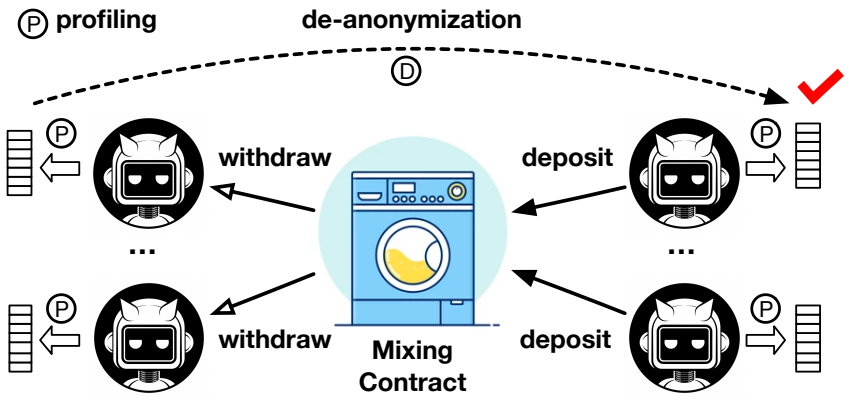
BERT4ETH: A Pre-trained Transformer for Ethereum Fraud Detection

Sihao Hu, Zhen Zhang, Bingqiao Luo, Shengliang Lu, Bingsheng He, Ling Liu

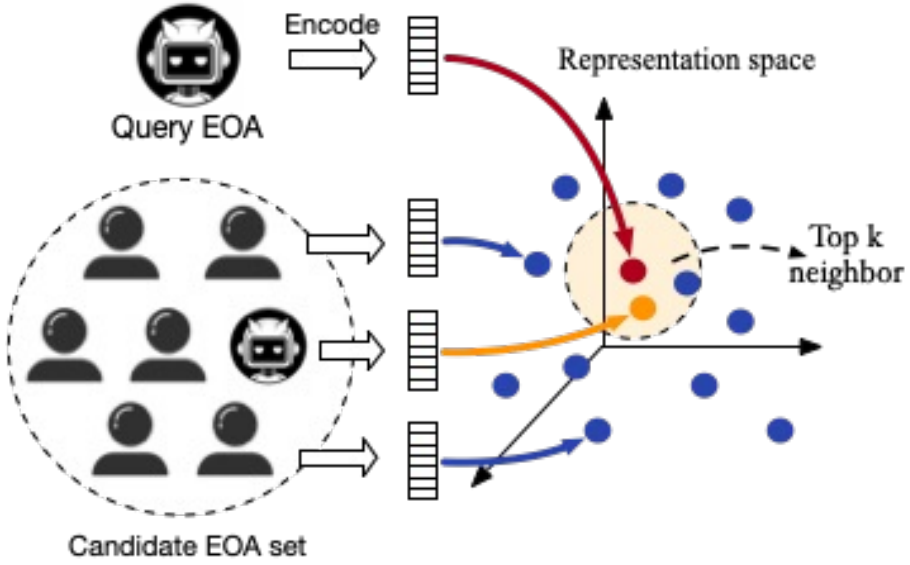
Background: Fraud activities proliferate on Ethereum

A universal account representation encoder can be used for multiple fraud detection tasks.

- Phishing scams
- Money laundering
- Identity inference
- Behavior analysis
- …



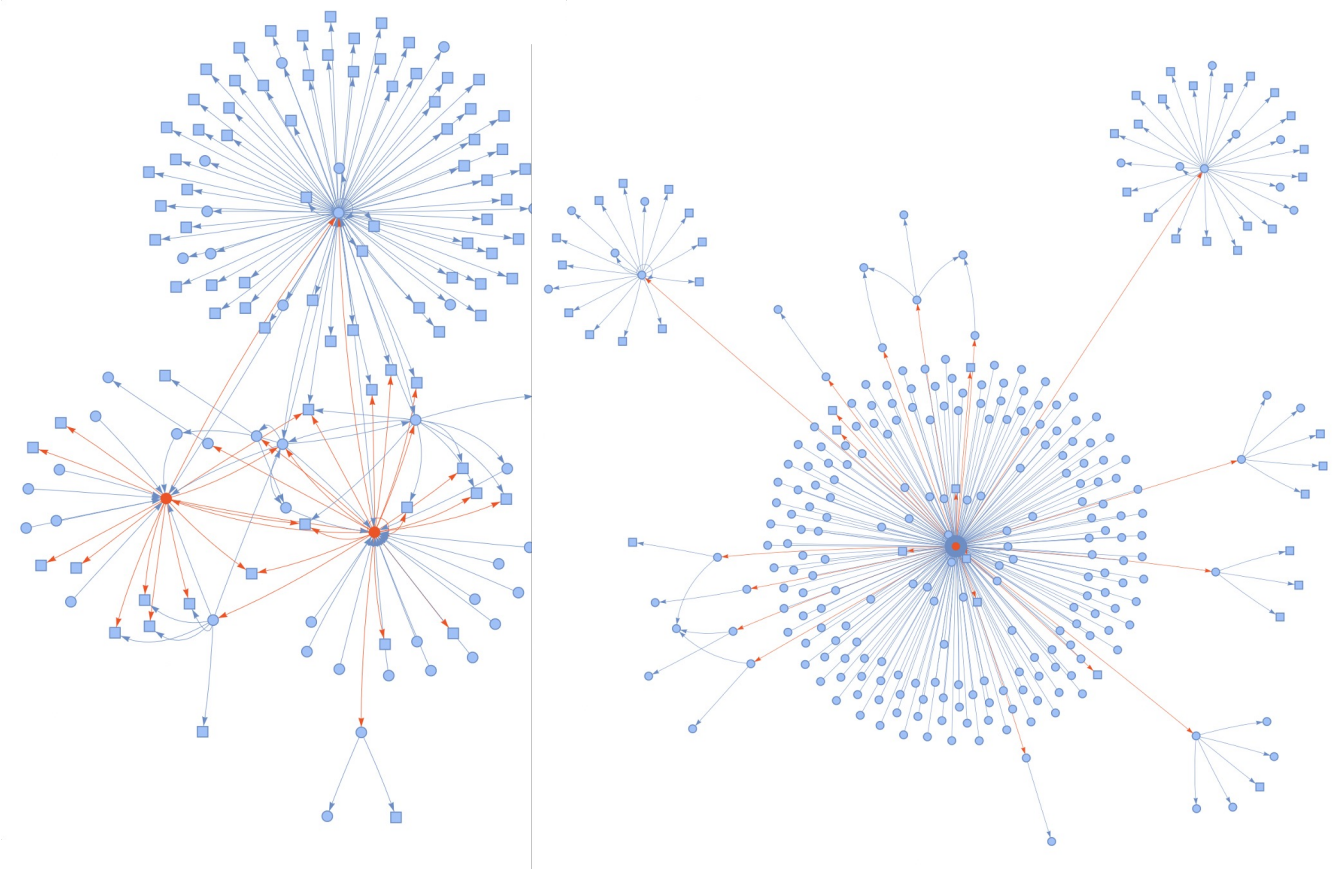
Tornado coin-mixing service



Link accounts controlled by the same person by measuring the similarity of accounts in the representation space.

Existing works exclusively focus on graph learning methods

It is intuitive to represent interactions between accounts as a graph.



