-0.01	83.45 \pm 0.13	76.72 \pm 0.14	68.97 \pm 0.14
	-0.1	83.37 \pm 0.13	76.71 \pm 0.14	69.07 \pm 0.14
	-1	83.12 \pm 0.11	76.60 \pm 0.13	68.97 \pm 0.13
Enron	0	90.05 \pm 0.09	85.94 \pm 0.11	81.13 \pm 0.13
	-0.01	90.02 \pm 0.11	85.86 \pm 0.13	80.92 \pm 0.16
	-0.1	90.08 \pm 0.11	85.96 \pm 0.13	81.02 \pm 0.17
	-1	89.60 \pm 0.11	85.40 \pm 0.13	80.39 \pm 0.16
GDELT	0	95.56 \pm 0.02	92.44 \pm 0.02	87.55 \pm 0.04
	-0.01	95.57 \pm 0.02	92.44 \pm 0.02	87.58 \pm 0.04
	-0.1	95.55 \pm 0.02	92.40 \pm 0.02	87.53 \pm 0.04
	-1	95.48 \pm 0.02	92.30 \pm 0.02	87.36 \pm 0.03

Sensitivity to α . As shown in Table 9, we observe that the choice of the time decay coefficient α affects the performance of the personalized algorithm selection module. Generally, across $\alpha \in \{0, -0.01, -0.1, -1\}$, the performance tends to peak at a mild negative value and degrades as $|\alpha|$ increases. For the Web 3.0 dataset Memo-Tx, as Web 3.0 transaction network lack strong temporal patterns due to the one-off, irregular, and non-periodic transactions, the temporal characteristics is weaker than the Web 2.0 dataset, thus applying $\alpha = 0$ that treats the historical interactions equally, can achieve the best performance. This trend reflects that how much the recent interactions should be prioritized when constructing the local subgraph for algorithm selection, according to the graph’s temporal characteristics.

Table 10: Effect of neighbor sample size γ on personalized algorithm selection performance (%). Each block reports the means and the standard deviations. Best values per dataset are in bold. All other parameters are fixed to their optimal values.

Dataset	γ	Acc@2	Acc@3	Acc@5
UCI	250	73.15 \pm 0.28	62.59 \pm 0.29	51.02 \pm 0.33
	500	72.76 \pm 0.24	62.33 \pm 0.29	50.83 \pm 0.30
	750	73.35 \pm 0.28	62.80 \pm 0.30	51.34 \pm 0.32
	1000	72.77 \pm 0.29	62.26 \pm 0.34	50.81 \pm 0.37
	1250	72.44 \pm 0.28	62.08 \pm 0.31	50.74 \pm 0.33
Memo-Tx	250	83.96 \pm 0.13	77.18 \pm 0.14	69.31 \pm 0.15
	500	83.81 \pm 0.13	77.19 \pm 0.14	69.52 \pm 0.14
	750	83.82 \pm 0.12	77.03 \pm 0.14	69.21 \pm 0.15
	1000	83.91 \pm 0.12	77.28 \pm 0.12	69.57 \pm 0.13
	1250	83.76 \pm 0.14	76.97 \pm 0.15	69.21 \pm 0.17
Enron	250	89.63 \pm 0.09	85.43 \pm 0.12	80.40 \pm 0.15
	500	89.80 \pm 0.09	85.65 \pm 0.10	80.69 \pm 0.12
	750	89.98 \pm 0.12	85.85 \pm 0.14	80.88 \pm 0.16
	1000	89.88 \pm 0.10	85.75 \pm 0.12	80.87 \pm 0.15
	1250	90.08 \pm 0.11	85.96 \pm 0.13	81.02 \pm 0.17
GDELT	250	95.54 \pm 0.02	92.41 \pm 0.03	87.54 \pm 0.04
	500	95.53 \pm 0.02	92.40 \pm 0.02	87.52 \pm 0.04
	750	95.54 \pm 0.02	92.39 \pm 0.02	87.51 \pm 0.04
	1000	95.57 \pm 0.02	92.44 \pm 0.02	87.58 \pm 0.04
	1250	95.55 \pm 0.02	92.42 \pm 0.02	87.55 \pm 0.04

Sensitivity to γ . Table 10 shows the performance given $\gamma \in \{250, 500, 750, 1000, 1250\}$. In DeSocial, γ controls the size of neighbor sample set to determine the most suitable algorithm. This directly affects the local structural context used for personalized algorithm selection. A small γ lead to insufficient information, making the selector unstable, while a large γ may introduce outdated or noisy neighbors. Therefore, an appropriate γ is needed for achieving the best performance.

