RAMA: Resistance-Aware Multi-Hop Aggregation Graph Representation Learning for Robust Ethereum Account Classification

Zhaowei Liu , Zengyang Zhang , Haitao Yang , Yao Shan *,
Xiangfu Zhao, Yongchao Song
Computer and Control Engineering College
Yantai University
{lzw, shanyao, ycsong}@ytu.edu.cn
{zhangzengyang, yanghaitao}@s.ytu.edu.cn
xiangfuzhao@gmail.com

Abstract

Fraudulent behaviors in Ethereum, such as Ponzi schemes and phishing attacks, pose persistent threats to blockchain security. Classifying high-risk accounts is crucial for mitigating these threats. Given the graph-structured nature of Ethereum transactions, Graph Neural Networks (GNNs) are widely adopted for this task. However, conventional GNNs rely on local aggregation and treat all edges equally, which amplifies structural noise and limits the ability to capture weak but semantically meaningful connections. This limitation arises from the inability to distinguish informative links from spurious ones during message passing, especially in noisy and long-range transaction graphs. To address this, we propose RAMA, a Resistance-Aware Multi-hop Aggregation framework that reweights edges using resistance distance to reduce the influence of noisy links while preserving important topological information. It also employs attention-based multi-hop aggregation to effectively integrate both local and global transaction patterns. Experiments on real-world Ethereum dataset show that RAMA consistently outperforms strong baselines in Macro-F1 and remains robust under edge perturbations, demonstrating its effectiveness for practical account classification in blockchain environments.

1 Introduction

Ethereum is a decentralized blockchain platform supporting Turing-complete smart contracts [1], enabling the development of composable decentralized applications without centralized control [2]. Its transparent environment, account-based model, and permissionless architecture allow open participation [3], fostering flexible interactions among heterogeneous protocols and pseudonymous users [4]. As a result, Ethereum has become a core infrastructure for decentralized finance and digital assets, attracting a large and rapidly expanding global developer and user base.

However, this openness and pseudonymity also create fertile ground for malicious activity [5]. According to a December 2023 report by Immunefi, cryptocurrency users collectively lost about $1.8 billion due to hacks and fraud. These incidents mainly arose from smart contract bugs, poor access control, and phishing attacks, posing serious threats to asset security and the integrity of the blockchain ecosystem. This escalating trend highlights the urgent need for effective account classification methods to detect and mitigate high-risk behaviors within Ethereum's transaction network.

*Corresponding Author

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

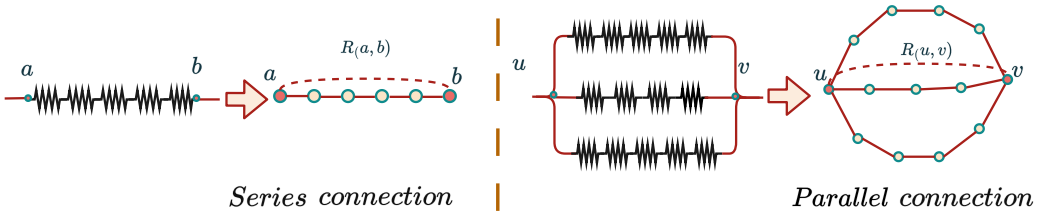


Figure 1: Two illustrative examples of effective resistance computation. For a pair of nodes u and v connected via multiple vertex-disjoint paths p, the effective resistance $R_{u,v}$ can be approximated by the reciprocal of the sum of the reciprocals of their lengths: $R_{u,v} = (\sum_{\text{uv-paths } p} \text{length}(p)^{-1})^{-1}$ Left: $R_{a,b} = 5$, equal to the total length of the single path between a and b. Right: $R_{u,v} = 20/13$, reflecting the effect of multiple redundant paths between u and v, which reduces the overall resistance.

A promising direction is transaction network-based account classification [6, 7, 8, 9], which models accounts as nodes and transactions as directed edges, framing the problem as node classification over a large-scale, heterogeneous graph.

Existing approaches fall into two categories: handcrafted feature-based methods [10, 11, 12] and graph neural network models [13, 14, 15]. Feature-based methods extract metrics such as transaction volume, degree, and balance changes, then use conventional classifiers. Although efficient and interpretable, they often miss complex or obfuscated behaviors. In contrast, GNNs learn node representations by aggregating information from local neighborhoods, capturing both transactional dynamics and structural context.

Despite progress, two core challenges persist. First, most GNNs adopt uniform or shallow aggregation, making it difficult to distinguish meaningful interactions from noisy or routine transactions, and limiting their ability to model long-range dependencies [16, 17, 18, 19]. Some methods address noise via heuristic edge filtering, but this may discard weak yet semantically important links, potentially undermining behavioral signals. Second, fixed-hop aggregation restricts the receptive field, causing models to overlook subtle multi-hop dependencies crucial for identifying complex schemes like laundering and coordinated scams [20, 21]. Without principled multi-hop fusion, models risk oversmoothing or missing key patterns.

To address these issues, this work proposes RAMA, a Resistance-Aware Multi-hop Aggregation framework grounded in electrical network theory. Effective resistance, derived from circuit theory and random walk models [22, 23], offers a principled way to measure topological strength between nodes by considering the complete path structure. As illustrated in Figure 1, this metric enables continuous edge reweighting without hard filtering, effectively suppressing spurious links while retaining semantically meaningful ones. RAMA further integrates attention-based multi-hop aggregation to capture behavioral patterns across diverse relational distances, enabling the model to focus on critical paths such as multi-step laundering and dispersed scams. Together, these components enhance the robustness of representation learning in the face of noisy, irregular, and obfuscated transaction structures within Ethereum.

The principal contributions of this paper are summarized as follows:

- This study proposes RAMA, a graph neural network framework that incorporates resistance distance–based adjacency reweighting and hop-aware multi-hop aggregation for Ethereum account classification.
- A resistance-aware feature augmentation module leverages resistance distances between account pairs to guide message passing while preserving the transaction graph's structural integrity.
- To capture local and global relational patterns, a multi-hop aggregation mechanism integrates features from multiple hops, mitigating oversmoothing and enhancing representation.
- Experimental results validate RAMA's effectiveness, demonstrating consistent superiority over state-of-the-art baselines with up to 2.01% improvement in Macro-F1 score.

2 Related Work

On the basis of the work presented in this paper, a concise summary of the most relevant work on Account Classification on Ethereum, Effective resistance in Graph Neural Network is provided.

2.1 Account Classification on Ethereum

Graph-based modeling is now the de facto standard for Ethereum account classification, treating accounts as nodes and transactions as directed edges to capture complex behavioral patterns. Current research primarily spans three directions: shallow structural modeling, temporal or relational extensions, and data imbalance mitigation. Shallow models, such as GCN-based approaches [13] and FA-GNN [14], utilize basic topology and edge filtering to stabilize local features, yet often fail to capture the long-range dependencies inherent in obfuscated fraud.

Temporal and relational methods like TSGN [8] and PEAE-GNN [15] enrich edge semantics using timestamps and ego-centric motifs, but remain constrained by fixed-hop propagation. To address data skewness, EthGAN [24] and standardized benchmarks by Yang et al. [25] improve evaluation quality but do not fundamentally advance structural representation. RAMA addresses these collective limitations by replacing static filtering and fixed-hop aggregation with resistance-aware edge reweighting and adaptive multi-hop attention, significantly enhancing structural expressiveness in noisy environments.

2.2 Effective Resistance in Graph Neural Networks

Effective resistance [26], rooted in electrical circuit theory, serves as a robust global proximity metric that accounts for all possible paths between nodes, reflecting commute times and spectral properties. In the GNN domain, resistance has been integrated to guide message propagation in Affinity-GNN [27], improve long-range flow through graph rewiring [28, 29], and identify low-conductance subgraphs for clustering [30].