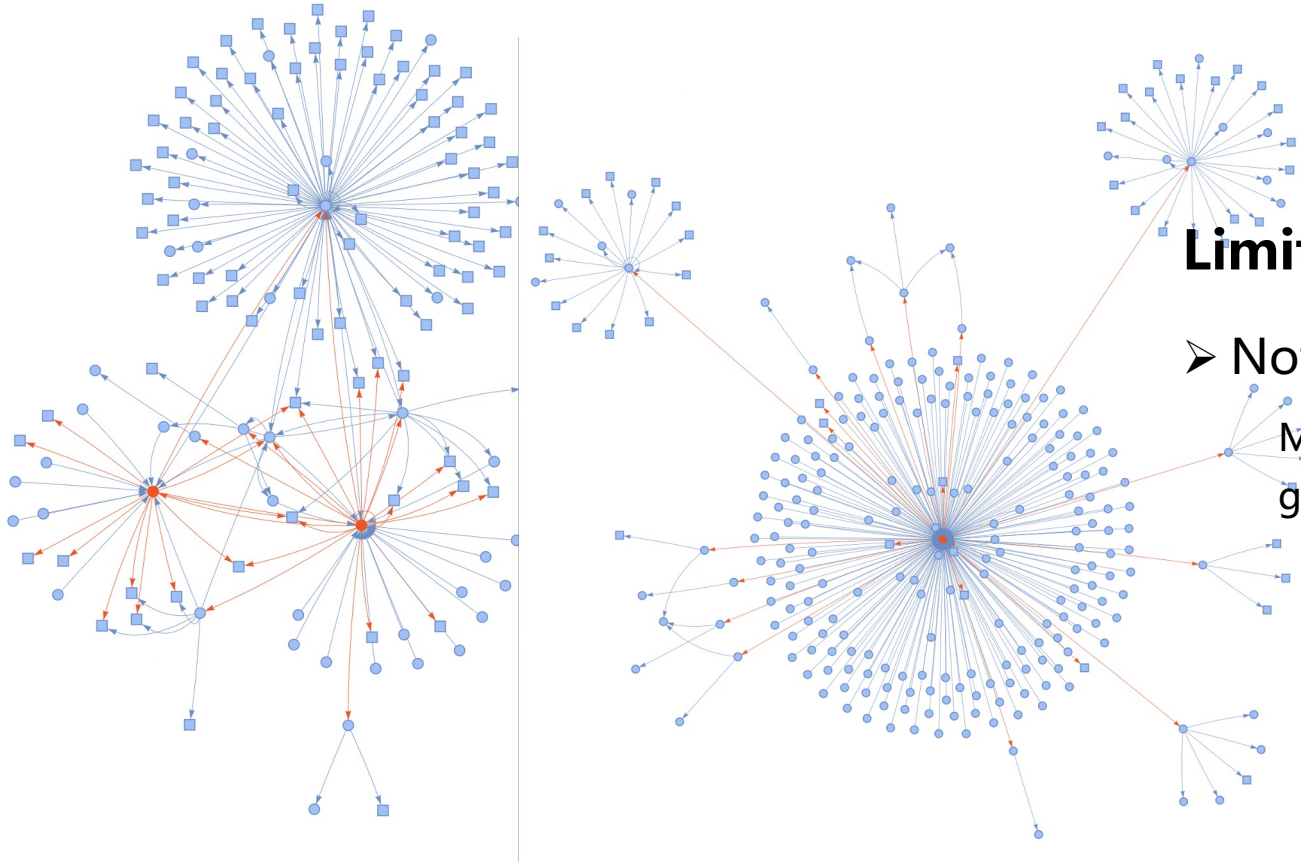
[1]Wu, Jiajing, et al. "Who are the phishers? phishing scam detection on Ethereum via network embedding." *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 52.2 (2020): 1156-1166.

[2]Béres, Ferenc, et al. "Blockchain is watching you: Profiling and deanonymizing Ethereum users." *2021 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)*. IEEE, 2021.

[3] Li, Sijia, et al. "TTAGN: Temporal transaction aggregation graph network for Ethereum phishing scams detection." *Proceedings of the ACM Web Conference 2022*. 2022.

Existing works exclusively focus on graph learning methods

It is intuitive to represent interactions between accounts as a graph.



Limitation 1

- Not suitable to capture sequential information.

  Merge multiple edges into one for graph computation like graph convolution or random walk.

[1]Wu, Jiajing, et al. "Who are the phishers? phishing scam detection on Ethereum via network embedding." *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 52.2 (2020): 1156-1166.

[2]Béres, Ferenc, et al. "Blockchain is watching you: Profiling and deanonymizing Ethereum users." *2021 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)*. IEEE, 2021.

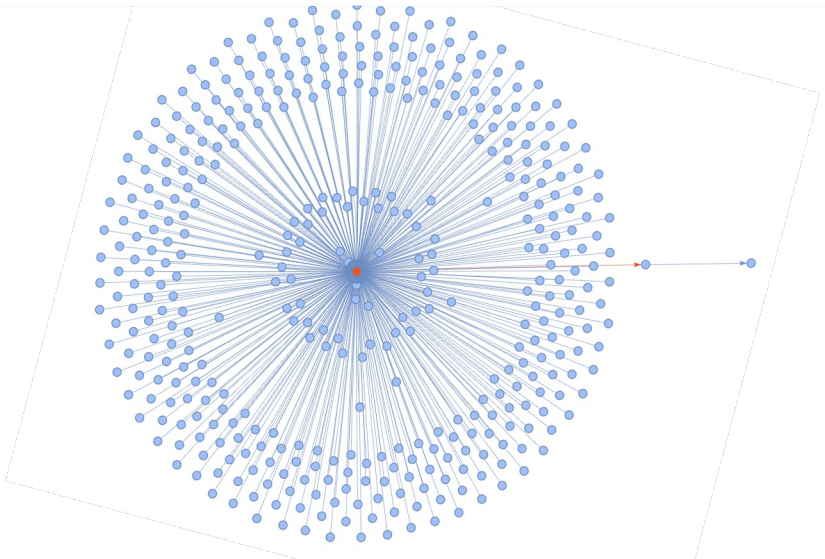
[3] Li, Sijia, et al. "TTAGN: Temporal transaction aggregation graph network for Ethereum phishing scams detection." *Proceedings of the ACM Web Conference 2022*. 2022.

Ethereum account follows an extreme power-law distribution

Power law: a minority of accounts appear at the majority of transactions.

Table 11: Top-10 popular accounts in randomly sampled 127,923,653 Ethereum transactions.

Rank	Account Address	Organization	Type	Tx Cnt	Portion
1	0xdac17f958d2ee523a2206206994597c13d831ec7	Bitfinex	Contract	6,596,673	5.16%
2	0x7a250d5630b4cf539739df2c5dacb4c659f2488d	Uniswap	Contract	3,938,795	3.08%
3	0xea674fdde714fd979de3edf0f56aa9716b898ec8	Ethermine	EOA	2,612,440	2.04%
4	0xb2930b35844a230f00e51431acae96fe543a0347	MiningPoolHub	EOA	1,920,792	1.50%
5	0xa090e606e30bd747d4e6245a1517ebe430f0057e	Coinbase	Contract	1,319,855	1.03%
6	0x7be8076f4ea4a4ad08075c2508e481d6c946d12b	OpenSea	Contract	1,286,292	1.00%
7	0x808b4da0be6c9512e948521452227efc619bea52	BlockFi	EOA	1,167,233	0.91%
8	0x52bc44d5378309ee2abf1539bf71de1b7d7be3b5	Nanopool	EOA	1,100,078	0.86%
9	0xa0b86991c6218b36c1d19d4a2e9eb0ce3606eb48	Centre	Contract	1,087,793	0.85%
10	0x3f5ce5fbfe3e9af3971dd833d26ba9b5c936f0be	Binance	EOA	922,156	0.72%
-	-	-	-	21,952,107	17.16%