Sensitivity to \mathcal{F} . Table 11 reports the Acc@2 of different backbone algorithm combinations \mathcal{F} across four datasets. We analyze the impact of the backbone pool \mathcal{F} from several perspectives and highlight potential implications for blockchain-based decentralized social network prediction frameworks. Specifically, we address the following questions:

- **Q1:** Which \mathcal{F} yield the best performance?
- **Q2:** Do different datasets exhibit distinct preferences for specific backbones?
- **Q3:** Does the "less is more" phenomenon occur, where increasing the number of backbones leads to degraded performance?

For **Q1**, from the aspect of Acc@2, the best \mathcal{F} in UCI is $\{\text{MLP}, \text{GCN}, \text{GraphSAGE}, \text{SGC}\}$ and $\{\text{GAT}, \text{GraphSAGE}, \text{SGC}\}$, with an Acc@2 of 73.35%. The best \mathcal{F} in Memo-Tx is $\{\text{GraphSAGE}, \text{SGC}\}$ with an Acc@2 of 83.96%. The best \mathcal{F} in Enron is $\{\text{GAT}, \text{GraphSAGE}\}$ with an Acc@2 of 90.08%. The best \mathcal{F} in GDELT is $\{\text{GraphSAGE}, \text{SGC}\}$ and $\{\text{GAT}, \text{SGC}\}$ with an Acc@2 of 95.56%. We observe that GraphSAGE and SGC appears in all four top-performing combinations, making them core backbones across diverse graph types. For each dataset, the best \mathcal{F} always includes the best centralized backbone model. This highlights the foundational role of core models in supporting the performance of decentralized social network algorithms.

For **Q2**, Yes. Although SGC and GraphSAGE are the core contributors to the performance, UCI may need to incorporate with GCN and MLP, while Enron and GDELT may need to incorporate with

GAT. It shows that different social network structures prefer different model combinations, which also demonstrates the importance of personalized algorithm selection in real-world deployments.

For **Q3**, Yes, the "less is more" phenomenon occurs in all datasets. In UCI, while selecting MLP and GraphSAGE can improve the centralized performance, adding GAT makes a worse performance. In Memo-Tx, while selecting GraphSAGE and SGC performs the best performance, adding GCN degrades the performance. In Enron, while selecting GAT and GraphSAGE achieves performs the best, adding one of the SGC, MLP, or GCN can hinder the improvement. In GDELT, when combining MLP and GAT can achieve 91.36% Acc@2, adding GCN can let Acc@2 drops back to 91.29%. Also, combining more backbones may perform worse than the centralized ones. Therefore, it is unrealistic to consider as many models as possible to combine, and increasing the number of models may not linearly improve performance, but may also introduce noise or redundancy.

Our analysis verifies that a reasonable combination of models is more effective than simply piling up more models. For future work on blockchain-based decentralized social networks, more research is needed to explore the complementarities and conflicts between backbone algorithms. Whether to adaptively choose combinations based on graph structure is the next question to be investigated.

Table 11: Performance comparison over different \mathcal{F} . Acc@2 (%) is reported with mean and standard deviation. "Y" indicates improvement over all centralized models in \mathcal{F} , while "N" indicates not.