However, most existing methods modify the graph topology by adding or removing edges, which can compromise the semantic integrity of transactional networks where edge existence represents specific behavioral intent. Furthermore, these designs typically apply resistance as a static modification rather than an adaptive component of the learning process. In contrast, RAMA treats effective resistance as a continuous reweighting signal, preserving the original topology while enriching node features with global structural context. Combined with attention-based aggregation, our approach selectively prioritizes informative paths and improves resilience against the structural obfuscation prevalent in Ethereum networks.

3 Preliminaries

This section establishes the theoretical foundations of RAMA, focusing on effective resistance in directed graphs and its application in guiding adaptive message passing.

3.1 Effective Resistance in Directed Graphs

Effective resistance measures the ease of information flow between nodes. In a weighted directed graph $G = (V, E, W)$, let $d_G = \sum_i \sum_j W_{ij}$ be the graph volume, $P = D^{-1}W$ the transition matrix, and $\Pi = \mathrm{Diag}(\pi)$ the diagonal matrix of the stationary distribution π.

Definition 1 (Directed Laplacian and Resistance) *The directed Laplacian is defined as* $L = d_G\Pi(I - P)$. *The effective resistance between nodes* u *and* v *is given by:*

$$\Omega(u, v) = (\mathbf{e}_u - \mathbf{e}_v)^\top L^\dagger (\mathbf{e}_u - \mathbf{e}_v) = \frac{1}{d_G} C(u, v), \tag{1}$$

where $L^\dagger$ *is the Moore-Penrose pseudoinverse of L, and* $C(u, v)$ *denotes the expected commute time between* u *and* v *in a random walk.*

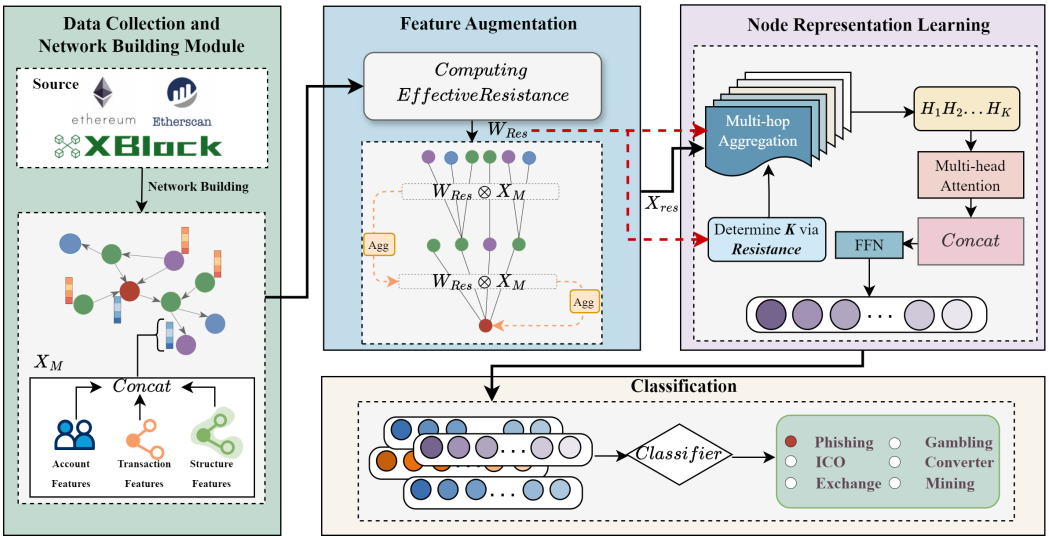


Figure 2: Original features, positional encodings, and clustering coefficients are concatenated to initialize representations. The Effective Resistance Feature Augmentation module then reweights the adjacency matrix via resistance distances, guiding the Multi-Hop Aggregation module to fuse hop-wise features via attention for final Ethereum account classification.

Definition 2 (Resistance-Guided Affinity) *To highlight globally coherent and directionally consistent node pairs, we define the resistance-guided affinity matrix* $\widetilde{A}$ *as:*

$$\widetilde{A}_{uv} = \exp(-\lambda \cdot \Omega(u,v)^{\gamma}), \quad \lambda > 0, \tag{2}$$

where λ *and* γ *are hyperparameters controlling the scale and sensitivity of the structural similarity.*

3.2 Resistance-Guided Hop Truncation

To mitigate over-smoothing and over-squashing in deep GNNs, we introduce a truncation mechanism that adaptively restricts the receptive field based on structural decay.

Definition 3 (Average Hop Affinity) *Let* $\mathcal{P}_k = \{(u,v) \mid dist(u,v) = k\}$ *be the set of node pairs whose shortest path length is exactly* k*. The average affinity at hop* k *is:*

$$\mathbb{E}[W_{\text{res}}^{(k)}] = \frac{1}{|\mathcal{P}_k|} \sum_{(u,v)\in\mathcal{P}_k} \widetilde{A}_{uv}. \tag{3}$$

Definition 4 (Optimal Hop Depth) *Given a structural tolerance parameter* $\tau > 0$*, the adaptive truncation depth* $K^\star$ *is defined as the maximum hop distance where the cumulative affinity decay remains within* τ*:*

$$K^\star = \max\{k \mid \mathbb{E}[W_{\text{res}}^{(1)}] - \mathbb{E}[W_{\text{res}}^{(k)}] \le \tau\}. \tag{4}$$

This truncation rule ensures that message passing is restricted to structurally significant neighborhoods, effectively filtering out distant and potentially noisy interactions that do not contribute to the semantic consistency of the transaction graph.

4 Methodology

This section introduces RAMA, a framework designed to address structural noise and long-range dependencies in Ethereum account classification. As shown in Figure 2, the architecture integrates resistance-based feature augmentation to refine topology via edge reweighting, and multi-hop attention aggregation to fuse multi-scale features. This design preserves informative weak links while mitigating oversmoothing, enabling the robust capture of local and global transaction patterns.

4.1 Problem Formulation

In this paper, the Ethereum account classification task is formulated as a graph node classification problem in a multi-class setting. Let the Ethereum transaction network be represented as $G = (V, E, X, Y)$, where V denotes the set of nodes corresponding to Ethereum addresses, and E represents directed edges indicating transactions. Each node $v \in V$ is associated with a feature vector, forming a node feature matrix $X \in \mathbb{R}^{|V| \times d}$, where d is the feature dimension. The label matrix is denoted as $Y \in \mathbb{R}^{|V| \times C}$, with C representing the number of account categories. The objective is to learn a classification function $f : V \to \mathcal{Y}$, where $\mathcal{Y} = \{y_1, y_2, \ldots, y_C\}$. For each node $v \in V$, the predicted label is $f(v) = y_c$, with $y_c \in \mathcal{Y}$. Training is performed in a semi-supervised setting, where labels are available only for a subset of nodes $V_L \subset V$.

4.2 Dataset and Feature Construction

The Ethereum transaction graph is constructed from the XBlock dataset [13], comprising 1,402,220 nodes and 2,815,028 edges. As detailed in Appendix A, the dataset includes 816 labeled accounts across six classes and provides 12 raw features per node. To capture structural signals, the raw feature matrix X is augmented with global Laplacian positional encodings $U \in \mathbb{R}^{n \times s}$ [31] and local clustering coefficients C. The final enhanced feature matrix X_M is obtained via:

$$X_M = X \parallel U \parallel C. \tag{5}$$

where $\parallel$ denotes concatenation, yielding a representation $X_M \in \mathbb{R}^{n \times d'}$ that fuses both semantic attributes and topological contexts.

4.3 Effective Resistance Feature Augmentation Module

This module captures the underlying global structural information of the graph by leveraging effective resistance as a measure of connectivity. Figure 3 illustrates the computation procedure, a central component in our framework. The computed resistance distances are used to adaptively reweight the adjacency matrix, emphasizing structurally important connections while preserving the original topology.

