计算机科学
COMPUTER SCIENCE

Instruct-Malware:基于控制流图的大型语言模型恶意软件分析

周昱辰, 李鹏, 韩科技

引用本文

周昱辰, 李鹏, 韩科技. Instruct-Malware:基于控制流图的大型语言模型恶意软件分析[J]. 计算机科学, 2025, 52(11): 40-48.

ZHOU Yuchen, LI Peng, HAN Keji. Instruct-Malware:Control Flow Graph Based Large Language Model Analysis of Malware [J]. Computer Science, 2025, 52(11): 40-48.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.241100118

Instruct-Malware:基于控制流图的大型语言模型恶意软件分析

周昱辰[1]　李　鹏[1,2]　韩科技[1,2]
1 南京邮电大学计算机学院　南京 210023
2 南京邮电大学网络安全和可信计算研究所　南京 210023
(zyc1715385293@163.com)

摘　要　恶意软件检测与分类面临复杂性和隐蔽性的挑战。图神经网络(Graph Neural Networks,GNNs)虽能有效建模控制流图,提升行为模式捕捉精度,但其"黑盒"特性限制了可解释性。此外,现有方法依赖大量标注数据,泛化能力较弱,难以应对新型恶意软件。大型语言模型(Large Language Models,LLMs)具备强大的特征提取和上下文理解能力,能够有效处理少样本数据,实现多模态信息融合,从而增强分析精度与泛化性。受大型语言模型的启发,结合对比学习策略,同时学习控制流图的结构和汇编指令,以提高恶意软件分析的效果和灵活性。基于此,设计了 Instruct-Malware 框架。该框架采用轻量级图-文本对齐投影,通过双阶段指令优化,显著增强了恶意软件分析的灵活性和鲁棒性;此外,提升了模型的解释能力,透明化了决策过程。实验结果表明,所提出的框架在恶意软件分类和子图识别任务中展现了显著的性能提升,超越了现有的主流方法,并大幅缩小了与专业模型之间的差距,为构建高效且可靠的恶意软件分析系统提供了新的思路。

关键词:恶意软件分析;图神经网络;大语言模型;对比学习
中图分类号　TP319

Instruct-Malware:Control Flow Graph Based Large Language Model Analysis of Malware

ZHOU Yuchen[1],LI Peng[1,2] and HAN Keji[1,2]
1 School of Computer Science,Nanjing University of Posts and Telecommunications,Nanjing 210023,China
2 Institute of Network Security and Trusted Computing,Nanjing University of Posts and Telecommunications,Nanjing 210023,China

Abstract　Malware detection and classification face challenges due to their complexity and stealthiness. Although GNNs can effectively model control flow graphs,thereby enhancing the accuracy of behavioral pattern recognition,their "black-box" nature limits interpretability. Moreover,existing methods rely heavily on large amounts of labeled data,resulting in weaker generalization capabilities and difficulties in addressing novel malware. LLMs possess strong feature extraction and contextual understanding abilities,capable of efficiently processing few-shot data and achieving multimodal information fusion,thus enhancing analytical precision and generalizability. Inspired by large language models and leveraging contrastive learning strategies,this paper aims to simultaneously learn the structure of control flow graphs and assembly instructions,thereby enhancing the effectiveness and flexibility of malware analysis. Based on this,this paper designs the Instruct-Malware framework,which employs lightweight graph-text alignment projection and two-stage instruction optimization,significantly enhancing the flexibility and robustness of malware analysis. Additionally,the interpretability of the model has been improved,clarifying the decision-makingprocess. Experimental results demonstrate that the proposed framework exhibits significant performance improvements in malware classification and subgraph recognition tasks,surpassing current mainstream approaches and substantially narrowing the gap with specialized models. This provides new insights into building an efficient and reliable malware analysis system.