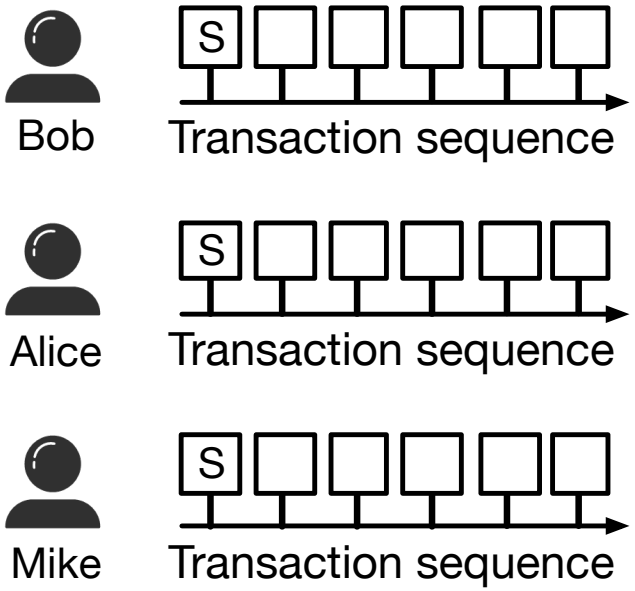


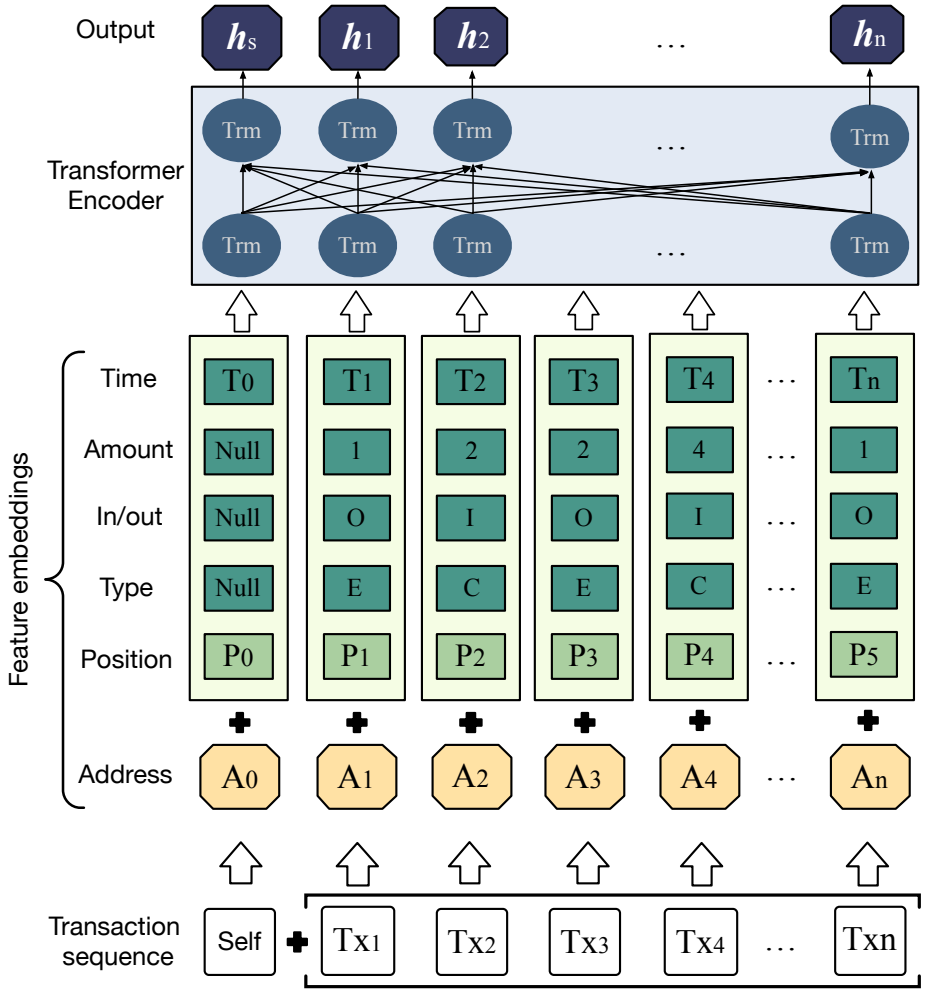
Limitation 2

1. Multi-hop convolution can introduce noise.
2. Model capability is limited (# of GNN layers = # of hop)

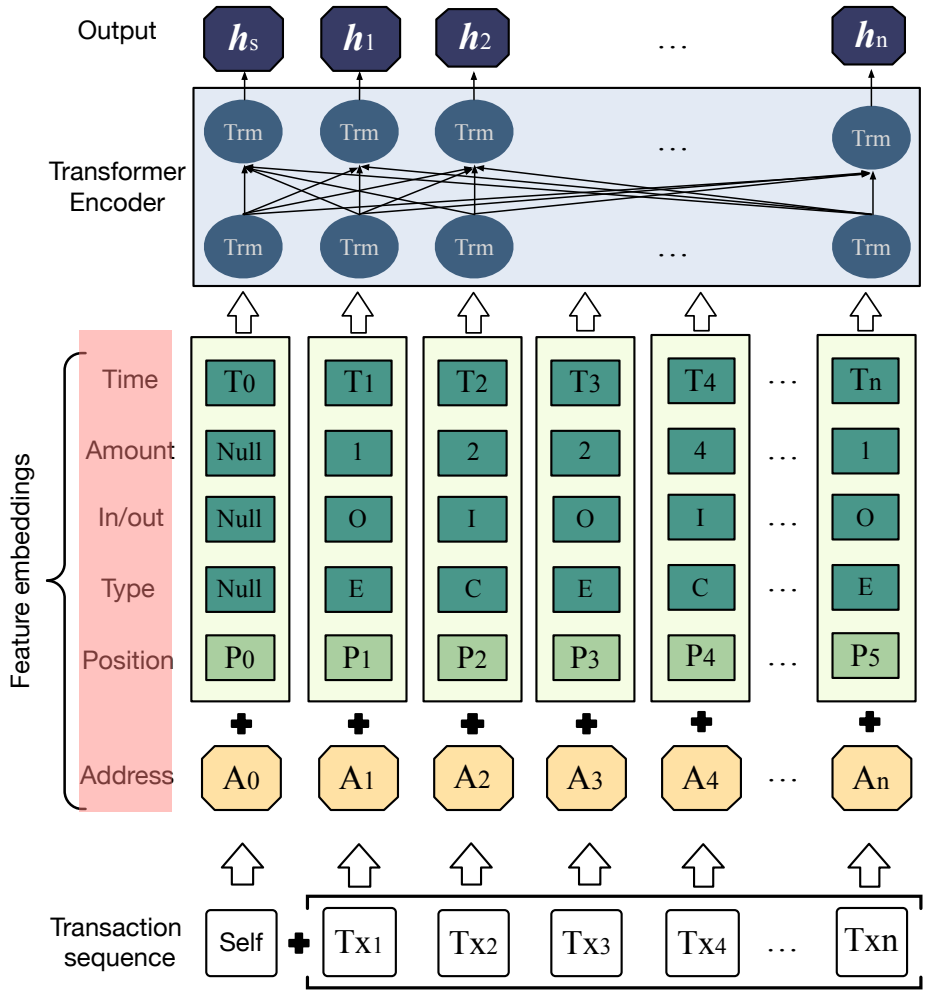
Sequential Transformer Encoder

1. Transaction Sequence Construction
2. Embedding Layer
3. Transformer Encoder

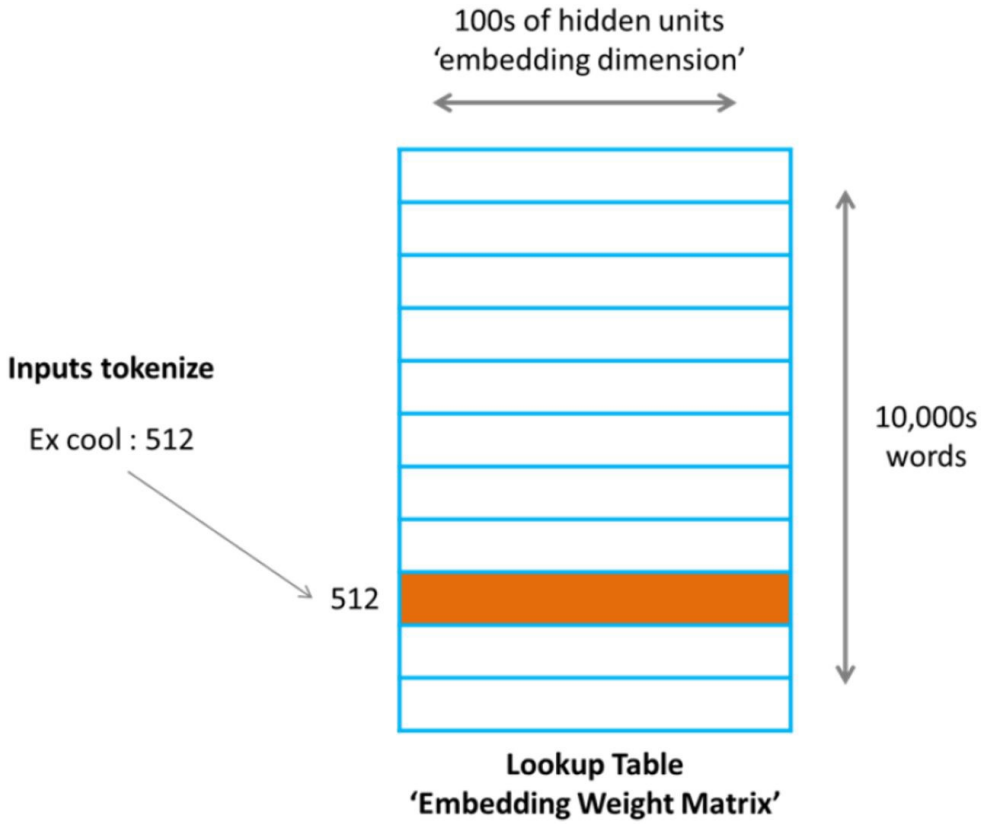




Model Architecture

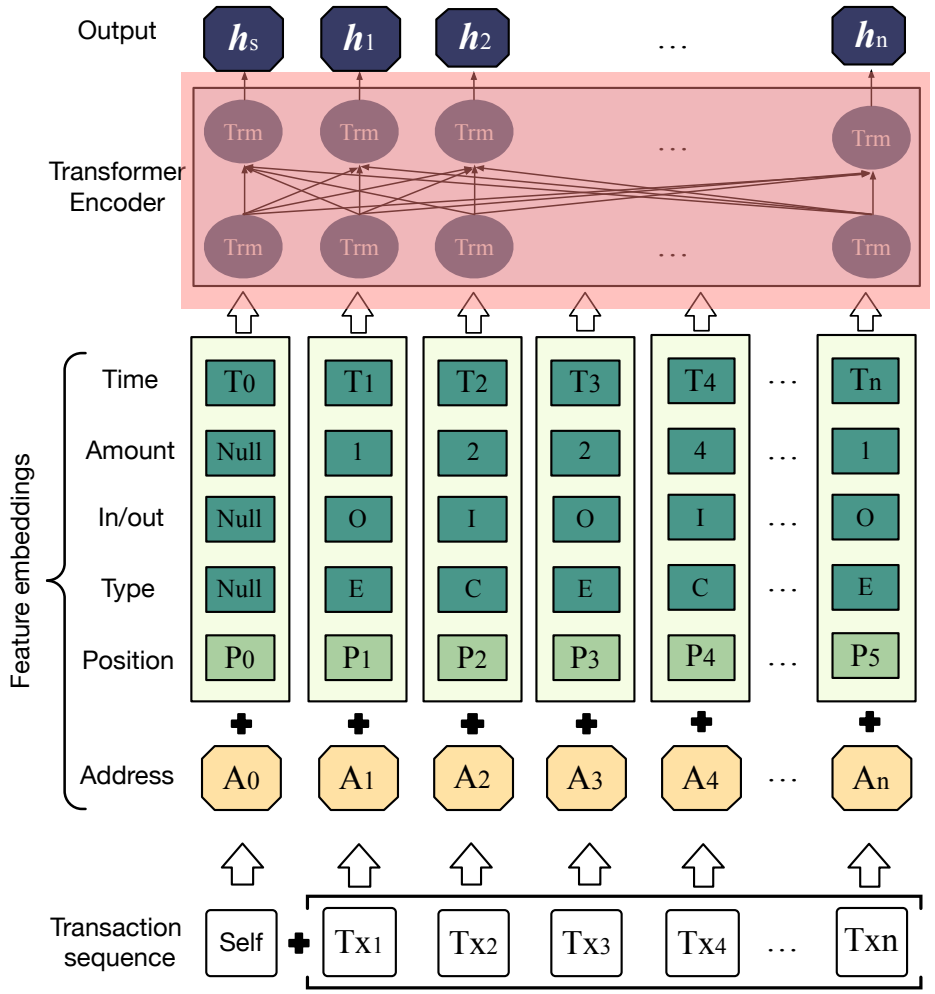


1. Transaction Sequence Generation
2. Embedding Layer
3. Transformer Encoder



An example of embedding lookup table

Model Architecture



1. Transaction Sequence Generation
2. Embedding Layer
3. Transformer Encoder

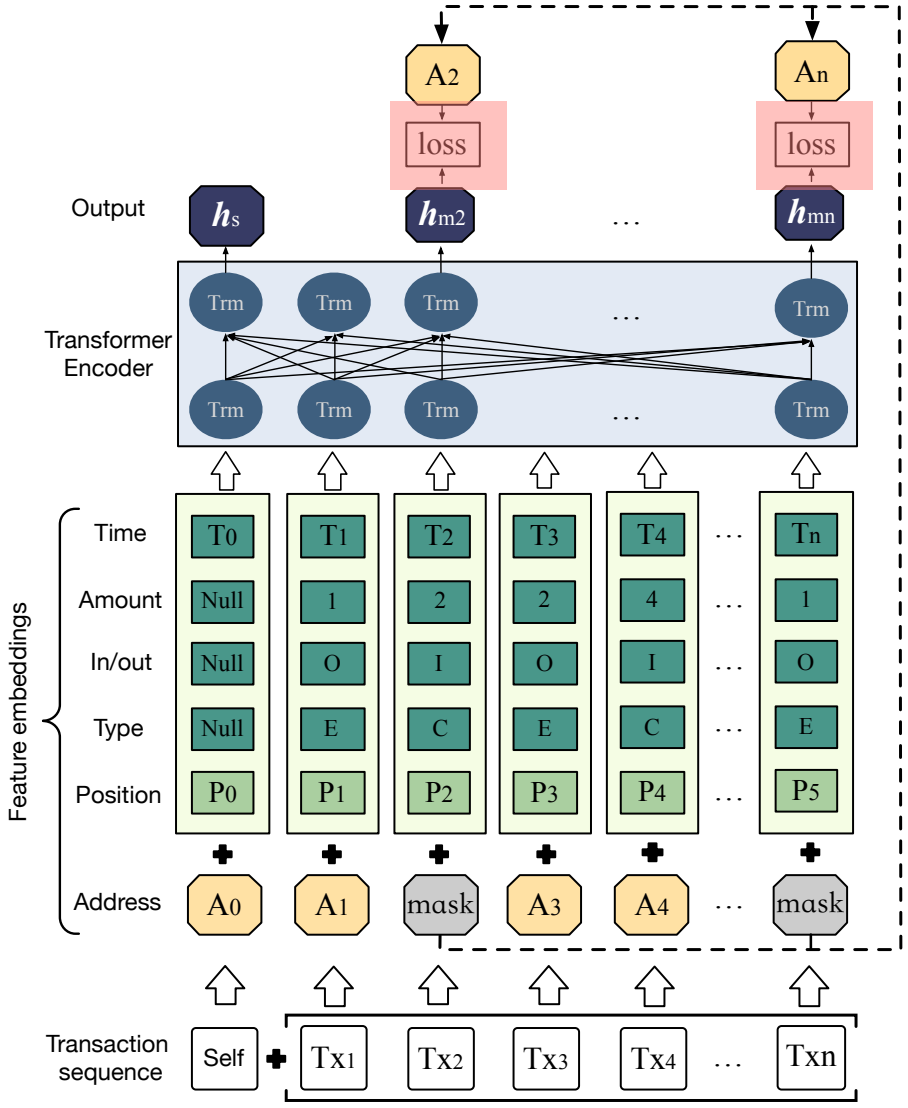
$$\boldsymbol{H}^{(l)} = \text{Attention}\left(\boldsymbol{H}^{(l)}\boldsymbol{W}_Q^{(l)}, \boldsymbol{H}^{(l)}\boldsymbol{W}_K^{(l)}, \boldsymbol{H}^{(l)}\boldsymbol{W}_V^{(l)}\right) \quad (1)$$

