

# 以太哨兵

基于图大模型的区块链安全一体化检测平台

INTEGRATED BLOCKCHAIN SECURITY DETECTION PLATFORM BASED ON GRAPH LARGE MODELS

为区块链生态安全保驾护航



ETHER SENTINEL

## 以太哨兵

——基于图大模型的区块链安全一体化检测平台

作者：李昊林 & 汪宇恒 & 邱嘉希 & 尚萧然 & 廖杨旭

组织：武汉大学

版本：2025年5月5日



# 目录

<b>第1章 作品概述</b>	<b>1</b>
1.1 选题背景 . . . . .	1
1.2 难点与挑战 . . . . .	3
1.3 相关工作 . . . . .	3
1.3.1 区块链交易风险检测方法 . . . . .	3
1.3.2 基于 Transformer 的图学习方法 . . . . .	4
1.4 特色描述 . . . . .	4
1.5 应用前景分析 . . . . .	5
<b>第2章 作品设计与实现</b>	<b>6</b>
2.1 系统总体设计 . . . . .	6
2.1.1 设计目标 . . . . .	6
2.1.2 系统整体架构 . . . . .	6
2.2 理论研究 . . . . .	7
2.2.1 区块链平台交易与智能合约数据 . . . . .	7
2.2.2 区块链交易的威胁模型 . . . . .	8
2.3 模块设计与实现 . . . . .	9
2.3.1 交易图构建模块 . . . . .	9
2.3.2 交易图分析模块 . . . . .	11
2.3.3 综合判别模块 . . . . .	18
2.4 用户端设计 . . . . .	19
2.4.1 网页架构 . . . . .	19
2.4.2 系统交互 . . . . .	20
<b>第3章 作品测试与分析</b>	<b>22</b>
3.1 测试方案 . . . . .	22
3.2 测试数据 . . . . .	22
3.2.1 预训练数据集 . . . . .	22
3.2.2 下游任务数据集 . . . . .	22
3.3 测试环境搭建及测试设备 . . . . .	24
3.4 系统功能测试 . . . . .	24
3.4.1 模块功能测试 . . . . .	25
3.4.2 可靠性测试 . . . . .	29
3.4.3 易用性测试 . . . . .	30
3.5 系统性能测试 . . . . .	31
3.5.1 评估指标 . . . . .	31
3.5.2 方法基线 . . . . .	32
3.5.3 恶意交易检测测试 . . . . .	32
3.5.4 地址风险评估测试 . . . . .	33
3.5.5 合约代码函数名恢复测试 . . . . .	33
3.5.6 平台迁移性测试 . . . . .	33
3.5.7 敏感性实验 . . . . .	35

---

3.5.8 消融实验 . . . . .	36
3.6 用户端测试 . . . . .	37
3.6.1 首页及功能选择页面 . . . . .	37
3.6.2 恶意交易检测页面 . . . . .	38
3.6.3 地址风险评估页面 . . . . .	40
3.6.4 合约代码函数名恢复页面 . . . . .	41
3.6.5 实时交易数据展示页面 . . . . .	42
3.6.6 系统响应延时 . . . . .	43
<b>第4章 创新性与实用性说明</b>	<b>44</b>
<b>第5章 总结</b>	<b>45</b>
<b>参考文献</b>	<b>46</b>

# 第1章 作品概述

## 内容提要

- 选题背景
- 难点与挑战
- 相关工作
- 特色描述
- 应用前景分析

### 1.1 选题背景

区块链技术最初因比特币的诞生而受到广泛关注，现已成为独立的研究领域。中共中央政治局曾对此进行集体学习，习近平总书记强调区块链在技术革新与产业变革中的关键作用，并提出将其作为自主创新的重点。全球区块链生态不断进化，涌现出 ICO、以太坊等新形态，我国也推出了基于区块链的数字人民币（E-CNY）[70]。

DeFi（去中心化金融）是建立在区块链技术上的金融架构，允许无需传统金融中介机构即可进行资产交易、借贷、投资等金融活动。去中心化金融提供的金融交互方式使得用户可以直接进行资金交易和金融服务，不受地理和监管的限制。这种开放和无门槛的特性，虽然极大地促进了金融创新和包容性，但同时也暴露了严重的安全风险。自 2018 年以来，DeFi 生态系统中的资产价值急剧增加，高峰时期锁定的总价值超过 2530 亿美元。这种快速增长吸引了大量用户和资本，但也成为了攻击者的主要目标。从 2018 年到 2022 年，DeFi 攻击导致的总损失估计至少为 32.4 亿美元 [9]。这些攻击事件不仅带来了经济损失，还可能破坏对区块链技术的信任和市场稳定性。

以常见的 DeFi 交易平台——以太坊为例。截至 2024 年 5 月，以太坊拥有超过 9600 万的账户数量、日平均 117 万的交易数量以及超 4100 亿的总估值 [71]。然而，在以太坊、BSC 等 DeFi 平台蓬勃发展的同时，也经历了令人触目惊心的安全事故。从基于区块链技术的所有金融方向来看，DeFi 是发生安全事件最多，损失最大的领域。纵观所有的区块链交易平台，2023 年整年共有 DeFi 安全事件 282 件，占区块链安全事件总数的 60.77%，损失高达 7.73 亿美元，对比 2022 年（共 183 件，损失约 20.75 亿美元），2023 年 DeFi 安全事件的损失虽然降低了 62.73%，但是事件数量却上升了 54.64%，意味着 DeFi 领域在防范和处理安全问题上仍然面临严峻挑战。

区块链交易平台安全事件的数量、涉案金额仍然在以较高的趋势增长，图 1.1.1 总结了自 2018 年 5 月至 2022 年 4 月的 DeFi 安全事件每月份的数量与损失金额 [69]。随着 DeFi 生态系统中资产的急剧增加，DeFi 安全事件的数量也随之爆发。虽然每月平均发生的安全事件数量约为 10 件，但由于 DeFi 平台的特性，往往单次案件涉案金额极为巨大，均以百万美元计算，因此总体损失规模也使得 DeFi 平台备受困扰。

DeFi（去中心化金融）领域的安全事件在近年来给用户和项目带来了巨大的资产损失。以下是一些典型的区块链交易安全事件，其总结如表 1.1.1 所示。

攻击事件	攻击手段	损失金额（以美元计）
Poly Network 跨链协议黑客攻击	跨链协议攻击	610,000,000
Zapper 漏洞	合约漏洞	7,500,000
Platypus Finance 事件	闪电贷攻击	85,000,000
RariCapital 安全事件	可重入漏洞	80,000,000
OneCoin 事件	加密骗局	4,000,000,000
Warp Finance 攻击	预言机攻击	8,000,000

表 1.1: 攻击事件实例总结

Poly Network 跨链协议黑客攻击（2021 年 8 月 10 日）[45]: 跨链协议 Poly Network 遭遇黑客攻击，使用该协议的 O3 Swap 损失惨重。黑客在 30 多分钟内带走了总计 6.1 亿美元的加密资产，包括 3.02 亿枚 USDT、5.5 万枚以太坊、2 千枚比特币等若干代币。这次攻击的损失是 2020 全年 DeFi 黑客攻击损失的 4.7 倍。

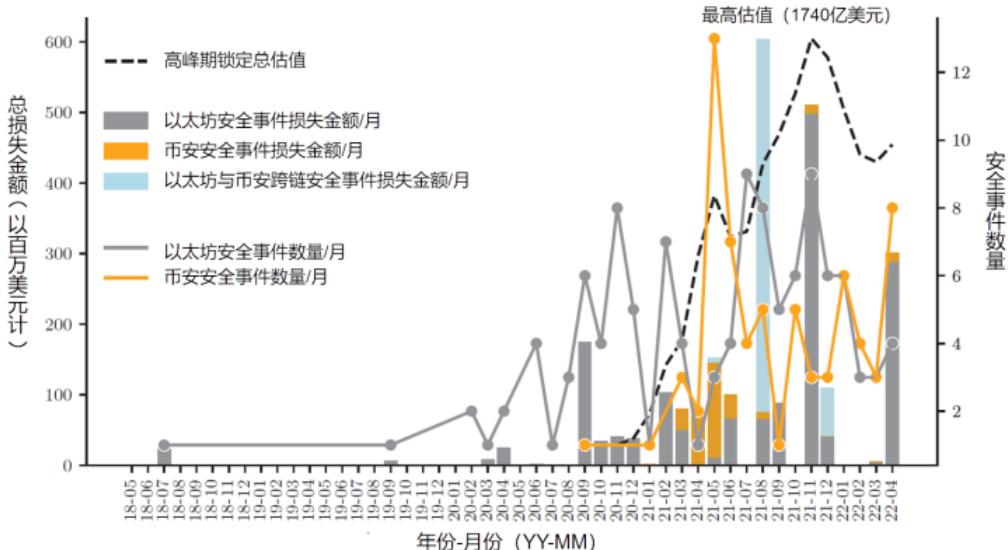


图 1.1: Defi 安全事件每月数量与损失金额

Rari Capital 和 Fei Protocol 安全事件（2022 年 4 月 30 日）[56]: 与 DeFi 借代平台 Rari Capital 和算法稳定币协议 Fei Protocol 相关的池遭到黑客攻击，损失超过 8000 万美元。遭受攻击的根本原因是典型的可重入性漏洞。

Zapper 漏洞事件（2021 年 6 月）[62]: DeFi 资产管理平台 Zapper 在旧版“Polygon Bridge”智能合约中发现了一个漏洞，该漏洞允许攻击者窃取无限批准的资金。

Platypus Finance 事件（2023 年 2 月 17 日）[57]: Avalanche 上的 DeFi 平台 Platypus Finance 遭遇闪电贷攻击，被盗走约 850 万美元。攻击者通过闪电贷方式从 Aave 借了 44,000,000 个 USDC，然后将所借的 USDC 全部存入 USDC 池中，同时获得相等数量的 LP-USDC。接着，攻击者将 LP-USDC 存入 MasterPlatypusV4 合约，并借得超 40 亿个 USP。最终，攻击者成功偿还了闪电贷款，总共获利约为 850 万美元。

OneCoin 事件（2017 年）[26]: 基于庞氏骗局的加密货币案件，由 Ruja Ignatova 领导。该项目在筹集了 40 亿美元后于 2017 年突然消失，成为最大的加密骗局。

Warp Finance 预言机攻击（2020 年 12 月 18 日）[7]: DeFi 借贷协议 Warp Finance 遭受黑客攻击，攻击者利用 Uniswap 交易对的相对价格作为预言机的喂价源。通过操纵 Uniswap 交易对的价格，攻击者破坏了 Warp Finance 的借款价值判断标准，损失金额近 800 万美元。表 1.1 展示了一些造成了合约重大损失的异常交易事件，总损失价值超过 2.76 亿美元。

综合研判 21 年初到 24 年初的区块链交易平台安全生态事件 [6, 17, 33, 40, 68]，从损失金额、攻击手法、审计情况等多角度来分析安全事件特征，本团队对区块链交易平台安全事件有以下洞察和发现：

1. 从损失金额来看：区块链交易平台的安全事件的涉案金额往往十分巨大，单次事件均以百万美元计算。例如，在 2022 年第一季度，发生 2 起 Solana 链典型攻击事件，损失金额高达 3 亿 7400 万美元，Solana 链因此遭受了重创。
2. 从攻击手法来看：合约漏洞利用和闪电贷攻击是黑客最常使用的手段。超过半数的区块链交易平台安全事件，攻击手法为合约漏洞利用。并且，超 80% 情况下，黑客都会将立即将盗取资金转移，进行混币或多次交易，以提高追踪难度。
3. 从审计情况来看：在所有被攻击的区块链项目中，大部分项目均经过了第三方安全公司审计。然而，剩余的未经过审计的小部分区块链项目，其因攻击事件遭受的损失占所有被攻击区块链项目总体损失金额的 60%[8, 67]。
4. 从链平台来看：Ethereum 和 BNB 依旧是被攻击频次最高的两条链，但高攻击频次并不意味着高损失金额。以 Solana 链为例，2022 年第一季度仅发生了 2 起安全事件，就造成了超 3 亿美元的损失，远远多于 BNB Chain 上的损失。

综上，区块链交易平台的安全事件，往往单次案件涉及巨额资金，并且链平台遭受攻击频率低并不意味着

低损失金额，同时，未进行安全审计的区块链项目案发金额占比超过 60%，意味着不同链平台的不同区块链项目均需要高效准确的交易风险监测系统。从攻击手法来看，大多数攻击均是利用合约漏洞或闪电贷来完成。其次，为了对抗攻击者的反追踪手段，需要对交易涉及到的账户地址及历史交易信息作全面的分析。因此，恶意交易检测和地址风险评估是区块链平台风险监测中的重要任务。

所以，如何构建一个具有良好迁移性的智能化区块链交易风险监测系统，使其能够部署在不同的区块链交易平台中，捕获更加全面的交易信息并高效准确的处理和分析，是保证去中心化交易生态安全性的一个亟待解决的重要问题。

## 1.2 难点与挑战

当前的区块链异常交易、风险地址检测方法往往依赖于历史数据和人工经验，对于新出现的、复杂的异常交易行为，检测效果可能不佳。本团队设计了具有良好迁移性的智能化区块链交易风险监测系统，其存在以下难点与挑战：

### 1. 海量复杂异构的交易数据

如何有效地处理海量、异构的信息以服务于分析金融风险是当前的一大难点。区块链行业的迅猛发展带来了海量的交易数据，这些交易数据往往已经达到了百万、千万级别，并且交易数据量仍在以极快的速度增长 [43, 50, 63]。而由于区块链平台本身是一个复杂的生态，其中包含多种角色与相应的信息，数据信息的结构也是多种多样，例如账户本身信息包含余额、历史交易、账户代码等，交易则包括金额、交易日志、时间戳等等。由于搜索空间的限制和所需的大量人工工作，用于区块链交易的实时入侵检测系统仍然具有挑战性。而对于数据的异构性

### 2. 实时动态智能风险监测

传统的静态分析方法已无法适应不断发展的区块链技术和演变的安全威胁 [51, 52]。需要结合人工智能技术，实现不依赖预定义规则和模式的动态、实时检测，能够应对更广泛的异常情况。系统需要具备快速响应能力，在检测到风险后，能够迅速生成应对措施，如生成反代码混淆脚本，拦截危险交易。

### 3. 高效、鲁棒的数据特征表示学习

目前的基于模式的方法主要是通过对目标区块链平台进行大规模调研和统计分析，最后总结出有效特征进行的，然而在不同的区块链平台上这些被总结出的特征也许会起到不同的作用，因而无法直接套用 [29–31, 41]。另一方面，对于区块链平台大规模的调研和统计极其耗时，因而几乎不可能对小的区块链平台专门做分析，这意味着小规模的区块链平台或项目难以得到有效的安全保障。

## 1.3 相关工作

随着区块链技术及 DeFi 平台的迅猛发展，针对 DeFi 交易平台的安全事故层出不穷并造成严重后果。国内外相关部门和研究人员提出各类应对措施以防范和打击针对或基于区块链的犯罪事件。本系统着眼于利用图神经网络和 Transformer 来实现对区块链交易的智能化风险监测，由此我们在下面梳理相关的技术。

### 1.3.1 区块链交易风险检测方法

DeFi 安全事件都以异常交易的形式体现，从大量交易数据集中发现疑似的异常交易并进行评估、分类，是抵御 DeFi 安全事件的重要手段。目前，区块链异常交易的检测方法主要有以下几种：

**(1) 基于交易奖励的方法** 这类工作主要通过寻找有利可图的交易从而在交易层面寻找漏洞，但这类工作大多无法进一步解读区块链交易之间的图结构关系，因而无法对交易的风险进行具体归因。APE [46] 是一种利用动态程序分析技术的通用模仿攻击方法，支持对抗性智能合约的自动合成，APE 基于寻找有利可图的交易模式来运行，通过分析和模仿交易中的智能合约调用的动态控制流图以及交易相关的账户状态来确定模仿交易的盈利能力，随后采用动态污染分析来合成可盈利交易的智能合约。然而，APE 存在诸多限制。由于 APE 不包括任

何前设交易分析相关知识，而只是对交易进行模拟，因此不适用于需要复杂语义推理的情况。此外，APE 无法使用非原子策略，也即只能实施交易而无法识别该交易在交易图中的位置，忽视了区块链交易网络图中的结构信息。

**(2) 基于交易执行模式的方法**这类方法通过对交易的执行模式进行分析，分辨出异常的执行模式并据此报警。但这类方法往往忽略交易的图结构信息，因此也无法准确分析交易的风险来源，在准确率上也较为有限。加州大学的 Yu Gai 等人提出了基于交易执行模式的 BLOCKGPT [21] 系统，其由事务跟踪器、训练模块和检测模块组成。事务跟踪器能够捕获由用户与 dApp 交互发起的事务的执行跟踪，包括智能合约函数调用的顺序、相关输入输出数据，并提供执行路径的详细视图。该跟踪用作检测模块的输入，训练好的模型可对新交易生成对数似然度评分，最后使用排名或阈值判断是否报警。然而，该方法虽然考虑了交易本身的执行情况，却未考虑交易与其他交易之间的联系，导致准确率较低。

**(3) 基于交易图结构的方法**一些工作注意到交易信息隐含的图结构，并基于此研究区块链异常交易检测。TTAGN [55] 通过时序图神经网络增强以太坊上的网络钓鱼诈骗检测性能。该方案通过建模节点之间的历史交易时间关系来构建边表示，并通过图神经网络将其与拓扑结构信息一起融合到节点特征中。然而，该方案虽然能够基于图拓扑信息进行学习，但依赖于预定义的统计特征，而时间信息往往可被攻击者操控。

**(4) 基于账户状态的方法**新加坡国立大学的 Siyao Hu 等人提出了 BERT4ETH [54] 模型，基于以太坊数据，采用 BERT 预训练方法，利用 Transformer 优秀的序列建模能力捕捉用户账户的行为模式。该方法的局限在于依赖节点历史状态，在无许可区块链平台（如比特币、以太坊）中，攻击者可以轻易创建新节点以混淆行为轨迹，增加追踪难度。

### 1.3.2 基于 Transformer 的图学习方法

在本系统中，将获取的交易数据构建为图结构并学习节点与边的特征，以获取异常交易与正常交易的特征差异并训练分类器。因此，如何学习图中的信息并捕捉节点之间的复杂关系，是系统实现的关键基础。

联想研究所的 Jin Yuan 等人提出了图注意力网络(Graph Attention Networks, GAT) [28]，是最早将 Transformer 的注意力机制应用于图表示学习的工作之一。GAT 使用多头注意力机制学习图中节点间的关系并进行信息融合。与图卷积神经网络 (GCN) [42] 相比，GAT 使用注意力分数代替拉普拉斯矩阵，更好地捕捉节点间的依赖关系。

Dexiong Chen 等人提出了 SAT [11] 架构，在注意力机制中引入以节点为中心的子图表示，扩展节点特征的指数核以考虑局部结构。

伊利诺伊大学芝加哥分校的 Jinwoo Kim 等人 [27] 证明，即使未针对图结构做修改，标准 Transformer 在图学习理论与实践中仍有良好表现。

GraphGPS [12] 将局部消息传递机制与 Transformer 架构分开，以克服特征过度平滑问题。两者作为独立模型输出，经多层次感知机融合进行最终预测。

## 1.4 特色描述

本系统探讨了如何利用图神经网络和 Transformer 技术来进行区块链交易信息的检测与甄别。通过将区块链交易信息建模成图结构，并应用图神经网络和 Transformer 架构进行特征抽取和特征学习，可以提高对区块链异常交易信息（恶意交易、欺诈行为等）的检测精度。本系统相比于现有的检测技术，具有以下特点：

1. **实时监测：**传统的检测系统通常是基于规则和模式的静态分析，已经无法适应目前不断发展的区块链技术和不断演变的安全威胁与欺诈行为。静态的规则和模式一旦遇上异常行为的新型变种，能够起到的作用就十分有限。本系统结合人工智能技术和传统方法，提出了不依赖预定义规则和模式的 EtherSentinel 架构，实现对异常区块链交易动态、实时的检测，能够检测更广泛的异常情况。
2. **高准确性：**为提高区块链异常交易数据、交易平台风险账户的准确性，本系统结合了人工智能技术，应用了 Graph Transformer 架构，融合了区块链交易信息的语义特征和交易数据的图结构信息，进行多角度的精