Effective resistance is a well-established metric that quantifies the ease of information flow between nodes. Unlike direct adjacency which is sensitive to individual spurious edges, effective resistance aggregates information over all possible paths. According to electrical network theory, the presence of multiple parallel paths reduces the overall resistance. Thus, sporadic noisy edges have minimal impact compared to structurally consistent connectivity, providing a theoretical basis for robustness. To compute this for a target node v, a random walk is performed on the original graph G to construct a subgraph G_s containing v and its associated higher-order neighbors, thereby capturing both direct and indirect interactions. Finally, the Laplacian of this subgraph is computed, and the effective resistance is derived from its Moore-Penrose pseudoinverse:

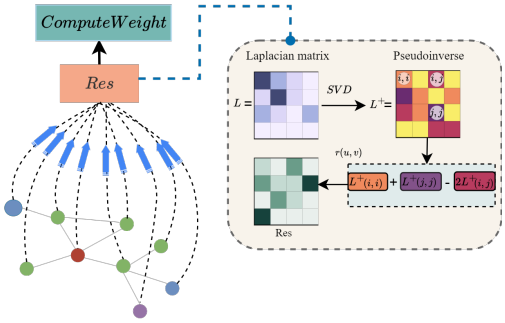


Figure 3: The procedure for computing effective resistance. By extracting a local subgraph via random walks and deriving the Laplacian pseudoinverse, the framework calculates the resistance distance $\Omega(v, u)$ between node pairs, providing a robust global proximity metric for subsequent resistance-aware edge reweighting.

$$L_s^+ = \text{pinv}(L_s), \tag{6}$$

where L_s^+ is the Moore-Penrose pseudoinverse of the Laplacian matrix for the subgraph G_s. The effective resistance $\Omega(v, u)$ between any two nodes v and u is then defined as:

$$\Omega(v, u) = L_s^+(i, i) + L_s^+(j, j) - 2L_s^+(i, j), \tag{7}$$

where i and j are the indices of nodes v and u in the Laplacian matrix. This formula calculates the effective resistance between nodes, quantifying their structural relationship in terms of global connectivity as defined in Section 3.1.

Once the resistance distance between node pairs is computed, it is used to reweight the adjacency matrix. To control the influence of the resistance distance in the subsequent neighbor feature aggregation process, we introduce two hyperparameters $\lambda > 0$ and $\gamma > 0$, and a small constant ϵ for numerical stability. The normalized weight $w(v, u)$, reflecting the strength of the connection between nodes v and u, is defined as:

$$w(v,u) = \frac{\exp\left[-\lambda(\Omega(v,u)+\epsilon)^{\gamma}\right]}{\sum_{u' \in \mathcal{N}_{G_s}(v)} \exp\left[-\lambda(\Omega(v,u')+\epsilon)^{\gamma}\right]}, \tag{8}$$

where $\mathcal{N}_{G_s}(v)$ represents the set of neighbors of node v in G_s, and ϵ is a numerical offset used only in the nonlinear mapping from the effective resistance $\Omega(v, u)$ to the unnormalized weight. These weights are then used to reweight the adjacency matrix, which ensures that shorter and stronger connections (indicated by smaller resistance distances) have higher weights. The enhanced feature representation for the target node v is then obtained by concatenating its input feature $X_M[v]$ with the aggregated neighbor features:

$$x'_v = \text{Concat}\left(X_M[v], \sum_{u \in \mathcal{N}_{G_s}(v)} w(v,u) X_M[u]\right), \tag{9}$$

where x'_v is the new feature representation for node v. The concatenation step ensures that the target node's feature is enriched with information from its neighbors, weighted by their effective resistance distance, which helps to capture both local and global structural information. The complete algorithmic procedure for this module is summarized in Appendix B.

4.4 Resistance-Guided Multi-Hop Aggregation Module

To capture complex relational dependencies in Ethereum transaction networks, a resistance-weighted multi-hop aggregation module is introduced. Building upon the enhanced features $X_{\text{res}} \in \mathbb{R}^{n \times 2d'}$ obtained from resistance-aware augmentation, this module constructs hop-wise features using the normalized resistance-weighted adjacency matrix $\hat{A}_{\text{res}}$. These hop-wise features allow the model to incorporate both local and global dependencies in the graph. The multi-hop features are computed as follows:

$$X_k = \hat{A}^k_{\text{res}} X_{\text{res}}, \quad k = 0, 1, \ldots, K^{\star}, \tag{10}$$

where $K^{\star}$ is the maximum hop depth guided by resistance-aware truncation. Specifically, the effective hop depth is determined from the decay of average resistance-weights at different hop distances, ensuring that message passing is truncated when further propagation no longer contributes meaningfully:

$$K^{\star} = \max\left\{k \mid \mathbb{E}[W^{(1)}_{\text{res}}] - \mathbb{E}[W^{(k)}_{\text{res}}] \leq \tau\right\}. \tag{11}$$

Here, $W^{(k)}_{\text{res}}$ denotes the k-hop resistance-weighted adjacency statistics, and $\mathbb{E}[W^{(k)}_{\text{res}}]$ is the average affinity over node pairs whose shortest-path distance is exactly k. This mechanism controls the receptive field by focusing on semantically meaningful neighborhoods, while avoiding high-resistance paths that are likely to introduce noise.

The features for each node v across different hops are then projected into a hidden space to form a sequence of representations. Let $z^{(k)}_v = X_k[v]\mathbf{W}_e \in \mathbb{R}^{d_m}$ denote the representation of node v at hop k, where $\mathbf{W}_e$ is a trainable projection matrix. The sequence is assembled as:

$$Z^{(0)}_v = \left[z^{(0)}_v, z^{(1)}_v, \ldots, z^{(K^{\star})}_v\right]. \tag{12}$$

This sequence captures the evolution of features across hop distances, enabling the model to distinguish between short- and long-range transactional patterns. A multi-layer aggregation process with residual connections and non-linear transformations is applied to model cross-hop dependencies and refine intermediate representations. Final node embeddings are obtained via attention-based fusion across hop-wise features. Specifically, each k-hop embedding $z_v^{(k)}$ is weighted relative to the 0-hop embedding $z_v^{(0)}$:

$$\alpha_{v,k} = \frac{\exp(\text{LeakyReLU}(\mathbf{a}^\top [z_v^{(0)} \parallel z_v^{(k)}]))}{\sum_{j=0}^{K^\star} \exp(\text{LeakyReLU}(\mathbf{a}^\top [z_v^{(0)} \parallel z_v^{(j)}]))}, \tag{13}$$

$$h_v = \sum_{k=0}^{K^\star} \alpha_{v,k} z_v^{(k)}. \tag{14}$$

This hop-level attention mechanism allows the model to assign adaptive importance to different structural scales. In Ethereum, where fraudulent behaviors often span multiple indirect paths, this aggregation mechanism enhances the model's ability to identify subtle long-range interactions and improves classification accuracy for both typical and minority account types.

4.5 Complexity Analysis

RAMA computes exact Laplacian pseudoinverses on sampled local subgraphs. Let s_v denote the number of nodes in the local subgraph sampled for target node v. Dense pseudoinversion requires $O(s_v^3)$ time and $O(s_v^2)$ working memory per subgraph, yielding a cumulative time complexity of $O(\sum_{v \in V} s_v^3)$ across all target nodes. The local pseudoinverses are discarded after the resistance weights are obtained. For a fixed graph, node features, and sampled subgraphs, the resulting resistance-weighted adjacency matrix $\hat{A}_{\text{res}}$ and augmented feature matrix X_{res} are precomputed and reused throughout training. Thus, the pseudoinversion cost is incurred during preprocessing rather than at every training epoch.

Multi-hop feature construction proceeds through the recurrence $X_{k+1} = \hat{A}_{\text{res}} X_k$. Let m_{res} denote the number of nonzero entries in $\hat{A}_{\text{res}}$ and d_{res} the feature dimension of X_{res}. Each propagation step involves a sparse-dense matrix multiplication with a time complexity of $O(m_{\text{res}} d_{\text{res}})$. Computing features up to hop $K^\star$ therefore requires $O(K^\star m_{\text{res}} d_{\text{res}})$ time. This propagation cost excludes the subsequent projection, attention, and classification operations.

