Expert Systems With Applications 260 (2025) 125391

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Using attentive temporal GNN for dynamic trust assessment in the presence of malicious entities

Besat Jafarian[a], Nasser Yazdani[a,*], Mohammad Sayad Haghighi[a,b]

[a] School of Electrical and Computer Engineering, University of Tehran, Tehran 1439957131, Iran
[b] School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran 19395-5746, Iran

A R T I C L E I N F O

Keywords:
Trust assessment
Complex networks
Graph convolutional networks
Security

A B S T R A C T

Trust networks are built on the premise that there is a network of trust among entities. Precisely evaluating pairwise trust in such networks is essential for enhancing the quality of service in various applications, including online marketing, recommender systems, and IoT-based applications. However, it remains challenging to consider dynamic trust and account for malicious behavior in a scalable and effective manner.

Recently, Graph Neural Networks (GNN)-based methods have emerged as powerful and scalable tools for trust estimation. Nevertheless, they lack thorough consideration for the time-dependent nature of trust, especially in the presence of malicious behaviors, leading to inaccurate trust predictions. To address this gap, we propose a novel trust assessment model called MATA, which handles the time dependency of trust and exhibits robustness against malicious behaviors. MATA leverages an extended Graph Convolutional Network (GCN) integrated with Recurrent Neural Networks (RNNs) and an attention mechanism to jointly capture time dependencies and trace entities' behavior over time in complex networks. Additionally, by clustering the learned trust representations over time, we assign reputation values to entities. This allows us to monitor their dynamic behavior and mitigate potential adverse effects on trust assessment.

In our experiments on real-world datasets, MATA outperforms the state-of-the-art methods by up to 8.4 % in accuracy. This improvement is a testament to the effectiveness of the proposed approach. Furthermore, our work enhances the robustness of trust assessment models, as evidenced by consistently higher performance metrics in the face of various trust-related attacks.

1. Introduction

Due to the open nature of complex networks such as social networks and Internet of Things (IoT), they provide an opportunity for malicious entities to deliberately generate fake information or distribute illegal messages for personal gain (Jiang et al., 2023). To address this challenge, some networks incorporate a trust network among entities, allowing each entity to assign explicit values indicating the trustworthiness of their direct friends. Precise evaluation of trustworthiness between two entities who lack a direct connection, which plays a crucial role in the functionality and operation of networks, has emerged as a crucial area of research (Huo et al., 2022). This, in turn, can enhance the quality of service across various domains, including online marketing, recommender systems and IoT-based applications.

Trust, usually refers to a situation where one party (trustor) is willing to rely on the actions of another party (trustee) (Massa & Avesani, 2007). There are two major categories of inferring trust for networks (Jiang et al., 2023; Massa & Avesani, 2007); implicit and explicit trust models. In implicit trust models, the system lacks explicit trust information, and the objective is to develop a mathematical model based on existing trust information in the networks (such as profile similarity, rating history, etc.) to construct a network with trust values between entities (Arabsorkhi et al., 2016; Chen et al., 2019; Ebrahimi et al., 2022; Jafarian et al., 2020; Nitti et al., 2012, 2014). On the other hand, explicit trust models assume the existence of a trust network where nodes represent entities and edges between nodes represent trust relationship indicating their explicit trust value. Explicit trust models aim to establish a trust propagation mechanism based on observed trust relationships among nodes and then aggregate them to predict trust ratings between unobserved nodes. Significant work has been reported in the literature on predicting explicit pairwise trust. Some of them employ breadth-first search (BFS) algorithms in trust networks to establish trust relationships

* Corresponding author.
E-mail addresses: bst.jafarian@ut.ac.ir (B. Jafarian), yazdani@ut.ac.ir (N. Yazdani), sayad@ut.ac.ir (M. Sayad Haghighi).

https://doi.org/10.1016/j.eswa.2024.125391
Received 15 November 2023; Received in revised form 30 June 2024; Accepted 11 September 2024
Available online 14 September 2024

between nodes who do not have a direct trust connection (Golbeck, 2005; Liu et al., 2017). In reference Golbeck (2005) trust values propagated from multiple paths between two nodes are summed up to estimate the trust value, while reference (Liu et al., 2017) takes a different approach by utilizing subjective logic operators for a more precise estimation of trust values.

Recently, Graph Convolutional Neural Networks (GCNs) (Hamilton et al., 2017; Kipf & Welling, 2016; Ying et al., 2018) have emerged as powerful tools for analyzing graph-structured data. GCNs leverage the node feature information propagation and aggregation guided by the graph structure. Additionally, through multiple convolution layers, local node information can be effectively propagated across the entire graph (Fan et al., 2019). Therefore, GCNs which are capable of extracting both node features and topological structures to model dependencies among graph nodes, have been increasingly utilized for trust assessment in recent research efforts (Huo et al., 2022; Jiang et al., 2023; Lin & Li, 2021; Lin et al., 2020). For instance, Guardian Lin et al. (2020), proposed a model based on GCN to capture propagation and aggregation rules of trust. By leveraging GCNs, this method integrates the graph structure of social networks and trust relationships among nodes, enabling the propagation of trust information throughout the entire network. Accordingly, pairwise trust can be predicted.

Existing GNN-based models (Huo et al., 2022; Jiang et al., 2023; Lin & Li, 2021; Lin et al., 2020) for trust prediction have a limitation in assuming that the trust network remains static, disregarding the dynamic and time-dependent nature of trust. The predictive efficiency of these models is hampered when they ignore the evolving dynamics and historical behavior of trustor/trustee relationships, leading to inaccurate trust predictions. GCNs demonstrate proficiency in representing graph information; however, they lack the ability to effectively handle temporal features. To address this limitation, researchers in related fields have explored combining Recurrent Neural Networks (RNNs) like LSTM (Hochreiter & Schmidhuber, 1997) and GRU (Cho et al., 2014) with GCNs to capture dynamism as well as structural and content features (Manessi et al., 2020; Pareja et al., 2020; Seo et al., 2018).

RNNs offer two key advantages for improved prediction. Firstly, they have the ability to retain past information in a hidden state, enabling them to capture long-term dependencies. Secondly, RNNs assume that temporal dependencies exhibit a monotonic change with input time steps, giving more significance to the current hidden state compared to the previous ones (Liu et al., 2016). This assumption allows for the consideration of short-term sequential effects. However, the correlations between short-term behaviors are more complex, and the monotonic assumption struggles to accurately distinguish the importance of recent time steps (Cui et al., 2017). Therefore, there is a need for mechanisms that can effectively learn and capture these intricate correlations to model short-term trust. This will make trust prediction more sensitive to recent behaviors. For instance, malicious nodes may occasionally engage in anomalous operations while behaving normally most of the time, aiming to conceal their long-term anomalous behaviors and increase the difficulty of detection. Consequently, despite the recent emergence of powerful and scalable GNN-based methods, they fall short of considering the time-dependent nature of trust and the significant effect of malicious behaviors on trust prediction. Our paper innovatively addresses these two essential features of trust assessment. Firstly, to tackle the dynamic nature of trust, we adopt a sequential approach, using input data as progressive views to trust networks instead of a single snapshot. This allows us to model the temporal dependencies between these sequences, effectively capturing the evolving dynamics of trust relationships over time. Secondly, we tackle two significant attacks related to trust in the design of the problem and mitigate their adverse effects on trust assessment in trust networks by introducing innovative techniques integrated with our proposed model.

In summary, our contribution is a novel malicious aware trust assessment (MATA) for explicit trust prediction in dynamic complex networks such as the social networks and IoT networks. MATA addresses two main challenges of traditional trust prediction methods that are their failure to account for the dynamic nature of trust relationships over time and their lack of consideration for malicious behaviors. MATA achieves accurate prediction of pairwise trust while effectively characterizing trust properties and accounting for malicious behaviors. The proposed approach is based on an extended GCN model integrated with RNN (e.g., GRU) and attention mechanisms to jointly capture time dependencies and trace nodes behavior over time. Additionally, the proposed method involves clustering nodes based on their embeddings in each snapshot and assigning a reputation value to each node. Pairwise trustworthiness is obtained by using a feedforward neural network. Throughout all stages of the model, the primary focus is placed on effectively accounting for malicious behavior.