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d}}\right)V \quad (2)$$

$$\boldsymbol{H}^{(l+1)} = [\text{FFN}(\boldsymbol{h}_1^{(l)}); \cdots ; \text{FFN}(\boldsymbol{h}_t^{(l)})] \quad (3)$$

$$\text{FFN}(\boldsymbol{x}) = \text{GELU}(\boldsymbol{x}\boldsymbol{W}_1^{(l)} + \boldsymbol{b}_1^{(l)})\boldsymbol{W}_2^{(l)} + \boldsymbol{b}_2^{(l)} \quad (4)$$

Masked Address Prediction (Pre-training Task)



Contrastive loss on sampled addresses

address embedding

masked transaction representation

$$L = -\frac{1}{|\mathcal{M}|} \sum_{m \in \mathcal{M}} \log \left(\frac{\exp(\boldsymbol{h}_m^{\mathrm{T}} \cdot \boldsymbol{a}_p)}{\exp(\boldsymbol{h}_m^{\mathrm{T}} \cdot \boldsymbol{a}_p) + \sum_{n \in \mathcal{N}} \exp(\boldsymbol{h}_m^{\mathrm{T}} \cdot \boldsymbol{a}_n)} \right)$$

negative address embedding

Encourage $\boldsymbol{h}\_m$ to move closer to $\boldsymbol{a}\_p$ in the latent space, and away from $\boldsymbol{a}\_n$.