确甄别。在恶意交易检测功能达到了 92% 的准确率，账户风险检测功能达到了 85% 的准确率，合约代码函数名恢复达到了 0.82 的 BLEU 指标。

3. **高兼容性：**相比于传统方法，本系统不依赖于固定的模式特征和预定义规则，仅采用将账户的代码和交易信息作为字符串输入，利用大模型的文字处理能力，从而不会受到不同区块链交易平台间的交易格式和账户状态格式的影响，因此可以方便地部署在不同的区块链交易平台上。
4. **代码函数名恢复功能：**与常规的自动检测平台不同，本系统在交易风险监测和地址风险检测的功能基础上，还增加了代码函数名恢复功能。该功能可以将账户刻意混淆的函数名或由于账户合约代码反编译失败而丢失的函数名恢复，有助于用户通过分析合约代码的函数名来对该账户进行更加深入的安全分析。
5. **综合特征分析：**相比于传统的方法，本系统使用的数据集中包含了非常丰富的交易信息，在原来的数字类信息的基础上，引入了文本类的特征，例如合约代码、合约调用记录等。使用提出的特征提取方法收集了交易元数据特征、合约调用记录特征、账户特征、交易图结构特征。通过图神经网络抽取到特征后，将其输入到 Transformer 架构中，学习节点的重要性、交易的相关性、交易图拓扑结构等深层次信息，为综合判别提供更加全面、有效的源信息。

## 1.5 应用前景分析

本系统是一个具有良好迁移性的智能区块链交易风险监测系统，适用于基于区块链技术的不同去中心化交易平台的相关安全服务，能够有效保障智能合约安全、用户数字资产安全，以及为金融服务监管和合规审查提供有力支持。

本系统与国家运用大数据推进区块链行业创新发展的方针相契合，与当下区块链交易市场的需求相契合。随着区块链数字资产市场的不断扩大，金融机构和用户对数字资产的安全保护需求也越来越高。面对日益增长的交易规模和复杂的风险环境，本团队研发的智能化风险监测系统具备关键的下游任务能力，包括交易风险检测、地址风险评估以及合约代码函数名恢复。这些功能的具体应用将极大地提升 DeFi 交易的安全性和透明度，同时降低因安全漏洞或欺诈行为导致的风险，具有良好的应用前景：

1. **去中心化金融（DeFi）平台安全：**本系统具有很好的可迁移性，可部署在不同的去中心化金融平台上，并对所有交易进行实时监控，实现实时交易风险检测。系统能够检测钓鱼攻击、欺诈交易等异常交易行为，并及时发出警报或自动中止可疑交易，保障资金安全。同时，本系统能够评估账户地址的历史行为和关联性，标记高风险地址并进行黑名单管理，以防范异常交易行为，提升平台的安全性和可信度。
2. **去中心化应用（DApps）安全：**去中心化应用，也即智能合约，常以合约账户的形式部署在区块链交易平台上。本系统能够对智能合约进行审查，恢复和解释合约函数名，帮助用户理解合约逻辑并提供合约账户的风险评估报告，避免恶意合约和代码漏洞带来的风险。针对不断变化的区块链生态环境，本系统可以提供动态的合约账户风险评估，保障去中心化应用的安全运行。
3. **数字资产保护：**以 NFT（非同质化代币）商品为代表的基于区块链技术的虚拟商品近年来愈发火热。非同质化代币本质是可标明所有权的电子凭证，针对 NFT 商品等区块链数字资产的攻击事件近年来不断增长。通过部署本系统，可以迅速识别并响应异常的数字资产交易行为，为用户提供科学的风险管理建议，提升资产保护水平，以应对不断变化的数字资产安全威胁。
4. **金融服务监管与合规支持：**全球区块链生态不断进化，我国也推出了基于区块链技术的数字人民币（E-CNY）。本系统可作为区块链金融服务的自动化风险管理工具，通过提供详细的交易跟踪记录和风险报告，使得交易过程更加透明，便于审计和监管，以支持国家监管机构的工作，推动区块链行业的合法合规发展。

# 第2章 作品设计与实现

## 内容提要

- 系统总体设计
- 理论研究

- 模块设计与实现
- 用户端设计

## 2.1 系统总体设计

### 2.1.1 设计目标

近年来，基于区块链的去中心化交易平台总共遭受了数十亿美元的损失 [64]，亟待更加通用、动态和可扩展的方法来检测异常区块链交易。然而，由于搜索空间的限制和所需的大量手动工程工作，用于区块链交易的智能化风险监测系统仍然具有挑战性。存在如海量异构交易数据、实时动态智能风险监测、高效鲁棒的数据特征表示学习等难点。系统部署图如图 2.1 所示。

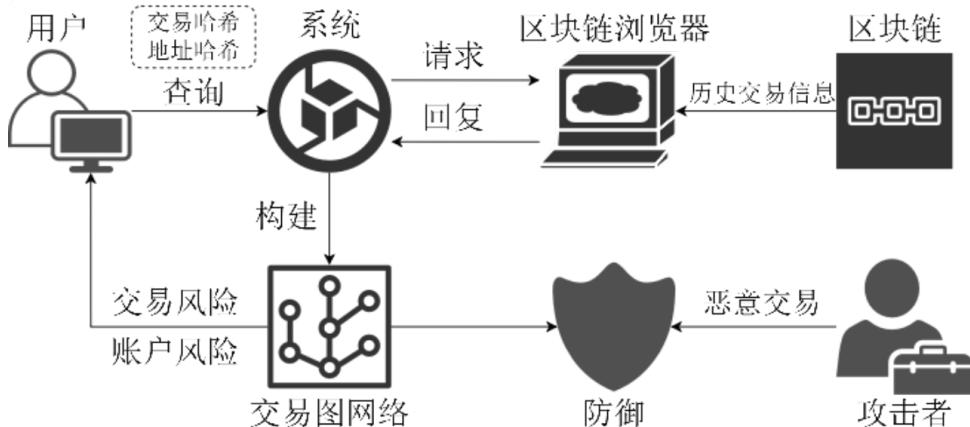


图 2.1: 系统部署图

因此，本作品旨在设计 EtherSentinel 系统，通过图神经网络和 Transformer 技术深入分析交易数据，以识别潜在风险并采取预防措施，提供全面的区块链智能化风险监测服务，从而增强区块链交易平台的安全性。本系统设计为用户提供的功能如下：

- **实时构建交易网络图**：本系统利用区块链的交易数据构建交易网络图，这些交易网络图展现了账户之间的交易关系和交互模式。
- **恶意交易检测**：基于图神经网络和 Transformer 的深度学习分类模型，系统能够识别和分类可能的恶意交易，如钓鱼、欺诈或其他非法活动。
- **地址风险评估**：评估和打分区块链地址的信誉，依据历史交易行为和网络关系来预测地址可能涉及的风险等级，为用户或者监管机构提供关键信息和决策依据。
- **合约代码函数名恢复**：对于智能合约的字节码，系统能够恢复出原始的合约代码函数名，有助于进一步的代码审计和安全分析。

### 2.1.2 系统整体架构

面向区块链的图大模型交易风险监测系统由三个模块组成，分别为交易图构建模块、交易图分析模块和综合判别模块。系统总体架构如图 2.2 所示。

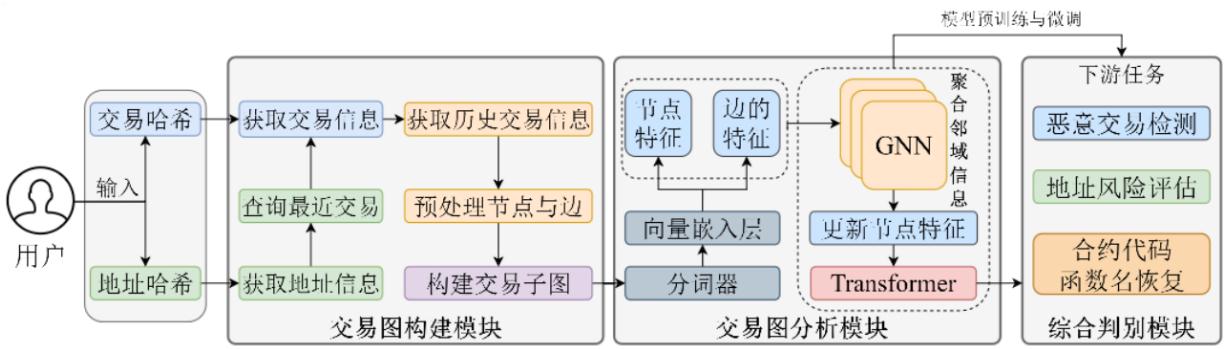


图 2.2: 系统整体架构

- 交易图构建模块:** 在区块链交易平台上, 存在大量的账户, 账户之间的交易无时无刻不在发生。如果将某一时间段内的所有交易数据获取下来, 该数据集就隐含了图的结构, 交易双方的账户可以视为图中的节点, 交易以及合约调用是账户节点之间的交互关系, 可以作为边。交易图构建模块基于交易哈希或地址哈希, 通过获取历史交易信息来构建相关的交易子图, 该图展现了账户之间的交易信息、交易关系和交互模式。
- 交易图分析模块:** 对于构建的交易子图, 交易图分析模块首先将节点和边中的信息作为文本信息进行分词和向量嵌入, 而后, 通过图神经网络 GNN[16] 来聚合邻域信息并更新节点特征。之后, 采用 Transformer[1] 架构来学习交易数据信息中的复杂模式和依赖关系, 进而获得区分正常交易和恶意交易的能力。此外, 通过采用预训练策略, 模型能够在多个相关任务上同时进行训练和优化, 提高了模型的泛化能力和下游任务表现。
- 综合判别模块:** 本系统针对恶意交易检测、账户风险检测以及合约代码函数名恢复这三个核心功能进行了深度优化与定制, 以期达到高效、准确的安全防护目的。通过对预训练模型进行下游任务微调, 使模型在恶意交易检测、账户风险检测、合约代码函数名的下游任务中获得更好的表现效果和实时性能。

## 2.2 理论研究

### 2.2.1 区块链平台交易与智能合约数据

下面将以常见的区块链交易平台以太坊为例, 介绍与区块链交易相关的概念。

(1) **区块链交易:** 交易是由账户发出, 带密码学签名的指令。账户将发起交易以更新以太坊网络的状态。最简单的交易是将 ETH 从一个账户转到另一个账户。以太坊交易是指由外部持有账户发起的行动, 外部持有账户是指由人管理而不是智能合约管理的账户。

在以太坊中, 交易的过程如下: 首先是交易创建, 用户通过钱包或应用创建并签名交易。而后广播交易, 将签名后的交易被广播到以太坊网络。随后进行交易验证, 节点验证交易的合法性, 包括签名验证、nonce 检查和余额检查。验证完成后, 执行交易打包, 矿工将交易打包到区块中, 执行交易代码并记录结果。最后, 确认交易, 区块被添加到区块链上, 交易被确认并不可更改。

一笔以太坊交易通常由三个部分组成, 包括元数据 (metadata)、缓存 (cache)、载荷数据 (payload)。其中, 元数据包含交易所需的关键字段信息, 包括交易金额、签名数据、时间戳等等。而缓存, 则存储了交易预计会使用的账户和私钥的列表。载荷数据中常常携带的是交易所调用的智能合约代码或者 API 信息。元数据信息和载荷数据是分析区块链交易的重要依据, 其包含的主要内容如表 2.1 所示。

(2) **区块链账户:** 一个以太坊账户 [13] 是一个具有以太币 (ETH) 余额的实体, 可以在以太坊上发送交易。账户可以由用户控制, 也可以作为智能合约部署。以太坊有两种账户类型: 一种为 EOA 账户, 由用户控制的账户, 需要用户设定私钥; 另一种为合约账户, 指的是部署到网络上的智能合约。以太坊账户的关键字段如表 2.2 所示。

EOA 账户的特点是由私钥控制, 私钥生成公钥, 再从公钥生成账户地址。其具备发送交易、持有以太币 (ETH) 和调用智能合约等功能。同时, 以太坊中的 EOA 账户发送交易需要支付燃油费用。EOA 账户地址是由 20 字节

表 2.1: 区块链交易关键字段

元数据字段	字段含义	阐述
Hash	交易哈希	唯一地标识一个交易
Blocknumber	交易区块序号	交易所在的区块号
From	交易创建者地址	交易发起者的地址哈希
To	交易接收者地址	交易接收者的地址哈希
Value	转账金额	转账的 ETH 数量
Gas	燃油费单位	量化执行交易或合约所需的计算工作量的单位
Gasprice	燃油价格	交易发起者愿意为处理交易而支付的每个 gas 的价格
MaxFeePerGas	最大 GAS 值	愿意为交易支付的最大 gas 值
MaxPriorityFeePerGas	优先级最大 GAS 值	愿意为交易费用的优先级部分支付的最大 gas 值
TransactionIndex	交易索引	指向在区块中每个交易的序号
Type	交易类型	标识交易类别：常规交易、合约部署交易或执行合约
ChainID	以太坊链 ID	表示以太坊不同的链
Payload 字段	字段含义	阐述
Code	合约代码	交易附带的合约代码
API	交易调用的 API	交易调用的合约代码 API

构成的字符串，例如 0x742d35cc6634c0532925a3b844bc454e4438f44e。

合约账户是由代码控制的账户，通过部署智能合约创建，创建合约存在成本，因为需要使用网络存储空间。合约账户的控制权由智能合约代码掌握而不是某个用户。合约账户同样可以持有以太币(ETH)、接受和发送交易，同时合约账户可以执行部署在账户中的合约代码。合约账户地址形式上也为 20 字节字符串，该字符串会在合约部署后自动由系统生成。在智能合约部署时，其代码被存储在区块链上，并且能够保存合约的内部状态，代码的形式是由函数名组成的，而函数名由开发者定义。合约创建者通过交易将智能合约代码发送到以太坊网络中，该部署交易被区块链认可后则智能合约成功部署，此后，外部账户或者其它合约账户就可以通过发送交易来调用该智能合约的函数。

(3) 智能合约：前述的合约账户与 EOA 账户最大的区别在于，合约账户部署了智能合约。在以太坊中，智能合约是以太坊平台上运行的自动化脚本，用于执行预定的指令，具有不可篡改和自动执行的特点。其具有以下特点：(i) 自动执行：合约一旦部署，则按照预定义的规则自动执行。(ii) 不可篡改：合约部署后代码和状态不可更改。(iii) 透明性：合约代码公开可见，任何用户都可以对其进行审计。

表 2.2: 以太坊账户关键字段

字段	字段含义	阐述
Balance	账户余额	地址拥有的 Wei 数量，每个 ETH 币有 $1e+18$ 个 Wei。
Nonce	计数器	显示外部账户发送的交易数量或合约账户创建的合约数量。
codeHash	代码哈希	表示以太坊虚拟机(EVM)上的账户代码，供状态数据库检索。
storageRoot	存储哈希	编码了账户的存储内容，默认情况下为空。

## 2.2.2 区块链交易的威胁模型

下面介绍区块链交易的威胁模型，并且本系统能够对以下异常交易进行甄别：

(1) 恶意合约：恶意合约 [15] 是攻击者利用智能合约代码中的漏洞进行攻击的一种手段。这些合约通常看似合法，但包含隐藏的恶意代码，能够在特定条件下触发，导致资产被盗。例如，攻击者可以在智能合约中设置后门，使其在不被察觉的情况下转移资金。这种攻击方式由于其隐蔽性，往往难以被发现和防范。

(2) 欺诈交易：区块链平台上的欺诈行为 [22] 形式多样，包括虚假 ICO 和庞氏骗局。攻击者通过发布虚假项目吸引投资者投入资金，随后卷款潜逃。由于区块链交易的匿名性和不可撤销性，一旦发生欺诈行为，受害者难以追回损失。例如，一些项目在 ICO 结束后立即关闭网站和社交媒体账户，导致投资者蒙受巨大损失。

(3) **黑名单机制**: 黑名单机制通过记录和共享已知的恶意地址来防范攻击。区块链平台通常维护一个黑名单，包含被标记为恶意的地址，用户在交易时可以查询该名单，避免与这些地址进行交易。然而，黑名单的维护和更新需要高度的及时性和准确性，才能有效防范攻击，因此，黑名单往往缺乏动态性。

(4) **蜜罐**: 蜜罐 [64] 是一种主动防御手段，通过部署看似存在漏洞的合约，吸引攻击者进行攻击，从而监控其行为并收集证据。研究人员可以部署含有明显漏洞的智能合约，当攻击者尝试利用该漏洞时，蜜罐会记录其行为并采取相应防护措施。因此不仅能够有效防范攻击，还能提供有价值的情报，帮助改进安全措施。

(5) **钓鱼攻击**: 钓鱼攻击 [10] 是区块链技术面临的一大威胁，利用其开放性和去中心化特性，攻击者伪装成可信实体，通过电子邮件、网站或在线聊天等方式，诱骗用户提供敏感信息如用户名、密码和私钥。在区块链领域，钓鱼攻击形式多样，不仅通过虚假网站获取信息，还通过发送恶意地址的电子邮件直接骗取资金。在 Bee Token 的 ICO 期间，攻击者通过虚假电子邮件诱使投资者转账，25 小时内骗取近 100 万美元。由于这类攻击不依赖传统钓鱼网站，传统检测方法难以应对。

(6) **黑客攻击**: 黑客攻击 [59] 可以采取多种形式，包括智能合约漏洞利用、网络节点攻击以及矿工作恶行为等。例如，2016 年的 DAO 攻击和 2017 年的 Parity 钱包攻击分别导致了超过 5000 万美元的损失。此外，矿工利用交易排序和前置运行等技术获取不公平利益。更复杂的攻击如日食攻击 (eclipse attacks) [25]，攻击者通过控制受害者的网络连接来隔离其与区块链网络的正常通信。

## 2.3 模块设计与实现

### 2.3.1 交易图构建模块

交易图构建模块基于交易哈希和地址哈希，获取其历史交易数据，从而扩展出若干账户节点和交易，将其构造为节点和边，从而构建了交易子图，为后续交易图分析模块作好基础准备。交易图构建模块的整体流程如图 2.3 所示：

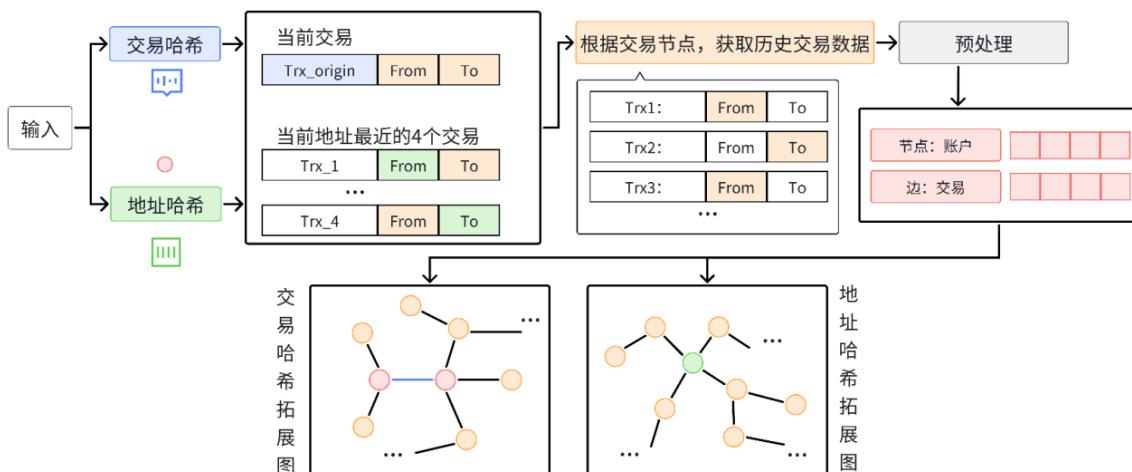


图 2.3: 交易图构建模块整体流程

区块链交易平台上，存在大量的账户，账户之间的交易无时无刻不在发生。如果将某一时间段内的所有交易数据获取下来，该数据集就隐含了图的结构，交易双方的账户可以视为图中的节点，交易是账户节点之间的交互，就可以视为边。因此，交易图就是交易数据隐含的图结构的具象化，交易图构建模块将图结构从抽象具象化为能够被程序处理的形式。在构建过程中，交易图构建的基本思路如下：

- **节点表示**: 交易的双方可以作为图中的节点。每个节点可以包含一些属性，例如地址、公钥或账户标识符等。这些节点代表了参与区块链交易的实体，例如个人、组织或智能合约。
- **边表示**: 每一次转账交易可以表示为图中的一条边。这条边连接着交易的出点和入点，表示了资金的流动路径。边可以包含一些属性，如转账金额、Gas 费用、时间戳等。这些属性描述了交易的细节信息 [20]。

- 图结构：将所有交易的节点和边连接起来，即可形成一张图，其中节点代表参与交易的实体，边代表交易行为。这样的图结构能够反映出交易之间的关系和依赖。

将区块链交易数据构建为图，能够更好地利用图神经网络来分析和处理这些数据。图神经网络可以利用节点和边的属性、拓扑结构以及上下文信息来进行图级别的推理、节点分类、边预测等任务。这种图结构的表示能够更准确地捕捉交易之间的复杂关系，并提供更全面的信息来支持区块链数据的分析和应用。

在交易图构建模块的输入中，存在两种类型，输入一个交易哈希生成子图，或输入一个账户地址来生成子图。在用户输入交易哈希或地址哈希后，会首先获取当前交易或当前账户的基本信息，交易的信息包括交易哈希、交易所在区块位置、交易双方的地址等等，账户地址的信息包括账户余额等等。