The rest of this paper is organized as follows: Section 2 reviews the related work. Section 3 introduces our network model as well as the adversarial model. Additionally, it presents the proposed trust prediction model in detail. Performance evaluation and further discussions about the proposed system are provided in Section 4. Finally, Section 5 concludes the paper.

2. Related work

2.1. Trust evaluation

The concept of trust assessment has attracted significant attention in the literature. We provide a brief review of pairwise trust prediction models. A common approach taken in the prior research is to establish paths between the trustor and trustee, propagate trust along these paths, and aggregate the propagated values to estimate trust. Golbeck (2005) introduced TidalTrust for web-based social networks. She examined the trust network structure in a real network called FilmTrust and investigated aspects such as the distribution of trust values, the relationship between trust rank and accuracy, and the relationship between path length and accuracy. Based on these findings, the TidalTrust algorithm was proposed to infer the trust level between two nodes who do not have a direct friendship.

Trust relationships can be formalized using subjective logic (Josang, 2001). For instance, in Liu et al. (2017), trust opinions are utilized instead of trust values, and the topology of the trust network is modeled using an opinion matrix. This approach employs the breadth-first search method to establish trust relationships between nodes who do not have a direct connection. Trust propagation and aggregation are then modeled through discounting and combining operators applied to the opinion matrix. NeuralWalk (Liu et al., 2019) is a deep neural network-based solution that learns the discounting and combining operators for trust opinions among nodes. NeuralWalk iteratively utilizes the WalkNet component, which is trained, to perform single-hop trust inference for unknown trust relations. Based on the results of single-hop trust inference in each iteration, the algorithm traverses the entire STN to conduct a multi-hop trust assessment. However, it should be noted that this approach may suffer from computational and memory complexity, which may limit scalability in real-world networks.

In recent years, node embedding has emerged as a popular topic for learning low-dimensional representations of graph elements, facilitating subsequent processing tasks such as node classification and link prediction. Consequently, several node embedding-based methods have been proposed for trust prediction in trust networks. STNE (Xu et al., 2019) is a skip-gram based model that incorporates negative sampling to jointly learn latent factor features and trust transfer pattern features. Another notable work is Guardian (Lin et al., 2020), a GCN-based model and the current state-of-the-art method in deep learning-based trust prediction. Guardian employs GCN to learn trust propagation and aggregation rules, enabling the evaluation of trust relationships between nodes. By stacking multiple trust convolutional layers, Guardian can simultaneously capture both the social graph structure and trust relationships. It is important to note that both Guardian and STNE assume a static trust network and do not explicitly address the detection of malicious nodes. GATrust (Hamilton et al., 2017) shares similarities with Guardian as it also employs GCN to capture the aggregation and propagation rules of trust. However, a key difference is that GATrust utilizes Graph Attention Network (GAT) as the first layer to assign distinct attention coefficients to various aspects of nodes, such as node features, network structure information, and trust information. It is important to note that GATrust is a static approach and does not account for the dynamic properties of trust. TrustGNN (Huo et al., 2022) extends knowledge graph embedding methods to GNN to better model the propagative pattern of trust, and aggregate the information propagated from different chains. TrustGNN also neglects the dynamic nature of trust and is designed for a single snapshot.

To address the limitation of ignoring the dynamic property of trust in previous approaches, Medley (Lin & Li, 2021) leverages dynamic nature of trust for time-aware trust prediction by incorporating graph attention mechanisms and functional time encoding into a unified framework. This method applies the functional time encoding to map the continues-time domain into a vector space. Furthermore, attention mechanisms are employed to capture varying weights for interactions that evolve over time. Medley, though promising for trust prediction, has limitations that need addressing, particularly its disregard for malicious nodes, a crucial aspect in trust assessment. Additionally, the system's high memory consumption due to the timestamped interaction model poses challenges when dealing with frequently updated graphs (Zhu et al., 2022).

Recently, TrustGuard (Wang et al., 2024) proposed a GNN-based explainable trust evaluation method with support for dynamicity. TrustGuard addresses dynamic trust by incorporating changes in the network structure over time employing a position-aware attention mechanism. However, there are several key differences between TrustGuard and our proposed method, MATA. While TrustGuard utilizes a GNN-based approach for trust evaluation with dynamicity support, MATA specifically integrates temporal dependencies using a combination of GCN and GRU. This allows MATA to better capture the evolving nature of trust relationships over time. Moreover, MATA employs a more advanced and adaptive approach to defending against malicious attacks in trust networks. By utilizing a similarity-based attention mechanism, MATA dynamically assigns weights to neighbors based on their similarity, ensuring that the model focuses on the most relevant and trustworthy nodes. This significantly enhances robustness against recommendation attacks. Additionally, MATA's reputation assessment system clusters node embeddings to assign dynamic reputation values, allowing the model to continuously monitor and adjust trust assessments based on real-time observed behaviors. In contrast, TrustGuard relies on a more static approach, using cosine similarity to calculate node similarity and pruning edges below a predefined threshold. While effective, this method lacks the dynamic adaptability of MATA and may not be as responsive to rapid changes in node behavior or the emergence of new malicious patterns.

2.2. *Temporal GCN and GAT*

GCN and its extensions, learns the graph representation by combine structural and content features. The study in (Hamilton et al., 2017) proposes a framework, called GraphSAGE, that generalizes the GCN approach for inductive node embedding. The core idea behind this approach is the training of a set of aggregator functions that learn how to aggregate feature information from a node's local neighborhood.

Attention mechanisms allow the model to assign a relevance score to elements in the graph and ignore noisy ones (Lee et al., 2018; Veličković et al., 2017). GAT (Veličković et al., 2017), in contrast to GCN, assigns implicitly different importance to nodes in the same neighborhood by learning attention weights for them. Another type of graph attention is proposed in Thekumparampil et al. (2018), in which more attention is given to objects that share more similar hidden representations or features.

GCN and its extension generally focus on static graphs, while in real-world applications, graphs are dynamic and evolve over time. For example, in social networks, nodes extend their friendships over time, as a result, their representations should be updated to reflect their evolving social interactions. Furthermore, trustworthiness can fluctuate with new interactions and may even decay over time. Neglecting these evolving dynamics and the historical behavior of trustor/trustee diminishes the predictive efficiency of the model, hindering its ability to provide accurate trust predictions. To address this, future research should focus on developing dynamic graph representation techniques that can adapt to changing graph structures and incorporate historical trust information, enabling more effective trust prediction in evolving real-world applications.

In this direction, in recent years, some GCN-based methods have been proposed for dynamic graph representations. Since GCN is designed for structural and feature extraction and RNN is used for sequence modeling, its combination is an ideal method for dynamic graph representation learning. The works of Seo et al. (2018) and Manessi et al. (2020) use GCN to learn node embeddings and then feed them into GRU to learn the temporal features of node embeddings. EvolveGCN (Pareja et al., 2020) extends GCN to the dynamic setting by introducing a recurrent mechanism to update the GCN parameters instead of the node embeddings, for capturing the dynamism of the graphs. Although EvolveGCN maintains temporal information through evolving parameters, its link prediction performance is not outstanding.

2.3. *Anomaly detection in dynamic graphs with graph representation*

Anomaly detection aims to detect unusual samples which deviate from the majority of the data. More recently, deep learning methods have achieved improvements in anomaly detection in high-dimensional datasets. Dynamic graphs, in addition to structural information, also contain temporal information which makes anomaly detection on dynamic graphs more challenging. Ma et al. (2021) provide a systematic review of the deep learning-based techniques for graph anomaly detection. Since our goal is to develop a graph representation-based trust prediction method that is aware of the anomalous nodes, in the following, we review the two most relevant anomaly detection methods.

NetWalk (Yu et al., 2018) presents a graph representation-based technique for detecting anomalous nodes and edges in dynamic graphs using structural information. Node representations are learned using an autoencoder and are incrementally updated when new edges are added or removed. Based on the low-dimensional nodes representations, a clustering-based technique is employed to incrementally and dynamically detect network anomalies. However, NetWalk could detect anomalies in dynamic graphs, it simply updates edge representations without considering the patterns of long/short-term nodes and the structure of the graph over time. AddGraph (Zheng et al., 2019) attempts to learn the anomaly patterns by considering structural, content and temporal features using the temporal GCN combined with Gated Recurrent Units (GRU) with attention. It assigns an anomaly score to each edge in the temporal graph based on the nodes associated with it. After training, the scoring function identifies anomalous edges in the test data by assigning higher anomaly scores to them.