Preliminary Results

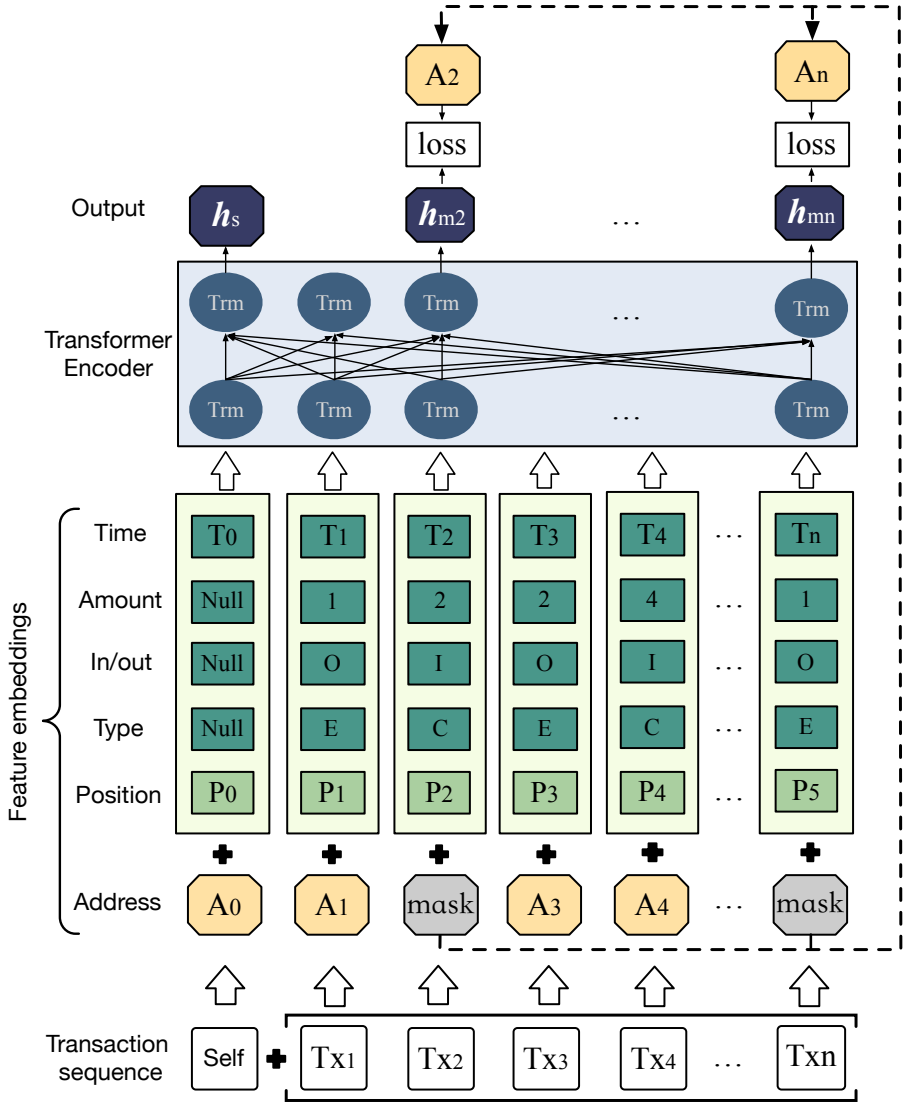


Table 3: Comparison for phishing detection w/ fixed training.

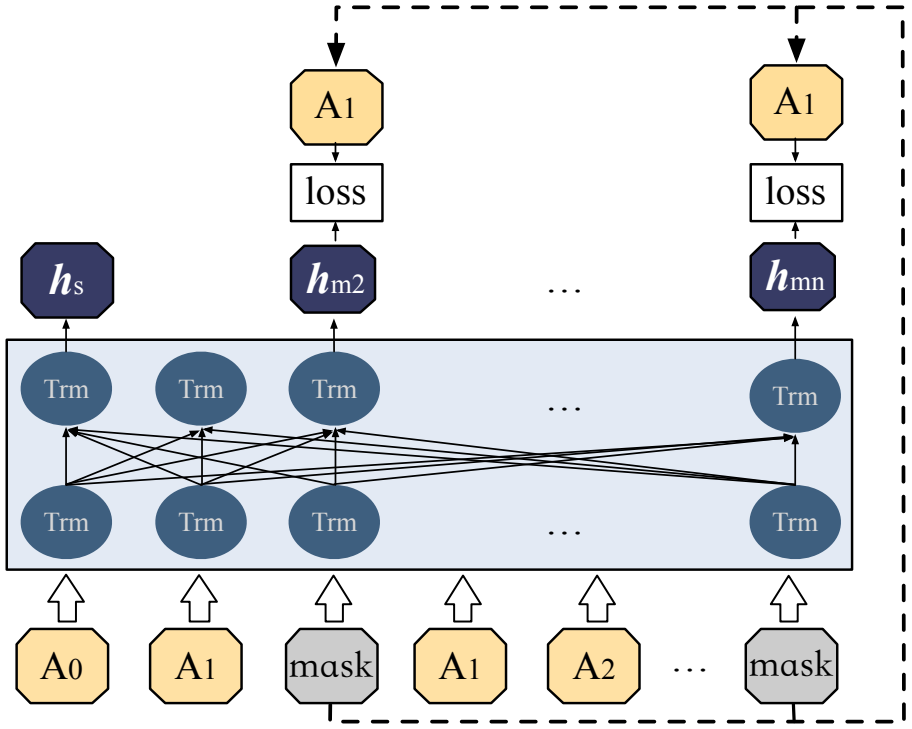
Method	Precision	Recall	F_1
DeepWalk	0.2293	0.1734	0.1974
Trans2Vec	0.1636	0.1432	0.1527
Diff2Vec	0.2184	0.2000	0.2088
Role2Vec	0.2520	0.2295	0.2402
GCN	0.3181	0.2180	0.2587
GSAGE	0.3023	0.2317	0.2623
GAT	0.3264	0.2581	0.2883
Ours			~ 0.1 (F1)

(° Д °) ???

Challenge 1: Repetitiveness

48.4% of transactions are repetitive (share the same addresses) with its one prior transaction.

High repetitiveness causes the **label leakage** problem.

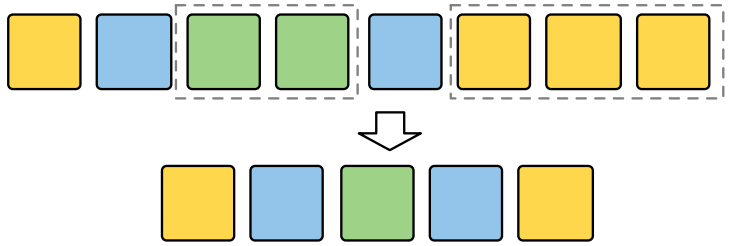


(15% masking ratio)

MAP on repetitive sequence

Strategy 1: Reduce Repetitiveness

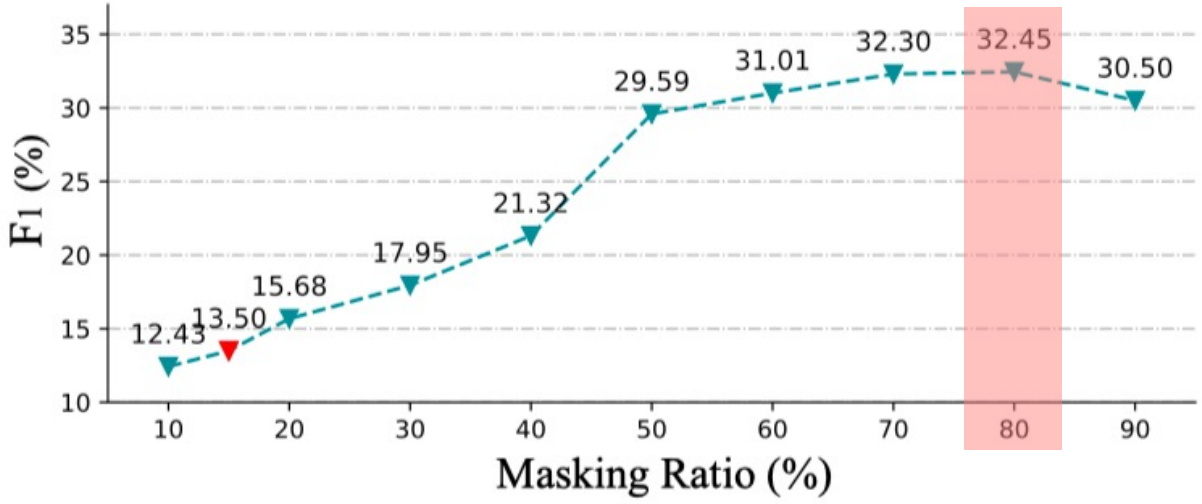
1. Continuous Repetitive Transaction Aggregation



Sum up the transaction amount and count the # of transactions

Still maintain the order of the sequence.