5 Experiment

This section presents a comprehensive evaluation of the proposed RAMA model on an Ethereum dataset. It includes experimental setup, comparison with baseline methods, ablation studies on key components, and sensitivity analysis of critical hyperparameters to validate the model's effectiveness and robustness.

5.1 Experimental Setup

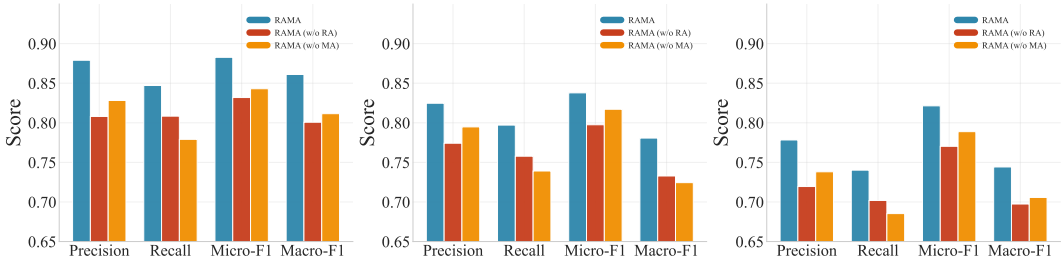
In this subsection, the datasets, baseline, and experimental specifics utilized in the experiment are described.

Dataset and Metrics: To ensure comparability with existing studies, the data splitting strategy proposed in [14] is adopted. Specifically, as detailed in Appendix A, three experimental splits are constructed, where 70%, 50%, and 30% of the labeled nodes are randomly selected as training data for Split 1, Split 2, and Split 3, respectively. The entire Ethereum transaction graph is utilized for representation learning. Following prior work, Precision, Recall, Micro-F1, and Macro-F1 are employed as evaluation metrics to comprehensively assess model performance.

Baselines and Implementation Details: RAMA is compared with comprehensive baselines across two categories: (i) General-purpose GNNs, including *GCN* [32], *GAT* [33], *GraphSAGE* [34], *Cluster-GCN* [35], H_2GCN [36], and the recent robust framework *GRANCE* [37]; and (ii) Blockchain-specific

Table 1: Performance Comparison of Methods on Ethereum Account Classification. In each column, the best result is indicated in bold, while the runner-up result is marked with an underline.

	Split 1				Split 2				Split 3			
Method	Prec	Rec	Micro-F1	Macro-F1	Prec	Rec	Micro-F1	Macro-F1	Prec	Rec	Micro-F1	Macro-F1
GCN	0.4549	0.5280	0.5440	0.4684	0.5972	0.6215	0.6648	0.6039	0.4952	0.5694	0.5726	0.5204
GAT	0.6984	0.6699	0.7366	0.6822	0.6768	0.6389	0.7592	0.6576	0.7086	0.6389	0.7301	0.6436
GraphSAGE	0.7181	0.7414	0.7526	0.7249	0.6974	0.6699	0.7319	0.6822	0.7284	0.6699	0.7730	0.6500
Cluster-GCN	0.7287	0.7003	0.7630	0.7043	0.6661	0.6696	0.7501	0.6636	0.6665	0.6488	0.7364	0.6514
H_2GCN	0.6596	0.6449	0.7432	0.6522	0.6312	0.6218	0.7137	0.6263	0.6290	0.6394	0.6772	0.5829
GRANCE	0.7925	0.7850	0.8245	0.7885	0.7540	0.7410	0.7980	0.7470	0.7150	0.7120	0.7750	0.7135
Trans2vec	0.7234	0.6949	0.7662	0.6972	0.6732	0.6439	0.7180	0.6585	0.6376	0.6038	0.6546	0.6206
HNRL	0.7537	0.7564	0.8235	0.7491	0.7359	0.7267	0.7720	0.7273	0.7124	0.7224	0.7425	0.7090
ABGRL	0.7743	0.7664	0.8145	0.7705	0.7359	0.7262	0.7859	0.7313	0.7015	0.7020	0.7651	0.7016
FA-GNN	0.8453	0.8345	**0.8860**	0.8343	0.7933	0.7772	0.8144	0.7372	**0.7843**	0.6939	0.7886	0.7182
ETGNN	0.8655	0.8110	0.8685	0.8380	0.7760	0.7720	0.8150	0.7610	0.7520	0.7050	0.8010	0.7250
RAMA (Ours)	**0.8790**	**0.8471**	0.8827	**0.8610**	**0.8250**	**0.7974**	**0.8386**	**0.7811**	0.7786	**0.7405**	**0.8223**	**0.7446**



(a) Split 1 (b) Split 2 (c) Split 3

Figure 4: Ablation study of RAMA for Ethereum account classification. The performance of the full RAMA model is evaluated against variants lacking resistance augmentation (w/o RA) and multi-hop aggregation (w/o MA) under three different training data proportions.

models, including *Trans2vec* [38], *HNRL* [39], *ABGRL* [40], *FA-GNN* [14], and the topological model *ETGNN* [41]. Training configurations and hyperparameters for RAMA and all baselines are detailed in Appendix C, and the performance comparison justifying the selection of the downstream classifier is provided in Appendix C.3.

5.2 Classification Experiment Results and Analysis

The performance of RAMA on node classification tasks for the Ethereum network is summarized in Table 1. The experimental outcomes indicate several key trends.

General GNNs, such as GCN, GAT, and GraphSAGE, underperform in Ethereum transaction networks due to their reliance on local neighborhood aggregation and the homophily assumption, which conflicts with the networks' heterophily and noise. Even recent robust frameworks like GRANCE, while mitigating noise via contrastive refinement, still lack the domain-specific inductive bias required for complex financial flows.

To address these limitations, blockchain-specific models like Trans2vec, HNRL, and ABGRL leverage temporal dynamics and heterogeneous node types for better adaptability. However, their heuristic structures often lead to cross-split generalization instability and poor performance on imbalanced data. More recently, ETGNN has advanced this domain by leveraging persistent homology to capture structural holes, emerging as a strong model. Yet, its fixed topological features lack the flexibility to adapt to diverse fraud patterns across different data splits compared to learnable representations. Consequently, these methods may struggle to generalize against sophisticated, evolving attack patterns that deviate from static topological norms.

Table 2: Per-class classification performance of RAMA under Split 1.

Account type	Precision	Recall	F1
Converter	1.0000	0.9643	0.9818
Exchange	0.8056	0.8529	0.8286
Gambling	0.7500	0.6000	0.6667
ICO	0.8000	0.8485	0.8235
Mining	1.0000	0.9167	0.9565
Phish/Hack	0.9184	0.9000	0.9091

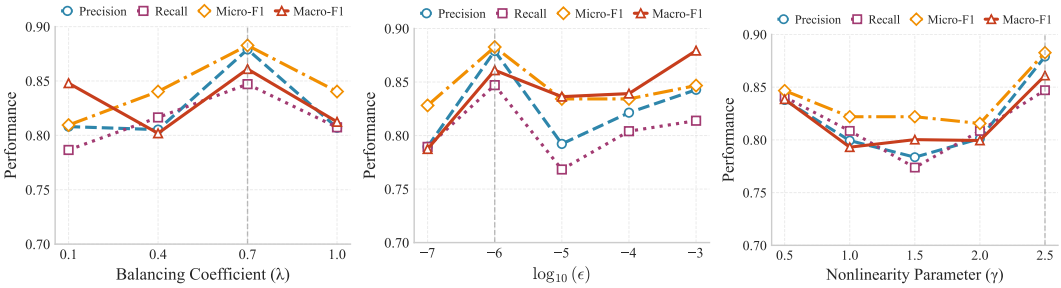


Figure 5: Effect of Resistance Distance Parameters

Comparatively, RAMA outperforms most baselines. Unlike the rigid topology in ETGNN, RAMA employs resistance augmentation to reweight edges. This simulates flow reliability to emphasize significant connections while suppressing noise, transforming the graph into a more informative structure. Furthermore, the Multi-Hop Aggregation module captures long-range dependencies via context-sensitive attention, mitigating oversmoothing and contributing to competitive overall classification performance, yielding discriminative embeddings resilient to structural perturbations.