3. Proposed method

3.1. *Problem statement*

3.1.1. *Network model*

We denote the dynamic trust network as a sequence of graph snapshots $G = \{G^1, G^2, \cdots, G^T\}$, where each $G^t = (V^t, E^t, W^t)$ represents a snapshot at time t, with $t \in \{1, 2, \cdots, T\}$. V^t represents the set of nodes representing entities (e.g., users in social networks, devices in IoT networks), E^t represents the set of edges indicating relationships, and W^t

Table 1
Summary of notations.

Notation	Description
$N_{in}(u)$	The set of In-degree neighbors of node u
$N_{out}(u)$	The set of out-degree neighbors of node u
n	The total number of nodes
d	The dimension of initial embeddings for nodes
$\mathbb{C}$	The number of trust levels
$x_{in}(u)$	The incoming trust vector of node u
$x_{out}(u)$	The outgoing trust vector of node u
$x(u)$	The static trust embedding of node u
w_{uv}	The trustworthiness of node u to node v
α_{uv}	The attention weight of node u to node v
h_u	The time-aware trust embedding for node u
$h_{c,u}$	The context trust embedding for node u
$h_{(u,v)}$	The pairwise trust embedding of node u towards node v
s_v	The malicious score of node v
R_v	The reputation score of node v
$T_{(u,v)}$	The overall pairwise trustworthiness of node u towards node v

represents the set of weights that denote the trustworthiness of the relationships in the edge set. In the major trust networks, trustworthiness is typically represented as a discrete value. The number of these values is denoted by $\mathbb{C}$ which corresponds to the number of classes in our prediction problem. For instance, in the Advogato dataset, $\mathbb{C} = 4$ while in the Bitcoin dataset $\mathbb{C} = 2$. To incorporate scalar trustworthiness w_{uv} into our model, we employ one-hot encoding. This encoding scheme represents each unique trust value as a binary vector of fixed length $\mathbb{C}$, where only one element is set to 1 and the rest are set to 0.

In this study, each graph snapshot is considered as a directed weighted network. So, each node $u \in V^t$ will have in-degree neighbors $N_{in}(u)$ and out-degree neighbors $N_{out}(u)$. Without loss of generality, we assume that the graph at any time has n nodes. In each snapshot of time t, there is two input data $A^t \in \mathbb{R}^{n\times n}$ and $H^{t-1} \in \mathbb{R}^{n\times d}$, where the former is the adjacency matrix and the latter is the hidden state of the previous snapshot, each row of which is a d-dimensional embedding of a node.

It's worth mentioning that before we delve into the explanation of our model, trust exhibits several noteworthy properties that serve as the foundation for trust prediction models (Gao et al., 2021). Firstly, trust is propagative and aggregable. The propagative nature of trust implies that trust can be passed from one node to another, creating chains of trust that connect two nodes who may not be explicitly linked. The aggregable nature of trust means that trust needs to be aggregated when multiple chains of trust exist. Secondly, trust is asymmetric. This asymmetry property signifies that the trustworthiness between two nodes is not necessarily mutual, resulting in a directed trust network. Thirdly, trust is personalized, implying that the trustworthiness of node C from the perspective of node A could differ from that of node B. Finally, due to the evolving nature of interactions over time, trustworthiness can vary with new interactions and may decay over time (Nitti, Girau, & Atzori, 2014). Hence, trust is dynamic and time-dependent. Dynamic trust not only accounts for changing trust values but also includes the addition and removal of nodes within the network.

Table 1 shows the notations we use in this paper. Our goal is to predict pairwise trustworthiness of nodes who do not have direct relationship in the trust network. In order to achieve this, the paper incorporates structural, content, and temporal features during the learning process while considering malicious behaviors.

3.1.2. *Adversarial model*

Graph representation-based approaches for predicting trust suffer from the limitation of considering malicious behaviors that are not ignorable. There are some trust-related attacks that are done by malicious nodes. We cover two most important trust-related attacks in our predicting trust method:

- *Recommendation attack (bad-mouthing/ballot-stuffing attack):* Malicious nodes may deliberately declare false trust values about another node. This behavior is called the recommendation attack. If a node gives a low trust value to the service provided by a benign node, such an attack is called bad-mouthing. On the other hand, if a node intentionally records high trust value for a bad service provider, it is launching a ballot-stuffing attack.
- *Opportunistic attack:* Malicious nodes perform anomalous operations occasionally in this attack, but they perform normally most of the time in order to obtain a high trust value and conceal their long-term anomalous behavior. This deceptive tactic aims to increase the difficulty of detection and undermine the accuracy of trust assessment within the network.

The detection of these attacks is not straightforward and poses a challenge. It requires a thoughtful approach and a combination of various features including structural, temporal and content features. In this paper, we employ effective strategies to mitigate the detrimental impact of these attacks. We utilize techniques such as clustering the learned trust representation of nodes and implementing two distinct attention mechanisms, which will be further explained in the upcoming section.

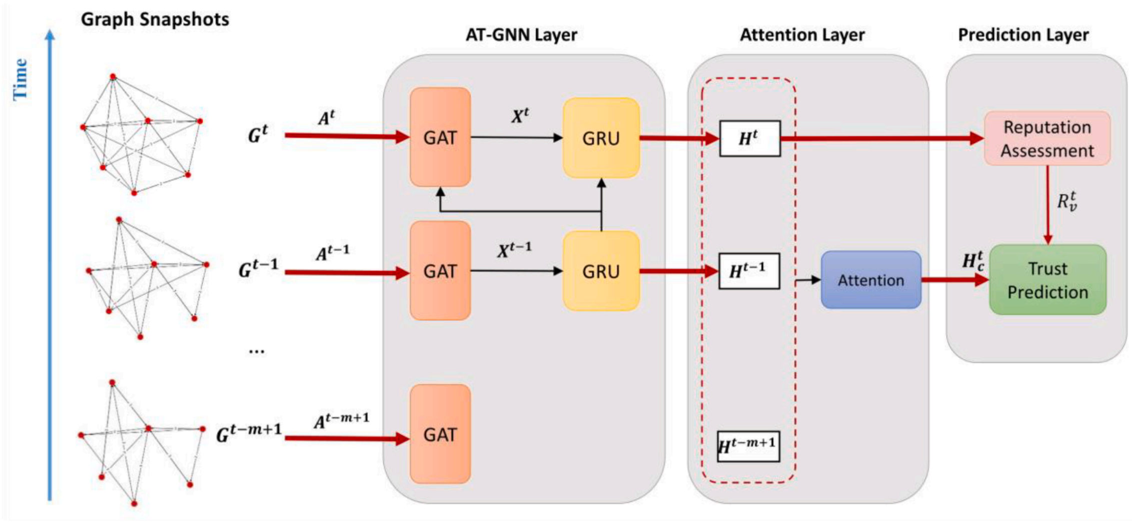

Fig. 1. Overall view of our MATA framework.

3.2. Method: MATA

Overall view of our method is illustrated in Fig. 1. As previously stated, our approach involves treating the input of the dynamic trust network as Discrete Time Dynamic Graph (DTDG). DTDG is a list of snapshots and each of them keeps the graph status at a certain moment (Zhu et al., 2022). Each snapshot contains an adjacency matrix and node embeddings. Our model contains three layers: 1) AT-GNN layer; 2) Trust attention layer; and 3) Trust prediction layer. AT-GNN extends the GCN technique to incorporate the examination of trust characteristics, during the propagation and aggregation of trust for more effective learning of nodes' trust latent factor. In this layer, GCN with attention mechanism is integrated with RNN for capturing both relational and temporal dependencies among nodes in the graph. By considering both structural proximity and node similarity, the model can effectively capture important patterns and relationships in aggregation of trust. To identify anomalous nodes who try to conceal their behavioral fluctuations over a long period, relying solely on the last hidden state is inadequate. Therefore, we incorporate an attention mechanism in the trust attention layer to consider recent hidden states with varying weights and creating short-term trust. After getting the hidden state H^t of all nodes, reputation assessment component in the prediction layer executes a clustering algorithm on the embedding space to measure the reputation of each node. Finally, trust prediction component first concatenates the node representations of the trustor and the trustee and then feeds it and the reputation to a fully connected layer followed by a softmax activation function to predict overall pairwise trustworthiness. The details of these layers are explained as follows.