Keywords　Malware analysis,Graph neural networks,Large language models,Contrastive learning

1　引言

恶意软件已成为网络安全领域的核心挑战。根据 AV-TEST[1]的报告,截至 2024 年 8 月,全球已发现超过 11.8 亿种恶意软件样本。为了有效应对这些高度复杂和隐蔽的威胁,研究者在恶意软件检测和分类方面投入了大量精力。传统的检测方法依赖特征工程与规则匹配,通常通过手动提取特征或基于预定义的规则来识别恶意软件。然而,这些方法在面对复杂、多变的恶意软件时存在明显不足。首先,手动提取特征耗时费力,难以应对大规模、多样化的恶意软件

到稿日期:2024-11-20　返修日期:2025-03-18
基金项目:江苏省六大人才高峰高层次人才项目(RJFW-111);江苏省研究生科研与实践创新计划项目(KYCX24_1227,KYCX23_1048)
This work was supported by the Six Talent Peaks Project of Jiangsu Province(RJFW-111) and Postgraduate Research and Practice Innovation Program of Jiangsu Province(KYCX24_1227,KYCX23_1048).
通信作者:李鹏(lipeng@njupt.edu.cn)

样本。其次，规则匹配依赖于已知威胁的模式，难以识别新型或变种恶意软件，导致检测的泛化能力较弱。随着深度学习技术的飞速发展，新兴方法逐渐崭露头角，能够自动学习特征，提升了检测的准确性和适应性。

近年来，计算机视觉[2]和序列分析[3]等技术在恶意软件检测中得到了广泛应用。例如，研究者将恶意软件的二进制文件转换为图像或序列数据，并利用深度神经网络进行分类，这些方法在实际应用中取得了显著成果。然而，此类技术尽管具备优异的分类性能，但常忽视恶意软件的结构化信息，而结构化信息对理解恶意软件的行为模式及传播机制至关重要。

为弥补这一不足，图神经网络（Graph Neural Networks，GNNs）[4]作为一种先进的深度学习模型，被引入恶意软件分析领域。通过将恶意软件表示为控制流图（Control Flow Graph，CFG）[5-6]，研究者能够更深入地解析恶意软件的执行逻辑，从而提升分类的准确性和鲁棒性。GNNs 的优势在于，其能够自然地捕捉图结构中的复杂关系，并在保留图的拓扑信息的同时进行高效学习。然而，尽管 GNNs 在恶意软件分类中展示出巨大潜力，但是其“黑盒”特性依然难以解释模型的决策过程。这引发了对 GNN 模型进行可解释性研究的强烈需求，以便更好地理解其内部机制并提升其在实际应用场景中的可信度。

为了满足对 GNN 解释能力的需求，研究者提出了多种解释方法，如 GNNExplainer[7]和 SubgraphX[8]，这些工具通过深入分析模型的内部机制，提供了对分类决策过程的有效解释。此外，CFGExplainer[9]和 PGExplainer[10]等工具被广泛用于解释 GNN 的分类结果。然而，这些解释过程通常依赖于在标注数据上训练的任务特定模型，这些模型的适应性有限，并且每个新任务都需要费时的额外训练。

近年来，大型语言模型（Large Language Models，LLMs）如 ChatGPT[12]的崛起[11]，展示了其在自我监督学习未标注文本数据方面的强大泛化能力。LLMs 在处理基于自然语言指令的任务时，展现出非凡的灵活性和开放性，其能力不仅限于传统文本处理，还在跨模态研究中取得显著进展[13-14]。例如，在图像处理领域，LLMs 推动了基于指令的图像生成和视觉问答任务的发展，改变了人们与视觉信息的互动方式[15]，同时也在图像[16]和视频数据[17]方面展现了日益强大的潜力。这种多模态能力的提升，使得 LLMs 能够处理和理解更复杂的信息结构。在控制流图的分析中，可以借鉴多模态大语言模型（Multimodal Large Language Models，MLLMs）的成功经验[18]，将 CFG 与 LLMs 相结合，进一步提升恶意软件分析的效率与准确性。通过跨领域的融合，LLMs 凭借其强大的语言处理与多模态学习能力，不仅能加深对复杂图结构的理解，还能有效处理和解释图中的结构化信息，从而提升恶意软件分类任务的准确性和可解释性。

然而，将 CFG 与 LLMs 相结合面临诸多挑战，包括结构信息的对齐和图结构的理解等。此外，基于图结构的提示可能增加输入的 token 数量，从而提高计算和内存成本。较长的 token 序列不仅加重了模型的计算负担，还增加了内存消耗，使得该方法在大规模应用中难以实现。同时，现有 LLMs 通常对 token 数量存在限制，这进一步限制了其在大规模图结构建模中的应用。这些局限性凸显了开发更高效且具扩展性的方法的迫切需求，以更好地将图结构信息融入 LLMs，实现更高效的图分类和关键子图识别任务。