\mathcal{F}	UCI			Memo-Tx			Enron			GDELT		
	Acc@2		↑	Acc@2		↑	Acc@2		↑	Acc@2		↑
MLP	66.38 ± 0.34	-		73.61 ± 0.11	-		81.48 ± 0.08	-		91.28 ± 0.02	-	
GCN	63.90 ± 0.17	-		69.62 ± 0.13	-		79.92 ± 0.09	-		82.94 ± 0.04	-	
GAT	61.15 ± 0.26	-		72.51 ± 0.26	-		85.52 ± 0.12	-		88.29 ± 0.28	-	
SAGE	69.00 ± 0.40	-		82.85 ± 0.15	-		90.27 ± 0.06	-		93.16 ± 0.02	-	
SGC	72.77 ± 0.24	-		80.37 ± 0.05	-		88.24 ± 0.04	-		95.59 ± 0.02	-	
MLP+GCN	66.51 ± 0.23	Y		76.20 ± 0.10	Y		83.73 ± 0.09	Y		91.02 ± 0.02	N	
MLP+GAT	65.73 ± 0.24	Y		74.22 ± 0.22	Y		86.57 ± 0.18	Y		91.36 ± 0.06	Y	
MLP+SAGE	70.04 ± 0.39	Y		82.08 ± 0.16	N		89.86 ± 0.07	N		92.76 ± 0.02	N	
MLP+SGC	73.19 ± 0.26	Y		80.73 ± 0.09	Y		88.29 ± 0.07	Y		95.42 ± 0.02	N	
GCN+GAT	64.12 ± 0.27	Y		74.29 ± 0.24	Y		85.55 ± 0.17	Y		87.70 ± 0.16	N	
GCN+SAGE	69.15 ± 0.39	Y		82.74 ± 0.16	N		89.88 ± 0.06	N		92.84 ± 0.02	N	
GCN+SGC	72.74 ± 0.23	N		80.15 ± 0.08	N		88.24 ± 0.06	N		95.56 ± 0.02	N	
GAT+SAGE	68.30 ± 0.44	N		81.83 ± 0.18	N		90.08 ± 0.11	N		92.96 ± 0.05	N	
GAT+SGC	72.45 ± 0.23	N		78.42 ± 0.17	N		88.68 ± 0.07	Y		95.56 ± 0.02	N	
SAGE+SGC	73.25 ± 0.33	Y		83.96 ± 0.13	Y		89.81 ± 0.04	N		95.57 ± 0.02	N	
MLP+GCN+GAT	65.92 ± 0.26	N		75.45 ± 0.19	Y		86.22 ± 0.17	Y		91.29 ± 0.05	Y	
MLP+GCN+SAGE	69.53 ± 0.33	Y		82.04 ± 0.12	N		89.66 ± 0.08	N		92.61 ± 0.02	N	
MLP+GCN+SGC	72.79 ± 0.26	Y		80.60 ± 0.08	Y		88.17 ± 0.06	N		95.41 ± 0.02	N	
MLP+GAT+SAGE	68.88 ± 0.34	N		81.08 ± 0.16	N		89.64 ± 0.10	N		92.74 ± 0.04	N	
MLP+GAT+SGC	72.87 ± 0.27	Y		78.64 ± 0.15	N		88.62 ± 0.07	Y		95.40 ± 0.02	N	
MLP+SAGE+SGC	73.25 ± 0.29	Y		83.57 ± 0.12	Y		89.79 ± 0.04	N		95.40 ± 0.02	N	
GCN+GAT+SAGE	68.48 ± 0.37	N		81.73 ± 0.15	N		89.89 ± 0.09	N		92.82 ± 0.04	N	
GCN+GAT+SGC	72.74 ± 0.22	N		78.20 ± 0.15	N		88.62 ± 0.07	N		95.55 ± 0.02	N	
GCN+SAGE+SGC	72.98 ± 0.28	Y		83.79 ± 0.13	Y		89.71 ± 0.05	N		95.56 ± 0.02	N	
GAT+SAGE+SGC	73.35 ± 0.27	Y		83.11 ± 0.15	Y		90.03 ± 0.06	N		95.55 ± 0.02	N	
MLP+GCN+GAT+SAGE	69.03 ± 0.38	Y		81.20 ± 0.14	N		89.57 ± 0.11	N		92.64 ± 0.02	N	
MLP+GCN+GAT+SGC	72.83 ± 0.28	Y		78.52 ± 0.17	N		88.70 ± 0.07	Y		95.40 ± 0.02	N	
MLP+GCN+SAGE+SGC	73.35 ± 0.27	Y		83.27 ± 0.13	Y		89.57 ± 0.05	N		95.39 ± 0.02	N	
MLP+GAT+SAGE+SGC	72.86 ± 0.26	Y		82.45 ± 0.16	N		89.77 ± 0.06	N		95.40 ± 0.02	N	
GCN+GAT+SAGE+SGC	72.77 ± 0.30	N		82.63 ± 0.14	N		89.86 ± 0.07	N		95.54 ± 0.02	N	
MLP+GCN+GAT+SAGE+SGC	73.07 ± 0.28	Y		82.36 ± 0.15	N		89.75 ± 0.06	N		95.39 ± 0.02	N	