3.2.1. AT-GNN layer

In this layer, we will enrich node embedding in such a way that it takes into account all the trust properties including propagative and aggregative ones, asymmetricity, personalization, and dynamicity. This comprehensive approach to enriching node embedding is referred to as Attentive and Temporal GNN (AT-GNN). In order to achieve this, we employ an elegant fusion of GCN with attention mechanism (GAT) along with a powerful recurrent architecture like GRU in each snapshot. This fusion of techniques results in an attentive and temporal GCN. This layer receives m snapshots of trust network data as input, which results in learning m hidden states H in the output. At each time snapshot t, the input data consists of the adjacency matrix $A^t \in \mathbb{R}^{n\times n}$ and the hidden state $H^{t-1} \in \mathbb{R}^{n\times d}$ at the time $t-1$. To summarize, the goal of this layer involves two steps:

$$X^t = GAT\left(A^t, H^{t-1}\right) \tag{1}$$

$$H^t = GRU\left(X^t, H^{t-1}\right) \tag{2}$$

In the following, we provide a detailed explanation of these two steps.

3.2.1.1. GCN with attention mechanism (GAT). Within this layer, we utilize GCN with an attention mechanism, known as Graph Attention Network (GAT), to explore the aforementioned trust properties (excluding dynamicity) and learn static trust embeddings $x(u)$ for each node. At first, we address the propagative, aggregative and asymmetric properties by adopting GCN with an approach similar to the ones mentioned in the literature (Jiang et al., 2023; Lin et al., 2020). GCN as a graph representation learning method encodes each node into an embedding vector by learning the structural and content features. A GCN includes multiple convolution layers that aggregate neighborhood information. Therefore, it is a good opportunity to use it for capturing propagative and aggregative properties of trust. In addition, trust is asymmetric, so the trust network is directed and each node plays two roles as a trustor or a trustee. Therefore, for each node u in the current snapshot, we generate two embedding vectors, an incoming trust vector $x_{in}(u)$, and an outgoing trust vector $x_{out}(u)$. The incoming trust means how much a node is trusted by others and the outgoing trust denotes how much a node as a trustor tends to trust others.

In each convolution layer of a pure GCN, nodes aggregate information from their neighbors. The weights assigned to the neighboring nodes are uniform, meaning that each neighbor has an equal influence on the aggregation process. On the other hand, GAT introduces attention weights α_{uv} to determine how much attention we want to give to a neighboring node v from node u's perspective. For addressing personalized recommendations, we use one type of attention named similarity-based attention to give more importance to nodes who share more similar representations or features (Lee et al., 2018; Thekumparampil et al., 2018). This is based on the assumption that nodes with similar features are likely to have similar taste about others compared to those with dissimilar features (collaborating filtering). This attention mechanism guides the model's attention towards nodes that are more likely to provide reliable trust information.

To illustrate this, the attention of node u to incoming node v in snapshot t and convolution layer ℓ, ($\alpha_{uv}^{\ell,t}$), can be learned via:

$$\alpha_{uv}^{\ell,t} = \frac{\exp\left(\beta.\cos\left(x_u^{\ell-1,t}, x_v^{\ell-1,t}\right)\right)}{\sum_{i\in N_{in}(u)}\exp\left(\beta.\cos\left(x_u^{\ell-1,t}, x_i^{\ell-1,t}\right)\right)} \tag{3}$$

where β is a trainable bias," cos " represents cosine-similarity, and $x_u^{\ell-1,t}$ is the learned hidden embedding of node u in the previous layer.

In addition of social relationships, there are trust information which should be considered in the aggregation process. We use linear transformation to map one-hot encoding of incoming trustworthiness w_{vu}^t into the embedding space by Eq. (4). By transforming the trustworthiness information into the same embedding space as the node features, we enable a seamless and effective aggregation process.

$$T_{vu}^t = W_{in}^t w_{vu}^t \tag{4}$$

where W_{in}^t is a trainable weight matrix that $W_{in}^t \in \mathbb{R}^{d\times\mathbb{C}}$.

Incoming trust embedding of node u in each convolution layer ℓ are formulated as aggregation of incoming neighbors' information that are weighted by both their associated trustworthiness and computed attention coefficient as follows:

$$x_{in}^{\ell,t}(u) = \sum_{i\in N_{in}(u)}\alpha_{ui}^t\left(x_i^{\ell-1,t}\otimes T_{iu}\right) \tag{5}$$

In a similar way, the attention of node u to outgoing node v' in snapshot t and convolution layer ℓ, can be expressed via:

$$\alpha_{uv'}^{\ell,t} = \frac{\exp\left(\beta.\cos\left(x_u^{\ell-1,t}, x_{v'}^{\ell-1,t}\right)\right)}{\sum_{i\in N_{out}(u)}\exp\left(\beta.\cos\left(x_i^{\ell-1,t}, x_u^{\ell-1,t}\right)\right)} \tag{6}$$

A linear transformation is used to map outgoing trustworthiness into the embedding space by:

$$T_{uv'}^t = W_{out}^t w_{uv'}^t \tag{7}$$

And then, *outgoing trust embedding* of node u is calculated as follows:

$$x_{out}^{\ell,t}(u) = \sum_{i\in N_{out}(u)}\alpha_{ui}^t\left(x_i^{\ell-1,t}\otimes T_{ui}\right)) \tag{8}$$

To obtain the **static trust embedding** of each node u in snapshot t, we first concatenate these two incoming and outgoing trust vectors and then apply an MLP as follows:

$$x^{\ell,t}(u) = \sigma\left(W^{\ell,t}.\left(x_{in}^{\ell}(u)\otimes x_{out}^{\ell}(u)\right) + b^{\ell,t}\right) \tag{9}$$

where $W^{\ell,t}$ represents a transformation matrix that should be trained, $b^{\ell,t}$ denotes a learnable bias, and σ indicates the non-linear activation function.

As a noteworthy point, the initial embedding H^0 can be derived either through one-hot encoding based on node degrees or by employing a random-walk method like node2vec (Grover & Leskovec, 2016). However, in feature-rich graphs we can use node features like node profile information. When processing each snapshot t in the first layer, the initial node embeddings $x_v^{0,t}$ are derived from the output h^{t-1} of the previous snapshot, except for the initial snapshot $t = 0$, where the node embeddings are directly initialized using the enriched features.

By employing multiple convolution layers, our approach not only captures structural features, but also enrich node embeddings with trust features from more similar neighborhoods. We show the final representation matrix for all nodes at depth L by X^t, where L represents the number of convolutional layers employed in our model.

3.2.1.2. GRU. To incorporate temporal features, we utilize GRU to create **time-aware trust embeddings** H^t for the nodes. By applying GRU we generate embeddings that encode the dynamic nature of trust relationships among nodes. GRU was introduced as a simplified variant of LSTM. The design of GRU aims to simplify the architecture while maintaining similar performance to LSTM. The formulation of GRU can be represented as follows:

$$Y^t = \sigma(W_{xr}X^t + b_{xr} + W_{hr}H^{t-1} + b_{hr}) \tag{10}$$

$$Z^t = \sigma(W_{xz}X^t + b_{xz} + W_{hz}H^{t-1} + b_{hz}) \tag{11}$$

$$\widetilde{H}^t = tanh(W_{xh}X^t + b_{xh} + Y^t * W_{hh}H^{t-1} + b_{hh})) \tag{12}$$

$$H^t = (1 - Z^t) * \widetilde{H}^t + Z^t * H^{t-1} \tag{13}$$

where Y^t and Z^t are the reset and update gates respectively, while $\widetilde{H}^t$ is the candidate hidden state. The weight matrices and bias vectors are denoted by W and b, respectively and σ is the sigmoid activation function. These equations describe how the GRU updates its hidden state to capture the dynamic nature of trust relationships over time.