2. High masking ratio (80%)



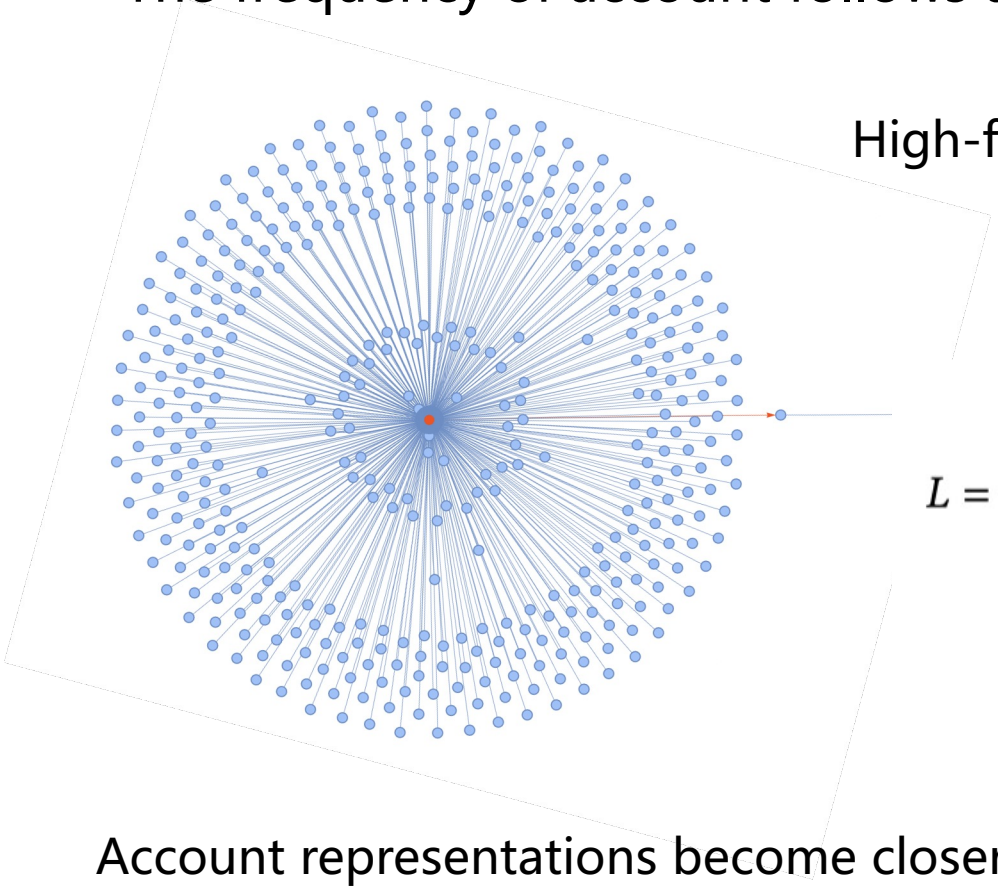
Validation results on the phishing account detection Task

Table 3: Comparison for phishing detection w/ fixed training.

Method	Precision	Recall	F_1
DeepWalk	0.2293	0.1734	0.1974
Trans2Vec	0.1636	0.1432	0.1527
Diff2Vec	0.2184	0.2000	0.2088
Role2Vec	0.2520	0.2295	0.2402
GCN	0.3181	0.2180	0.2587
GSAGE	0.3023	0.2317	0.2623
GAT	0.3264	0.2581	0.2883

Challenge 2: Skewed distribution

The frequency of account follows a **power-law** distribution.



High-frequency addresses are more likely be masked.

High-frequency address

$$L = -\frac{1}{|\mathcal{M}|} \sum_{m \in \mathcal{M}} \log \left(\frac{\exp(\boldsymbol{h}_m^{\mathrm{T}} \cdot \boldsymbol{a}_p)}{\exp(\boldsymbol{h}_m^{\mathrm{T}} \cdot \boldsymbol{a}_p) + \sum_{n \in \mathcal{N}} \exp(\boldsymbol{h}_m^{\mathrm{T}} \cdot \boldsymbol{a}_n)} \right)$$

Account representations become closer to the high-frequency addresses during optimization.

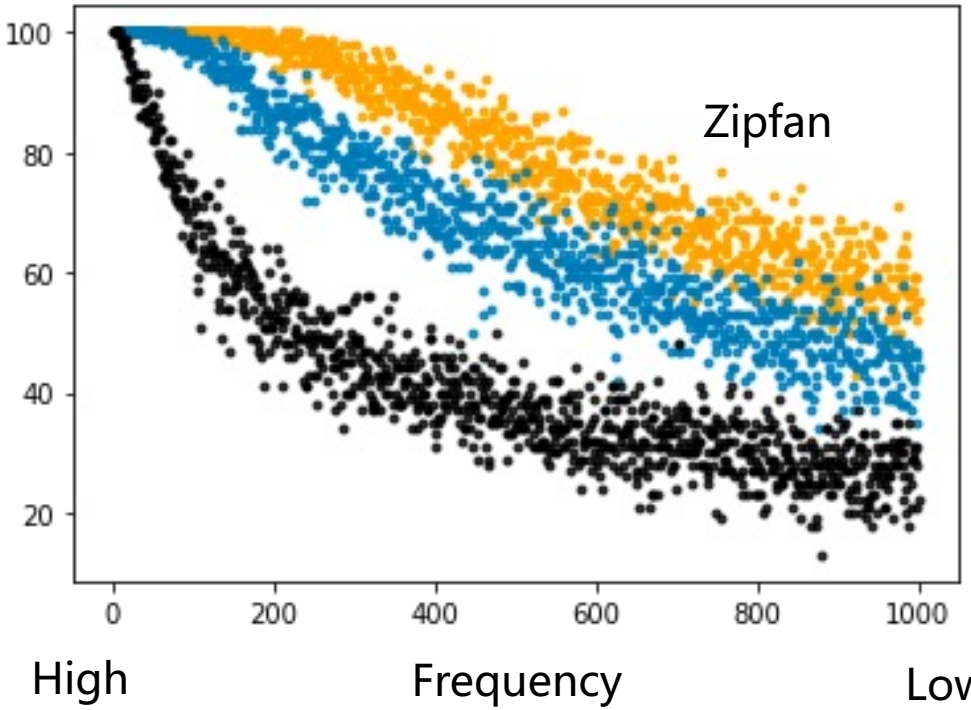
Two **irrelevant** accounts interacted with same popular accounts are close in the latent space.

Strategy: Skew Alleviation

Let the model pay less attention on high-frequency addresses.

1. Frequency-aware negative sampling

Take high-frequency addresses as **negative** samples to counteract the impact of high-frequency addresses being trained frequently.