C.2 Enhancements of Decentralized Consensus

To better understand decentralized multi-validator consensus mechanisms in DeSocial, we analyze the behavior of the proportions of the validator agreements and their relationship with accuracy in every testing period across UCI, Memo-Tx. For Enron and GDEL, we observe consistent results regarding the effectiveness of decentralized consensus. Figure 9 and Figure 10 illustrate the performances on three evaluation metrics and the proportions of different levels of agreements for UCI, respectively. Figure 11 and Figure 12 are for Memo-Tx.

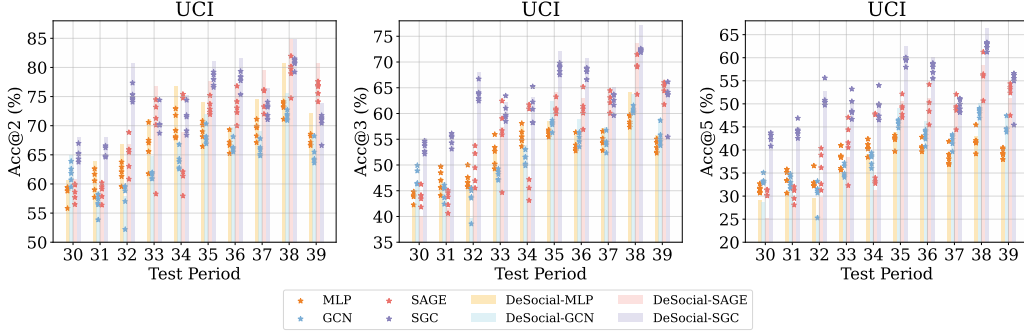


Figure 9: Testing performance at UCI dataset. The stars illustrate the centralized performances while the bars illustrate the decentralized performances in each testing period.

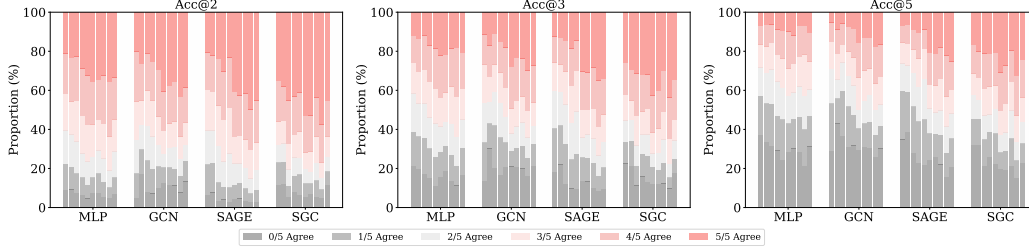


Figure 10: Distribution of validator agreement levels on the UCI dataset. Each set of ten bars corresponds to a single model evaluated over test periods 30 to 39.

Accuracy Threshold and Voting Effectiveness. As shown in Figure 9 and Figure 11, in general, when the centralized accuracy is above 0.5, decentralized consensus almost always improves or maintains the performance. However, when the centralized model falls below 0.5, there is a greater risk of performance degradation due to the amplification of poor local predictions in the voting process. Therefore, a minimum model quality threshold is needed to leverage decentralized voting.

Robustness Against Outliers and Noise. The decentralized consensus methods help improve the overall performance even when the centralized experiments exhibit outliers or noise. For example, notable outliers are observed at Acc@2 for MLP at period 30, SGC at period 39, GraphSAGE at periods 33 and 38 in Figure 9, and Acc@2 for GraphSAGE and GCN at period 31 in Figure 11. In these cases, the performance difference between the best and worst centralized results can be as high as 5%. However, in majority voting under decentralized consensus, the final performances not only do not decline with such extreme values but steadily improve. This indicates that decentralized consensus provides robust aggregation, effectively mitigating localized prediction failures.