The class-wise results in Table 2 show that RAMA achieves F1 scores above 0.80 for five of the six account categories under Split 1. Converter and Mining exhibit the strongest performance, combining perfect precision with high recall. RAMA also achieves high precision and recall for Phish/Hack, demonstrating effective identification of phishing and hacking accounts. In contrast, Gambling exhibits the lowest precision and recall, with the latter showing a more pronounced shortfall. This identifies missed Gambling accounts as a specific challenge for the model.

5.3 Ablation Analysis

To assess the contribution of each core component in RAMA, we evaluate two ablation variants: RAMA w/o RA, which removes resistance-based feature augmentation, and RAMA w/o MA, which retains the hop-wise features but replaces attention-based aggregation with mean pooling. As shown in Figure 4, the full model consistently outperforms both variants across all evaluation splits, demonstrating the contribution of both components under different training data proportions. Removing resistance augmentation leads to performance drops of up to 4.4% in Micro-F1 and 5.4% in Macro-F1. This degradation highlights the value of incorporating resistance-derived connectivity information into feature augmentation. In Ethereum transaction graphs, relevant behavioral signals may extend through sparse or indirect connections. Resistance-aware reweighting accounts for the surrounding path structure when aggregating neighboring features, enriching node representations with structural context. The results suggest that these connectivity cues are beneficial for account classification.

Replacing attention-based aggregation with mean pooling results in a more pronounced Macro-F1 drop of up to 7.5%, alongside a Recall drop of up to 7.8%. Since hop-wise features are retained, this comparison highlights the importance of adaptive fusion across hop levels. Mean pooling assigns equal importance to different hop-wise representations, whereas attention adjusts their contributions according to the node context. This flexibility allows the model to integrate information from different

relational distances without treating all hop levels as equally informative. Together, these results highlight the complementary roles of resistance-aware structural reweighting and multi-hop attention aggregation in capturing complex and long-range transaction patterns for robust Ethereum account classification. Detailed ablation settings and metric scores are provided in Appendix D.

5.4 Sensitivity Analysis

We evaluate the sensitivity of RAMA to core hyperparameters. Regarding the resistance parameters shown in Figure 5, optimal performance is obtained with $\lambda = 0.7$, $\epsilon = 10^{-6}$, and $\gamma = 2.5$. These findings indicate that moderate edge reweighting effectively mitigates topological noise while preserving structurally significant weak ties; deviations from this range risk compromising performance by either over-filtering semantic information or retaining irrelevant connections. This underscores the necessity of carefully calibrating λ to align with graph sparsity. Further sensitivity analyses concerning the number of attention heads and the learning rate η, which respectively demonstrate the model's capacity to capture richer multi-hop patterns and the optimal balance for convergence speed, are detailed in Appendix E.

5.5 Robustness Analysis

The robustness of RAMA under noisy conditions is evaluated by randomly adding 10%, 20%, and 30% spurious edges to the Ethereum transaction graph. As shown in Figure 6, traditional models like GCN and GraphSAGE suffer substantial performance degradation as noise increases, particularly in Macro-F1, due to indiscriminate neighborhood aggregation. Advanced frameworks like ABGRL and ETGNN exhibit improved resilience via self-supervised filtering and persistent homology, respectively. However, their performance still deteriorates under high-noise regimes due to a lack of dynamic noise-aware adaptability. In contrast, RAMA consistently achieves superior stability. Its resistance distance-based edge reweighting identifies and suppresses spurious links, while the Multi-Hop Aggregation module captures informative long-range dependencies. Together, these components enable RAMA to effectively filter out structural noise and preserve high-quality node representations. Detailed evaluations on other metrics (Micro-F1, Precision, and Recall) showing consistent trends are provided in Appendix F.

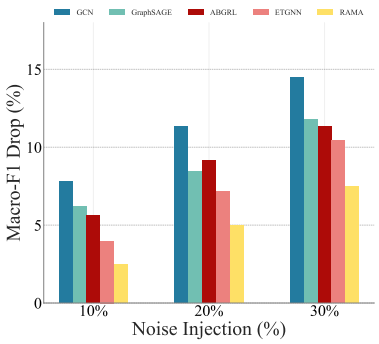


Figure 6: Robustness evaluation (Macro-F1 Drop) of RAMA and baselines under structural noise, simulated by the random addition of spurious edges.

6 Conclusion

This paper presents RAMA, a resistance-aware multi-hop aggregation framework for robust Ethereum account classification. By coupling effective resistance-based edge reweighting with adaptive multi-hop attention, RAMA effectively captures complex structural patterns and long-range dependencies. Experiments demonstrate consistent superiority over baselines, confirming its robustness and generalization, particularly under high noise and label scarcity. While empirically strong, RAMA's limitations regarding computational overhead and static graph assumptions, along with future directions, are discussed in Appendix G.

Acknowledgements

We thank all the reviewers and the area chair for their helpful feedback, which aided us in greatly improving the paper. This work was supported in part by the National Natural Science Foundation of China under Grant 62572420, the Shandong Province Key R&D Program under Grant 2024CXGC010801, Yantai Science and Technology Plan Project under Grant 2025ZDCX012 and 2025ZDCX020.

References

[1] Vitalik Buterin. A next-generation smart contract and decentralized application platform. *white paper*, 3(37):2–1, 2014.

[2] Junqin Huang, Linghe Kong, Guanjie Cheng, Qiao Xiang, Guihai Chen, Gang Huang, and Xue Liu. Advancing web 3.0: Making smart contracts smarter on blockchain. In *Proceedings of the ACM on Web Conference,(WWW)*, pages 1549–1560, 2024.

[3] Ilya Grishchenko, Matteo Maffei, and Clara Schneidewind. Foundations and tools for the static analysis of ethereum smart contracts. In *International conference on computer aided verification*, pages 51–78, 2018.

[4] Molud Esmaili and Ken Christensen. Performance modeling of public permissionless blockchains: A survey. *ACM Computing Surveys*, 57(7):1–35, 2025.

[5] Shivani Jamwal, José Cano, Gyu Myoung Lee, Nguyen H Tran, and Nguyen Truong. A survey on ethereum pseudonymity: Techniques, challenges, and future directions. *Journal of Network and Computer Applications*, 232:104019, 2024.

[6] Teng Hu, Xiaolei Liu, Ting Chen, Xiaosong Zhang, Xiaoming Huang, Weina Niu, Jiazhong Lu, Kun Zhou, and Yuan Liu. Transaction-based classification and detection approach for ethereum smart contract. *Information Processing & Management*, 58(2):102462, 2021.

[7] Sijia Li, Gaopeng Gou, Chang Liu, Chengshang Hou, Zhenzhen Li, and Gang Xiong. Ttagn: Temporal transaction aggregation graph network for ethereum phishing scams detection. In *Proceedings of the ACM Web Conference(WWW)*, pages 661–669, 2022.

[8] Jinhuan Wang, Pengtao Chen, Xinyao Xu, Jiajing Wu, Meng Shen, Qi Xuan, and Xiaoniu Yang. Tsgn: Transaction subgraph networks assisting phishing detection in ethereum. *IEEE Transactions on Dependable and Secure Computing*, pages 1–16, 2025.

[9] Jianguo Sun, Yifan Jia, Yanbin Wang, Ye Tian, and Sheng Zhang. Ethereum fraud detection via joint transaction language model and graph representation learning. *Information Fusion*, 120:103074, 2025.

[10] Liang Chen, Jiaying Peng, Yang Liu, Jintang Li, Fenfang Xie, and Zibin Zheng. Phishing scams detection in ethereum transaction network. *ACM Transactions on Internet Technology (TOIT)*, 21(1):1–16, 2020.

[11] Steven Farrugia, Joshua Ellul, and George Azzopardi. Detection of illicit accounts over the ethereum blockchain. *Expert Systems with Applications*, 150:113318, 2020.