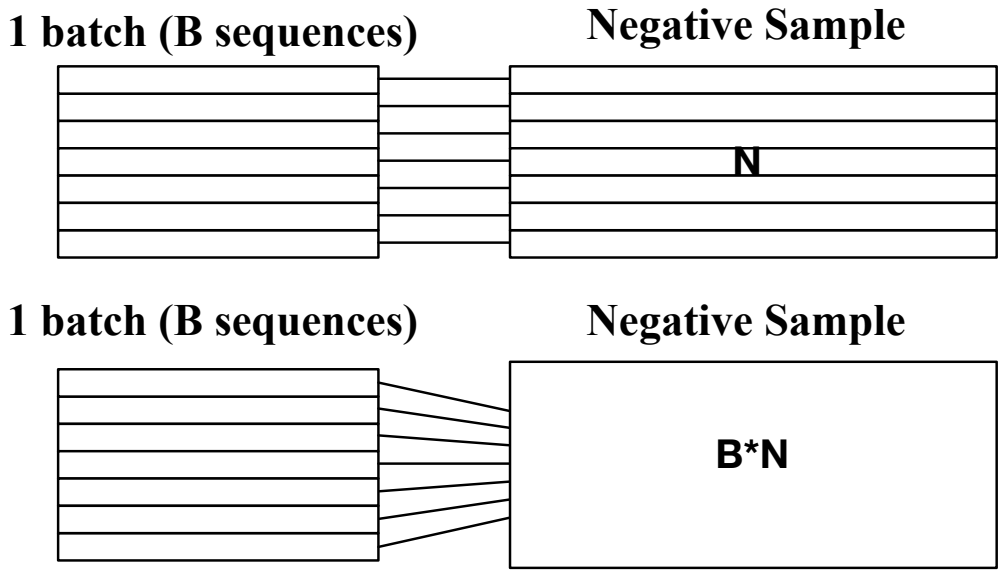


$$L = -\frac{1}{|\mathcal{M}|} \sum_{m \in \mathcal{M}} \log \left(\frac{\exp(\boldsymbol{h}_m^{\mathrm{T}} \cdot \boldsymbol{a}_p)}{\exp(\boldsymbol{h}_m^{\mathrm{T}} \cdot \boldsymbol{a}_p) + \sum_{n \in \mathcal{N}} \exp(\boldsymbol{h}_m^{\mathrm{T}} \cdot \boldsymbol{a}_n)} \right)$$

High-frequency address embedding
As negative address embedding

Strategy: Skew Alleviation

2. Intra-batch sharing



1. Reduce overlap
2. Increase negative-to-positive ratio

Table 1: Testing F_1 of phishing detection *w.r.t.* different skew alleviation strategies. † denotes intra-batch sharing.

P/N Ratio	1:20	1:1000†	1:5000†	1:10000†
Uniform	0.3245	0.3554	0.3804	0.3746
Freq(0.5)	0.3313	0.3939	0.4214	0.4203
Freq(1.0)	0.3770	0.4390	0.4365	0.4371
Zipfan	0.4239	0.4251	**0.5044**	0.5036

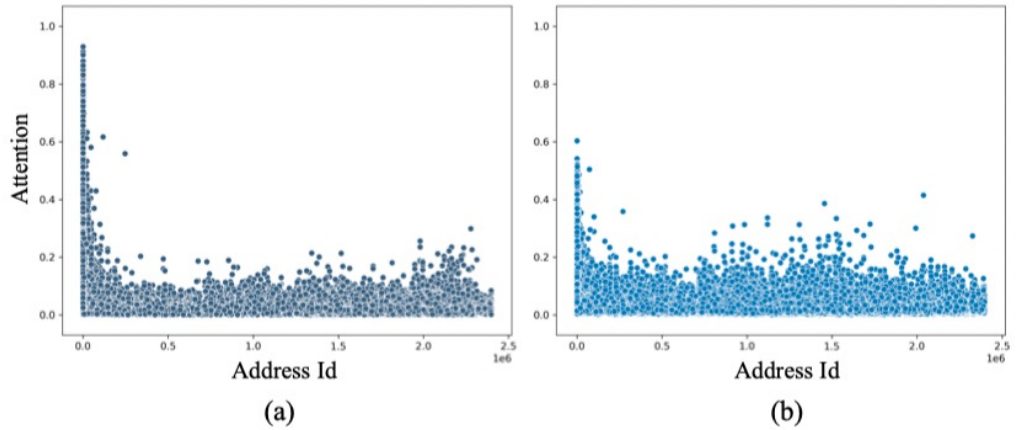
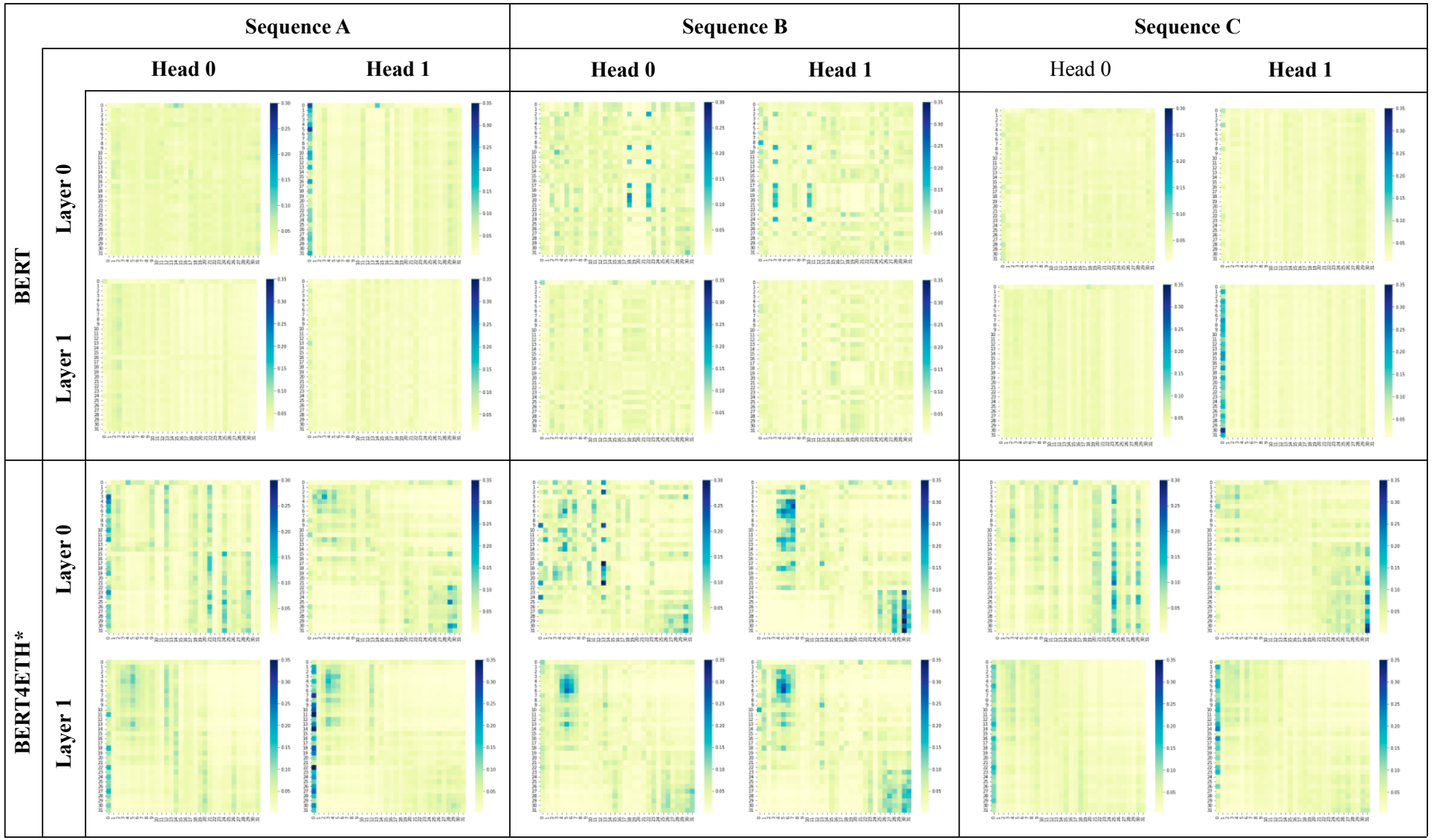


Figure 4: Attention distribution with uniform(a) or Zipfan(b) negative sampling. Address Id is ranked according to the descending frequency.

Visualization of Attention Map

Before

After



The attention mechanism becomes more attentive.

Experiment

Dataset

Table 2: Dataset statistics

Dataset	Phishing	ENS	Tornado	Normal
# EOA	3,220	1,335	2,301	594,038
# All address	151,415	57,526	139,178	2,609,855
# Tranx	328,261	821,140	1,056,674	11,350,640
# In tranx	182,356	60,258	150,157	3,159,707
# Out tranx	145,905	760,882	906,517	8,190,933

Competitor

- DeepWalk-based method: DeepWalk, Trans2Vec, Diff2Vec, Role2Vec
- GNN-based method: GCN, GraphSAGE, GAT **(2 layer = 2 hop)**
- BERT-based method: BERT4ETH **(8 layer)**

Increasing the depth of GNNs decreases the performance (# of GNN layers = # of hop)

Comparison on phishing account detection

Fixed-training: train a MLP classifier on the fixed representations.

Fine-tuning: train the model with a cascaded MLP classifier together.

Table 3: Comparison for phishing detection w/ fixed training.