为了解决这些问题，本文提出了一种新的框架 Instruct-Malware，该框架将控制流图与文本信息进行对齐，并采用两阶段训练策略。首先，训练一个轻量级且灵活的接口，将控制流图的节点级表示映射到 LLMs 能够理解的文本空间中。然后，在任务特定指令的微调阶段冻结图编码器，并在 LLMs 上训练低秩适配器（Low-Rank Adaptation，LoRA），使模型能够适应多种应用场景。这种方法实现了控制流图信息与文本信息的有效结合，显著增强了恶意软件分析的灵活性和稳健性，拓展了其在多样化任务中的应用潜力。

2 相关工作

2.1 图神经网络

GNNs 近年来在处理图结构数据方面取得了显著进展，能够通过迭代地聚合邻居节点的信息来学习节点和图的表示，已被广泛应用于社交网络分析、推荐系统和生物信息学等领域。GNNs 的核心思想是通过信息传播机制，将图中每个节点的特征向量与其邻居节点的特征向量进行聚合，从而生成每个节点的嵌入表示。这一过程可以表示为：

$$\boldsymbol{m}_i^{(t+1)}=\sum_{j\in N(i)}M(\boldsymbol{h}_i^{(t)},\boldsymbol{h}_j^{(t)},\boldsymbol{e}_{ij}) \tag{1}$$

其中，$\boldsymbol{m}_i^{(t+1)}$表示节点v_i在第$t+1$轮中接收到的消息，$N_{(i)}$表示节点v_i的邻居节点集合，M是一个消息传递函数，$\boldsymbol{h}_i^{(t)}$为节点i在时刻t的表示，$\boldsymbol{h}_j^{(t)}$为节点j在时刻t的表示，$\boldsymbol{e}_{ij}$表示节点i与节点j形成的边的权重。

$$\boldsymbol{h}_i^{(t+1)}=U(\boldsymbol{h}_i^{(t)},\boldsymbol{m}_i^{(t+1)}) \tag{2}$$

其中，U是一个更新函数，通常是一个神经网络，如一个全连接层或一个 LSTM 单元。

2.2 图的可解释性

在 GNNs 的应用中，图的可解释性已经成为一个重要的研究方向。尽管 GNNs 在各种任务中展示了强大的性能，但其复杂的结构使得模型的决策过程难以理解。为了解决这一问题，研究者提出了多种可解释性方法和模型，这些方法通过识别和分析图结构的关键部分来解释 GNN 的决策过程。

GNNExplainer 通过学习一个掩码来识别对分类结果最重要的子图，从而提供模型的可解释性。SubgraphX 则利用蒙特卡罗树搜索（Monte Carlo Tree Search）和 Shapley 值来探索和评估不同的子图组合，并识别对模型分类贡献最大的子图。PGExplainer 采用了生成对抗网络（Generative Adversarial Network，GAN）框架，通过生成边缘掩码来解释图结构的重要性，从而识别出对分类最有贡献的子图。

此外，针对基于 CFG 的恶意软件分类任务，提出了一种新的解释方法——CFGExplainer。CFGExplainer 通过识别对分类最重要的控制流子图，并提供这些子图中节点的重要性评分，帮助分析人员深入理解恶意软件的分类过程。实验表明，CFGExplainer 在多个恶意软件家族的数据集上表现出色，能够识别出较小的子图，这些子图在分类准确性上超过了 GNNExplainer，SubgraphX 和 PGExplainer。

3 本文方法

Instruct-Malware 总体框架如图 1 所示，该框架旨在通过两大模块解决 CFG 中的图学习任务。第一模块是结构建模，将原始汇编文件转换为控制流图，使用预训练的 GNN 提取基本块的语义嵌入，为后续的图分类和子图识别任务做准备。第二模块是双阶段指令调优，在第一阶段的对齐预训练中，使用基本块-汇编指令对进行训练，构建一个轻量且适应性强的接口，旨在将基本块的节点级表示投影到 LLMs 可以理解的文本空间；第二阶段通过多个任务特定的指令进行微调。在这个过程中，预训练图编码器，并在 LLM 上训练一个低秩适配器，使所提模型能够适应不同的场景。通过这种方法，模型能够生成更加准确且符合上下文的表示，在后续任务中更高效地识别和处理恶意软件等图结构数据。