3.2.2. *Attention layer*

This layer employs an attention mechanism to deal with monotonic assumption of GRU. Instead of solely relying on the last trust embedding, this mechanism allows us to consider the influence of recent trust embeddings. By incorporating attention, the model can assign varying weights to several recent time steps, enabling a more robust detection of short-term behavioral patterns. This improvement plays a vital role in effectively identifying opportunistic attacks, as the model becomes more adept at capturing and analyzing subtle variations in node behavior over shorter time intervals. This layer calculates context trust embedding that attend m recent hidden states as follows:

$$H_c^t = ATT(H^{t-m+1}; \cdots; H^t) \tag{14}$$

We use a contextual attention-based model proposed by Cui et al. (2017) to model nodes' context trust. The calculation details of context trust embedding are shown in Eqs. (15)–(18).

$$C_{h,v}^t = \left[h_v^{t-m+1}; \cdots; h_v^t\right] \qquad C_{h,v}^t \in \mathbb{R}^{m\times d} \tag{15}$$

$$e_{h,v}^t = p^T tanh\left(Q_h\left(C_{h,v}^t\right)^T\right) \qquad e_{h,v}^t \in \mathbb{R}^m \tag{16}$$

$$a_{h,v}^t = softmax\left(e_{h,v}^t\right) \qquad a_{h,v}^t \in \mathbb{R}^m \tag{17}$$

$$h_{c,v}^t = \left(a_{h,v}^t C_{h,v}^t\right)^T \qquad h_{c,v}^t \in \mathbb{R}^d \tag{18}$$

where h_u^t is the time-aware trust embedding of node u, Q_h and p are weights for attention and $h_{c,u}^t$ is the context trust embedding of node u which catches the recent trust information. This process should be done for all nodes to acquire the output **context trust embedding** matrix H_c^t.

3.2.3. *Prediction layer*

The prediction layer in the model consists of two inputs: the last hidden state and the context trust embedding. These inputs are learned network representation that can be useful for consequent tasks. We serve the last hidden state input to assess the malicious score and subsequently determine the reputation. On the other hand, the context trust embedding input is utilized for primary trust prediction. To arrive at the overall trustworthiness, the reputation assessment and the primary trust assessment are combined. we provide further elaboration and detail on these processes in the following.

3.2.3.1. Reputation assessment. We assess the reputation score of nodes based on the last hidden state to reveal their acceptance by others over time. This is achieved by employing DBSCAN clustering algorithm (Ester et al., 1996) to group nodes' representations into clusters in the latent space in each snapshot. At first, we determine nodes malicious score by analyzing the placement of their representations within the clusters. For example, if a node does not stay in any clusters in the majority snapshots will very likely indicate certain type of maliciousness (Yu et al., 2018). To address this issue, we utilize the following approach:

We assign a binary value r_v^t to each node v at time t, where 1 indicates that the node is not placed in any clusters (noisy node), and 0 indicates placement. When a node representation is not placed in any cluster, it can have two possible meanings: either the node is a malicious node, isolated from others in the trust embedding space, or it is a newly added node. However, these two cases cannot be determined solely based on the current snapshot; further analysis of subsequent snapshots is required. Let r_v^t represent the outcome of a Bernoulli experiment with success probability θ_v^t. θ_v^t can be modeled using a Beta distribution $Beta(\alpha_v, \beta_v)$, where the parameters are updated as follows:

$$\alpha_v^t = e^{-\varphi}.\alpha_v^{t-1} + r_v^t \tag{19}$$

$$\beta_v^t = e^{-\gamma}.\beta_v^{t-1} + 1 - r_v^t \tag{20}$$

Here, φ and γ are decay factors over time for an exponentially-weighted moving average. We specially set $\gamma > \varphi$, to ensure that malicious forgetting occurs at a slower pace. The value of α_v^t represents the weighted number of times the node was not placed in any clusters, while β_v^t is related to number of times the node was placed in a cluster. The malicious score of node v (s_v) at time t is calculated as the expected value of θ_v^t, *i.e.*,

$$s_v^t = E\left(\theta_v^t\right) = \frac{\alpha_v^t}{\alpha_v^t + \beta_v^t} \tag{21}$$

We set the initial value of α_v^0 and β_v^0 to 0.5. In the end, we define the reputation of each node by:

$$R_v^t = 1 - s_v^t \tag{22}$$

Obviously R_v resides within the range of 0 and 1.

3.2.3.2. Trust prediction. The paper addresses overall trustworthiness through two key components: direct trust latent factor and reputation. Direct trust is determined by considering both long-term and short-term trust information of the trustor and trustee. On the other hand, reputation focuses on the global behavior of the trustee in the trust latent space, which is calculated based on last hidden state.

In order to obtain direct trust latent factor among any two nodes (u, v), we concatenate the learned representations of those nodes into a single vector. This concatenated vector is then passed through a standard fully connected layer alongside the reputation value of node v to enhance the representation of the trust relationship, facilitating trust prediction. The equation for this process is as follows:

$$h^t_{(u,v)} = \sigma\left(W_{dt}\cdot\left(h^t_{c,u}\otimes h^t_{c,v}\right) + W_r.R^t_v + b\right) \tag{23}$$

In this equation, W_{dt} represents the weight parameter matrix associated with the concatenated trust latent factor, W_r is the weight matrix associated with the reputation input R^t_v, b is the bias vector for the fully connected layer, σ denotes the softmax function and $h^t_{(u,v)}$ is the pairwise trust embedding.

To determine the pairwise trustworthiness of node u towards node v at time t, we utilize the following equation:

$$T_{(u,v)} = argmax(h^t_{(u,v)}) \tag{24}$$

This equation allows us to identify the trust level that has the maximum probabilistic value within the pairwise trust embedding $h^t_{(u,v)}$. By employing this approach, we can effectively predict and evaluate the trust level between any two nodes in a given network.

This approach allows us to incorporate the reputation of a node as a factor that influences the predicted trust.

3.2.4. Model training

In order to train our model and learn the parameters of it, we apply a cross-entropy loss function. This function tunes the model's parameters via minimizing difference between the predicted values and the ground-truth trustworthiness. We employed the Adam SGD optimizer as our model's optimizer. Furthermore, we implemented an early stopping strategy to optimize training efficiency and prevent overfitting.

4. Experiments

In this section, we elaborate on the different data sets utilized, outline the baseline models employed for comparative analysis, and present the outcomes derived from these comparisons.

4.1. Data description

We use three real-world datasets for evaluation of MATA, named Advogato,[1] Bitcoin-Alpha[2] and Bitcoin-OTC.[3] The details of these datasets are here:

4.1.1. Advogato

The Advogato dataset is a well-known dataset in the field of trust and reputation systems. It was collected from the Advogato online community platform, which is an online social network for free and open-source software developers. It includes relationships between users, and trust values assigned by users to indicate their level of trustworthiness which signify four distinct levels of trustworthiness: Observer, Apprentice, Journeyer, and Master. To facilitate evaluation, we have chosen a subset of daily snapshots from the extensive collection.

4.1.2. Bitcoin-alpha/OTC

These datasets consist of a collection of transactions involving Bitcoin, where users participate in the transfer of funds and assign positive (trust) or negative (distrust) trustworthiness rate to each other. The users involved in these transactions remain anonymous throughout the dataset, therefore a trust network between users can be helpful in having more secure transactions. In addition to user transactions and trustworthiness ratings, the dataset includes a timestamp for each entry, reflecting the dynamic nature of the data.

We preprocess the bitcoin dataset in order to create snapshots. At first, we arrange the edges in the order of their timestamp and determine the number of snapshots to be divided. The first half of the edges, sorted by timestamp, is selected as the first snapshot and so the time value of the edges in this snapshot is set to 0. After doing it, according to the determined number of snapshots (we set it to 16 snapshots), we divide the second half into equal parts. The time value for each edge within a snapshot is set to the same value, indicating their belonging to the same snapshot. By following these steps, the Bitcoin dataset is preprocessed, resulting in multiple snapshots that represent different periods of time, with edges arranged in chronological order within each snapshot. We have used these two datasets for evaluation and comparison in both scenarios: without the presence of attacks and in the presence of attacks.

The statistics of these datasets in the last snapshot is summarized in Table 2.

Table 2
Statistics of datasets.

	#Nodes	#Edges
Advogato	2853	28,135
Bitcoin-Alpha	3783	24,186
Bitcoin-OTC	5881	35,592

4.2. Experiment setup

For the implementation of our proposed model, we utilized PyTorch. To generate initial embeddings for each node, we employed node2vec (Grover & Leskovec, 2016). Across all datasets, we maintained a fixed embedding dimension of 100. The number of GCN layers set to 4. We split datasets into 70/15/15 parts along the time as training/validation/test sets. The learning rate is 0.005. All the experiments were performed on a machine with Intel Core i5 4-core 1.6 GHz CPU, 16 GB RAM, and 500 GB SSD.