基于输入的当前交易/当前账户扩展交易图网络的过程中，遵循以下基本思路：当前交易存在交易双方的账户地址，交易双方账户也可能存在和其它账户节点的交易，因此，可以在区块链中搜寻当前交易双方账户参与的交易，从中抽取交易的另一端，从而扩展出交易图网络的第一轮节点，以此类推，可以扩展多轮。当前账户显然也可以从区块链中获取其参与过的交易历史，并从交易的字段值从获得交易的另一方账户地址，从而扩展出交易图网络的第一轮节点，而后与输入交易哈希的情况类似，可继续进行扩展，得到多轮节点。相关采样方式如图 2.4 所示。

在以太坊网络中构建交易图时，采样是至关重要的一环，旨在有效处理大规模的交易数据，以揭示网络中的交易模式和行为趋势。因此，本研究采用区块采样的方法来获取最近的以太坊交易子集。本系统选择采用区块采样的原因如下：首先，基于时间临近采样后的区块中搜索相关交易可显著降低内存资源的消耗和时间复杂度。其次，基于时间临近采样后的区块中的交易与给定的交易联系最为密切，最大程度地保证了生成子图的表达能力。

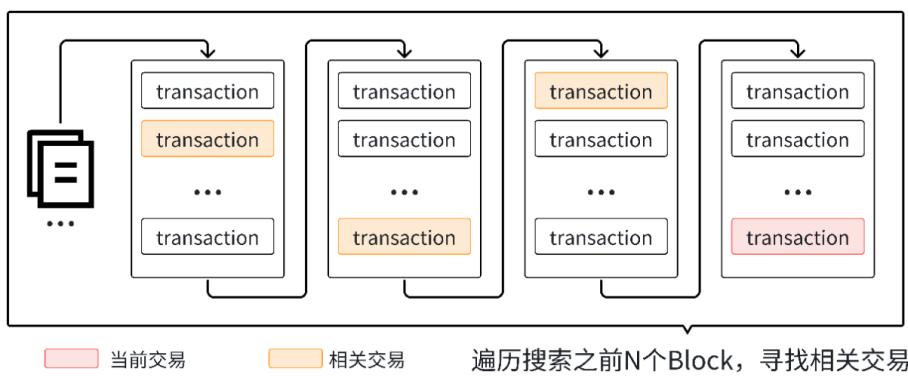


图 2.4: 时间邻近区块采样

因此，后续生成子图的工作将按照给定的区块跨度（默认为 30000 个区块），以获取与给定交易时间最接近的交易子集，并在该子集上进行进一步的图采样和分析，以揭示以太坊网络中的交易行为特征。

对于给定的交易，构建一个基于扩展轮数的账户队列，假设扩展的轮数为 hop，队列中的第 i 个元素是第 i 轮扩展出的账户，其中  $i=0$  时的队列元素是给定交易的 From address 以及 To address， $i=1$  时队列的元素是 From address 和 To address 在采样的区块中直接相关的账户，以此类推，第  $i=k+1$  时的队列中的元素即为  $i=k$  时队列中的账户直接相关的账户。多轮扩展操作如图 2.5 所示。

对于采样图上的节点，上述基于扩展轮数的账户队列中的每一个账户，我们首先将其分类成 EOA 账户和合约账户，接着分别对其构建用户画像。而后统一以账户节点结构存储。对于外部拥有账户（EOA）节点，只保留其 balance 和 nonce 字段，并添加 Goplus 字段作为属性。Goplus [23] 提供实时的安全分析和检测数据，这些信息能够帮助识别 EOA 账户的安全性。在处理合约账户节点时，需要获取该账户部署的合约代码，这通常涉及到从区块链上提取和解析合约的字节码。除了合约代码，我们还会将 Goplus 扫描结果作为属性加入，帮助识别合约的风险和特征。无论是 EOA 账户还是合约账户，都会统一以节点的形式存储在交易图中，并使用 is\_contract 字段来区分这两种类型的账户。节点属性如表 2.3 所示。

对于采样图上的边，在获取直接相关账户的多轮扩展过程中，无疑涉及到了许多交易，在扩展过程中，这些

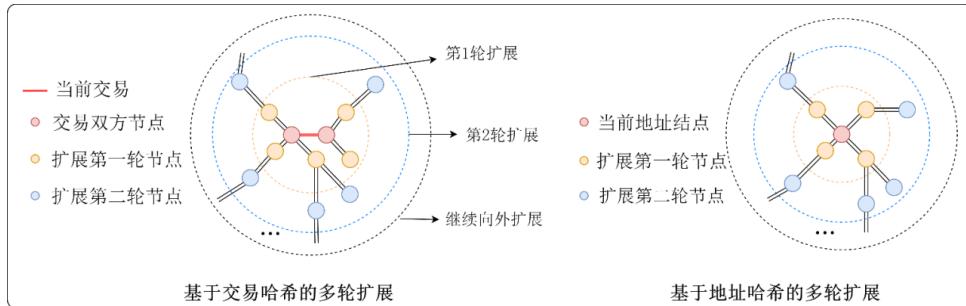


图 2.5: 基于交易哈希/地址哈希的多轮扩展

表 2.3: 节点属性

字段	字段含义	阐述
Is_contract	是否为合约账户	区分节点类型：EOA 账户节点或合约账户节点。
Goplus	Goplus 扫描结果	初步评估，提供安全性检测信息。
Balance	账户余额	仅 EOA 账户节点包含该属性，合约节点置为 none。
Nonce	账户随机数	仅 EOA 账户节点包含该属性，合约节点置为 none。
Code	合约代码	仅合约账户节点包含该属性，EOA 账户节点置为 none。

交易都被完整的保存下来。对于保存下来的交易，将去除没有具体含义的哈希字符串和 topics 相关的信息。剩下的属性将被用作定义节点间关系的边，增加了图的信息丰富程度，使得采样的子图具有更好的表达能力。针对目标地址 (target address)，需要一个相关的目标交易 (target trx)。因此，会选择区块链上最新的有关交易作为目标交易。接着，将该交易处理为边 (edge) 结构，其中边的属性来源于交易的元数据信息和交易的合约调用记录。这样，交易的详细信息包括调用流程、参数和顺序都会被记录和分析。这种方法通过将交易转化为边结构，确保了交易图中的节点和边都能准确地反映交易的实际情况和逻辑关系。

此外，针对目标交易 (target trx)，系统会附加一个 calltrace[48] 属性。该属性在交易执行期间捕获发生的函数调用依赖关系。当智能合约函数调用其他函数时，calltrace 会记录这些合约间和合约内的调用。这样，calltrace 能够详细解释合约的调用流程，概述交易中发生的事件，包括调用的智能合约、相应的参数、调用顺序和存储信息。这为理解合约功能和交易行为提供了新的视角。获取 calltrace 信息的过程涉及在交易执行期间监听和记录所有函数调用。即当一个智能合约函数调用另一个函数时，系统会捕捉并记录该调用的详细信息。calltrace 将作为交易图中边 (edge) 的属性，直接以字符串形式保存。Call Trace 的示例如表 2.4 所示，边属性如表 2.5 所示。

表 2.4: Call Trace 示例

CALL, from:0x99d..., to:0xe59..., data:c4f...
– DELEGATECALL, from:0xe59..., to:0xe..., data:f39...
– READ, 0x95c..., 0x67a
– LOG1, 0xb8..., 0x699
...

### 2.3.2 交易图分析模块

实现交易图分析模块时，本团队构建了一种结合图神经网络 [16] 和 Transformer[1] 的图大模型 (graph transformer) 架构，用于分析和处理交易图。这种架构特别适用于处理具有复杂关系和属性的图数据，例如区块链交易图。交易图分析模块的整体架构如图 2.6 所示。

根据交易图分析模块的流程，可将其细分为以下子模块：

- 特征嵌入子模块：在交易图构建完成后，图中的每个节点可能代表一个账户或智能合约，节点属性（如“is\_contract”，“Balance”等）描述了节点的特征。边代表交易，边的属性（如“value”，“type”等）描述了交易的细节。对交易图中的节点信息和边信息作特征嵌入，将节点和边的原始属性通过嵌入技术转换为连续

表 2.5: 边属性

属性字段	字段含义	阐述
Value	转账金额	转账的 ETH 数量。
Gas	燃油费单位	量化执行交易或合约所需的计算工作量的单位。
Gasprice	燃油价格	交易发起者愿意为处理交易而支付的每个 gas 的价格。
MaxFeePerGas	最大 GAS 值	愿意为交易支付的最大 gas 值。
MaxPriorityFeePerGas	优先级最大 GAS 值	愿意为交易费用的优先级部分支付的最大 gas 值。
TransactionIndex	交易索引	指向在区块中每个交易的序号。
Type	交易类型	标识交易类别：常规交易、合约部署交易或执行合约。
ChainID	以太坊链 ID	表示以太坊不同的链。
Calltrace	合约调用流程	展示交易所涉及的合约节点及代码，以字符串存储。

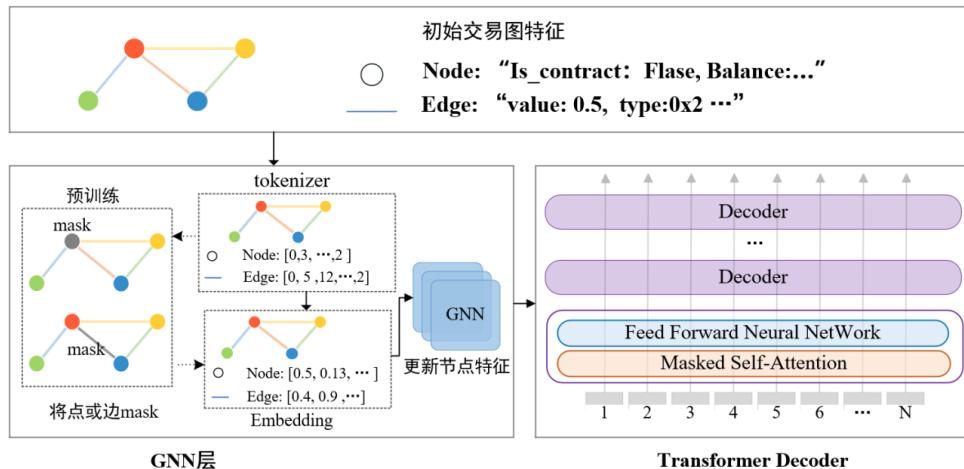


图 2.6: 交易图分析模块整体架构

的向量表示。这为后续的图神经网络处理提供了适合的输入格式。

- GNN 子模块：使用图神经网络（GNN）对图数据进行处理。GNN 通过考虑节点的邻域信息，能够捕捉图中的局部结构特征。在这个过程中，每个节点的嵌入会根据其邻居的信息进行更新和聚合。
- Transformer 子模块：引入 Transformer 架构进一步处理从 GNN 输出的节点特征。Transformer 通过自注意力机制能够捕捉节点间的复杂依赖关系，允许模型在全图范围内动态地重新加权节点特征。Transformer 层的输出可以进一步被用于不同的下游任务，通过训练输出层，可使得模型获得恶意交易检测、地址风险评估的能力。
- 预训练子模块：预训练子模块仅仅在构建模型时使用，通过采用针对图结构的 MLM（掩码语言模型）[14] 预训练策略，对节点特征以及边特征进行遮蔽，增强模型的基本性能以及泛化能力。预训练完成后，在系统实际运行时，分词器的输出结果不再需要经过预训练子模块，而是直接进入特征嵌入层。

综上所述，交易图分析模块能够在保持图结构信息的基础上，通过学习节点间的细微相似性来增强特征表达的丰富性和预测的准确性。Graph Transformer 架构的应用，使得模型在处理大规模和复杂的交易图数据时更为高效和精确。下面，将分别详细介绍交易图分析模块子模块的实现方案。

### (1) 特征嵌入子模块：基于分词器的节点特征与边特征表示

在交易图的分析模块中，节点和边的信息将视为文本信息，通过分词器切分为词并进行特征表示。这一特征表示方法启发于 NLP（自然语言处理）领域，在 NLP 任务中，原始文本必须转换为数值型数据才能被计算机处理。常见的方法之一是 one-hot 编码，但其存在向量维度过高和丢失语义信息的问题。词向量（word embedding）[34][53] 解决了这些问题，通过将词转换为向量，保留了语义信息。

为了获得词向量，首先需要对文本进行分词，这就需要使用分词器。现在大多数分词器不仅将句子分成单词，还将单词转换为唯一编码，以便在词向量矩阵中查找对应的词向量。交易图分析模块中，使用 Byte Pair Encoding (BPE)[58] 进行特征信息的分词。BPE 的主要思想是通过迭代地将频繁出现的字符或字符序列合并为一个单元，

从而优化词汇表，使常见词汇由较少的子词构成，而罕见词汇则由更多的子词组成。这个过程不仅减少了词汇表的大小，还提高了模型处理未登录词（即训练集中未出现的词汇）的能力。基于子词单元的 BPE 算法能够有效处理训练集中未出现的词汇，大大提高了模型的泛化能力，使其能够应对更多未知或罕见的词汇。

在使用 BPE 算法的分词器时，首先从训练数据中获取单词集合及其频数。然后，BPE 利用这些符号创建基本词表，并通过学习合并规则迭代更新词表，直到达到预定的规模。这种方法确保了交易图节点和边特征的高效性和准确性，为交易图的特征嵌入提供了强大的支持。在节点属性中，code 属性包含了合约账户的代码，其中 function、\_fallback 是最常见的单词，分别表示函数、返回值。图 2.7 中，展示了 BPE 分词器对 function、\_fallback 的分词示例。

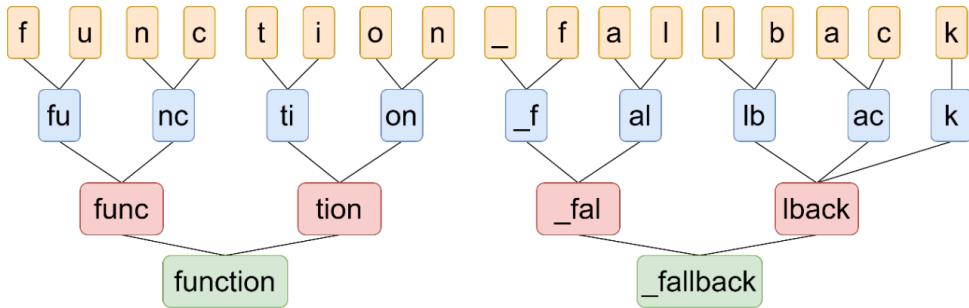


图 2.7: BPE 分词示例图

BPE 算法的关键原理可分为三个步骤：

定义压缩效用函数：BPE 算法的目标是最大化压缩效用。给定一个字符串  $x$ ，定义有效合并序列  $\mu$  应用于  $x$  的压缩效用为公式 (1.1) 所示，其中， $\text{APPLY}_\mu(x)$  表示将合并序列  $\mu$  应用于字符串  $x$  之后的结果。

$$\kappa_x(\mu) = |x| - |\text{APPLY}_\mu(x)| \quad (1.1)$$

压缩增益：定义两个合并序列相对于彼此的压缩增益。给定合并序列  $\mu$  和  $\mu'$ ，相对于  $\mu$  的  $\mu'$  的压缩增益定义为：

$$\kappa_x(\mu\mu') - \kappa_x(\mu) \quad (1.2)$$

组合优化：该步骤的目标是在给定字符串  $x$  和指定的合并次数  $M$  的情况下，找到一个有效的合并序列  $\mu$ ，使得压缩效用最大化。

$$\mu^* = \arg \max_{\mu \in M_Y, |\mu|=M} \kappa_x(\mu) \quad (1.3)$$

在特征嵌入子模块中，BPE 算法基于下面的流程处理输入的节点与边信息：

输入节点与边信息：将节点与边中的信息视为文本，以序列形式输入，设一个序列为  $x$ ，并设定 BPE 合并次数  $M$ 。首先初始化合并序列  $\mu : \mu \leftarrow \langle l \rangle$ ，而后对于  $i \in \{0, \dots, M\}$ ，执行后续步骤：对于最频繁的非重叠对，更新合并序列：

$$\mu \leftarrow \arg \max_{(\mu', \mu'') \in \text{set}(x)^2} \text{PAIRFREQ}(x, (\mu', \mu'')) \quad (1.4)$$

之后，应用选择的合并对： $x = \text{APPLY}(\mu, x)$ ，并更新合并序列： $\mu \leftarrow \mu \circ \langle \mu \rangle$ 。最后即可得到输出结果，包括合并序列  $\mu$ ，以及输入序列  $x$ 。

## (2) GNN 子模块：图结构信息聚合

根据交易图分析的实际需求，本作品希望构建一个能够将图结构特征和邻域信息（包括边信息和邻近节点信息）的 GNN 网络。在项目中实现的图神经网络（GNN）架构 [66] 基于 Weihua Hu 等人的研究。该 GNN 基于 GIN 实现，能够将边特征聚合到节点特征中，特别适用于边属性显著影响图结构的场景。GNN 模块流程如图 2.8 所示。

一般而言，图表示学习的主要任务是将结点映射为矢量表示的时候尽可能多地保留图的拓扑信息。基于图特征的表示学习对结点的矢量表示既包含了图的拓扑信息（邻接矩阵表达的图结构），也包含了已有的特征矢量（包含结点特征的矢量）。一般而言，图表示学习的主要任务是将结点映射为矢量表示的时候尽可能多地保留图的

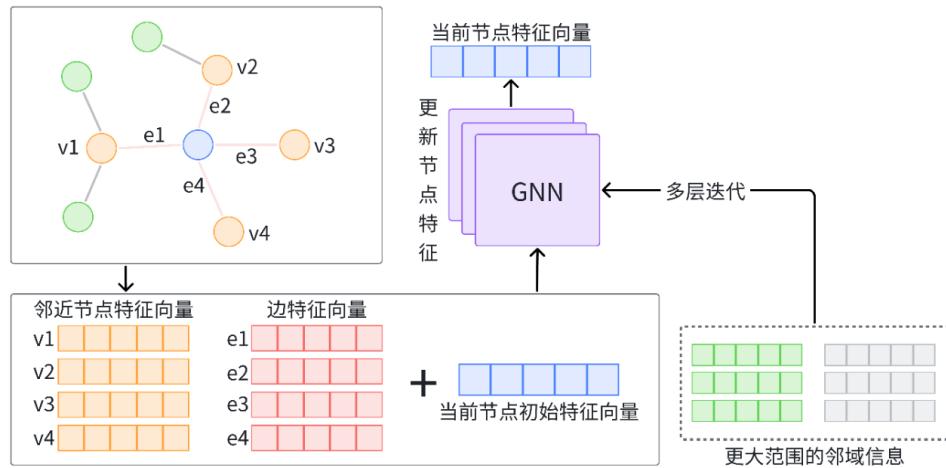


图 2.8: GNN 子模块总体流程

拓扑信息。基于图特征的表示学习对结点的矢量表示既包含了图的拓扑信息（邻接矩阵表达的图结构），也包含了已有的特征矢量（包含结点特征的矢量）。最传统的图结构特征和节点特征融合的方法，即是邻接矩阵和节点特征矩阵融合。

不同的图神经网络架构的差异在于其消息传递机制的差别。GNN 子模块采用的架构对于边（实体间的关系，例如交易）中包含的信息，采取的策略是聚合到节点特征中而不是直接忽略。交易图中的边实体（交易），包含了账号节点间重要的交互信息，对学习过程和模型整体性能可能有重大的影响。因此，该架构契合交易图分析模块的需求。该 GNN 架构基于下面的（2.1）公式来更新节点特征：

$$x'_i = h_\theta((1 + \epsilon) \cdot x_i + \sum_{j \in \mathcal{N}(i)} \text{ReLU}(x_j + e_{j,i})) \quad (2.1)$$

公式 (2.1) 中， $x_i$  是节点  $i$  的特征， $\mathcal{N}(i)$  表示节点  $i$  的邻居节点集合。 $\epsilon$  是一个可学习的参数，用于平衡节点自身特征和邻居特征的影响。 $e_{j,i}$  是连接节点  $j$  与  $i$  的边特征。 $h_\theta$  是一个参数化的神经网络，通常是一个或多个全连接层，用于变换节点特征。

$h_\theta$  层采用 MLP（多层感知机）实现，其表达公式为：

$$y = f(x) = \sigma(W_2(\text{ReLU}(W_1x + b_1)) + b_2) \quad (2.2)$$

公式 (2.2) 表明  $h_\theta$  的结构由两层线性层和一层 ReLU 层组成，其中  $x$  是输入向量， $W_1$  和  $W_2$  是权重矩阵， $b_1$  和  $b_2$  是偏置向量。 $\sigma$  是输出层的激活函数。