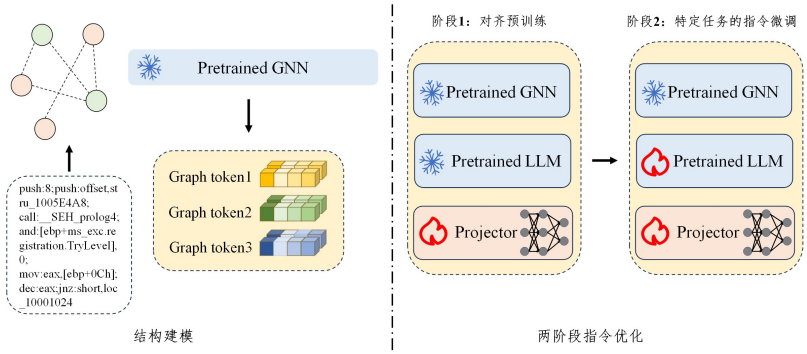

图 1 Instruct-Malware 模型架构设计

Fig. 1 Instruct-Malware model architecture design

3.1 基于控制流图的特征提取与子图构建

本节介绍了控制流图的构建、语义建模的应用以及关键子图的识别方法，为大语言模型提供针对恶意软件分析领域的指令微调[19]，进一步提升模型在该领域的应用效能。基于控制流图的特征提取与子图构建的结构流程如图 2 所示。

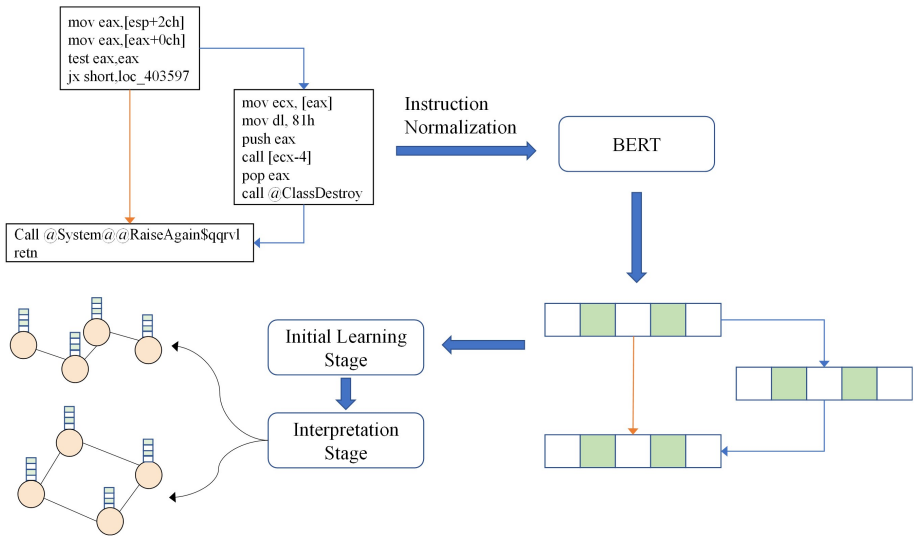

图 2 基于控制流图的特征提取与子图构建的结构流程图

Fig. 2 Structural flowchart of feature extraction and subgraph construction based on control flow graph

3.1.1 控制流图的构建

在静态恶意代码分析中，传统手段（如反汇编、字符串分析和签名匹配）被广泛用于理解恶意软件的行为。然而，这些方法存在明显的局限性。例如，反汇编结果可能因混淆技术而变得难以解读，字符串分析无法有效捕捉代码的逻辑结构，而签名匹配检测机制对变种恶意软件的识别能力有限，容易被规避。为了弥补这些不足，CFG 作为一种更为有效的分析手段被提出。通过从代码中提取控制流图，可以更好地捕捉程序执行的顺序和逻辑关系，从而为静态分析提供更深层次的洞察。

在此背景下，本文基于 CFG 提取算法，采用先进的工具（如 IDA Pro）从恶意软件代码中提取原始控制流信息。该过程首先将输入文件预处理为从排序地址到汇编指令的一对一映射。接着，通过两遍迭代过程对指令进行标记，以创建基本块并连接这些块。在第一次遍历中，为能够改变执行顺序的指令添加标签，以识别潜在的控制流转移；第二次遍历则依据第一次遍历时添加的标签创建基本块，并记录这些基本块之间的连接关系，从而完成控制流图的构建。通过这种方法，CFG 的构建不仅增强了对恶意软件行为的理解，也为后续分析提供了坚实的基础，如恶意软件行为识别、特征提取与分类等。