4.3. Baselines

In our evaluation, we compare MATA with three baseline models, all of which are based on GNNs. Among these baselines, one is a static trust prediction model, while the other two are dynamic models. One of the dynamic models does not take into account trust properties, whereas the other is specifically designed as a trust model.

Guardian (Lin et al., 2020). This approach is a trust prediction model for online social networks. Guardian focuses on a single snapshot of interaction graph, disregarding its time-varying features. It also assigns equal weights to the neighbors during the aggregation of information. A single GCN model is used for all time steps, and the loss is accumulated over the time to apply this static approach on the snapshot-based datasets.

EvolveGCN (Pareja et al., 2020). This method uses a single GCN model as a feature extractor with a recurrent model (GRU and LSTM) to regulate GCN parameters at every time step. It offers a dynamic GCN model, but it does not take into account trust features in generating node embeddings. Additionally, similar to Guardian, this method assigns equal weights to neighbors during the aggregation phase.

Medley (Lin & Li, 2021). This approach is a time-aware trust prediction model. This framework applies the functional time encoding to represent the social interactions and the timing information into a latent space. The core of Medley is the attentive-trust propagation network, which is equipped with attention mechanisms to capture various importance weights for interactions that evolve over time.

4.4. Results

Performance. The performance of all baselines is assessed on the test set using three significant evaluation metrics, average precision (AP), F1-weighted and area under ROC curve (AUC). The F1-weighted strikes a balance between precision and recall, offering a

[1] https://www.trustlet.org/datasets/advogato/.
[2] https://snap.stanford.edu/data/.
[3] https://snap.stanford.edu/data/.

Table 3
Accuracy assessment on Advogato dataset.

	Guardian	EvolveGCN	Medley	MATA
F1-weighted	0.772	0.747	0.788	0.884
AP	0.61	0.821	0.835	0.907

Table 4
Accuracy assessment on Bitcoin-Alpha dataset.

	Guardian	EvolveGCN	Medley	MATA
F1-weighted	0.821	0.791	0.919	0.953
AP	0.845	0.811	0.892	0.936
AUC	0.854	0.823	0.904	0.941

Table 5
Accuracy assessment on Bitcoin-OTC dataset.

	Guardian	EvolveGCN	Medley	MATA
F1-weighted	0.817	0.775	0.869	0.933
AP	0.836	0.788	0.841	0.927
AUC	0.843	0.792	0.864	0.937

comprehensive assessment of a model's performance. AUC is a measure of a model's ability to distinguish between classes across all threshold levels. For the binary classification datasets (Bitcoin-Alpha and Bitcoin-OTC), we used all three metrics, including AUC. However, for Advogato with four classes, we did not use AUC due to its complexities and potential for producing less interpretable results. Instead, we focused on AP and F1-weighted, which are more suitable for evaluating multiclass datasets.

A maximum epoch of 100 is set for all experiments. Tables 3–5 present the outcomes across different datasets. MATA exhibited the highest accuracy among the methods tested. This outcome highlights our approach's effective utilization of trust-related information. Guardian and EvolveGCN rank the lowest in performance metrics. Guardian disregards the essential property of dynamism in trust, while EvolveGCN captures temporal dependencies; however, it wasn't specifically designed for trust prediction. On the other hand, Medley takes into account trust-related properties, such as dynamism. It employs an attention mechanism to assign greater weights to recent social interactions, yet it ignores giving attention toward more similar social interactions that could contribute to a powerful trust representation.

Attack resistance. In order to assess the system's resistance to trust-related attacks, we manually inject anomalous nodes into the two Bitcoin datasets due to the unavailability of suitable attack data. We randomly selected 5 % and 10 % of nodes as attackers. For recommendation-based attacks, attackers provide fake ratings to their neighbors. The assignment of positive or negative trust in the injected edges depended on the average rating of the neighbor, with positive averages leading to assigned distrust and vice versa. In the case of opportunistic attacks, we select attackers from the pool of trustworthy nodes, as this enables a more accurate simulation of such attacks. trustworthy nodes refer to those nodes with a positive average trust value. Opportunistic attackers do malicious behavior at time t, where $t = 10, 12$ and at other times, it behaves normally. Here, we show the accuracy of the model's prediction on the final time snapshot. To demonstrate the resistance of the MATA model against attacks, we compare it with baselines in two distinct modes. The first mode represents the main version of MATA, which includes all the original malicious-aware techniques integrated into our model. For the second mode, we intentionally exclude these malicious-aware techniques from the model. These techniques, which are deliberately omitted in the second mode, encompass:

1. Removing similarity-based attention for recommendation attacks (MATA-SimAtt)
2. Removing contextual attention for opportunistic attacks (MATA-ContAtt)
3. Removing reputation for opportunistic attacks (MATA-Rep)

In Figs. 2 and 3, we observe the resilience of our methods against various attacks. The opportunistic attack is a time-dependent attack; therefore, our model cannot be compared with the Guardian method for this type of attack. Notably, MATA outperforms the other baseline methods, including its own version without malicious-aware techniques. In contrast, the performance of the other baseline methods is compromised by these attacks due to their lack of attack defense solutions. For recommendation attacks (bad-mouthing/ballot stuffing attacks) in

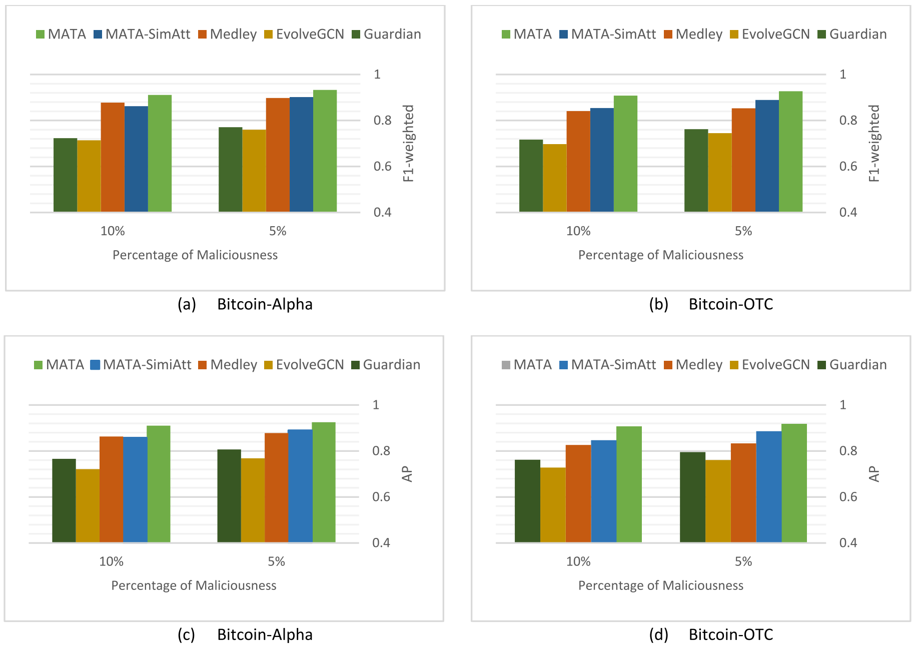

Fig. 2. Attack resistance for various percentages of maliciousness under recommendation attacks.