Method	Precision	Recall	F_1
DeepWalk	0.2293	0.1734	0.1974
Trans2Vec	0.1636	0.1432	0.1527
Diff2Vec	0.2184	0.2000	0.2088
Role2Vec	0.2520	0.2295	0.2402
GCN	0.3181	0.2180	0.2587
GSAGE	0.3023	0.2317	0.2623
GAT	0.3264	0.2581	0.2883
BERT4ETH	0.5483	0.4670	0.5044

Table 4: Comparison for phishing account detection w/ fine-tuning. BERT4ETH w/o pre-training equals to Transformer.

Method	Precision	Recall	F_1
BERT4ETH	0.7153	0.5984	0.6516
w/o pre-training			
BERT4ETH	0.5114	0.3845	0.4389

Pre-training and Fine-tuning are both important.

Comparison on de-anonymization

Euclidean distance is adopted to measure the proximity of two accounts.

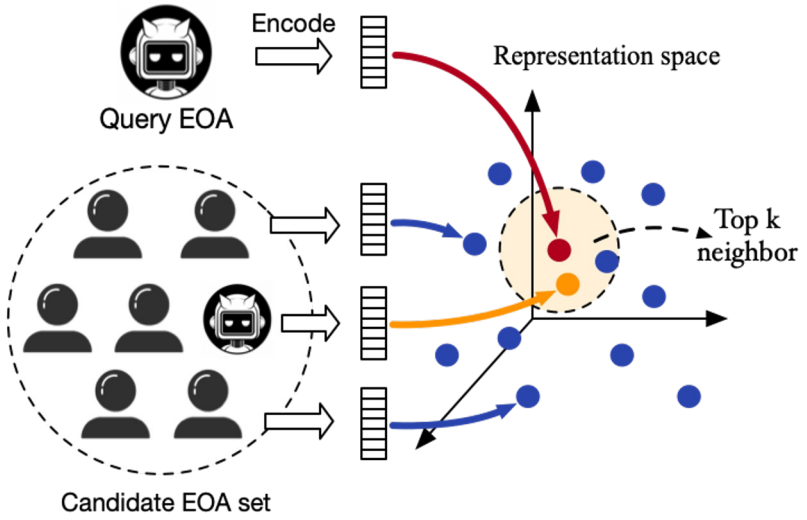


Table 5: Comparison for de-anonymization on the ENS dataset (%).

Method	HR@1	@3	@5	@10	@100	@1000
DeepWalk	2.43	5.21	5.56	7.64	15.28	41.32
Trans2Vec	6.60	8.33	9.03	11.46	19.44	24.65
Diff2Vec	3.82	5.90	6.25	9.03	22.22	43.75
Role2Vec	4.17	5.56	6.25	7.30	17.71	35.07
GCN	1.74	2.78	3.13	4.51	11.46	31.94
GSAGE	5.90	9.03	10.07	12.85	26.39	55.90
GAT	3.13	4.51	5.21	6.25	18.40	47.57
BERT4ETH	16.32	23.26	28.47	32.64	47.92	65.28

Table 6: Comparison for de-anonymization on the Tornado dataset.

Mixer	0.1 ETH			1 ETH		
	# Candidate		# Pair	# Candidate		# Pair
Statistic	405.2		102	263.7		80
Method	**Rank↓**	**HR@1**	**HR@3**	**Rank↓**	**HR@1**	**HR@3**
DeepWalk	112.64	17.65%	20.59%	63.59	26.25%	37.50%
Trans2Vec	100.69	18.63%	28.43%	70.57	26.25%	35.00%
Diff2Vec	82.99	26.47%	37.25%	54.00	35.00%	43.75%
Role2Vec	100.99	21.57%	30.39%	64.65	15.00%	31.25%
GCN	153.75	13.73%	17.65%	87.01	16.25%	25.00%
GSAGE	89.46	30.39%	37.25%	58.01	35.00%	48.75%
GAT	91.25	21.57%	27.45%	60.42	25.00%	38.75%
BERT4ETH	67.96	40.20%	56.86%	42.26	45.00%	62.50%

Ablation Study

Different strategies have different impacts on downstream tasks.

Table 7: Ablation study for phishing detection w/ fixed-training.

Method	Precision	Recall	F_1
BERT4ETH	0.5483	0.4670	0.5044
w/o de-duplication	0.4661	0.3289	0.3857
w/o 80% masking	0.4177	0.2431	0.3073
w/o freq. sampling	0.4554	0.3266	0.3804
w/o batch-sharing	0.5034	0.3661	0.4239
w/o tranx. features	0.5076	0.3607	0.4217

Reducing repetitiveness is more important to phishing account detection.

Table 8: Ablation study for de-anonymization on ENS dataset (%).

Method	HR@1	@3	@5	@10	@100	@1000
BERT4ETH	16.32	23.26	28.47	32.64	47.92	65.28
w/o de-duplication	18.06	26.39	30.21	32.29	46.18	59.38
w/o 80% masking	16.67	23.96	27.43	31.60	41.33	52.78
w/o freq. sampling	4.52	9.03	10.07	11.11	27.78	48.61
w/o batch-sharing	0.69	1.74	3.82	5.90	21.88	43.40
w/o tranx. features	15.28	21.53	26.74	27.09	42.71	62.85

Skew alleviation is more important to de-anonymization.

Visualization (phishing account detection)

(a) DeepWalk

(b) Diff2Vec

(c) Role2Vec

(d) Trans2Vec

(e) BERT (50)

(f) GSAGE

(g) GCN

(h) GAT

(i) BERT4ETH

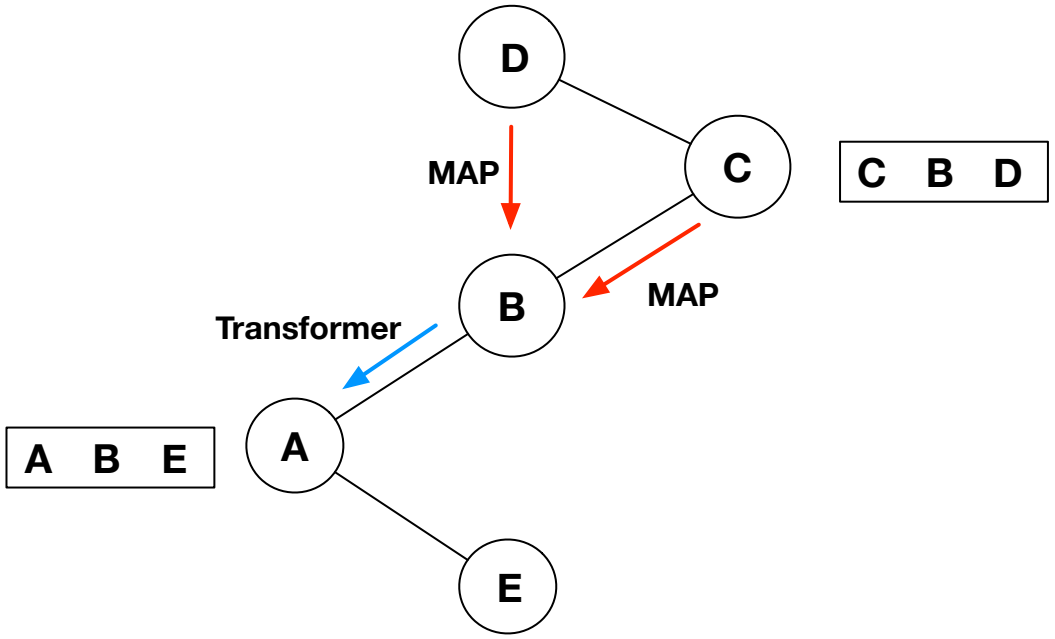
(j) BERT4ETH (normal account)

BERT4ETH still captures multi-hop information

Masked Address Prediction captures 2-hop neighbor information

Transformer captures 1-hop neighbor information

BERT4ETH captures 3 hop neighbor information



BERT4ETH **Address Embedding**

Phishing account detection

BERT4ETH	0.5483	0.4670	0.5044
Addr embed	0.3775	0.2604	0.3084

De-anonymization (ENS dataset)

BERT4ETH	16.32	23.26	28.47	32.64	47.92	65.28
Addr embed	13.32	19.26	24.47	25.64	30.92	39.28

Only using address embedding decreases the performance but still works.

BERT4ETH Paper

BERT4ETH GitHub

Conclusion:

1. A universal pre-trained language model for Ethereum
2. Captures sequential & transaction-level information
3. More resistent to noise & better model capability
4. Still captures multi-hop information

contact: sihaohu@gatech.edu