3.1.2 控制流图的语义建模

近期研究设计的 MAGIC 系统采用人工定义的基本块属

性来计算节点嵌入，但这一方法忽视了汇编指令的深层语义信息。相对而言，MCBG 系统利用 BERT 模型[20]生成类似于句子嵌入的基本块语义嵌入，实验证明半人工特征学习模型在特征学习效果上优于完全依赖人工特征的方法，展现出良好的性能与通用性。通过上述方法，可以获得控制流图中基本块的语义嵌入，从而构建一个带有语义建模的控制流图，记作$\mathcal{G}=(V,E,\boldsymbol{A},\boldsymbol{X})$。其中，$V=\{v_1,v_2,\cdots,v_N\}$是基本块（节点）的集合，而$|V|=N$表示基本块总数量；边的集合$V=\{(v_i,v_j)\}\subseteq V\times V$代表了程序执行路径之间的转移关系；$\boldsymbol{A}\in\mathbb{R}^{N\times N}$是邻接矩阵，定义了基本块之间的连接关系：

$$\boldsymbol{A}_{ij}=\begin{cases}1, & \text{存在边}(v_i,v_j)\\ 0, & \text{不存在边}\end{cases}$$

$\boldsymbol{X}\in\mathbb{R}^{N\times F}$包含了每个节点的相关属性，其中$F$是基本块嵌入（特征维度）。这种结构不仅增强了对控制流图的表达能力，也为后续的恶意软件分析提供了更加丰富的语义信息。

3.1.3 针对恶意软件分类的子图解释

基于图神经网络的控制流图分类器常被视为黑盒模型，这给恶意软件分析人员在验证分类结果和识别潜在恶意模式时带来了严峻的挑战。为提高分类器的可解释性，本文采用文献[9]中提出的 GNN 解释算法。该算法分为两个关键阶段：初始学习阶段和解释阶段。

初始学习阶段训练了两个核心组件：节点评分组件和基于 GNN 的节点嵌入生成组件。节点评分组件负责计算每个节点对分类结果的重要性，并为每个节点生成重要性分数$\boldsymbol{\Psi}\in[0,1]^{1\times N}$，量化节点在分类决策中的贡献。节点嵌入生成组件通过 GNN 模型生成图$\mathcal{G}$中每个节点$v_i\in V$的嵌入表示，从而捕捉节点间的复杂关系。解释阶段利用已训练好的节点评分组件和节点嵌入生成组件，生成可解释性结果。该过程不仅能够识别对恶意软件分类任务贡献最大的子图$G_s\subset G$，还能为子图中的基本块分配重要性得分$\boldsymbol{\Psi}$，为进一步的分析提供了有力支持。

3.2 控制流图与汇编指令的模态融合

控制流图与汇编指令的模态融合研究的核心在于，将控制流图的结构信息与汇编指令的语义信息进行有效整合，使得两种模态能够在同一特征空间中统一表示。对于控制流图$\mathcal{G}=(V,E,\boldsymbol{A},\boldsymbol{X})$，采用预训练带有跳跃知识的图同构网络（GIN-JK）作为图编码器f_G，用于生成结构级别的图表示$\boldsymbol{H}$。这一编码器通过多层信息聚合，能够捕捉节点间的复杂关系和全局结构特征，从而提高对控制流图的理解。同时，利用预训练的文本编码器（如 BERT）来生成汇编指令$\boldsymbol{C}$的语义表示$\boldsymbol{T}$，以确保其信息能够与图表示进行有效对齐。通过两种编码器，可以获得图表示$\boldsymbol{H}$和语义表示$\boldsymbol{T}$：

$$\boldsymbol{H}=f_G(\boldsymbol{X}),\boldsymbol{T}=f_T(\boldsymbol{C}) \tag{3}$$