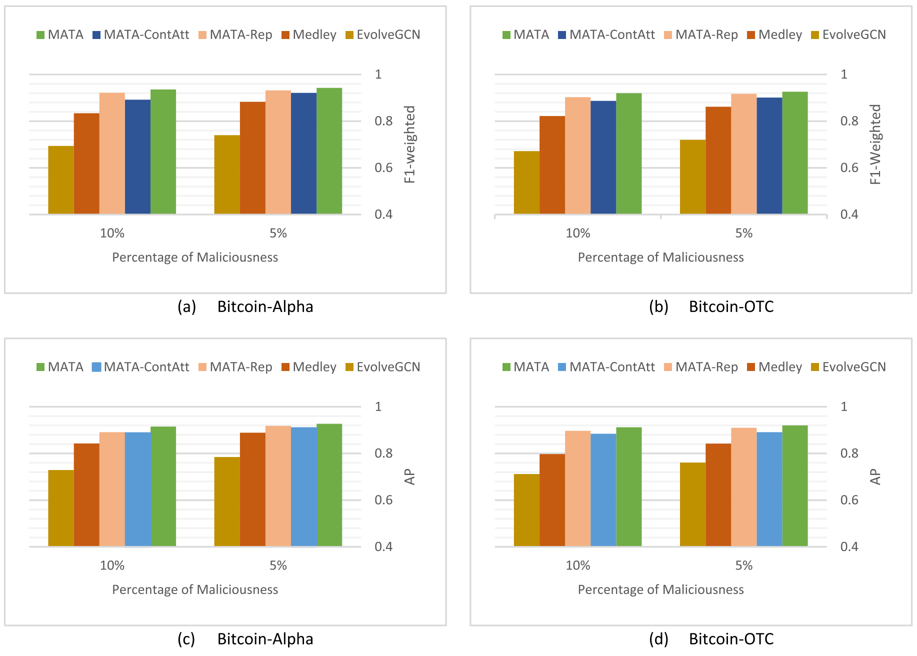

Fig. 3. Attack resistance for various percentages of maliciousness under opportunistic attacks.

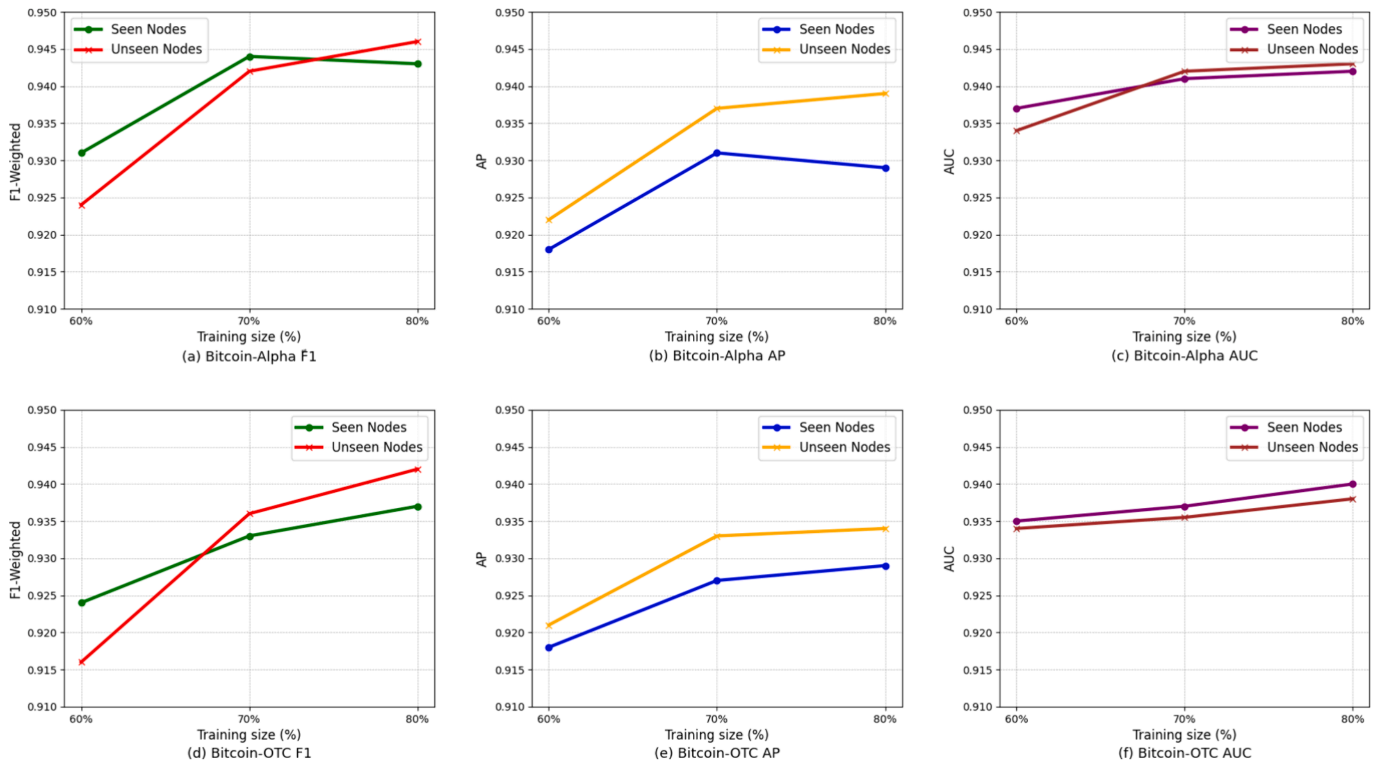

Fig. 4. Performance on seen and unseen nodes vs. percentages of the training set.

Fig. 2, MATA exhibits a notable improvement over MATA-SimAtt. For the Bitcoin-Alpha dataset under 10 % of maliciousness, it shows a 4.9 % increase in F1-weighted and a 5.2 % increase in AP. Similarly, for the Bitcoin-OTC dataset against 10 % of maliciousness, it achieves a 5.4 % improvement in F1-weighted and a 5.91 % increase in AP. This validates the effectiveness of our attention mechanism during the aggregation phase under the assumption that more similar friends during the aggregation leads to learning a more robust and efficient trust embeddings.

In MATA, we propose two effective techniques against opportunistic attacks: contextual attention and reputation. Removing either of these techniques results in a reduction in our model's resilience to opportunistic attacks, as illustrated in Fig. 3. It's noteworthy that, when evaluating the model with F1-weighted and AP metrics, the performance of the MATA method experiences a negligible decrease in the presence of attacks compared to scenarios without any attacks. In other words, our proposed techniques for reducing the consequences of opportunistic attacks have managed to yield expected effect. This is because, contextual attention is used to learn the significance of recent snapshots when creating nodes' trust embeddings. Meanwhile, the reputation value reflects changes in nodes' behavior, with a longer-lasting memory of their negative actions compared to their trustworthy behavior.

Generalization capability evaluation: This section aims to evaluate the model's ability to generalize to unseen data that was not part of the training set. By testing the model on previously unseen data, we can assess how well the model can apply the learned patterns to new and previously unobserved instances. This is crucial for evaluating the model's performance in making accurate predictions on real-world data that may include nodes or entities not present in the training dataset.

To this end, we applied trust prediction to two categories of nodes: nodes that were not seen during the training phase and nodes that were seen. This comparison was conducted across different percentages of the training and testing sets on the Bitcoin-Alpha and Bitcoin-OTC datasets. As shown in Fig. 4, the performance of the MATA model on unseen nodes does not significantly differ across various training set sizes. This indicates that our model has successfully learned meaningful patterns from the training data and can generalize to new data.

5. Conclusion

In response to the limitations of existing trust prediction models, we propose a novel approach called MATA (Malicious-Aware Trust Assessment), which aims to overcome the challenges of trust prediction in dynamic complex networks in the presence of malicious behavior. MATA utilizes a fusion of an extended GCN along with RNNs and an attention mechanism to effectively capture time dependencies and trace nodes' behavior over time in the evolving graph. By integrating these components, MATA can generate more comprehensive and accurate trust representations for nodes, enabling us to assign reputation values and dynamically monitor their behaviors for precise and malicious-aware trust predictions. The experimental results demonstrate that our proposed model, MATA, surpasses state-of-the-art methods and effectively mitigates the negative impact of malicious behavior. We have examined MATA on several real-world datasets, and this evaluation showcases MATA's robustness and effectiveness in trust prediction, even in the presence of malicious nodes. As future work, we intend to focus on identifying different types of attacks by examining the learned representations of the graph across various snapshots.

CRediT authorship contribution statement

Besat Jafarian: Conceptualization, Methodology, Software, Validation, Writing – original draft, Writing – review & editing, Visualization. **Nasser Yazdani:** Project administration, Writing – review & editing, Validation, Supervision. **Mohammad Sayad Haghighi:** Conceptualization, Methodology, Writing – review & editing, Validation, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The authors do not have permission to share data.

References

Arabsorkhi, A., Sayad Haghighi, M., & Ghorbanloo, R. (2016). A conceptual trust model for the Internet of Things interactions. In *8th international symposium on telecommunications* (pp. 89–93). https://doi.org/10.1109/ISTEL.2016.7881789

Chen, I. R., Guo, J., Wang, D. C., Tsai, J. J. P., Al-Hamadi, H., & You, I. (2019). Trust-based service management for mobile cloud IoT systems. *IEEE Transactions on Network and Service Management, 16*, 246–263. https://doi.org/10.1109/TNSM.2018.2886379

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. http://arxiv.org/abs/1406.1078.