Correlation Between Ambiguity and Improvement. For each testing period, we compute the level of agreement, which is defined as the number of validators (out of five) that correctly predict the ground truth label, for each edge and report the proportions of each agreement level. In Figure 10 and Figure 12, darker colors indicate stronger consensus among validators. In UCI dataset shown in Figure 10, we observe that the ambiguous predictions (lighter colors) appear in similar proportions across four backbone models. As a result, the performance improvements from centralized to decentralized variants are also relatively similar (1.76% in average). In contrast, Figure 12 reveals that for Memo-Tx dataset, GCN has a higher proportion of ambiguous decisions compared to the

other backbones. This leads to a larger performance gain (10.89%) under decentralized voting for GCN, while the performance gain for other three backbones is 2.98% in average.

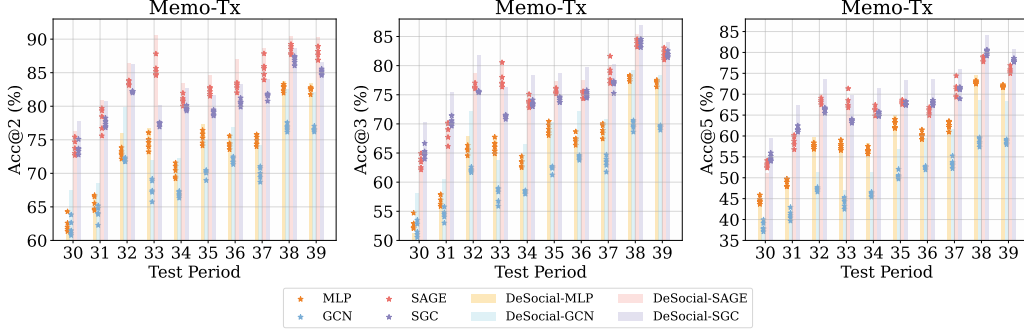


Figure 11: Testing performance at Memo-Tx dataset. The stars illustrate the centralized performances while the bars illustrate the decentralized performances in each testing period.

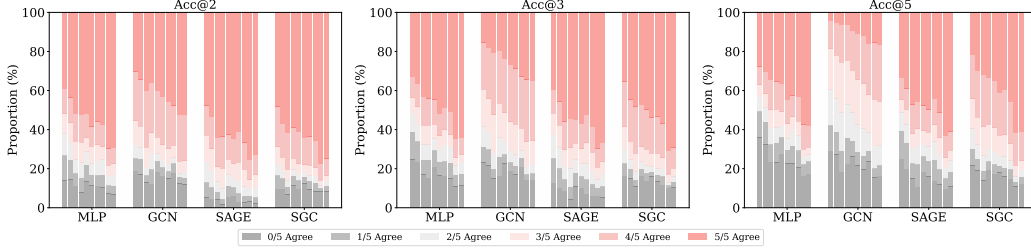


Figure 12: Distribution of validator agreement levels on the Memo-Tx dataset. Each set of ten bars corresponds to a single model evaluated over test periods 30 to 39.

C.3 Consensus Analysis of Different Node Groups

In each temporal snapshot, nodes with higher degrees are considered more active, while those with lower degrees are less active. To analyze performance across different activity levels, we divide the graph nodes into four quartiles in each test period based on their degrees. The lowest 25% nodes are assigned to Q1, while the highest 25% fall into Q4.

Figure 13 illustrates the proportion of the 5/5 agreement (i.e., of all the agreements in this quartile, how many of them have all the validators given true votes) at Acc@2 on each quartile. Q1 nodes exhibit the lowest proportion of 5/5 agreement due to insufficient neighborhood information. Q2 and Q3 nodes gain more neighbors, enabling more reliable feature aggregation and increasing the proportion of 5/5 agreement. Q4 nodes gain the highest proportion of 5/5 agreement in most cases, because the nodes with the most neighborhoods have enough local information to make accurate predictions. However, for all backbones on UCI and GCN on GDELT, the proportion of 5/5 agreement in Q4 is lower than that of Q3 because the information learned by the model is redundant or even conflicting, and multiple models understand these structures in different ways, the disagreement increases instead.