基于已有的研究成果[21-22]，本文采用了一种对比学习的方法 MoleculeSTM[23]，旨在将文本信息有效融入图结构编码过程中。对比学习的核心思想在于，最小化同一样本的控制流图结构与其对应的汇编指令描述之间的表示距离，同时最大化来自不同样本对之间的表示距离。例如，EBM-NCE[24]和 InfoNCE[25]方法均采用了这一思路，分别实现了同一样本的结构与文本的对齐，并对不同样本的结构与文本进行了有效对比。具体而言，EBM-NCE 和 InfoNCE 通过对同一数据样本的结构-文本对齐来促进模型的学习，同时确保模型能够区分不同样本之间的结构-文本特征。为此，引入跨模态的文本-结构对比损失，其计算式如下：

$$\begin{aligned}L_{\text{EBM-NCE}}=-\frac{1}{2}(&\mathbb{E}_{\boldsymbol{H},\boldsymbol{T}}[\log\sigma(E(\boldsymbol{H},\boldsymbol{T}))]+\mathbb{E}_{\boldsymbol{H},\boldsymbol{T}'}[\log(1-\\&\sigma(E(\boldsymbol{H},\boldsymbol{T}')))]+\mathbb{E}_{\boldsymbol{H},\boldsymbol{T}}[\log\sigma(E(\boldsymbol{H},\boldsymbol{T}))]+\\&\mathbb{E}_{\boldsymbol{H}',\boldsymbol{T}}[\log(1-\sigma(E(\boldsymbol{H}',\boldsymbol{T})))])\end{aligned} \tag{4}$$

$$L_{\text{InfoNCE}}=-\frac{1}{2}\mathbb{E}_{\boldsymbol{H},\boldsymbol{T}}\left[\log\frac{\exp(E(\boldsymbol{H},\boldsymbol{T}))}{\exp(E(\boldsymbol{H},\boldsymbol{T}))+\sum_{\boldsymbol{T}'}\exp(E(\boldsymbol{H},\boldsymbol{T}'))}+\log\frac{\exp(E(\boldsymbol{H},\boldsymbol{T}))}{\exp(E(\boldsymbol{H},\boldsymbol{T}))+\sum_{\boldsymbol{H}'}\exp(E(\boldsymbol{H}',\boldsymbol{T}))}\right] \tag{5}$$

其中，σ是 Sigmoid 激活函数；$\boldsymbol{H}$和$\boldsymbol{T}$代表每个样本的结构文本对；$\boldsymbol{H}'$和$\boldsymbol{T}'$则是从噪声分布中随机选取的负样本，基于经验数据分布进行采样。在联合学习空间中采用点积，即$E(\boldsymbol{H},\boldsymbol{T})=\langle p_c\circ f_c(\boldsymbol{H}),p_t\circ f_t(\boldsymbol{T})\rangle$。

3.3 两阶段训练

训练过程包含两个阶段：对齐预训练和特定任务的指令微调。通过借鉴最近提出的指令微调概念，增强语言模型在特定领域的适应性。预训练阶段的目标是将模型的语言处理能力与图学习任务的具体要求结合起来，使其能够为图结构数据生成更加准确且符合上下文的表示。微调阶段则侧重于高效适应后续的特定任务。

3.3.1 阶段 1：对齐预训练

第一阶段旨在对齐汇编指令与节点模态，从而使大型语言模型能够有效理解图结构中的上下文信息。如图 3 所示，所采用的恶意软件控制流图$\mathcal{G}=(V,E,\boldsymbol{A},\boldsymbol{X})$中，每个节点对应不同的汇编指令，形成节点-汇编指令对。为此，设计了一套指令，用于将图标记与汇编指令关联，同时考虑节点的重要性。具体而言，这些指令由 3 部分组成：1）人类指令；2）带有节点重要性得分的图信息；3）GPT 生成的回答。从控制流图中提取节点重要性得分$\boldsymbol{\Psi}_i\in[0,1]^{1\times N}$，将其归一化后与原始的基本块嵌入$\boldsymbol{x}_i$拼接，形成$\boldsymbol{x}_i'=[\boldsymbol{x}_i,\boldsymbol{\Psi}_i]$。这样做的目的是通过结合节点的重要性得分和节点的原始特征，将图结构中的关键信息直接融入节点嵌入表示中。这种拼接能够增强模型对节点之间相对重要性的感知，从而更有效地捕捉与图分类或子图识别相关的关键信息。指令包括图标记$\boldsymbol{X}_G$和打乱的汇编指令，图标记$\boldsymbol{X}_G$代表控制流图中的节点，而汇编指令列表对应节点的文本描述，通过线性投影器将图信息经过图编码器f_g映射到图标记$\boldsymbol{X}_G=f_g(\mathcal{G})=[\boldsymbol{x}_i',\boldsymbol{\Psi}_i]_{i=1}^N$，实现图标记与汇编指令的有效对齐。接下来，使用 GPT 模型生成的响应$\boldsymbol{X}_A$对图标记与汇编指令的关联进行优化。指令可表示为 Human：〈cls〉$\boldsymbol{X}_I$〈graph〉$\boldsymbol{X}_G$〈eos〉GPT：$\boldsymbol{X}_A$。其中，$\boldsymbol{X}_I$是由人类指令生成的文本嵌入；$\boldsymbol{X}_A$为 GPT 回答标记；$\boldsymbol{X}_G$为通过线性投影的图标记，具体为节点嵌入与重要性得分的拼接信息，即$\boldsymbol{X}_G=[\boldsymbol{x}_i,\boldsymbol{\Psi}_i]_{i=1}^N$。对于长度为$L$的输入序列，计算输出$\boldsymbol{X}_A$的概率如下：