[12] Rabia Musheer Aziz, Mohammed Farhan Baluch, Sarthak Patel, and Abdul Hamid Ganie. Lgbm: a machine learning approach for ethereum fraud detection. *International Journal of Information Technology*, 14(7):3321–3331, 2022.

[13] Tao Huang, Dan Lin, and Jiajing Wu. Ethereum account classification based on graph convolutional network. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 69(5):2528–2532, 2022.

[14] Jieli Liu, Jiatao Zheng, Jiajing Wu, and Zibin Zheng. Fa-gnn: Filter and augment graph neural networks for account classification in ethereum. *IEEE Transactions on Network Science and Engineering*, 9(4):2579–2588, 2022.

[15] Hexiang Huang, Xuan Zhang, Jishu Wang, Chen Gao, Xue Li, Rui Zhu, and Qiuying Ma. Peae-gnn: Phishing detection on ethereum via augmentation ego-graph based on graph neural network. *IEEE Transactions on Computational Social Systems*, 11(3):4326–4339, 2024.

[16] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI open*, 1:57–81, 2020.

[17] 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, 2021.

[18] Zhanghao Wu, Paras Jain, Matthew Wright, Azalia Mirhoseini, Joseph E Gonzalez, and Ion Stoica. Representing long-range context for graph neural networks with global attention. *Advances in neural information processing systems*, 34:13266–13279, 2021.

[19] Zhaowei Liu, Yixian Wang, Shenqiang Wang, Xiangfu Zhao, Haiyang Wang, and Haoyu Yin. Heterogeneous graphs neural networks based on neighbor relationship filtering. *Expert Systems with Applications*, 239:122489, 2024.

[20] Johannes Gasteiger, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *International Conference on Learning Representations(ICLR)*, 2019.

[21] Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. In *International Conference on Learning Representations(ICLR)*, 2021.

[22] Istvan Lukovits, Sonja Nikolić, and Nenad Trinajstić. Resistance distance in regular graphs. *International Journal of Quantum Chemistry*, 71(3):217–225, 1999.

[23] Francois Fouss, Alain Pirotte, Jean-Michel Renders, and Marco Saerens. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Transactions on knowledge and data engineering*, 19(3):355–369, 2007.

[24] Xuehai Tang, Zhongjiang Yao, Huazhen Zhong, Guan Wang, Yuanshu Zhao, Xiaodan Zhang, and Jizhong Han. Ethgan: Improving ethereum account classification accuracy via data augmentation. In *International Joint Conference on Neural Networks*, pages 1–8, 2024.

[25] Jingjing Yang, Jieli Liu, Dan Lin, Jiajing Wu, Baoying Huang, Quanzhong Li, and Zibin Zheng. Who stole my nft? investigating web3 nft phishing scams on ethereum. *IEEE Transactions on Information Forensics and Security*, 19:9301–9314, 2024.

[26] Gustav Kirchhoff. Ueber die auflösung der gleichungen, auf welche man bei der untersuchung der linearen vertheilung galvanischer ströme geführt wird. *Annalen der Physik*, 148(12):497–508, 1847.

[27] Ameya Velingker, Ali Sinop, Ira Ktena, Petar Veličković, and Sreenivas Gollapudi. Affinity-aware graph networks. *Advances in Neural Information Processing Systems*, 36:67847–67865, 2023.

[28] Mitchell Black, Zhengchao Wan, Amir Nayyeri, and Yusu Wang. Understanding oversquashing in gnns through the lens of effective resistance. In *International Conference on Machine Learning(ICML)*, pages 2528–2547, 2023.

[29] Xu Shen, Pietro Lio, Lintao Yang, Ru Yuan, Yuyang Zhang, and Chengbin Peng. Graph rewiring and preprocessing for graph neural networks based on effective resistance. *IEEE Transactions on Knowledge and Data Engineering*, 36(11):6330–6343, 2024.

[30] Shota Saito and Mark Herbster. Multi-class graph clustering via approximated effective p-resistance. In *International Conference on Machine Learning, (ICML)*, pages 29697–29733, 2023.

[31] Devin Kreuzer, Dominique Beaini, Will Hamilton, Vincent Létourneau, and Prudencio Tossou. Rethinking graph transformers with spectral attention. *Advances in Neural Information Processing Systems*, 34:21618–21629, 2021.

[32] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations(ICLR)*, 2017.

[33] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero-Soriano, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations(ICLR)*, 2018.

[34] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[35] Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 257–266, 2019.

[36] Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. *Advances in neural information processing systems*, 33:7793–7804, 2020.

[37] Shuman Zhuang, Zhihao Wu, Zhaoliang Chen, Hong-Ning Dai, and Ximeng Liu. Refine then classify: Robust graph neural networks with reliable neighborhood contrastive refinement. In *Proceedings of the AAAI Conference on Artificial Intelligence(AAAI)*, volume 39, pages 13473–13482, 2025.

[38] Jiajing Wu, Qi Yuan, Dan Lin, Wei You, Weili Chen, Chuan Chen, and Zibin Zheng. Who are the phishers? phishing scam detection on ethereum via network embedding. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 52(2):1156–1166, 2020.

[39] Yixian Wang, Zhaowei Liu, Jindong Xu, and Weiqing Yan. Heterogeneous network representation learning approach for ethereum identity identification. *IEEE Transactions on Computational Social Systems*, 10(3):890–899, 2022.

[40] Haojie Sun, Zhaowei Liu, Shenqiang Wang, and Haiyang Wang. Adaptive attention-based graph representation learning to detect phishing accounts on the ethereum blockchain. *IEEE Transactions on Network Science and Engineering*, 11(3):2963–2975, 2024.

[41] Shuhui Fan, Haoran Xu, Shaojing Fu, Yuchuan Luo, and Ming Xu. Edge-feature modeling-based topological graph neural networks for phishing scams detection on ethereum. In *2024 IEEE/ACM 32nd International Symposium on Quality of Service (IWQoS)*, pages 1–10. IEEE, 2024.

A Dataset Details

This section provides detailed statistics and distribution metrics for the Ethereum transaction network dataset used in our experiments.

Table 3 summarizes the overall scale of the graph, including the total number of nodes, edges, and node features. To illustrate the severe data imbalance typical in real-world blockchain fraud detection scenarios, Table 4 presents the exact distribution of the 816 manually verified accounts across six distinct categories. Finally, Table 5 details the three data splitting strategies (Split 1, 2, and 3) employed to rigorously evaluate the models' performance and robustness under varying supervision ratios.

Table 3: Dataset Summary of the Ethereum Transaction Graph

Nodes	Edges	Labeled Nodes	Classes	Features
1,402,220	2,815,028	816	6	12

Table 4: Label Distribution of Verified Accounts in the Ethereum Dataset

Mining	Phish/Hack	ICO	Exchange	Converter	Gambling
60	252	167	171	139	27

Table 5: Training-Validation-Test Set Partitions

Dataset	Train Set	Validation Set	Test Set
Split 1	70%	10%	20%
Split 2	50%	20%	30%
Split 3	30%	30%	40%

B Algorithm of Feature Augmentation

In this section, we present the step-by-step pseudocode for the Effective Resistance Feature Augmentation module discussed in Section 4. Algorithm 1 outlines how the resistance-weighted adjacency matrix $\hat{A}_{\text{res}}$ and the augmented feature matrix X_{res} are computed. For each node, a local subgraph is extracted via random walks, and the Laplacian pseudoinverse is derived to compute the effective resistance between node pairs. These resistance distances guide the adaptive reweighting of edges, enriching the node representations with robust global structural contexts before they are passed to the multi-hop aggregation module.

Importantly, RAMA uses the resistance-derived weights in two complementary ways before multi-hop aggregation. First, the normalized weights $w(v, u)$ are used to aggregate neighboring features and construct the resistance-enhanced feature matrix X_{res}. Second, the same weights define the resistance-weighted adjacency matrix $\hat{A}_{\text{res}}$, which controls the propagation paths in the subsequent multi-hop aggregation module. The effective resistance $\Omega(v, u)$ is therefore not directly concatenated with the original node features as an independent scalar. Instead, it is first transformed into normalized edge weights, which are used to construct both X_{res} and $\hat{A}_{\text{res}}$. The multi-hop representations are then computed as $X_k = \hat{A}^k_{\text{res}} X_{\text{res}}$.