$h_\theta$  的 ReLU 层也是激活函数，定义为：

$$\text{ReLU}(z) = \max(0, z) \quad (2.3)$$

通过  $h_\theta$  层的结构，其第一层线性层（全连接层）负责将输入节点或边的特征线性变换到一个新的特征空间。这个线性变换可以捕捉输入特征的重要线性组合，但为了使网络能够学习到更复杂的模式，全连接层之后添加的 ReLU 激活函数引入非线性处理，使得网络不仅仅是简单的线性变换，而能够模拟更复杂的函数关系。在第一个全连接层和 ReLU 之后，第二个全连接层进一步将特征从 ReLU 激活的输出转换到最终的输出空间。这种多层次的结构使得 MLP 能够从数据中学习到深层的特征表示，这些表示对于图结构数据中的复杂关系尤为重要。

由此，该 GNN 子模块的输入是节点特征矩阵、边特征矩阵及其节点索引和边索引，而输出是聚合了邻域信息与图拓扑结构信息的节点特征矩阵。

该 GNN 架构允许模型学习更复杂的节点表示，能够捕捉节点间的复杂交互和图结构的深层特性。通过堆叠多个这样的节点更新网络层，可以增加模型的深度，让每个节点的特征表示能够逐渐包含更多的局部图结构信息，使最终的特征表示可以有效地反映节点在图中的结构和属性。在多层迭代之后，可以使用一个输出层来生成后续任务（节点分类、边分类等）的节点表示。综上，该 GNN 结构允许更细致和上下文感知的特征学习，这

对于后续下游任务中准确的预测恶意交易分类和评估风险地址此类基于图的机器学习任务至关重要。

### (3) Transformer 子模块：基于 Transformer 的特征学习

Transformer 子模块的架构基于原始 Transformer 的 Decoder 组件，通过堆叠若干个 Decoder 组件并利用其自注意力机制，能够进一步处理从 GNN 输出的节点特征，捕捉节点间的复杂依赖关系。Transformer 子模块的输出可被进一步的用于下游任务。

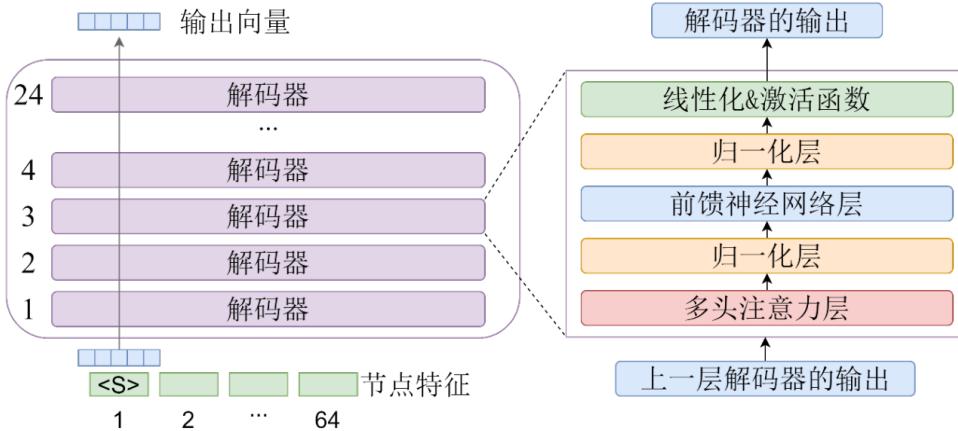


图 2.9: Transformer 子模块架构图

堆叠的 decoder 组件按如下方式处理输入序列 [3]: 最底下第一个 decoder 组件, 要处理的 token 先经过自注意力层的处理, 再经过神经网络层的处理, 处理之后把结果传输给下一个 decoder 组件。每个 decoder 的处理流程保持一致, 但是每个组件的自注意力层和神经网络层都有自己的权重。Decoder 组件示意图如图 2.3.7 所示。

然而, Transformer 最初被设计用于序列任务, 如机器翻译、文本生成等等, 而交易图网络并不是序列信息。伊利诺伊大学芝加哥分校的 Jinwoo Kim、Tien Dat Nguyen 等人证明了即使没有针对图结构进行修改 transformer, 其在图学习的实践中依然有良好的结果 [33]。因此, Transformer 子模块将交易图网络中的节点以构建时的序号为顺序, 将其对应的节点特征向量以类似序列的形式输入到 Transformer 架构中。下面根据 Decoder 组件的自注意力层和前馈神经网络层阐述 Transformer 子模块如何处理交易图数据。首先, 定义图为公式 (3.1) 所示:

$$G = (V, E, X) \quad (3.1)$$

其中, 节点  $u \in V$  的节点属性由  $x_u \in X \subset \mathbb{R}^d$  表示, 所有节点的节点属性存储在节点特征矩阵  $X \in \mathbb{R}^{n \times d}$ ,  $n$  表示该交易图有  $n$  个节点,  $E$  表示该图中的边构成的矩阵。

1. 多头自注意力层: 输入为节点特征矩阵  $X$ , 首先通过线性投影得到查询 ( $Q$ )、键 ( $K$ ) 和值 ( $V$ ) 矩阵, 再通过公式计算自注意力分数:

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V \quad (3.2)$$

$$\text{Attn}(X) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_{out}}}\right)V \in \mathbb{R}^{n \times d_{out}} \quad (3.3)$$

其中,  $d_{out}$  值是  $Q$  的维度, 是一个缩放因子, 防止点积运算过大导致梯度消失。 $W_Q$ 、 $W_K$ 、 $W_V$  是可训练的参数。在自注意力层中, 采用多头注意力机制, 旨在通过并行处理信息以捕捉序列中不同子空间的信息。首先, 对于每个头, 查询矩阵 ( $Q$ )、键矩阵 ( $K$ ) 和值矩阵 ( $V$ ) 都被线性投影到较低维度的子空间中:

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \quad (3.4)$$

其中,  $W_i^Q \in \mathbb{R}^{d_{model} \times d_q}$ ,  $W_i^K \in \mathbb{R}^{d_{model} \times d_k}$ ,  $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$  是对应与第  $i$  个头的参数矩阵,  $d_{model}$  是节点向量矩阵的原始维度,  $d_q$ 、 $d_k$ 、 $d_v$  分别是参数矩阵  $Q$ 、 $K$  和  $V$  的维度。与节点特征进行相应计算后就可

以得到各注意力头的查询矩阵 ( $Q$ )、键矩阵 ( $K$ ) 和值矩阵 ( $V$ ):

$$Q_i = XW_Q, K_i = XW_K, V_i = XW_V \quad (3.5)$$

而后，再使用公式 (3.3) 计算每个头的自注意力  $\text{Attn}(X)_i$ ，最后，所有的自注意力头的输出将被拼接起来，并通过线性投影转换回原始维度  $d_{\text{model}}$ :

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O \quad (3.6)$$

其中， $W^O \in \mathbb{R}^{d_{\text{model}} \times hd_v}$  是输出投影的参数矩阵。通过采用多头自注意力机制，模型能够在不同的表示子空间中捕捉到序列的不同特征，从而增强了模型的表达能力。每个头可能专注于序列中不同类型的信号，如某些头可能更关注交易图结构，而其他头则可能关注交易文本信息。这种并行处理方式不仅提高了效率，也使得模型能更全面地理解数据。

2. 前馈神经网络层 (FFN): 与自注意力层相连接，共同组成一个 Decoder 组件。其通过以下公式来给出一个 Transformer 层的输出:

$$\begin{aligned} X' &= X + \text{Attn}(X), \\ X'' &= \text{FFN}(X') = \text{ReLU}(X'W_1)W_2. \end{aligned} \quad (3.7)$$

可以堆叠多个层以形成 Transformer 模型，最终提供图的节点级表示。由于自注意力与输入节点的排列是等变的，因此如果仅仅只有 Transformer 架构，将始终为具有相同属性的节点生成相同的表示，即无论它们在图中的位置和周围结构如何，将输出相同的特征向量。因此，在交易图分析模块设计中，基于 Graph Transformer 技术，通过 GNN 来将节点的位置和周围结构信息合并到 Transformer 中。

综上，通过将 GNN 和 Transformer 连接起来，交易图分析模块不仅能够利用 GNN 的能力来捕捉局部的拓扑结构，从而有效地识别节点间的直接连接和邻域关系，也能利用 Transformer 来使模块在全局层面上分析和理解节点间深层次关联和相互作用。

然而，GNN 仅能聚合图结构信息和邻域信息至节点特征中，无法更新边特征，但在后续基于 Transformer 的特征学习过程中，希望边特征也能够包含图结构信息，以便于模型针对边特征来对交易进行深入的分析和学习，使模型在恶意交易分类任务中具有更好的表现。为此，本团队设计适应性的方法来处理边特征问题，将边两端的节点特征相加作为边特征。此方法的依据是，在使用 GNN 来聚合邻域信息和图结构信息至节点内部时，边的信息也被嵌入到两端节点中。因此，将两端节点特征相加作为边特征，既能表示边的自身信息，又包含了图结构信息。如图 2.10 所示。

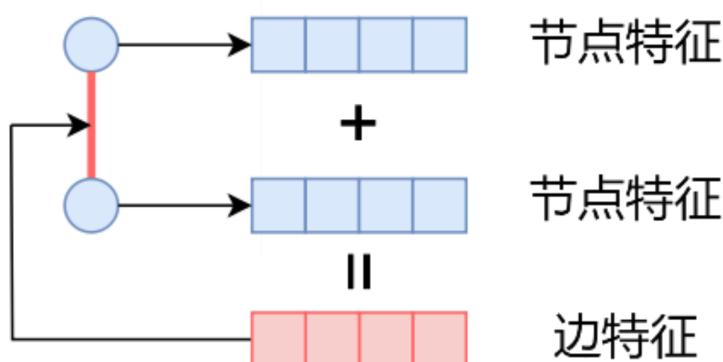


图 2.10: 边特征更新

#### (4) 基于 MLM (掩码语言模型) 的预训练任务: 增强模型泛用性

为增强交易图分析模块对区块链交易数据的理解和处理能力，使其具备在多个任务场景下的泛用性，本作品采用预训练任务中的 MLM 策略对交易图分析模块所使用的模型进行的预训练，得到模型称为 EtherSentinel。

预训练模型能够在大规模未标记数据上学习通用的表示，之后可以针对特定任务进行微调。

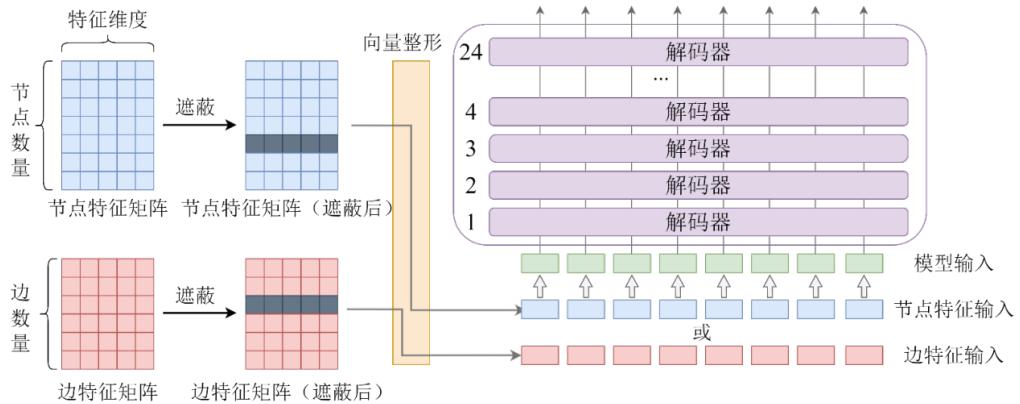


图 2.11: 基于 MLM 任务的预训练整体流程

对于交易图数据，直接应用传统的 MLM 方法（随机遮蔽多个节点或边）可能可能破坏图的结构信息，使得学习过程中难以捕捉到节点间的依赖关系，以致于训练效果不够理想。因此，本作品针对于交易图的需要，设计了一种特定于图结构的 MLM 方法，其特点是单一遮蔽，即在每次训练迭代中，随机选择并遮蔽图中的一个节点或一条边，优点是可以保持图的大部分结构不变，同时允许模型专注于推断遮蔽节点或边的属性。防止在多重遮蔽的情况下，模型过度依赖于少数几个高度连通的节点或边，而忽略其他部分的学习。预训练的整体流程如图 2.11 所示。

在设定的 MLM 任务中，采用了动态掩码策略 [35]，其中以 50% 的概率进行节点遮蔽，50% 的概率进行边遮蔽。具体来说，动态掩码策略不仅灵活地在节点和边之间切换，还能根据当前批次数据的特点实时调整掩码位置和比例，显著提升了模型的适应能力和学习效率。相比于 BERT 的静态掩码方法，动态掩码策略可以覆盖更广泛的特征空间，避免模型过度依赖特定的掩码模式。

节点遮蔽是指在节点数目 N 和 embedding 长度的向量矩阵中，根据动态掩码策略生成相应的 mask 矩阵，选择并遮蔽特定的节点特征。同样地，边遮蔽是在边数目 M 和 embedding 长度的向量矩阵中，通过生成动态 mask 矩阵，对特定的边特征进行遮蔽。这种动态掩码策略能够在不同的训练迭代中灵活调整掩码位置，确保模型学习到更多样化和全面的特征信息。节点与边的 mask 处理如图 2.12 所示。

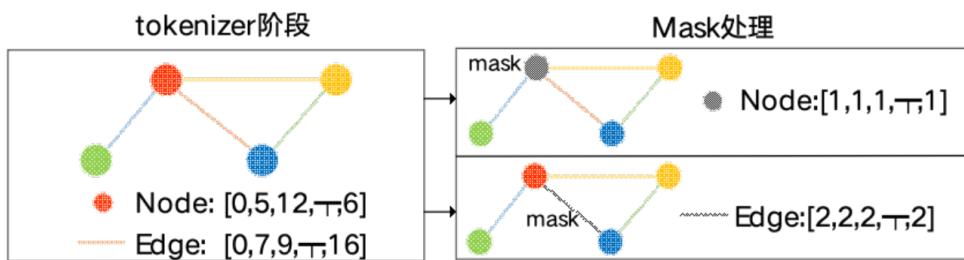


图 2.12: 节点与边的 mask 处理

因 Transformer 起初是处理序列结构的，因此通常需要以序列形式输入节点特征或边特征。节点排序和边排序直接使用其标识符序号，该标识符序号仅仅表示了交易图构建过程中节点扩展、边创建的先后顺序，没有特殊含义，因此本系统中的 Transformer 架构对序列顺序不敏感，以节点遮蔽为例，仅仅遮蔽了一个节点的特征，然后将所有节点以标识符顺序 [49] 输入到 Transformer 中。该 MLM 训练过程关键目的在于通过自注意力机制，从其它未被遮蔽的节点特征出发，去预测被遮蔽的节点特征，并以该被遮蔽的节点特征的原始向量作为标签对 Transformer 架构内部的可训练参数进行迭代更新。图结构中节点是无序的，而图结构信息以及嵌入在节点特征中，因此只需要将节点特征向量输入进 Transformer 架构即可，不受输入的序列顺序影响。

### 2.3.3 综合判别模块

综合判别模块针对恶意交易识别、账户风险检测以及账户函数名恢复这三个功能进行了深度优化，以期达到高效准确的安全防护目的。综合判别流程如图 2.13 所示。

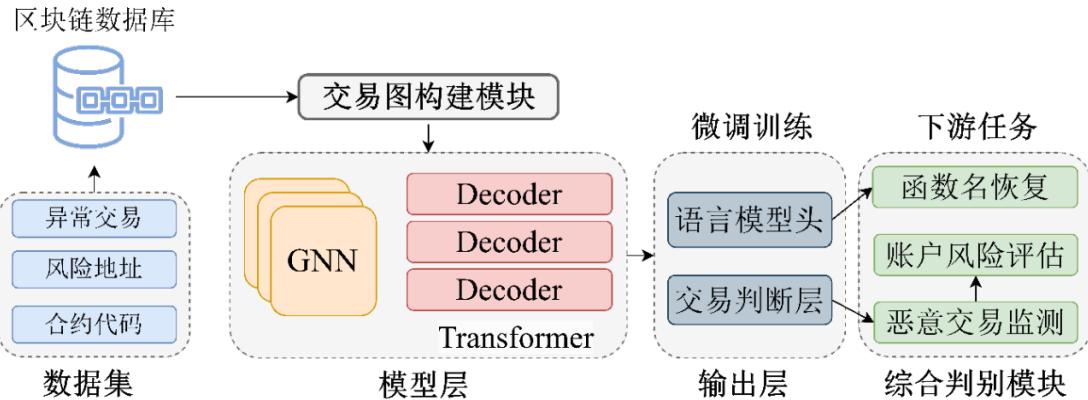


图 2.13: 综合判别模块实现流程

#### (1) 恶意交易识别:

此功能模块聚焦于快速准确地从海量交易数据中甄别异常行为。本团队利用了图分析技术，通过对交易网络的深入挖掘，捕捉交易间的复杂关联模式。具体实施时，随后，图分析模块会对这些分词信息进行编码，形成高维向量表示，其中最后一个分词的向量输出被专门用于后续的分类决策，这是因为最后一个分词聚合了前面所有分词的信息。最终本团队使用线性层来进行风险分类，有效提升了识别的精准度和效率。模型在交易识别任务上达成了 92% 的正确率。

#### (2) 账户风险检测:

为了更全面地评估账户的安全状况，本作品采取了一种综合历史行为分析的方法。这个想法来自本团队对账户风险挖掘的朴素直觉，即账户的风险应该体现在过往的每一笔交易当中如果一个账户曾经参与过恶意活动，那么该账户很有可能会继续参与因此过往交易风险的综合就是账户风险。因此不同于单一交易的即时判断，该模块会收集并分析账户下所有交易的历史分类结果。每一条过往交易的潜在风险都会被量化为一个概率值，反映其涉及恶意活动的可能性。当过往的交易太多时，采用随机抽样的方法，只抽取出最多二十条。在此基础上，系统进一步执行风险综合，即从所有交易的风险概率中选取最大值，作为该账户的整体风险评分。该策略能够有效突出偶发但高度危险的交易行为，避免因个别低风险交易而低估账户的整体威胁水平，确保对高风险账户的及时识别与干预。模型在账户风险识别任务上达成了 85% 的正确率。

#### (3) 合约代码函数名恢复:

在构建交易图的过程中，发现合约账户节点的代码中常常出现某些代码功能的函数名缺失，如表 2. 所示。可以看到第 6、7 行中，代码函数名的位置被 0x5b1e3b51 此类十六进制字符串代替。分析其原因，可能是因为合约账户隐藏了函数名，也可能是网络波动导致函数名缺失。函数名缺失为进一步的分析合约账户节点的安全风险带来了阻碍，因此，本系统使用预训练策略来实现函数名恢复功能，从而克服该难点。

因此，本作品设计了一个创新性的账户函数名恢复机制：在交易图中将所有 function 0x5b1e3b51 或者 function withdraw 的函数名字符串洗去，作为标签来进行训练，以生成的 function 字符串是否符合被洗去的真实功能作为评判标准，最终使模型获得代码生成的能力。由于函数名本身就包含在模型的输入中，因此该机制充分利用了生成式预训练任务，仅仅需要少量微调即可实现功能。系统中的图模块分析输出只有向量，为了从这些复杂的向量表示中解码出具体的函数名称，系统引入了一个专门的语言模型头。将图分析的抽象输出映射到一个庞大的函数名称词汇表上，为每一个潜在的函数名分配一个概率值。通过最大化这些概率值，系统能自动推断出最有可能被执行的函数名，这对于深入理解账户行为模式、追踪异常操作路径具有重大意义。这种方法不仅提高了对账户活动的解析精度，也为安全审计和威胁追踪提供了强有力的工具支持。模型在代码函数名恢复任务中达到 0.82 的双语评估分数。

表 2.6: 合约账户函数名缺失示例

```
{
  'is_contract': True, 'goplus': 'stealing attack',
  'code': {
    '_dispatcher': 'function _dispatcher;',
    '_fallback': 'function _fallback payable view pure;',
    'withdraw()': 'function withdraw();',
    '0x5b1e3b51': 'function 0x5b1e3b51 payable view;',
    '0x5fe2e4d1': 'function 0x5fe2e4d1 payable view;',
    'withdraw(address,uint256)': 'function withdraw() view;'
  }
}
```

#### (4) 构建预训练数据集 TrxNets:

在项目中，为了对构建的 Graph Transformer 模块进行有效的模型预训练，团队采取了系统化的方法来收集和准备预训练数据，搭建了预训练数据集 TrxNets。团队选取了具有高活跃度和丰富交易类型的以太坊 [71] 区块链交易平台，获取了 213857 条以太坊区块链交易数据。具体数据如表 2.3.5 所示。通过采用该预训练数据集进行训练，模型最终大小超过 20GB。

为确保数据的多样性和覆盖度，上述 20 万余条交易数据，是从以太坊的所有区块中随机抽取的 7 万个区块中，随机拓展得到的。这种随机抽样方法可以增加训练数据的广度，帮助模型学习到更多类型的交易行为和模式。