Cui, Q., Wu, S., Huang, Y., Wang, L. (2017). A hierarchical contextual attention-based GRU network for sequential recommendation. http://arxiv.org/abs/1711.05114.

Ebrahimi, M., Tadayon, M. H., Haghighi, M. S., & Jolfaei, A. (2022). A quantitative comparative study of data-oriented trust management schemes in Internet of Things. *ACM Transactions on Management Information Systems, 13*, 1–30. https://doi.org/10.1145/3476248

Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD-96, KDD-96* (pp. 226–231).

Fan, W., Ma, Y., Li, Q., He, Y., Zhao, E., Tang, J., & Yin, D. (2019). Graph neural networks for social recommendation. In *The web conference 2019 – proceedings of the world wide web conference, WWW 2019* (pp. 417–426). Association for Computing Machinery, Inc. https://doi.org/10.1145/3308558.3313488.

Gao, X., Xu, W., Liao, M., & Chen, G. (2021). Trust prediction for online social networks with integrated time-aware similarity. *ACM Transactions on Knowledge Discovery from Data, 15*. https://doi.org/10.1145/3447682

Golbeck, J. A. (2005). *Computing and applying trust in web-based social networks* (Ph.D. dissertation). College Park, MD, USA: Dept. Comput. Sci., Univ. Maryland.

Grover, A., & Leskovec, J. (2016). node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 855–864). New York, NY, USA: ACM. https://doi.org/10.1145/2939672.2939754.

Hamilton, W. L., Ying, R., Leskovec, J. (2017). Inductive representation learning on large graphs. http://arxiv.org/abs/1706.02216.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*, 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Huo, C., Jin, D., Liang, C., He, D., Qiu, T., Wu, L. (2022). TrustGNN: Graph neural network based trust evaluation via learnable propagative and composable nature. http://arxiv.org/abs/2205.12784.

Jafarian, B., Yazdani, N., & Sayad Haghighi, M. (2020). Discrimination-aware trust management for social internet of things. *Computer Networks, 178*, Article 107254. https://doi.org/10.1016/j.comnet.2020.107254

Jiang, N., Wen, J., Li, J., Liu, X., & Jin, D. (2023). GATrust: A multi-aspect graph attention network model for trust assessment in OSNs. *IEEE Transactions on Knowledge and Data Engineering, 35*, 5865–5878. https://doi.org/10.1109/TKDE.2022.3174044

Josang, A. (2001). A logic for uncertain probabilities. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 09*, 279–311. https://doi.org/10.1142/S0218488501000831

Kipf, T. N., Welling, M. (2016). Semi-supervised classification with graph convolutional networks. http://arxiv.org/abs/1609.02907.

Lee, J. B., Rossi, R. A., Kim, S., Ahmed, N. K., Koh, E. (2018). Attention models in graphs: A survey. http://arxiv.org/abs/1807.07984.

Lin, W., Gao, Z., Li, B. (2020). Guardian: Evaluating trust in online social networks with graph convolutional networks. http://networkrepository.com/arenas.

Lin, W., & Li, B. (2021). Medley: Predicting social trust in time-varying online social networks. In *IEEE INFOCOM 2021 – IEEE conference on computer communications* (pp. 1–10). IEEE. https://doi.org/10.1109/INFOCOM42981.2021.9488814.

Liu, G., Chen, Q., Yang, Q., Zhu, B., Wang, H., & Wang, W. (2017). OpinionWalk: An efficient solution to massive trust assessment in online social networks. In *IEEE INFOCOM 2017 – IEEE Conference on Computer Communications* (pp. 1–9). IEEE. https://doi.org/10.1109/INFOCOM.2017.8057106.

Liu, Q., Wu, S., Wang, L. (2016). Multi-behavioral sequential prediction with recurrent log-bilinear model. http://arxiv.org/abs/1608.07102.

Liu, G., Li, C., & Yang, Q. (2019). NeuralWalk: Trust assessment in online social networks with neural networks. In *IEEE INFOCOM 2019 – IEEE conference on computer communications* (pp. 1999–2007). IEEE. https://doi.org/10.1109/INFOCOM.2019.8737469.

Ma, X., Wu, J., Xue, S., Yang, J., Zhou, C., Sheng, Q. Z., Xiong, H., Akoglu, L. (2021). A comprehensive survey on graph anomaly detection with deep learning. doi:10.1109/TKDE.2021.3118815.

Manessi, F., Rozza, A., & Manzo, M. (2020). Dynamic graph convolutional networks. *Pattern Recognition, 97*, Article 107000. https://doi.org/10.1016/j.patcog.2019.107000

Massa, P., & Avesani, P. (2007). Trust-aware recommender systems. In *Proceedings of the 2007 ACM conference on recommender systems* (pp. 17–24). New York, NY, USA: ACM. https://doi.org/10.1145/1297231.1297235.

Nitti, M., Girau, R., & Atzori, L. (2014). Trustworthiness management in the social internet of things. *IEEE Transactions on Knowledge and Data Engineering, 26*, 1253–1266. https://doi.org/10.1109/TKDE.2013.105

Nitti, M., Girau, R., Atzori, L., Iera, A., & Morabito, G. (2012). A subjective model for trustworthiness evaluation in the social Internet of Things. In *2012 IEEE 23rd International Symposium on Personal, Indoor and Mobile Radio Communications – (PIMRC)* (pp. 18–23). IEEE. https://doi.org/10.1109/PIMRC.2012.6362662.

Pareja, A., Domeniconi, G., Chen, J., Ma, T., Suzumura, T., Kanezashi, H., Kaler, T., Schardl, T., & Leiserson, C. (2020). EvolveGCN: Evolving graph convolutional networks for dynamic graphs. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 5363–5370). https://doi.org/10.1609/aaai.v34i04.5984

Seo, Y., Defferrard, M., Vandergheynst, P., Bresson, X. (2018). Structured sequence modeling with graph convolutional recurrent networks. pp. 362–373. doi:10.1007/978-3-030-04167-0_33.

Thekumparampil, K. K., Wang, C., Oh, S., Li, L. -J. (2018). Attention-based graph neural network for semi-supervised learning. http://arxiv.org/abs/1803.03735.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y. (2017). Graph attention networks. http://arxiv.org/abs/1710.10903.

Wang, J., Yan, Z., Lan, J., Bertino, E., & Pedrycz, W. (2024). TrustGuard: GNN-based Robust and Explainable Trust Evaluation with Dynamicity Support. *IEEE Transactions on Dependable and Secure Computing*, 1–18. https://doi.org/10.1109/TDSC.2024.3353548

Xu, P., Hu, W., Wu, J., Liu, W., Du, B., & Yang, J. (2019). Social trust network embedding. In *Proceedings – IEEE international conference on data mining, ICDM* (pp. 678–687). Institute of Electrical and Electronics Engineers Inc.. https://doi.org/10.1109/ICDM.2019.00078

Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W. L., & Leskovec, J. (2018). Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the ACM SIGKDD international conference on knowledge discovery and data mining, association for computing machinery* (pp. 974–983). https://doi.org/10.1145/3219819.3219890

Yu, W., Cheng, W., Aggarwal, C. C., Zhang, K., Chen, H., & Wang, W. (2018). NetWalk: A flexible deep embedding approach for anomaly detection in dynamic networks. In *Proceedings of the ACM SIGKDD international conference on knowledge discovery and data mining, association for computing machinery* (pp. 2672–2681). https://doi.org/10.1145/3219819.3220024

Zheng, L., Li, Z., Li, J., Li, Z., & Gao, J. (2019). AddGraph: Anomaly detection in dynamic graph using attention-based temporal GCN. In *Proceedings of the twenty-eighth international joint conference on artificial intelligence, international joint conferences on artificial intelligence organization, California* (pp. 4419–4425). https://doi.org/10.24963/ijcai.2019/614

Zhu, Y., Lyu, F., Hu, C., Chen, X., Liu, X. (2022). Encoder-decoder architecture for supervised dynamic graph learning: A survey. http://arxiv.org/abs/2203.10480.