C Baselines and Implementation Details

C.1 Baselines

To evaluate the performance of the proposed RAMA, the following state-of-the-art methods were considered.

Algorithm 1 Effective Resistance Feature Augmentation

Input: Graph $G = (V, E)$; Node feature matrix X_M; Parameters $\lambda, \gamma, \epsilon$.

Output: Resistance-weighted adjacency $\hat{A}_{\text{res}}$; Augmented features X_{res}.

for each $v \in V$ **do**
  $G_s \leftarrow \text{RandomWalk}(v, G)$; //Extract local subgraph
  $L_s^+ \leftarrow \text{pinv}(\text{Laplacian}(G_s))$; //Compute Laplacian pseudoinverse
  for each $u \in \mathcal{N}_{G_s}(v)$ **do**
    $\Omega(v, u) \leftarrow L_s^+(v, v) + L_s^+(u, u) - 2L_s^+(v, u)$; //Compute effective resistance
    $\tilde{w}(v, u) \leftarrow \exp\left[-\lambda(\Omega(v, u) + \epsilon)^\gamma\right]$; //Calculate unnormalized weights
  end for
  $w(v, u) \leftarrow \tilde{w}(v, u) / \sum_{u'} \tilde{w}(v, u')$; //Normalize weights
  $x'_v \leftarrow X_M[v] \parallel \sum_{u \in \mathcal{N}_{G_s}(v)} w(v, u) X_M[u]$; //Update node representation
end for
$\hat{A}_{\text{res}} \leftarrow \{w(v, u)\}$; // Construct resistance-weighted adjacency matrix
$X_{\text{res}} \leftarrow \{x'_v\}_{v \in V}$; // Construct augmented feature matrix

- GCN. As applied in previous works [32] for Ethereum fraud detection, GCN leverages graph convolution operations based on the normalized Laplacian to update node embeddings using local neighborhood features.
- GAT. GAT [33] introduces attention mechanisms into graph neural networks, allowing the model to assign different importance weights to neighbors during aggregation. This contrasts with GCNs, where aggregation weights are fixed by graph structure.
- GraphSAGE. GraphSAGE [34] aggregates feature information from a sampled set of neighboring nodes and applies a linear transformation. The combined representation serves as the node embedding, enabling inductive generalization to unseen nodes.
- Cluster-GCN. Cluster-GCN [35] improves training efficiency by partitioning the graph into dense subgraphs and performing neighborhood aggregation only within each cluster. This design significantly reduces memory usage and accelerates stochastic gradient descent optimization.
- H_2GCN. H_2GCN [36] targets graphs with low homophily by separating ego and neighbor features, aggregating multi-hop neighborhoods, and combining intermediate representations, enabling more expressive learning in heterophilic or noisy network structures.
- GRANCE. GRANCE [37] is a robust graph neural network framework designed to mitigate the impact of structural noise. It employs a graph refinement mechanism combined with contrastive learning to purify spurious edges and enhance the quality of learned representations.
- Trans2vec. Trans2vec [38] identifies phishing scams on blockchain networks by analyzing transaction behaviors, incorporating both transfer values and temporal patterns derived from transaction records.
- HNRL. HNRL [39] performs heterogeneous network representation learning by extracting latent information from Ethereum transactions, enabling the projection of the transaction graph into a low-dimensional embedding space.
- ABGRL. ABGRL [40] is an adaptive attention-based graph learning method that fuses multi-channel features and enhances tail node representation through self-supervised regression for improved phishing detection on Ethereum.
- FA-GNN. FA-GNN [14] targets Ethereum's heterogeneous transaction network by filtering out noisy neighbors and augmenting features with high-order information, thereby enhancing feature representation and improving classification performance in noisy and heterophilic network structures.
- ETGNN. ETGNN [41] focuses on edge-centric modeling by treating transactions as primary entities. It incorporates persistent homology to capture topological features such as structural holes and effectively models the temporal dynamics of transaction flows.

C.2 Implementation Details

In the model implementation, the feature augmentation module, which enhances node features using random walks and effective resistance, employs 55 random walks per node with a walk length of 5, and a small constant $\epsilon = 1 \times 10^{-6}$ to stabilize the nonlinear mapping from effective resistance values to resistance-aware weights when the resistance approaches zero. The original and aggregated features are concatenated, doubling the feature dimension. Structural features, including normalized node degree and local clustering coefficients, are computed using the `igraph` library and appended to the feature matrix. In the node embedding learning stage, this study uses 8 attention heads, with a per-head dimension of $d_k = \frac{d_h}{h}$, where d_h is the hidden dimension. The model applies a dropout rate of 0.2 (both for attention and feed-forward layers) and an attention dropout rate of 0.2. The model uses AdamW as the optimization method with a weight decay of 0.00001. Early stopping with a patience of 50 epochs is employed during training, monitoring validation accuracy and loss. Laplacian positional encodings with a dimension of 15 are concatenated with node features to incorporate structural information. For evaluation, we use a multi-layer perceptron as the downstream classifier on the learned RAMA embeddings. All experiments are conducted using a machine equipped with four NVIDIA GeForce RTX 4090 GPUs. Reproducibility is ensured by fixing the random seed to 3407.

C.3 Downstream Classifier Evaluation

To ensure that the embeddings generated by RAMA are robust and effectively separable, we evaluated the performance of several standard machine learning classifiers on the learned node representations. As summarized in Table 6, we compared Naive Bayes, Logistic Regression, Support Vector Machines (SVM), K-Nearest Neighbors (KNN), and a Multi-Layer Perceptron (MLP). While simpler linear models like Logistic Regression achieve highly competitive results, the MLP classifier yields the highest performance across all evaluation metrics. This demonstrates the high quality and non-linear separability of the representations learned by the proposed framework.

Table 6: Performance Comparison of Various Classifiers Using Learned RAMA Embeddings

Classifier	**Precision**	**Recall**	**Micro-F1**	**Macro-F1**
Naive Bayes	0.7233	0.7498	0.7593	0.7262
Logistic Regression	0.8337	0.8262	0.8642	0.8262
SVM	0.8684	0.8089	0.8580	0.8315
KNN	0.8617	0.8107	0.8519	0.8329
MLP	**0.8790**	**0.8471**	**0.8827**	**0.8610**

D Ablation Analysis

To thoroughly evaluate the contributions of key components within the Effective Resistance with Multi-Hop Aggregation framework, two ablation variants are designed for comparison. The first variant excludes the resistance distance-based structural augmentation module (RAMA w/o RA), while the second retains the hop-wise features but replaces attention-based aggregation with mean pooling (RAMA w/o MA). As shown in Figure 4, the full RAMA model outperforms both variants across all evaluation metrics, highlighting the critical roles of the resistance distance-based augmentation and the multi-hop aggregation mechanism in enhancing Ethereum account classification performance.

D.1 Evaluate the Validity of Feature Augmentation

After removing this module and retraining the model under identical experimental settings, Macro-F1 and Micro-F1 scores drop by approximately 4.2% and 3.6%, respectively. This performance degradation suggests that relying solely on local aggregation strategies is insufficient for capturing information from distant or weakly connected nodes. In Ethereum transaction graphs, complex behaviors such as obfuscation and coordinated operations often propagate through sparse or indirect paths, which are difficult to identify using conventional methods. By incorporating resistance distance

as a global connectivity measure, the model reweights adjacency structures, emphasizing connections that are subtle but semantically important. This structural awareness significantly enhances the expressiveness of node representations and aids in detecting nuanced behavioral differences in high-risk accounts.