相比于其它交易数据集，TrxNets 具有以下优势：

- 更丰富的交易信息：相比于现有的交易数据集，TrxNets 收集的交易数据包含了更多交易元数据字段，例如 Gas 值。并且，还收集了交易数据包含的文本信息，例如交易所调用的智能合约代码，以及智能合约的调用跟踪记录。更加丰富的交易信息能够使模型更深入的理解交易行为和潜在特征。
- 更全面的数据分布：TrxNets 从以太坊中随机抽取了总共 20 万余条的交易数据。因此交易数据规模更大，时间跨度大，涵盖的交易行为更广泛。因此模型在此数据集上预训练能够学习到更多样的交易行为与交易分布，提高模型性能。

表 2.7: TrxNets 数据集

数据来源	数据规模	数据特征
以太坊	213,857 条交易数据	丰富的交易特征，时间跨度大

## 2.4 用户端设计

### 2.4.1 网页架构

前端可视化界面使用 JavaScript 语言的 Vue 框架开发；后台服务器使用基于 Python 语言的 Django 框架开发，MySQL 作为数据库存储数据；

根据用户友好的原则以及系统实现的功能，设计了首页及功能选择页面、实时交易数据展示页面、恶意交易检测页面、地址风险评估页面以及合约代码函数名恢复页面。用户端网页的实际效果将在第三章作品测试与分析的 3.6 节用户端测试中详细阐述。系统首页如图 2.14 所示，其它页面详情见 3.6 节。

用户界面提供输入框，允许用户输入交易哈希或地址哈希。输入框旁边提供提交按钮，用户点击后提交输入信息。交易图展示模块：根据处理后的历史交易数据构建交易子图，并在浏览器中展示。交易图构建模块处理历史交易数据，生成节点和边的结构，而后使用图形渲染库将交易子图渲染在页面中。允许用户在图中进行



图 2.14: 用户端首页

交互，如点击节点查看详细信息等。浏览器接收综合判别模块返回的结果，而后将结果展示在页面中，用户可以查看恶意交易类别、地址风险评估、合约函数名等信息。

为优化用户体验，用户端设计针对错误处理、加载动画、交互反馈作了深入的调整和优化。错误处理指的是在用户输入无效哈希或服务器响应失败时，提供清晰的错误提示信息。在数据请求和处理期间，使用加载动画提示用户操作正在进行中。网页提供了许多交互反馈，例如点击节点查看详细信息，显示弹窗或侧边栏展示详细信息。

通过上述设计和实现，前端系统可以有效地完成用户输入处理、数据交互、交易图展示、数据分析和结果展示的全部功能。

#### 2.4.2 系统交互

系统交互的总体流程为：用户在输入框中输入交易哈希或地址哈希，点击提交按钮。浏览器将输入信息发送到区块链服务器，获取历史交易信息。区块链服务器返回历史交易信息，浏览器接收并传递给交易图构建模块。交易图构建模块处理数据并构建交易子图，在页面中展示。交易图构建完成后，数据传递给交易分析模块进行分析。交易分析模块分析数据，并将分析结果发送到综合判别模块。综合判别模块执行判别任务，并返回判别结果。浏览器接收判别结果，并在页面中展示给用户查看。具体系统交互流程如图 2.15 所示。

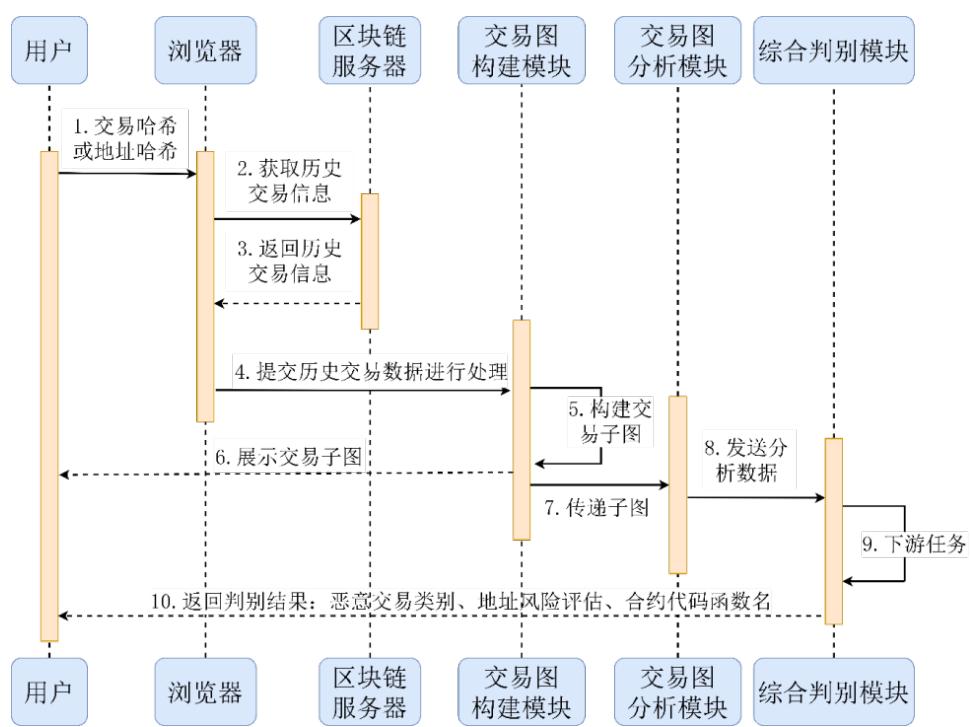


图 2.15: 系统交互图

# 第3章 作品测试与分析

## 内容提要

- 测试方案
- 测试数据
- 系统功能测试
- 系统性能测试
- 用户端测试

## 3.1 测试方案

作品测试与分析阶段主要对本系统进行功能测试、性能测试。系统功能测试包括验证系统的基本功能是否达到需求中要求实现的功能，以及测试系统的可靠性和可用性。系统性能测试部分设定了性能指标、方法基线，进行了有效性测试、敏感性实验以及消融实验，以证明本系统检测恶意交易、地址风险的能力以及良好的迁移性。

作品测试与分析部分的数据集与某 M lab 合作搭建。某 M lab 来自新加坡某高校，是 Web3 AI 安全工具和代码审计服务的领先提供商，致力于为 Web3 开发人员构建安全的基础设施。本团队与某 M lab 合作，搭建了一个高质量的交易数据集，这些数据经过严格验证，确保了其准确性和可靠性。

## 3.2 测试数据

### 3.2.1 预训练数据集

本团队从某 M lab 的数据平台中获取了从 2020 年到 2024 年约 213857 个来自以太坊的交易。这些交易数据不仅涵盖了各种类型的区块链交易，还包括了详细的交易日志。此 20 万条交易数据作为预训练数据集使用，将用于训练交易图分析模块中的图大模型架构。预训练数据集的时间分布如图3.1所示。

预训练数据集的所有交易均来自于以太坊，也即本系统仅在以太坊数据集上进行预训练。因本作品设计了适应性的掩码语言模型动态遮蔽方法来对模型进行预训练，由此系统在不同区块链平台上具有较强的兼容性。在后续测试部分将详细阐述。

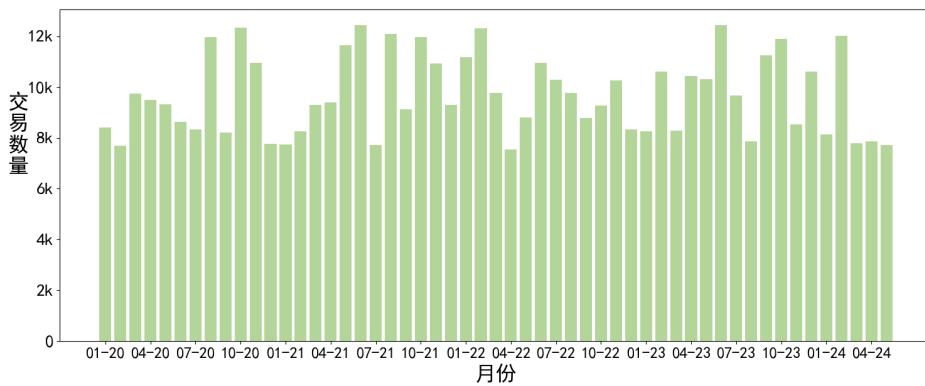


图 3.1：预训练数据集的时间分布

### 3.2.2 下游任务数据集

为测试本系统在恶意交易识别、账户风险检测以及账户函数名恢复这三个核心功能中的性能，本团队还构建了专用于下游任务的微调数据集。数据集将以训练集、验证集、测试集的形式随机划分，对本作品提出的模

型进行微调训练以及性能测试。

(1) 恶意交易数据集：该数据集由 83170 个来自于 5 个不同区块链交易平台的犯罪交易组成，其中包含了各类犯罪标签：恶意合约 [15]、欺诈 [22]、黑名单、蜜罐 [64]、钓鱼 [10]、黑客攻击 [59]。通过对这些异常交易进行交易图扩展以及交易图分析，模型可以更全面的学习和捕捉到交易之间的关系和模式。微调训练完成后，使用未纳入训练集的验证集和测试集根据异常交易的原始标签，评估系统分析结果的准确性和其它指标。恶意交易数据集的各项信息如图3.2-图3.6所示。

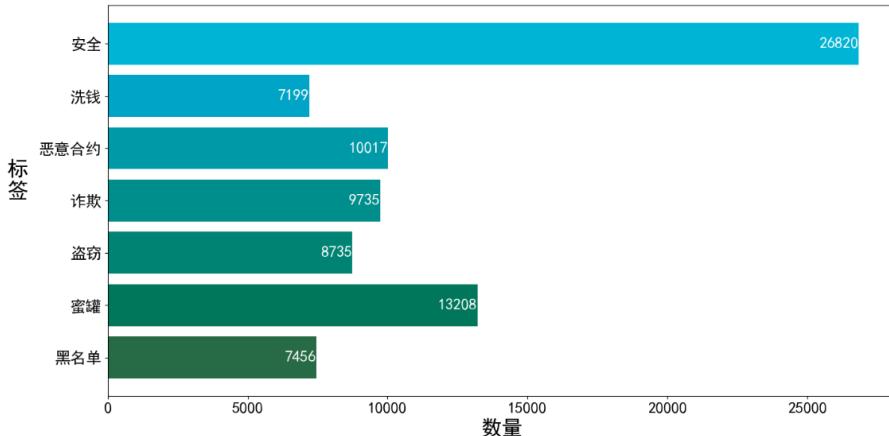


图 3.2: 恶意交易数据集标签分布

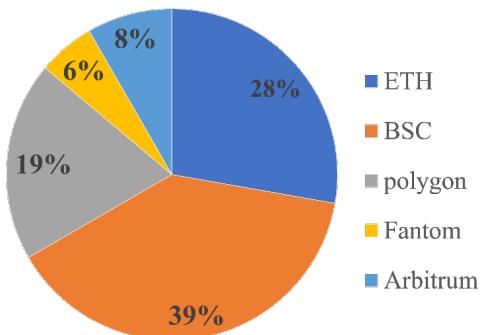


图 3.3: 恶意交易数据集平台分布

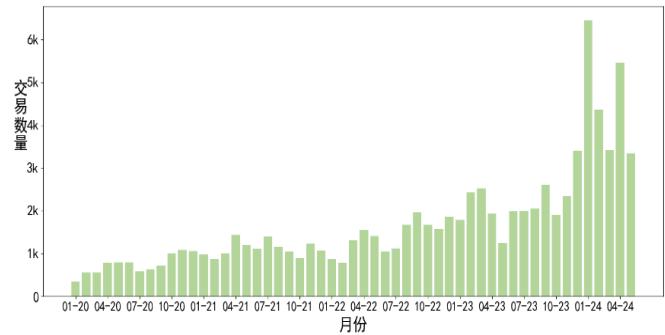


图 3.4: 恶意交易数据集时间分布

图3.2展示了数据集中的标签分布。数据集中包含六类标签，其中“安全”标签的样本数最多，为 26820 个；其次是“蜜罐”标签，样本数为 13208 个；“恶意合约”和“欺诈”标签样本数分别为 10017 个和 9735 个；“钓鱼”标签样本数为 8726 个；“洗钱”标签样本数为 7199 个；“黑名单”标签样本数为 7456 个。这样的标签分布有助于全面评估模型在不同类型交易检测中的性能和鲁棒性，确保模型在处理多样化的交易风险时具有高效的识别能力。

图3.3展示了恶意交易数据集的平台分布。主要来自五个区块链平台，其中 BSC[4] 占比最大，为 38.9%；其次是 ETH[71]，占比 27.8%；Polygon[61] 占比 19.4%；Fantom[60] 占比 8.3%；Arbitrum[2] 占比 5.6%。多样化的来源分布有助于评估模型在不同区块链平台上的性能和适应性，确保模型能够在各种环境下保持高效和稳定的表现。

从图3.6可以看出，测试集的数据覆盖了从 2020 年 1 月到 2024 年 4 月的多个时间段。2021 年和 2022 年的数据量相对较少，而 2023 年和 2024 年的数据量显著增加，尤其是 2024 年 1 月和 4 月的数据量达到峰值。这种时间分布特点表明，测试集的数据在不同时间段内具有较大的差异性，有助于全面评估模型在处理不同时间范围内交易数据时的性能和鲁棒性。数据量的逐年增加反映了近年来交易数据的增长趋势，为模型提供了更多的训练和测试样本，提升了模型的泛化能力和稳定性。

(2) 风险地址数据集：本团队从某 M lab 的数据平台中获取了来自五个不同平台的 5000 个账户地址及其评分数据。该评分数据是基于基于用户举报，人工验证和研究来区分异常交易和正常交易，从而对该地址进行风险评分。对于每一个地址，都有相应的风险评分，评分越高则该地址风险越高。

本团队基于这 5000 个账户地址，在各区块链交易平台的公开数据库中爬取地址的历史交易信息并构建相应的交易子图，为了避免有些账户历史交易太多而导致交易图构建时间过长，当账户历史交易超过 20 条时，我们从中随机抽样二十条进行交易图拓展，我们把每个交易在各个风险分类上的评分去最大值作为账户的方向标签，换言之，只要这 20 条历史交易中有一条涉及风险就决不会被忽视。

图3.5展示了地址数据集的平台分布。主要来自五个区块链平台，其中 ETH 占比最大，为 41.61%；其次是 BSC，占比 19.24%；Polygon 占比 13.16%，Fantom 占比 9.47%；Arbitrum 占比 16.51%。图3.6展示了地址数据集的标签分布，安全地址为 2302 条，风险地址为 2698 条，六类标签的风险地址详细数量见图中数据。

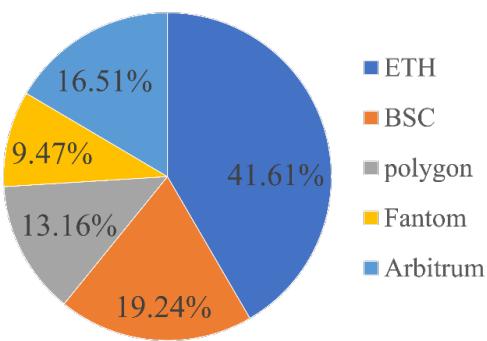


图 3.5: 风险地址数据集平台分布

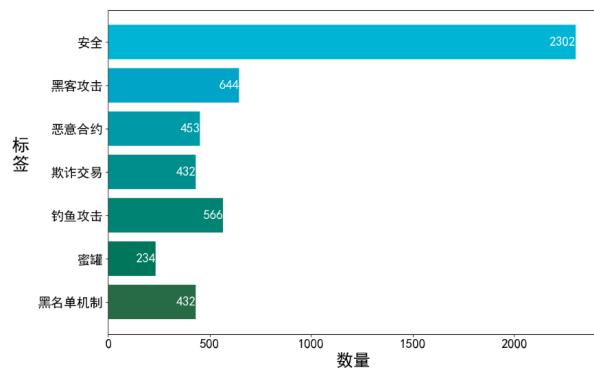


图 3.6: 恶意交易数据集时间分布

### (3) 合约代码函数名数据集：

我们从前述的交易数据集中滤除了所有包含无效函数名（即十六进制字符串而非自然语言文本）的交易，将剩余的约 5 万条交易数据用于构建合约代码函数名任务数据集。在测试时，效仿预训练的做法，在交易图中随机掩码一个合约节点的所有特征让模型做生成式预测，在测试时则将中心节点进行掩码让模型进行节点特征的预测。

## 3.3 测试环境搭建及测试设备

本系统可以部署于具备 GUI 能力的 Linux 系统或 Windows 系统中，推荐部署于 Linux 系统。硬件环境说明如表??所示，软件环境说明如表??所示。

表 3.1: 硬件环境说明

硬件	版本	说明
操作系统	Ubuntu18.04	具备 GUI 能力的 LINUX 系统或 Windows 系统
处理器	Xeon(R) Gold 6348	支持多线程计算和数据处理的多核处理器
显卡	NVIDIA A800	显存 6GB 及以上，支持模型的高效训练和推理
内存	三星 ECC 32GB ×8	应对大规模数据处理和模型训练所需的高内存需求
硬盘	PM883 SSD 2TB	快速读取和写入，支持区块链数据存储和处理

## 3.4 系统功能测试

本部分的测试主要是对系统的功能进行验证，以确保系统能够正常运作。测试包括基本功能测试、可靠性测试和可用性测试，每个测试部分都有不同的重点和目标。

表 3.2: 软件环境说明

依赖库	版本	依赖库	版本
evm_cfg_builder	0.3.1	goplus	0.2.2
Numpy	1.26.4	pandas	2.2.2
requests	2.31.0	tokenizers	0.15.2
torch	2.2.1	torch_geometric	2.5.2
Web3	5.13.4	tqdm	4.66.2
transformers	4.37.2	wandb	0.16.6

(1) 模块功能测试：针对本系统的设计目标所设计的基本功能进行测试，包括交易子图构建、恶意交易检测、地址风险评估、合约代码生成。关注系统是否按设计需求正确运转，完成用户提交的多种任务并及时响应。

(2) 可靠性测试：关注系统在各种情况下的反应和表现。测试包括屏蔽用户误操作、提供正确的错误提示信息以及对异常情况的处理。这些测试旨在验证系统能够稳定运行，并能正确地处理用户的输入和操作。

(3) 易用性测试：关注系统的用户体验和界面设计。测试中评估系统是否提供必要的操作指导信息，界面是否简洁、美观、实用，并且在不同浏览器下渲染页面是否一致。这些测试有助于确保系统易于使用，用户能够方便地进行操作和获取所需信息。通过这些测试部分的综合评估，可以全面了解系统的功能表现、稳定性和用户体验，为系统的进一步优化和改进提供重要的参考依据。

### 3.4.1 模块功能测试

针对本系统的设计目标所设计的基本功能进行测试，包括交易子图构建、恶意交易检测、地址风险评估、合约代码生成。关注系统是否按设计需求正确运转，完成用户提交的多种任务并及时响应。表3.3展示了模块功能测试的总览。表3.4至表3.7展示了模块功能测试所有测试项目的详细测试内容。

表 3.3: 模块功能测试总览

测试项目	测试内容	测试目的	预期结果
实时交易子图构建	输入交易哈希或地址哈希，获取历史交易并构建交易图	测试系统是否能够构建交易子图	成功构建交易子图
恶意交易检测	输入交易哈希，系统预测其类别并展示，同时呈现相关信息	测试系统是否能够预测交易类别	成功预测交易类别
地址风险评估	输入地址哈希，模型预测其危险程度并展示，同时呈现相关信息	测试系统是否能够预测地址风险	成功评估地址风险
合约代码函数名恢复	输入未知合约代码函数名，模型预测其函数名并展示	测试系统是否能够恢复合约代码函数名	成功恢复合约代码函数名

#### 1. 实时交易子图构建测试

测试实时构建交易子图功能，验证系统能否根据输入的交易哈希或地址哈希成功获取历史交易并构建交易子图。前提条件、测试步骤及实际结果如表3.4所示。系统运行结果如图3.7所示。

#### 2. 交易风险评估功能测试

测试恶意交易检测功能，验证系统能否根据输入的交易哈希，使用模型预测其类别，并输出正确的交易类别。前提条件、测试步骤及实际结果如表3.5所示。系统运行结果如图3.8所示。

表 3.4: 实时交易子图构建测试

用例编号	TC001
用例描述	实时构建交易子图测试
测试目的	验证系统能否根据输入的交易哈希或地址哈希成功获取历史交易并构建交易子图
前提条件	1. 系统已成功部署并运行。 2. 数据库中存在与输入哈希相关的历史交易记录。 3. 用户已登录并具有相应的访问权限
测试步骤	1. 通过浏览器访问本系统。 2. 导航到交易甄别模块。 3. 以输入交易哈希为例，测试是否能构建交易子图。 4. 点击“预测交易类别”按钮。
预期结果	系统成功获取与输入哈希相关的历史交易记录并构建交易子图，显示在用户界面上。
实际结果	成功构建交易子图。