As higher quartiles achieve higher proportions of 5/5 agree, we can conclude that the contributions to 5/5 agree are mainly from the nodes with higher degrees, i.e., from more active users.

D Broader Impacts

DeSocial aims to empower users on social platforms by allowing them to choose personalized social network algorithms and validate predictions through decentralized voting. This shift from centralized to decentralized decision-making has many potential social implications.

Positive Social Impacts. Firstly, by giving the user rights to select personalized social network algorithm, DeSocial reduces the risk of algorithmic centralization, which often serves platform

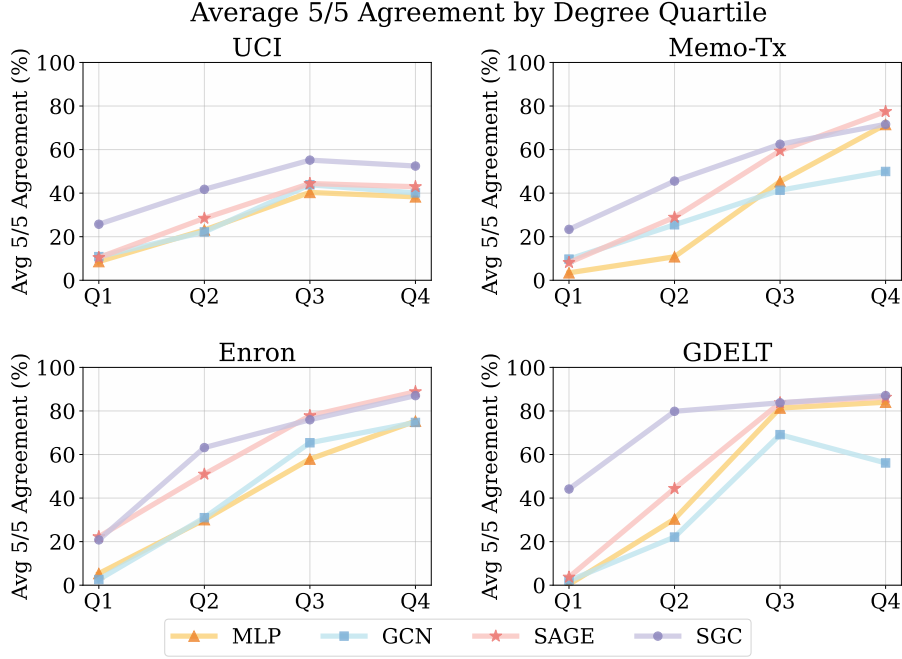


Figure 13: 5/5 agreement across degree node quartiles for different models and datasets.

interests over user needs. Secondly, blockchain-based voting ensures that prediction results can be traced and verified, encouraging more transparent and auditable AI decisions. Thirdly, We give the community a novel solution to improve the personalized recommendations, giving an additional way to extend the existing state-of-the-art recommendation algorithms. Lastly, DeSocial aligns with the values of Web 3.0 infrastructure, contributing to ethical AI deployment in emerging digital economies.

Negative Social Impacts. Firstly, improper algorithm selection may reduce prediction quality. Secondly, misuse of Web 3.0 techniques like speculative behavior unrelated to genuine social interaction may undermine the utility of decentralized social media. Particularly, domination of the malicious validators can game or bias the outcomes, especially without proper incentive or trust mechanisms.

Given the associated risks, we only implement this algorithm in a locally simulated blockchain currently. For future deployment, we need to solve the common problems in blockchain environment, such as trustworthy, fairness, security, and privacy, to ensure a broadly accepted and reliable social infrastructure for technical users.

E Ethics Statement

This work contributes to the design of decentralized AI frameworks by combining graph learning with blockchain-enabled consensus mechanisms. It emphasizes user autonomy through personalized model selection and multi-node validation, offering an alternative to traditional centralized decision-making pipelines. All experiments are carried out in a controlled local blockchain simulation environment, without involving real blockchain deployments or human participants. The datasets used are publicly available and free of sensitive information like human subjects. Our work is committed to building socially responsible AI infrastructure that respects user autonomy and supports trustworthy deployment in Web 3.0 environments.