D.2 Evaluate the Validity of Multi-Hop Aggregation Mechanism

Replacing attention-based aggregation with mean pooling over hop-wise representations leads to a performance decline, with Macro-F1 decreasing by an average of 3.9% and Recall by 4.7%. Mean pooling computes a uniform average of neighbor features across different hop levels without adapting to the heterogeneous and context-dependent relational structures prevalent in Ethereum networks. In contrast, the multi-hop aggregation mechanism integrates information from neighbors across various hop levels, using attention-weighted summation to combine hop-wise features. This process adjusts weights based on feature similarity, structural positioning, and contextual cues, enabling the model to capture fine-grained transactional patterns and model multi-level dependencies. By mitigating over-smoothing and enhancing the model's ability to discern complex account behaviors, this mechanism contributes to more robust and discriminative node representations.

E Sensitivity Analysis

This section examines the sensitivity of RAMA to the number of attention heads, the learning rate, and the resistance distance parameters. By systematically varying these hyperparameters, we evaluate their effects on classification performance and identify suitable settings for Ethereum account classification.

E.1 Effect of Attention Heads

This study examines the impact of attention expressiveness by varying the number of attention heads $h \in \{2, 4, 6, 8, 10\}$. As shown in Figure 7, increasing the number of heads generally improves performance across all evaluation metrics, as it enables the model to better capture fine-grained patterns and heterogeneous relationships in the transaction graph. The performance rises steadily from 2 to 8 heads, indicating that attention diversity enhances representation quality. However, a slight decline is observed at 10 heads, suggesting that excessive heads may introduce redundancy or noise, which offsets the benefits of multi-head attention.

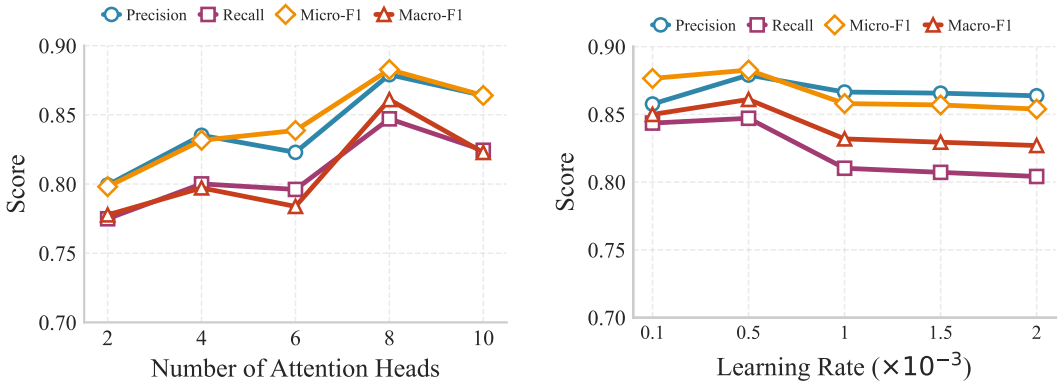


Figure 7: Effect of Attention Heads and Learning Rate.

E.2 Effect of Learning Rate

We evaluated the influence of the learning rate $\eta \in \{0.0001, 0.0005, 0.001, 0.0015, 0.002\}$ on classification performance under a polynomial decay schedule, with the final learning rate fixed at $\eta_{\text{end}} = 0.0001$. As shown in Figure 7, $\eta = 0.0005$ achieved the highest Precision and Micro-F1. This setting also yielded the highest Recall and Macro-F1. The smallest rate, $\eta = 0.0001$, produced

lower Precision and F1 scores than this intermediate setting. Larger rates of 0.001, 0.0015, and 0.002 reduced performance relative to $\eta = 0.0005$, particularly Recall and Macro-F1. These two metrics declined further at $\eta = 0.0015$ and 0.002, whereas Precision and Micro-F1 showed smaller declines. Overall, $\eta = 0.0005$ provided a favorable balance across metrics for multi-class Ethereum account classification.

E.3 Effect of Resistance Distance Parameters

The impact of the scaling factor $\lambda \in \{0.1, 0.4, 0.7, 1.0\}$, the stability constant $\epsilon \in \{10^{-7}, 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}\}$, and the nonlinearity parameter $\gamma \in \{0.5, 1.0, 1.5, 2.0, 2.5\}$ on the resistance distance-based enhancement module is evaluated by varying their values independently. As shown in Figure 5, $\lambda = 0.7$ achieves the highest Micro-F1 score, indicating an optimal weighting of structurally significant connections. For ϵ, a value of 10^{-6} yields the best performance among the tested settings. This small positive offset keeps the input to the power transformation away from zero, moderating its local sensitivity when $0 < \gamma < 1$. Additionally, $\gamma = 2.5$ achieves the highest Micro-F1 score among the tested values. These results highlight the importance of tuning λ, ϵ, and γ to enhance the module's effectiveness in capturing global structural information.

F Additional Robustness Analysis

In addition to the Macro-F1 drop evaluated in the main text, we present the robustness evaluation of RAMA and baseline models across other classification metrics, specifically Precision, Recall, and Micro-F1, under varying levels of structural noise. As illustrated in Figure 8, increasing the proportion of spurious edges (from 10% to 30%) naturally degrades the performance of all models. However, RAMA exhibits the least performance drop across all three metrics compared to general GNNs and blockchain-specific baselines. This consistent stability further validates that the resistance-guided edge reweighting mechanism effectively filters out topological noise and prevents the propagation of misleading signals during multi-hop aggregation.

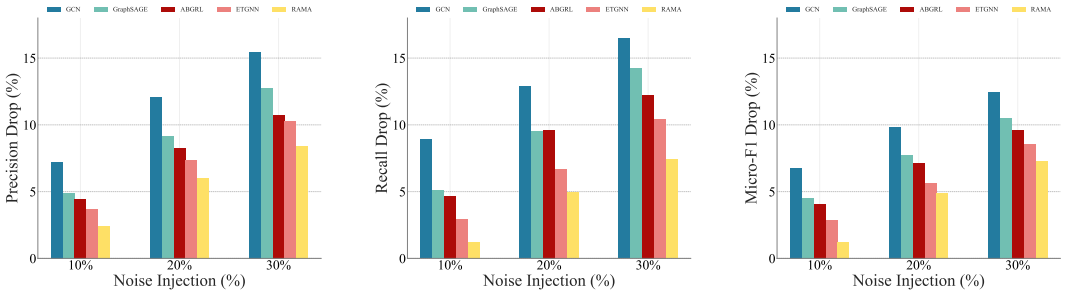


Figure 8: Additional robustness evaluation of RAMA and baselines under structural noise. The plots illustrate the performance drop in Precision, Recall, and Micro-F1 when spurious edges are randomly added to the transaction graph.

G Limitations and Future Work

While RAMA demonstrates strong performance and robustness in Ethereum account classification, it has certain limitations that provide directions for future research. First, calculating the effective resistance requires computing the Moore-Penrose pseudoinverse of the Laplacian matrix. Although we mitigate the computational complexity by restricting the operation to local subgraphs extracted via random walks, it still introduces an additional computational overhead during the feature augmentation phase compared to standard message-passing GNNs. Future work could explore scalable approximation algorithms for effective resistance to further improve training efficiency on massive graphs. Second, our current framework models the Ethereum transactions primarily from a structural topology perspective. Real-world blockchain transactions are inherently dynamic, with critical temporal patterns. Extending RAMA to a continuous-time dynamic graph framework that jointly models resistance-aware topology and temporal evolution remains an important next step.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduction accurately reflect the paper's ontributions and scope.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss our limitations.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: This paper does not present any new theoretical results.All mathematical results used are drawn from the existing literature.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Necessary information to reproduce our method and appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: All data used in this work is open-source.Unfortunately,due to organizational policy,we are unable to release our source code at submission time.However,we plan to make it publicly available in the near future once administrative challenges are resolved.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide details in C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The experiments are conducted with a fixed random seed and no error bars are reported.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We detail the hardware resources in C.2

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper complies with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: The proposed method focuses on the technical aspects of the problem and do not include societal impacts.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: This paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have properly cited all code, data, and models used in our research and complied with the licensing agreements and terms of use set by the original authors.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: There are no new assets introduced in the paper.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: We used an LLM solely for editing assistance.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.