表 3.5: 恶意交易检测测试

用例编号	TC002
用例描述	测试恶意交易检测功能。
测试目的	验证系统能否根据输入的交易哈希，使用模型预测其类别，并输出正确的交易类别
前提条件	1. 系统已成功部署并运行。 2. 数据库中存在与输入哈希相关的历史交易记录。 3. 用户已登录并具有相应的访问权限
测试步骤	1. 通过浏览器访问本系统。 2. 导航到恶意交易检测模块。 3. 输入有效的交易哈希。 4. 点击“预测交易类别”按钮。
预期结果	系统成功输出交易的类别，显示在用户界面上。
实际结果	成功输出交易类别。



图 3.8: 恶意交易样例测试检测结果图



图 3.7: 实时构建交易子图测试结果图（以输入交易哈希为例）

### 3. 地址风险评估功能测试

测试地址风险评估功能，验证系统能否根据输入的地址哈希，使用模型评估其风险，并输出正确的地址风险评估。前提条件、测试步骤及实际结果如表3.6所示。系统运行结果如图3.9所示。

表 3.6: 地址风险评估测试

用例编号	TC003
用例描述	测试地址风险评估功能。
测试目的	测试地址风险评估功能。
前提条件	1. 系统已成功部署并运行。 2. 数据库中存在与输入哈希相关的历史交易记录。 3. 用户已登录并具有相应的访问权限
测试步骤	1. 通过浏览器访问本系统。 2. 导航到地址风险评估模块。 3. 输入有效的地址哈希。 4. 点击“评估地址风险”按钮。
预期结果	系统成功输出地址的风险评估，显示在用户界面上。
实际结果	成功输出地址风险评估结果。



图 3.9: 地址风险评估样例测试结果图

#### 4. 合约代码函数名恢复功能测试

测试合约代码函数名恢复功能，验证系统能否根据输入的未知合约代码功能，使用模型预测并输出正确的函数名。前提条件、测试步骤及实际结果如表3.7所示。系统运行结果如图3.11所示。

表 3.7: 合约代码函数名恢复测试

用例编号	TC004
用例描述	测试合约代码函数名恢复功能
测试目的	验证系统能否根据输入的未知合约代码功能，使用模型预测并输出正确的函数名。
前提条件	1. 系统已成功部署并运行。 2. 用户已登录并具有相应的访问权限。 3. 合约代码生成模型已训练并部署。
测试步骤	1. 通过浏览器访问本系统。 2. 导航到合约代码生成模块。 3. 输入合约代码功能描述。 4. 点击“生成函数名”按钮。
预期结果	系统成功输出与输入功能描述相对应的代码函数名，显示在用户界面上。
实际结果	成功输出代码函数名。



图 3.10: 合约代码函数名恢复样例测试结果图

### 3.4.2 可靠性测试

可靠性测试关注系统在各种情况下的反应和表现。测试包括屏蔽用户误操作、提供正确的错误提示信息以及对异常情况的处理等等。这些测试旨在验证系统能够稳定运行，并能正确地处理用户的输入和操作。如表3.8所示。

表 3.8: 可靠性测试

序号	测试内容	测试结果
1	是否能屏蔽用户的误操作	系统给出相应的提示
2	是否对错误有正确提示	系统给出相应的提示信息
3	输入错误地址时，系统不崩溃、不异常退出	系统给出相应的提示信息
4	资金流地址最近交易记录不足 50 条时取全部交易信息	系统正常工作
5	地址最近无历史交易记录	系统正常工作，不报错
6	获取历史交易信息时出现交易双方中某一方地址为空	系统正常构建子图并提示
7	交易哈希涉及地址最近无历史交易记录	系统正常工作并提示



图 3.11: 合约代码函数名恢复样例测试结果图

### 3.4.3 易用性测试

易用性测试关注系统的用户体验和界面设计。测试中评估系统是否提供必要的操作指导信息，界面是否简洁、美观、实用，并且在不同浏览器下渲染页面是否一致。这些测试有助于确保系统易于使用，用户能够方便地进行操作和获取所需信息。通过这些测试部分的综合评估，可以全面了解系统的功能表现、稳定性和用户体验，为系统的进一步优化和改进提供重要的参考依据。如表3.9 所示。

表 3.9: 易用性测试

序号	测试内容	测试结果
1	具有必要的信息，指导用户使用对应功能	系统界面提示操作信息，方便指导用户操作
2	人机界面简洁、美观，风格相对一致	系统界面显示简洁易懂，方便使用
3	在不同浏览器下渲染网页	页面风格一致，未出现字符图片错位情况
4	页面风格一致，整体风格与系统功能匹配	系统与页面风格一致
5	用户是否能顺利注册并登录系统	注册、登录过程快速且无障碍
6	导航栏布局合理，能快速找到所需功能	导航栏布局依据功能设计，清晰明了
7	检查表单填写的提示信息和错误处理机制	填写过程中有明确提示，错误信息清晰明了
8	评估系统响应时间和操作反馈	系统响应迅速，操作后有及时反馈

## 3.5 系统性能测试

### 3.5.1 评估指标

为了更好的评价实验结果，本作品选择以下指标进行测试，分别是：精确率、召回率、F1 分数、AUC、平衡准确率以及双语评估分数。在计算中，TP 表示真正例数，TN 表示真负例数，FP 表示假正例数，FN 表示假负例数。

1. 精确率 (Precision): 模型正确预测为正样本的样本数与预测为正样本的比例，衡量的是模型预测为正样本的准确性。计算公式如下：

$$P = \frac{TP}{TP + FP} \quad (3.1)$$

2. 召回率 (Recall): 模型正确预测为正样本的样本数与实际正样本数之比，衡量的是模型对于实际正样本的识别能力。计算公式如下：

$$R = \frac{TP}{TP + FN} \quad (3.2)$$

3. F1 分数 (F1 Score): 精确率和召回率的调和平均数，衡量的是模型在保持精度的同时提高召回率的能力。计算公式如下：

$$F1 = 2 \times \frac{P \times R}{P + R} \quad (3.3)$$

4. ROC 曲线下的面积 (Area Under the Curve, AUC): ROC 曲线是以不同的分类阈值为横轴，以真正例率 (True Positive Rate) 和假正例率 (False Positive Rate) 为纵轴所绘制的曲线。AUC 综合考虑真正例率和假正例率的二分类模型评价指标，衡量的是模型能够将正负样本正确分类的能力。计算公式如下：

$$AUC = \int_0^1 TPR(FPR^{-1}(t)) dt \quad (3.4)$$

其中，TPR 和 FPR 分别表示真正例率和假正例率， $FPR(t)$  表示当分类器的阈值为  $t$  时，对应的假正例率； $TPR(t)$  表示当分类器的阈值为  $t$  时，对应的真正例率。通过对 ROC 曲线下的面积进行积分得到 AUC 的值。

5. 双语评估分数 (BLEUscore): BLEU (Bilingual Evaluation Understudy) [44]，即双语评估替补。所谓替补就是代替人类来评估机器翻译的每一个输出结果。给定一个机器生成的翻译，双语评估将自动计算一个分数，以衡量机器翻译的好坏。分数取值范围是  $[0, 1]$ ，越接近 1，表明翻译质量越好。在本作品的测试中，双语评估分数将专门用于测试合约代码函数名恢复功能。计算公式如下：

$$\text{BLEU} = BP \cdot \exp \left( \sum_{n=1}^N w_n \log p_n \right) \quad (3.5)$$

其中，BP 是简短惩罚因子，惩罚一句话的长度过短，防止训练结果倾向短句的现象，其计算方法为：如果 MT 输出长度大于指向参考文本的输出长度，则  $BP = 1$ ，否则

$$BP = \exp \left( 1 - \frac{\text{MToutputlength}}{\text{referenceoutputlength}} \right) \quad (3.6)$$

是基于 n-gram 的精确度，其表达公式为：

$$P_n = \frac{\sum_{n\text{-gram} \in y} \text{CounterClip}(n\text{-gram})}{\sum_{n\text{-gram} \in y} \text{Counter}(n\text{-gram})} \quad (3.7)$$

## 3.5.2 方法基线

方法基线包括 CodeBERT[19]、GraphCodeBERT[24]、BERT4ETH[54]、XBlockFlow[65]。

(1) **CodeBERT**: CodeBERT 是 BERT 模型的扩展，是一个专门为编程语言 (PL) 和自然语言 (NL) 设计的双峰预训练模型。它能够执行各种下游的 NL-PL 任务，例如代码搜索、代码生成和代码文档生成等。CodeBERT 通过在大规模的自然语言和编程语言数据上进行预训练，学习到跨语言的表示，从而在处理 NL-PL 任务时表现出色。

CodeBERT 使用了六种主要的编程语言进行训练：Python、Java、JavaScript、PHP、Ruby 和 Go。这些编程语言涵盖了多种编程范式和应用场景，使得 CodeBERT 在不同的编程环境中都能有效工作。通过在这些语言的数据上进行掩码语言模型 (MLM) 和替换词预测 (RTD) 的训练，CodeBERT 不仅能理解编程语言的语法和结构，还能捕捉自然语言和编程语言之间的语义关联。

(2) **GraphCodeBERT**: GraphCodeBERT 是一个基于 BERT 的预训练模型，专注于利用代码的语义结构来学习代码表征。除了传统的掩码语言模型 (MLM) 任务外，GraphCodeBERT 引入了两个新的预训练任务：数据流边预测和源代码与数据流的变量对齐。这些任务帮助模型更好地理解代码的语义和数据依赖关系。

数据流边预测通过预测变量之间的数据流关系，增强了模型对代码逻辑的理解能力。源代码与数据流的变量对齐任务使模型能够更准确地捕捉代码结构和变量交互。通过这些任务，GraphCodeBERT 在代码搜索、代码克隆检测、代码补全和代码总结等四个下游任务中表现优异，显著提升了模型效果。

(3) **BERT4ETH**: BERT4ETH 是专门为以太坊设计的大模型，旨在检测以太坊上的交易风险。该模型将节点的状态序列输入 Transformer 中，通过这一方式，可以同时预测账户是否涉嫌各项风险以及交易的潜在风险。BERT4ETH 利用 Transformer 的强大序列建模能力，能够捕捉节点状态随时间变化的动态特征，从而更准确地识别和预测风险行为。BERT4ETH 通过分析以太坊节点的状态序列，学习到不同状态之间的复杂关系和潜在模式。这使得模型在进行账户风险评估和交易风险预测时，能够提供高效且准确的判断。BERT4ETH 在实验中展示了其卓越的性能，不仅提高了风险检测的准确性，还能有效地预防潜在的安全威胁。

(4) **XBlockFlow**: XBlockFlow 是一个用于检测以太坊洗钱活动的时间检测框架。该框架通过分析交易图的拓扑结构和资金流动图，从地址出发，逐层追踪和分析交易，以识别潜在的洗钱地址。XBlockFlow 利用污点分析的方法，对每一层交易进行详细检查，评估交易链中的可疑行为，从而得出是否存在洗钱活动的结论。

XBlockFlow 首先构建交易图和资金流动图，捕捉地址之间的交易关系和资金转移路径。然后，通过污点分析技术，框架能够逐层识别并标记可疑交易，追踪资金流动的源头和去向。这样的多层次分析方法，使得 XBlockFlow 在识别复杂的洗钱模式时，能够提供更准确和细致的检测结果。

## 3.5.3 恶意交易检测测试

下游任务数据集中包含了正常交易 (26820 条) 和恶意交易样本 (56350 条)，共 83170 条。将数据集随机划分为训练集 (80%) 和测试集 (20%)。使用训练集分别训练各基线模型以及本作品的模型 EtherSentinel，而后

使用测试集评估各模型的性能，计算精确率、召回率、Macro F1 和 Micro F1。各评估指标计算公式如 3.5.1 节所示。测试结果汇总如表 3.10 所示，可以直观的进行各模型在恶意交易检测任务中的性能对比。

表 3.10：恶意交易分类测试结果

方法	精确率	召回率	Macro F1	Micro F1
CodeBERT	0.6923	0.7235	0.7019	0.7161
GraphCodeBERT	0.7618	0.8124	0.7822	0.7989
BERT4ETH	0.5032	0.4796	0.4983	0.4871
XBlockFlow	0.7367	0.7248	0.7343	0.7309
EtherSentinel(our)	<b>0.9244</b>	<b>0.9017</b>	<b>0.9123</b>	<b>0.9122</b>

EtherSentinel 在所有测试指标上均表现出色，尤其在精确率和 Macro F1 方面显著优于其他模型，表明其在恶意交易检测任务中具有很高的准确性和稳定性。

### 3.5.4 地址风险评估测试

下游任务数据集中的风险地址数据集包含正常地址 2302 条，风险地址 2698 条。将数据集随机划分为训练集（80%）和测试集（20%）。使用训练集分别训练各基线模型以及本作品的模型 EtherSentinel，而后使用测试集评估各模型的性能，计算精确率、召回率、Macro F1 和平衡准确率。各评估指标计算公式如 3.5.1 节所示。测试结果汇总如表 3.11 所示，可以直观的进行各模型在恶意交易检测任务中的性能对比。

表 3.11：地址风险评估测试结果

方法	准确率	召回率	micro-F1	平衡准确率
CodeBERT	0.6347	0.5219	0.5863	0.6532
GraphCodeBERT	0.7008	0.6427	0.6859	0.6921
BERT4ETH	0.3104	0.2206	0.2543	0.3129
XBlockFlow	0.6502	0.6038	0.6289	0.6327
EtherSentinel(our)	<b>0.8546</b>	<b>0.8317</b>	<b>0.8432</b>	<b>0.8619</b>

### 3.5.5 合约代码函数名恢复测试

在合约代码函数恢复测试中，我们从下游任务数据集的交易数据集中滤除了所有包含无效函数名（即十六进制字符串而非自然语言文本）的交易图用于进行函数名恢复任务。我们效仿预训练的做法，在交易图中随机掩码一个合约节点的所有特征让模型做生成式预测，在测试时则将中心节点进行掩码让模型进行节点特征的预测。

而后，节点特征经过综合判别模块的语言模型头解码为自然语言文本，并比对相应函数名位置原始标签与生成结果，通过 BLEU 指标来评估合约代码函数名恢复结果的有效性。方法基线中仅 CodeBERT、GraphcodeBERT 以及我们的模型 EtherSentinel 具备生成式能力，合约代码函数名恢复测试结果如图 3.12 所示。

### 3.5.6 平台迁移性测试

现有的工作往往是针对特定的区块链交易平台特征进行训练和优化，因此在迁移至其它平台时，由于平台间特征的不兼容，导致迁移性较差，模型性能明显衰减。因此，本作品通过利用大模型对文本的处理能力，将所有特征转化为文本输入，从而摆脱对特定特征的依赖，并且设计了适应性的掩码语言模型动态遮蔽方法来对模型进行预训练，以期系统在不同区块链交易平台上具有较强的迁移性。

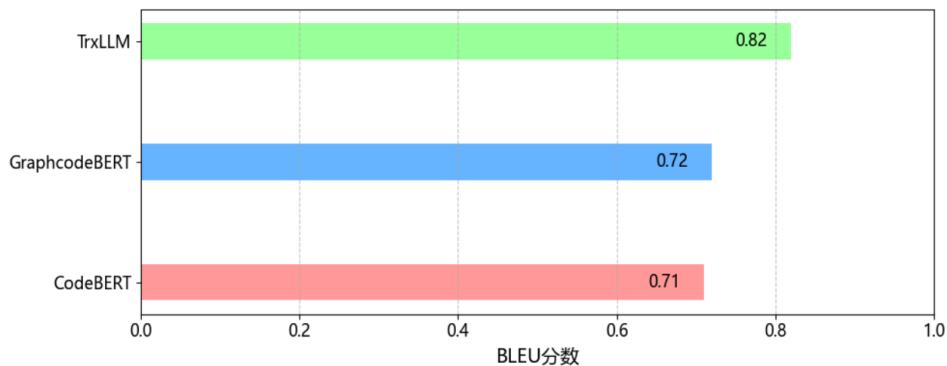


图 3.12: 合约代码函数名恢复测试

对于本作品提出的模型，我们使用 3.2 节测试数据中的预训练数据集（总交易量约为 20 万条）来对模型进行预训练。预训练数据集的所有交易均来自于以太坊。而后，我们使用 3.2 节测试数据中来自五个不同的平台的小规模数据集（总交易数量约为 7 万条）进行微调训练。微调训练后的模型在下游任务中的性能指标如表 3.12 至 3.14 所示。测试结果证明，Trx 仅需使用不同平台的小规模数据集进行训练，无需修改算法即可迅速迁移到其它平台，并且下游任务的性能不会显著衰减。

表 3.12: 不同平台的恶意交易检测准确率

方法	ETH	BSC	Fantom	Arbitrum	Polygon
CodeBERT	0.7104	0.7013	0.6802	0.6511	0.6342
GraphCodeBERT	0.6793	0.7184	0.7145	0.7406	0.6049
BERT4ETH	0.4235	0.2736	0.2471	0.2814	0.3139
XBlockFlow	0.8032	0.7247	0.6051	0.6523	0.7189
<b>EtherSentinel(our)</b>	<b>0.9381</b>	<b>0.9212</b>	<b>0.9134</b>	<b>0.8926</b>	<b>0.9078</b>

表 3.13: 不同平台的地址风险检测准确率

方法	ETH	BSC	Fantom	Arbitrum	Polygon
CodeBERT	0.6092	0.6017	0.5539	0.4513	0.6298
GraphCodeBERT	0.7774	0.8025	0.7483	0.7207	0.7051
BERT4ETH	0.5619	0.5234	0.4541	0.4072	0.3426
XBlockFlow	0.6634	0.6247	0.6143	0.6342	0.5699
<b>EtherSentinel(our)</b>	<b>0.8481</b>	<b>0.8012</b>	<b>0.7834</b>	<b>0.7356</b>	<b>0.7778</b>

表 3.14: 不同平台的合约函数名恢复的双语评估分数

方法	ETH	BSC	Fantom	Arbitrum	Polygon
CodeBERT	0.6905	0.7251	0.6544	0.5528	0.7302
GraphCodeBERT	0.7015	0.6928	0.7546	0.7251	0.7319
<b>EtherSentinel(our)</b>	<b>0.8223</b>	<b>0.8174</b>	<b>0.8037</b>	<b>0.8101</b>	<b>0.7959</b>

### 3.5.7 敏感性实验

敏感性实验（Sensitivity Analysis）是一种用于确定系统输出对输入参数变化的响应程度的分析方法。它评估系统的输入参数在一定范围内变化时，对输出结果的影响程度，从而识别出哪些参数对系统的性能或行为有重要影响。在本节实验中，我们测试了系统模型对最大序列长度、图神经网络层数、注意力头数的敏感性。

#### (1) 最大序列长度对各任务性能的影响

本实验研究了最大序列长度对各任务性能的影响。即测量了边与点特征所能容纳的最大特征总数与各个任务性能之间的关系。具体而言，考察了最大序列长度分别为 32、64、96 和 128 时，交易检测准确率、账户验证准确率以及代码函数名恢复双语评估分数的变化情况。其结果如图 3.13 所示。

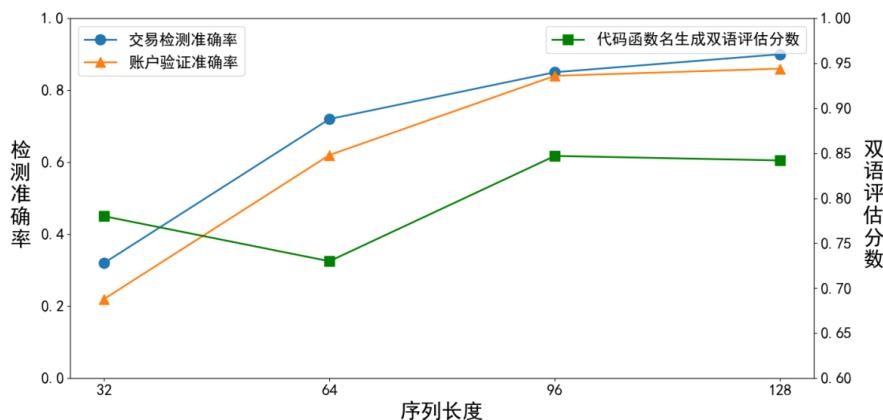


图 3.13: 最大序列长度对各任务性能的影响

从图 3.13 中可以看出，随着特征长度的增加，各任务的性能均有显著提升。交易检测准确率、账户验证准确率和代码函数名生成双语评估分数在特征长度达到 64 时有明显提升，且在长度为 128 时达到了最高值。这表明增加特征长度能够有效提升模型的性能，但过长的序列长度可能导致性能的提升趋于平缓。

因此，增加特征长度对提升各项任务性能具有显著作用，但需要控制在合理范围内以避免性能的边际收益递减。该实验结果为选择适当的最大序列长度提供了指导，有助于进一步优化模型在区块链交易风险监测系统中的应用。

#### (2) 图神经网络层数对各任务性能的影响

系统模型采用了图神经网络架构，本实验测试了图神经网络层数的增加如何影响各个任务的性能。如图所示，研究了图神经网络层数分别为 1、2、3、4 和 5 时，交易检测准确率、账户验证准确率以及代码函数名生成双语评估分数随图神经网络层数变化的情况。这一实验结果表明，增加图神经网络的层数能够有效提升模型在各任务上的表现。这可能是因为更多的网络层能够捕捉到更复杂和多样的特征，从而增强了模型的表示能力和泛化性能。其结果如图 3.14 所示。

#### (3) 注意力头数对各任务性能的影响

本次实验还测试了注意力头数对各任务性能的影响。具体而言，考察了注意力头数分别为 2、4、8 和 12 时，交易检测准确率、账户验证准确率以及代码函数名生成双语评估分数的变化情况。结果如图 3.15 所示。

随着注意力头数的增加，各任务的性能均有显著提升。当注意力头数增加至 8 时，交易检测准确率、账户验证准确率和代码函数名生成双语评估分数均达到了最高值。随后在注意力头数增加至 12 时，各项性能有所下降。这表明，适当增加注意力头数能够有效提升模型性能，但过多的注意力头数可能导致性能的边际收益递减。

总体而言，增加注意力头数对提升各项任务性能具有显著作用，但需要在合理范围内控制头数以避免性能下降。该实验结果为选择适当的注意力头数提供了指导，有助于进一步优化模型在区块链交易风险监测系统中的应用。

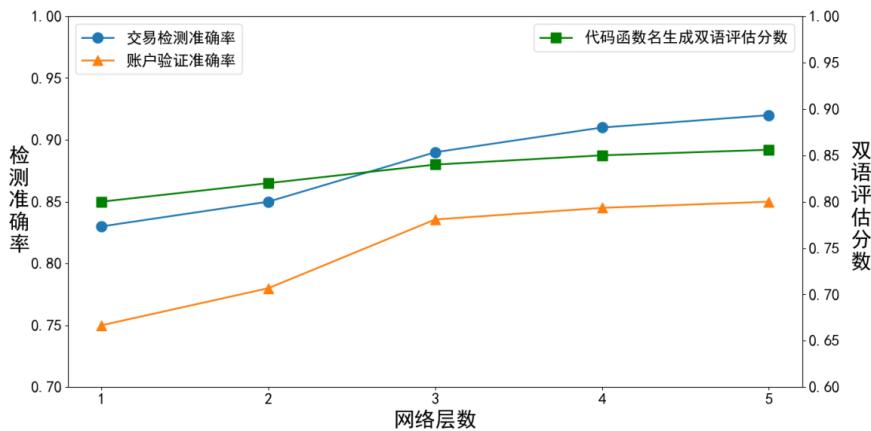


图 3.14: 各任务性能与图神经网络层数之间的关系

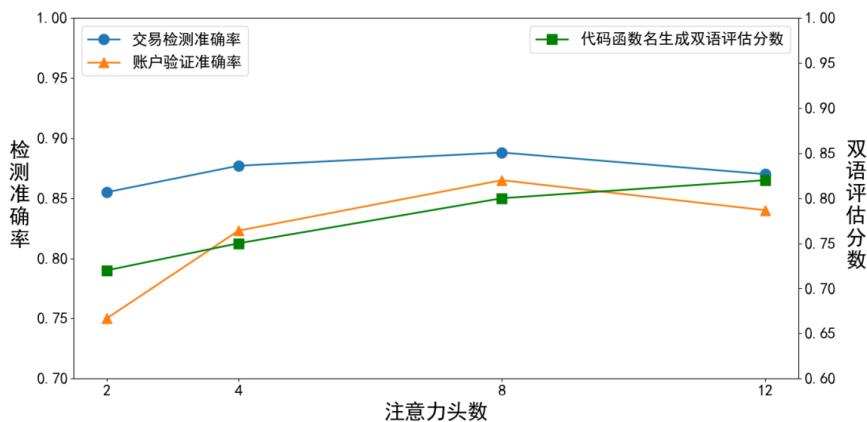


图 3.15: 各任务性能与注意力头数之间的关系

### 3.5.8 消融实验

消融实验（Ablation Study）是一种用于评估模型或系统中各组成部分重要性的方法。通过系统地移除或禁用模型的某些部分，观察对整体性能的影响，来确定各部分对模型整体性能的贡献。为了评估不同模型组合在区块链交易风险监测中各下游任务的性能，我们进行了消融实验。实验通过去除或组合不同的模型组件，通过观察对性能的影响，从而验证每个组件对模型整体性能的贡献。

在实验中，我们比较了三种模型组合：仅使用图神经网络（GNN）、仅使用 Transformer 以及使用 Graph Transformer 架构。评价指标包括代码函数名生成双语评估分数、账户验证准确率和交易检测准确率。实验结果如图 3.16 所示。

实验结果显示，在代码函数名生成双语评估分数方面，Graph Transformer 模型表现最佳，评分接近 0.9，显著优于仅 GNN 或仅 Transformer 的模型。在账户验证准确率方面和交易检测准确率方面，Graph Transformer 模型表现也显著优于其它组合。

t-分布随机邻域嵌入（t-distributed Stochastic Neighbor Embedding, t-SNE）[39] 是一种非线性降维技术，常用于高维数据的可视化。下面，我们将 GNN、Transformer、Graph Transformer 的抽取出的交易数据特征分布，通过 t-SNE 技术将其从高维映射至低维空间，从而可视化的揭示特征分布的内在结构和模型。

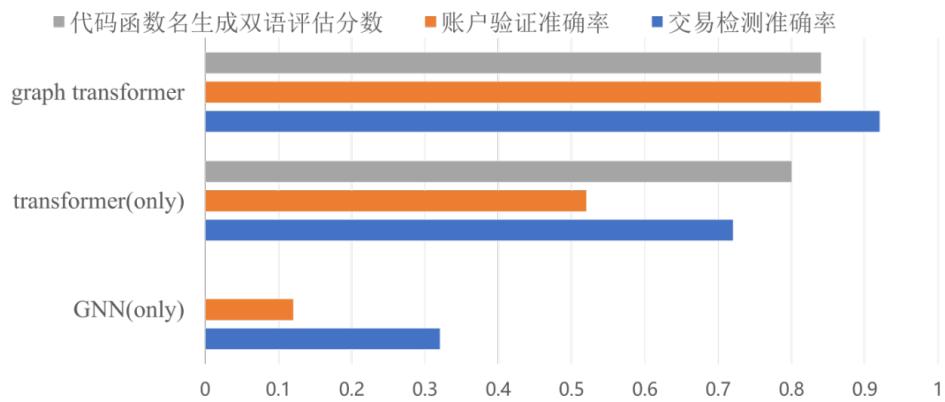


图 3.16: 消融实验结果

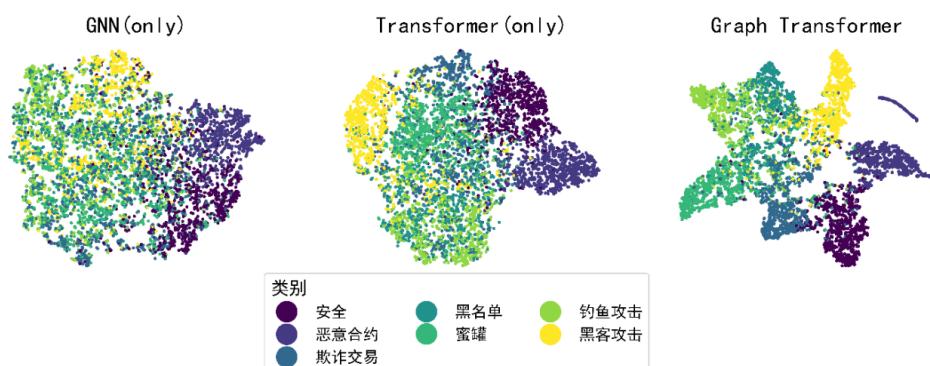


图 3.17: 交易检测的 t-SNE 特征分布

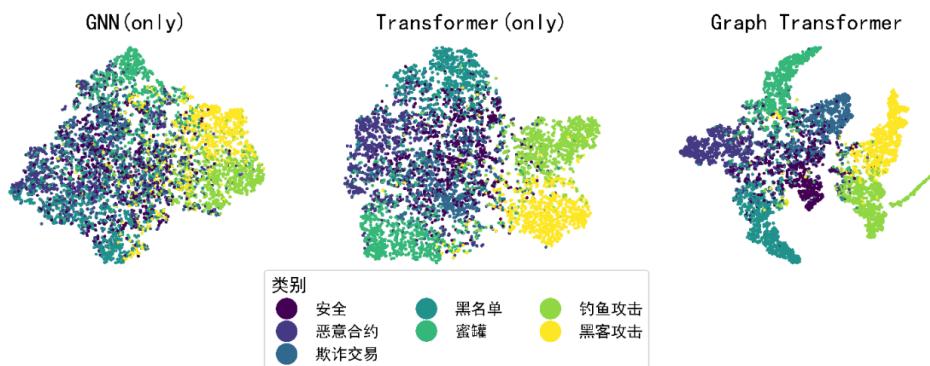


图 3.18: 地址检测的 t-SNE 特征分布

根据对交易和地址的特征抽取及低维映射，其 t-SNE 特征分布如图 3.17 和图 3.18 所示。从图中也可观察到，不同交易及不同地址的特征以聚类的方式分布，可以看出 Graph Transformer 对各类型交易和各类型地址的聚类分布最为集中，不同聚类间的界限更加明显，进一步证明了图 3.16 的消融实验结果。

## 3.6 用户端测试

### 3.6.1 首页及功能选择页面

为了展示区块链交易智能化风险监测系统的功能和实现流程，本团队特意开发了相应的前端展示系统。该系统首页提供了多个主要功能模块的入口，包括交易甄别、地址检测、代码生成和实时数据分析。通过这些模块，用户可以直观地了解和使用系统提供的多维度风险分析和智能监测服务。

在首页的功能选择部分，用户可以点击不同的功能模块进行详细操作。例如，通过“交易甄别”模块，用户能够实时检测和识别异常交易行为，保障交易的安全性；在“地址检测”模块，用户可以分析区块链地址的历史行为和风险等级，防范潜在威胁；“代码生成”模块则帮助用户生成和恢复智能合约代码，确保合约的完整性和安全性；“实时数据”模块提供区块链交易的实时监控与分析，支持用户做出及时准确的决策。该界面如图3.19所示。



图 3.19：系统首页及功能选择页面

### 3.6.2 恶意交易检测页面

恶意交易分类页面在用户输入交易哈希后，展示系统构建的交易图，提供全面的交易信息统计数据，包括交易数量、交易金额及交易所在区块等信息。此外，页面详细展示了交易的具体路径和参与地址，有助于用户追踪和分析每笔交易的流向和关联。

系统通过多维度的风险评估机制，识别交易中的潜在风险因素。例如，系统能够检测到可能存在的钓鱼攻击、黑客攻击、洗钱活动及与已知黑名单地址的交易互动。用户可以通过该页面获取详细的交易风险分析报告，并结合图示化的交易路径，迅速了解交易的安全性。为了帮助用户更好地理解和防范恶意交易，页面还提供了交互式的工具，用户可以对交易信息进行筛选和分析。同时，系统也会提供建议和警示，指导用户采取相应的安全措施，以降低交易风险。总之，该页面不仅提供了交易信息的可视化展示，还集成了强大的分析和预警功能，旨在为用户提供全面的交易风险甄别服务。页面如图3.20所示。



图 3.20: 恶意交易检测页面

在恶意交易检测页面，系统展示了输入交易哈希对应的交易图和相关信息，识别并标记了潜在的危险交易。通过交易图，用户可以直观地看到交易路径和各参与地址。交易图中的红色节点代表系统识别的危险交易，提示用户特别注意这些风险。

右侧区域列出了详细的交易信息，包括区块号、时间戳、交易发起者和接收者地址、交易金额以及燃料费用(Gas)，为用户提供了全面了解交易背景的基础，便于进一步的风险评估和决策。系统根据多维度风险评估模型，对交易进行了详细分析，结果包括涉嫌洗钱活动、与黑名单地址交易、异常高频交易等。每个风险指标旁边都有图标提示，帮助用户快速识别和理解风险类型和严重程度。对于被标记为危险的交易，系统会给出清晰的风险等级和详细说明。该页面如图3.21所示。

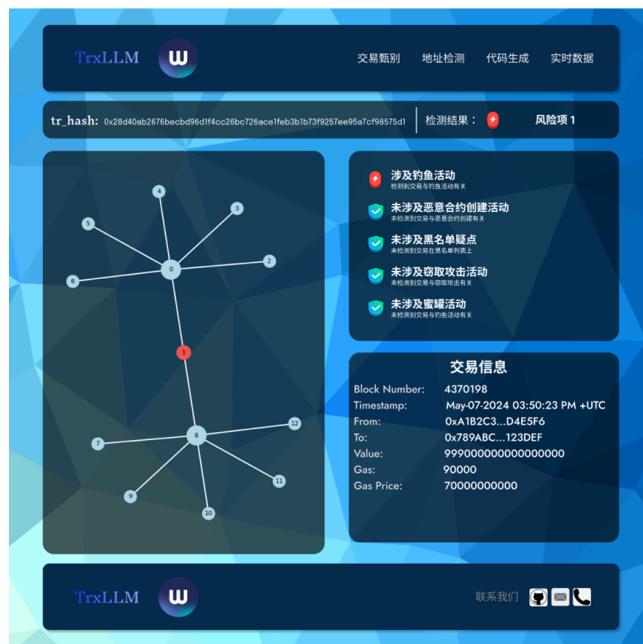


图 3.21: 恶意交易检测页面

如图3.22所示，在该页面系统展示了输入交易哈希所对应的交易图和详细信息，并确认该交易被识别为安全交易。交易图通过节点连接展示交易路径和所有参与地址，帮助用户清晰地了解交易的流向和参与方。

交易图中的节点显示为蓝色，表明系统未发现该交易存在任何风险。右侧区域提供了交易的详细信息，包

括区块号、时间戳、交易发起者和接收者地址、交易金额以及燃料费用（Gas），帮助用户全面了解交易背景和具体内容。系统对该交易进行了多维度风险评估，结果显示未发现洗钱活动、未与黑名单地址交易、未检测到异常交易频率或金额。这些安全评估结果通过图标和文字详细列出，确保用户能够清晰地了解交易的安全状况。



图 3.22: 恶意交易分类页面

### 3.6.3 地址风险评估页面

地址风险检测页面展示了用户输入地址哈希后的详细分析结果。首先，系统会构建并显示与该地址相关的交易图，用户可以通过该交易图直观地了解该地址的所有交易路径和交互关系。接着，页面展示了该地址的账户信息，包括账户余额、历史交易数量、历史交易金额等统计数据，帮助用户全面了解该地址的基本情况。

如果输入的地址是合约地址，系统还会展示该合约账户的详细信息，包括合约代码、合约函数、合约调用记录等。这些信息有助于用户了解合约的具体功能和行为模式，评估其安全性和潜在风险。此外，系统还会根据地址的历史行为进行风险评估，标记出可能存在的安全隐患，提供具体的风险指标和建议。页面如图3.23所示。



图 3.23: 地址风险检测页面

页面还提供了交互式工具，用户可以对交易图进行放大、缩小和详细查看，深入分析地址的交易关系和历史记录。通过这些功能，用户不仅可以快速识别地址的风险等级，还能获取到详细的分析报告和安全建议，帮助用户做出更明智的决策。

如图3.24所示，系统展示了输入地址哈希对应的交易图和详细分析结果，并标识出潜在的危险地址。交易图通过节点连接展示了该地址的所有交易路径和交互关系，红色节点代表被系统识别为高风险的交易，提示用户该地址存在安全隐患。右侧区域提供了详细的地址信息，包括ETH余额、ETH价值、地址标签、最后一次交易时间和第一次交易时间等，这些信息帮助用户全面了解该地址的背景和历史活动，便于评估其风险。最后，被检测为风险地址的账户节点会被系统记录下来。

图3.24展示了输入的账户地址被检测为风险地址的情况，如果账户地址被系统判定为安全地址，则所有节点显示为蓝色，同时也会展示交易子图及账户相关信息。



图 3.24: 地址风险检测页面

### 3.6.4 合约代码函数名恢复页面

合约代码恢复页面为用户提供了未知合约代码的深度分析功能，帮助用户更好地理解和分析合约的内部逻辑。用户在输入合约账户的地址后，系统将对该合约进行解析和分析，展示详细的预测结果，包括可能的函数名和功能描述。

如图3.25所示，该页面首先展示了原始合约的扫描结果，包含合约代码的基本结构和函数调用关系。通过这些信息，用户可以看到合约的内部构造和函数的调用路径。右侧区域展示了系统生成的结果，包括每个函数的预测名称和类型。



图 3.25: 合约代码函数名恢复页面

例如，函数名 0x5b1e3b51 被识别为 approveTransaction，0x5fe2e4d1 被识别为 finalizePayment。这些预测结果帮助用户更全面地了解合约代码的功能和潜在风险，为进一步的安全审计和分析提供了坚实的基础，如图3.26所示。

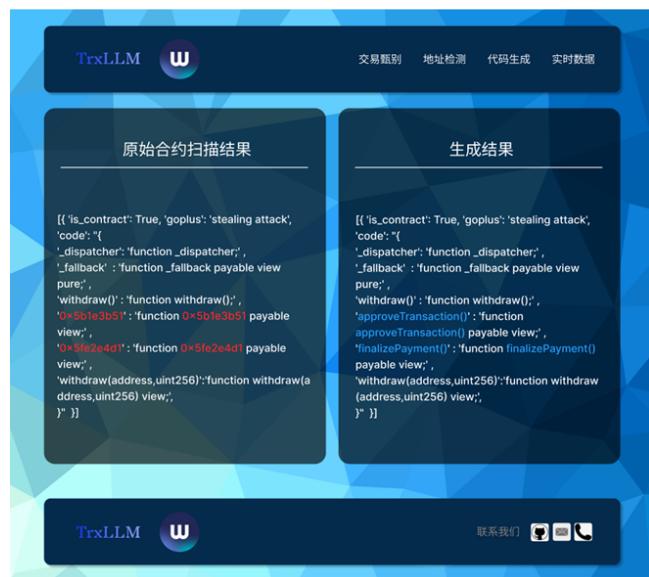


图 3.26: 合约代码函数名恢复页面

### 3.6.5 实时交易数据展示页面

实时交易数据展示页面为用户提供了选择交易平台和交易对的功能，实时展示特定区块中的所有交易信息，包括交易哈希、发起者、接收者等详细数据，同时展示系统对每笔交易的风险评估结果，帮助用户快速识别潜在风险。页面如图3.27所示。



图 3.27: 实时交易数据检测页面



图 3.28: 实时交易数据检测页面

### 3.6.6 系统响应延时

我们测试了用户输入交易哈希或地址哈希，并进行交易风险查询、地址风险查询、合约代码函数名恢复需求的系统响应延时。结果如表3.15所示。相应的，我们还测试了除去交易图构建时长的模型实际运行时间，结果也列于同表。测试结果显示，系统响应延时中，交易图构建花费的时间占主要地位。

表 3.15: 系统平均响应延时与模型平均运行时间

测试项	恶意交易检测	地址风险评估	合约代码函数名恢复
系统响应延时	3.27s	21.33s	3.58s
模型运行时间	0.12s	0.30s	0.13s

## 第4章 创新性与实用性说明

本作品的创新点主要表现在以下三个方面：

### 1. 面向区块链交易多元异构的统一嵌入表示技术

当前的恶意交易检测方法通常忽略了交易过程中产生的多种复杂特征，如事务日志（log）、函数调用记录（trace）等文本型特征，导致检测效果不佳。本作品创新性地设计了一种将文本型特征与数据型特征统一编码的架构，有效解决了交易全过程特征展示不全面的难题。通过这种统一的嵌入表示技术，不仅提高了交易异常检测的准确性，还使得交易行为和账户状态分析更加深入，为识别和分析恶意交易提供了更强的技术支持。此外，这种方法能够将多种特征统一为高维向量表示，便于后续的模型训练和特征提取。

### 2. 基于 Graph Transformer 的图大模型架构及图采样技术

当前的方法在处理大规模区块链交易数据时，通常采用简单随机游走采样，存在图学习复杂度高和重要信息丢失的问题。本作品创新性地设计了一种围绕检测交易构建子图的图采样策略，有效解决了多边和大量交易数据处理的难题，实现了更高效、更准确的模型学习。同时，本作品结合了 Transformer 和图神经网络（GNN）的优势，设计了一种图大模型架构——Graph Transformer，它能够同时处理交易数据的序列信息和图结构信息。通过这种深度融合的模型架构，系统能够更好地理解复杂交易图数据，提高恶意交易的检测精度。

### 3. 面向不同交易平台的迁移性增强技术

当前的区块链交易监测系统在面对不同交易平台时往往缺乏适应性，导致检测效果不佳。本作品针对这一不足，设计了一种基于图结构的通用下游任务预训练技术和账户函数名恢复方法，以提升系统在多种区块链平台上的适用性。在实际应用中，系统不仅能够检测恶意交易和评估账户风险，还具备合约代码函数名恢复功能。具体来说，本作品采用了图结构的掩码语言模型（MLM）预训练技术，使模型能够在大规模未标记数据上学习通用表示，之后针对特定任务进行微调。此外，系统还引入了账户函数名恢复技术，进一步提高了模型在不同平台上的迁移能力，从而实现更广泛的应用。通过这种迁移性增强技术，系统可以在多种区块链平台上高效部署，并适应不同平台的交易特性，提高了检测的准确性和适应性。

同时本系统的实用性主要表现以下几个方面：

**(1) 高准确率：**本作品在三个下游任务均取得了优秀的性能。该系统在交易风险检测任务中取得了 92% 的准确率，在账户检测任务中取得了 85% 的正确率，显著高于现有工作，这主要来源于我们采用的面向区块链交易多元异构信息的统一特征表示使得我们可以学习更多特征从而获得了更多有关交易图的信息以及图 transformer 的架构有效的学习了区块链交易图的数据信息；另一方面该系统在函数代码名恢复任务中取得了双语评估指数 0.82 正确率，这得益于我们所采取的生成式预训练任务使得模型有效的学习到了账户的特征。

**(2) 高兼容性：**本作品在进行区块链平台之间的迁移检测时具有高兼容性在平台细分交易正确率评估时最低正确率达到 89%，最高正确率则达到 94%，即性能不会随着平台迁移而发生剧烈波动，稳定性显著高于现有工作，这是因为我们采用了面向区块链交易多元异构信息的统一特征表示，这意味着该系统在区块链之间迁移时不需要依赖于某一特征，从而可以方便的迁移，同时我们采取的预训练方法允许我们将大规模平台数据集预训练的模型经过小规模平台数据集微调后迁移到模型上进一步提高了兼容性。

**(3) 实时性：**本系统在函数名恢复与恶意交易检测两个下游任务平均响应时间为 3.27 秒，在账户风险检测任务中具有 21.89 秒，具有实时高效完成各种下游任务的能力。我们的中心交易采样方法有效的限制了需要分析的信息以及子图的采样时间，这是至关重要的因为子图采样时间占响应时间的一大部分，从而使得模型可以更加高效的学习交易图信息。高实时性也意味着高模型可以方便的用于实时监测。

综上，本系统具有高准确率、高兼容性、以及实时性，可以部署在系统上用于实时监测以及预测区块链交易风险。

## 第5章 总结

总结来说，本团队推出了 EtherSentinel，一个基于图大模型的区块链风险检测平台。该平台能够实现对区块链交易的实时风险监测。本团队与新加坡某高校展开合作，构建了一个包含二十万个交易子图的数据集，对模型进行生成式任务预训练，并对模型进行微调，使其能够适应各种下游任务并取得了良好的性能。在预训练模型的基础上进行微调训练，可以支持恶意交易检测、账户风险检测和合约代码函数名恢复等功能，并且在 8 万规模的数据集上进行了测试和验证，分别达到了 0.92, 0.85 的准确率和 0.82 的 BLEU 指标分数，函数名恢复与恶意交易检测功能平均响应时间为 3.27 秒，账户风险检测系统评价响应时间为 21.33 秒。成为我们目前所有已知工作中表现最出色的解决方案，同时具备良好的实时性。

展望未来，EtherSentinel 将不断优化和扩展，以适应区块链技术和交易平台的快速发展。我们计划进一步提升系统的实时性和处理能力。在功能扩展方面，我们计划增加更多的风险检测和分析工具。例如，引入更多维度的风险评估指标，不仅限于交易和账户风险，还包括智能合约的漏洞检测。此外，将进一步开发用户友好的界面和交互工具，使系统的操作更加简便直观，用户能够更加高效地进行风险监测和管理。后续功能开发方向如下，针对本作品的功能模块作出优化：

(1) 增加风险分类：目前交易风险分析针对的具体类别尚有限，后续开发我们会逐步增加该系统能够检测的具体交易类别，如三明治攻击，洗钱等。

(2) 增加交易的周边信息：除了交易本身以及节点的账户信息和图结构信息之外，我们还希望考虑其他的信息，比如账户状态的变迁历史，交易前后的其他交易，这些信息可能会有助于我们检测其他类别的交易风险如抢跑等。

(3) 增强智能合约审计与验证能力：本系统现有对智能合约的审计能力主要基于对合约账户地址风险的评估以及代码函数名恢复功能，未对智能合约代码的逻辑与潜在漏洞进行更加深入的分析。后续工作希望集成智能合约的自动审计工具，帮助发现智能合约漏洞，使系统能够更加全面的服务于区块链交易平台的安全保障与监管机制。

总之，未来 EtherSentinel 将不断创新和进步，通过技术的不断突破和应用的广泛推广，成为区块链交易风险监测领域的领先解决方案，全面提升区块链生态系统的安全性和可靠性。

## 参考文献

- [1] Vaswani A, Shazeer N, Parmar N, et al. “Attention is all you need”. In: *Advances in Neural Information Processing Systems* 30 (2017).
- [2] Arbitrum. *Arbitrum —The Future of Ethereum*. Online. Available at: <https://arbitrum.io/>. June 2024.
- [3] Tom Brown, Benjamin Mann, Nick Ryder, et al. “Language models are few-shot learners”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 1877–1901.
- [4] BscScan. *BscScan*. Online. Available at: <https://bscscan.com/>. June 2024.
- [5] Charles T Carlstrom and Timothy S Fuerst. “Agency Costs, Net Worth, and Business Fluctuations: A Computable General Equilibrium Analysis”. In: *The American Economic Review* (1997), pp. 893–910. issn: 0002-8282.
- [6] Federico Cernera et al. “Token Spammers, Rug Pulls, and Sniper Bots: An Analysis of the Ecosystem of Tokens in Ethereum and in the Binance Smart Chain (BNB)”. In: *Proceedings of the 32nd USENIX Security Symposium (USENIX Security 23)*. Anaheim, CA: USENIX Association, 2023, pp. 3349–3366. URL: <https://www.usenix.org/conference/usenixsecurity23/presentation/cernera>.
- [7] CertiK. *Warp Finance* 闪电贷攻击事件分析. <https://mytokencap.com>. [EB/OL]. 2024-06-04.
- [8] J. Chen et al. “Defining Smart Contract Defects on Ethereum”. In: *IEEE Transactions on Software Engineering* (2020), pp. 1–1. doi: [10.1109/TSE.2020.2989002](https://doi.org/10.1109/TSE.2020.2989002). URL: <https://doi.org/10.1109/TSE.2020.2989002>.
- [9] CoinMarketCap and Spartan Labs. *State of the DeFi Industry Report*. <https://coinmarketcap.com/academy/zh/article/coinmarketcap-and-spartan-labs-state-of-the-defi-industry-report>. [EB/OL]. 2024-06-04. 2024.
- [10] Mauro Conti et al. “A survey on security and privacy issues of bitcoin”. In: *IEEE Communications Surveys & Tutorials* 20.4 (2018), pp. 3416–3452.
- [11] Chen D, O’ Bray L, and Borgwardt K. “Structure-aware transformer for graph representation learning”. In: *International Conference on Machine Learning*. 2022, pp. 3469–3489.
- [12] Kreuzer D, Beaini D, Hamilton W, et al. “Rethinking graph transformers with spectral attention”. In: *Advances in Neural Information Processing Systems*. Vol. 34. 2021, pp. 21618–21629.
- [13] Vujičić D, Jagodić D, and Randić S. “Blockchain technology, bitcoin, and Ethereum: A brief overview”. In: *17th International Symposium Infoteh-Jahorina (Infoteh)*. 2018, pp. 1–6.
- [14] Jacob Devlin et al. “BERT: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).
- [15] Gritti F, Ruaro N, McLaughlin R, et al. “Confusum contractum: confused deputy vulnerabilities in ethereum smart contracts”. In: *32nd USENIX Security Symposium (USENIX Security 23)*. 2023, pp. 1793–1810.
- [16] Scarselli F, Gori M, Tsoi A C, et al. “The graph neural network model”. In: *IEEE Transactions on Neural Networks* 20.1 (2008), pp. 61–80.
- [17] Yuzhou Fang, Daoyuan Wu, Xiao Yi, et al. “Beyond “Protected” and “Private”: An Empirical Security Analysis of Custom Function Modifiers in Smart Contracts”. In: *Proceedings of the ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA)*. 2023. doi: [10.1145/3597926.3598125](https://doi.org/10.1145/3597926.3598125). URL: <https://doi.org/10.1145/3597926.3598125>.
- [18] 方军雄. “所有制、制度环境与信贷资金配置”. In: *经济研究* 12 (2007), pp. 82–92. issn: 0577-9154.

- [19] Zhangyin Feng et al. “CodeBERT: A pre-trained model for programming and natural languages”. In: *arXiv preprint arXiv:2002.08155* (2020).
- [20] Wood G. *Ethereum: A secure decentralised generalised transaction ledger*. Tech. rep. 151. Ethereum Project, 2014, pp. 1–32.
- [21] Yu Gai, Liyi Zhou, Kaihua Qin, et al. *Blockchain Large Language Models*. arXiv preprint. 2024. URL: <https://doi.org/10.48550/arXiv.2304.12749>.
- [22] Guillermo Gomez, Pedro Moreno-Sanchez, and Juan Caballero. “Watch your back: Identifying cybercrime financial relationships in Bitcoin through back-and-forth exploration”. In: *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. 2022, pp. 1291–1305.
- [23] GoPlus Labs. *GoPlus Labs*. <https://gopluslabs.io/>. [Online; accessed 2024-06-04]. 2024.
- [24] Daya Guo et al. “GraphCodeBERT: Pre-training code representations with data flow”. In: *arXiv preprint arXiv:2009.08366* (2020).
- [25] Ethan Heilman et al. “Eclipse attacks on Bitcoin’s peer-to-peer network”. In: *24th USENIX Security Symposium (USENIX Security 15)*. 2015, pp. 129–144.
- [26] Investopedia. *What Happened to OneCoin, the \$4 Billion Crypto Ponzi Scheme?* <https://investopedia.com>. [EB/OL]. 2024-06-04.
- [27] Kim J, Nguyen D, Min S, et al. “Pure transformers are powerful graph learners”. In: *Advances in Neural Information Processing Systems*. Vol. 35. 2022, pp. 14582–14595.
- [28] Yuan J, Chen S, Zhang Y, et al. “Graph attention transformer network for multi-label image classification”. In: *ACM Transactions on Multimedia Computing, Communications and Applications* 19.4 (2023), pp. 1–16.
- [29] N. Leontiadis and N. Christin. “Empirically measuring WHOIS misuse”. In: *European Symposium on Research in Computer Security*. 2014, pp. 19–36.
- [30] C. Lever, P. Kotzias, D. Balzarotti, et al. “A lustrum of malware network communication: Evolution and insights”. In: *2017 IEEE Symposium on Security and Privacy (SP)*. 2017, pp. 788–804.
- [31] C. Lever, R. Walls, Y. Nadji, et al. “Domain-z: 28 registrations later measuring the exploitation of residual trust in domains”. In: *2016 IEEE Symposium on Security and Privacy (SP)*. 2016, pp. 691–706.
- [32] Qiang Li, Liwen Chen, and Yong Zeng. “The Mechanism and Effectiveness of Credit Scoring of P2P Lending Platform: Evidence from Renrendai.com”. In: *China Finance Review International* 8.3 (2018), pp. 256–274.
- [33] X. Li, A. Yepuri, and N. Nikiforakis. “Double and nothing: Understanding and detecting cryptocurrency giveaway scams”. In: *Network and Distributed Systems Security Symposium (NDSS)*. 2023. doi: [10.14722/ndss.2023.24584](https://dx.doi.org/10.14722/ndss.2023.24584). URL: <https://dx.doi.org/10.14722/ndss.2023.24584>.
- [34] Yiming Li and Tao Yang. “Word embedding for understanding natural language: a survey”. In: *Guide to Big Data Applications*. 2018, pp. 83–104.
- [35] Yinhan Liu, Myle Ott, Naman Goyal, et al. “RoBERTa: A robustly optimized BERT pretraining approach”. In: *arXiv preprint arXiv:1907.11692* (2019).
- [36] 刘凤良, 章潇萌, 和于泽. “高投资、结构失衡与价格指数二元分化”. In: *金融研究* 02 (2017), pp. 54–69. ISSN: 1002-7246.
- [37] 吕捷 and 王高望. “CPI 与 PPI “背离”的结构性解释”. In: *经济研究* 50.04 (2015), pp. 136–149. ISSN: 0577-9154.
- [38] Nadini M, Alessandretti L, Di Giacinto F, et al. “Mapping the NFT revolution: market trends, trade networks, and visual features”. In: *Scientific Reports* 11.1 (2021), p. 20902.

- [39] Laurens van der Maaten and Geoffrey Hinton. “Visualizing Data using t-SNE”. In: *Journal of Machine Learning Research* 9.11 (2008).
- [40] Robert McLaughlin, Christopher Kruegel, and Giovanni Vigna. “A Large Scale Study of the Ethereum Arbitrage Ecosystem”. In: *Proceedings of the 32nd USENIX Security Symposium (USENIX Security 23)*. Anaheim, CA: USENIX Association, 2023. URL: <https://www.usenix.org/conference/usenixsecurity23/presentation/mclaughlin>.
- [41] E. Medvet, E. Kirda, and C. Kruegel. “Visual-similarity-based phishing detection”. In: *Proceedings of the 4th international conference on Security and privacy in communication networks*. 2008.
- [42] Kipf T N and Welling M. “Semi-supervised classification with graph convolutional networks”. In: *arXiv preprint arXiv:1609.02907* (2016).
- [43] Tai D. Nguyen, Long H. Pham, et al. “sFuzz: An Efficient Adaptive Fuzzer for Solidity Smart Contracts”. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering (ICSE)*. 2020, pp. 778–788. doi: [10.1145/3377811.3380334](https://doi.org/10.1145/3377811.3380334). URL: <https://doi.org/10.1145/3377811.3380334>.
- [44] Kishore Papineni et al. “BLEU: a method for automatic evaluation of machine translation”. In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. 2002, pp. 311–318.
- [45] Poly Network. *Poly network and Hacker Communicate*. <https://sites.google.com/view/hackersconfession/home/communicate>. [EB/OL]. 2024-06-04.
- [46] K. Qin, S. Chaliasos, L. Zhou, et al. “The blockchain imitation game”. In: *USENIX Security Symposium*. 2023.
- [47] Vincenzo Quadrini. “Financial Frictions in Macroeconomic Fluctuations”. In: *FRB Richmond Economic Quarterly* 97.3 (2011), pp. 209–254.
- [48] QuickNode. *debug\_traceTransaction*. [https://guides.quicknode.com/docs/ethereum/debug\\_traceTransaction](https://guides.quicknode.com/docs/ethereum/debug_traceTransaction). [Online; accessed 2024-06-04]. 2024.
- [49] Alec Radford, Jeffrey Wu, Rewon Child, et al. “Language models are unsupervised multitask learners”. In: *OpenAI Blog* 1.8 (2019), p. 9.
- [50] Zubair Rafique, Tom Van Goethem, Wouter Joosen, et al. “It’s Free for a Reason: Exploring the Ecosystem of Free Live Streaming Services”. In: *Proceedings of the 23rd Network and Distributed System Security Symposium (NDSS)*. 2016. doi: [10.14722/ndss.2016.23030](https://doi.org/10.14722/ndss.2016.23030). URL: <https://doi.org/10.14722/ndss.2016.23030>.
- [51] Moheeb Abu Rajab et al. “The Nocebo Effect on the Web: An Analysis of Fake Anti-Virus Distribution”. In: *Proceedings of the 3rd USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET 10)*. San Jose, CA: USENIX Association, 2010. URL: <https://www.usenix.org/conference/leet-10/nocebo-effect-web-analysis-fake-anti-virus-distribution>.
- [52] R. Roberts, Y. Goldschlag, R. Walter, et al. “You Are Who You Appear to Be: A Longitudinal Study of Domain Impersonation in TLS Certificates”. In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 2019, pp. 2489–2504. doi: [10.1145/3319535.3363188](https://doi.org/10.1145/3319535.3363188). URL: <https://doi.org/10.1145/3319535.3363188>.
- [53] Xin Rong. “word2vec parameter learning explained”. In: *arXiv preprint arXiv:1411.2738* (2014).
- [54] Hu S, Zhang Z, Luo B, et al. “BERT4ETH: a pre-trained transformer for ethereum fraud detection”. In: *Proceedings of the ACM Web Conference*. 2023, pp. 2189–2197.
- [55] Li S, Gou G, Liu C, et al. “TTAGN: Temporal transaction aggregation graph network for ethereum phishing scams detection”. In: *Proceedings of the ACM Web Conference*. 2022, pp. 661–669. doi: [10.1145/3485447.3512226](https://doi.org/10.1145/3485447.3512226).

- [56] Security Affairs. *\$625M stolen from Axie Infinity's Ronin bridge, the largest ever crypto hack.* <https://securityaffairs.com>. [EB/OL]. 2024-06-04.
- [57] SharkTeam. 闪电贷攻击 + 业务逻辑漏洞: Platypus Finance 事件分析. <https://learnblockchain.cn>. [EB/OL]. 2024-06-04.
- [58] Yasuhiro Shibata, Takashi Kida, Seiji Fukamachi, et al. “Byte pair encoding: A text compression scheme that accelerates pattern matching”. In: (1999).
- [59] Matteo Taverna and Kenneth G Paterson. “Snapping snap sync: practical attacks on go Ethereum synchronising nodes”. In: *32nd USENIX Security Symposium (USENIX Security 23)*. 2023, pp. 3331–3348.
- [60] Fantom Team. *Fantom Overview*. Online. Available at: <https://fantom.foundation>. June 2024.
- [61] Polygon Technology. *Polygon Technology*. Online. Available at: <https://polygon.technology>. June 2024.
- [62] The Defiant. *Zapper Zaps Its Own Vulnerability Before Hackers Do*. <https://thedefiant.io/news/defi-zapper-zaps-its-own-vulnerability-before-hackers-do>. [EB/OL]. 2024-06-04.
- [63] Christof Ferreira Torres, Mathis Steichen, and Radu State. “The Art of the Scam: Demystifying Honeypots in Ethereum Smart Contracts”. In: *Proceedings of the 28th USENIX Security Symposium (USENIX Security 19)*. Santa Clara, CA: USENIX Association, 2019, pp. 1591–1607. URL: <https://www.usenix.org/conference/usenixsecurity19/presentation/ferreira>.
- [64] Christopher F Torres and Marc Steichen. “The art of the scam: Demystifying honeypots in ethereum smart contracts”. In: *28th USENIX Security Symposium (USENIX Security 19)*. 2019, pp. 1591–1607.
- [65] Jiahua Wu et al. “Towards Understanding Asset Flows in Crypto Money Laundering Through the Lenses of Ethereum Heists”. In: *IEEE Transactions on Information Forensics and Security* (2023).
- [66] Keyulu Xu et al. “How powerful are graph neural networks?” In: *arXiv preprint arXiv:1810.00826* (2018).
- [67] Shuo Yang, Jiachi Chen, and Zibin Zheng. “Definition and Detection of Defects in NFT Smart Contracts”. In: *Proceedings of the 32nd ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA ’23)*. Seattle, WA, USA, 2023. doi: [10.1145/3597926.3598063](https://doi.org/10.1145/3597926.3598063). URL: <https://doi.org/10.1145/3597926.3598063>.
- [68] P. Zhang, Z. Sun, S. Kyung, et al. “I’m SPARTACUS, no. I’m SPARTACUS: Proactively Protecting Users from Phishing by Intentionally Triggering Cloaking Behavior”. In: *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 2022, pp. 3165–3179. doi: [10.1145/3548606.3559334](https://doi.org/10.1145/3548606.3559334). URL: <https://doi.org/10.1145/3548606.3559334>.
- [69] L. Zhou, X. Xiong, J. Ernstberger, et al. “Sok: Decentralized finance (defi) attacks”. In: *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2023, pp. 2444–2461.
- [70] 中国人民银行数字人民币研发工作组. 中国数字人民币的研发进展白皮书. [https://www.gov.cn/xinwen/2021-07/16/content\\_5625569.htm](https://www.gov.cn/xinwen/2021-07/16/content_5625569.htm). [EB/OL]. 2024-06-04. 2021.
- [71] 以太坊. 以太坊首页. <https://ethereum.org>. [EB/OL]. 2024-06-04.