$$p(\boldsymbol{X}_A\mid\boldsymbol{X}_G,\boldsymbol{X}_I)=\prod_{i=1}^{L}p_\theta(\boldsymbol{x}_i\mid\boldsymbol{X}_G,\boldsymbol{X}_I,\boldsymbol{X}_{A,<i}) \tag{6}$$

在训练过程中，为了防止过拟合，减少计算资源的需求，需要冻结大语言模型和图编码器 f_g 的初始权重，让训练过程更加专注于投影器的优化，最终成功学习将图表示映射到图标记 $\boldsymbol{X}_G$ 中与汇编指令有效对齐。

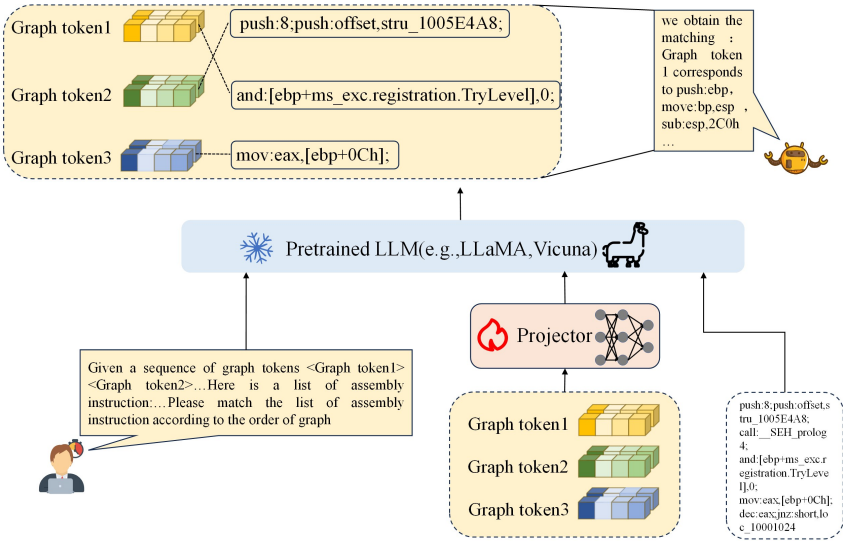

图 3 对齐预训练

Fig. 3 Alignment pretraining

3.3.2 阶段 2：特定任务的指令微调

第二阶段的目标是通过针对特定任务的指令微调，优化模型的基本能力，使其能够有效应对图分类和关键子图识别任务。特定任务的指令微调流程如图 4 所示。使用与第一阶段相同的指令模板，在第二阶段的图分类和关键子图识别任务中，依然将图表示通过线性投影器映射到图标记。图分类和关键子图识别任务的指令仍然包括图标记 $\boldsymbol{X}_G$ 和图中的汇编指令。在训练过程中，使用第一阶段训练得到的线性投影器参数作为初始状态，这有助于更好地针对图分类和关键子图识别任务进行指令微调，且只冻结图编码器 f_g 的参数，并继续更新投影器和大语言模型的初始权重。为了有效适应多样化的任务，在图分类和关键子图识别任务中，采用了低秩适应[26]。在图分类任务中，通过 LoRA 对图级别的图表示进行适配，以捕捉图的全局特征，从而对整体类别进行预测。而在子图识别任务中，则使用节点级别的图表示，利用 LoRA 识别出对分类贡献最大的关键子图。对于选定的语言模型中的权重矩阵 $\boldsymbol{W}\in\mathbb{R}^{d_1\times d_2}$，通过选择较小的秩参数 r，LoRA 不仅避免了全量微调可能导致的遗忘问题，还显著减少了内存占用。具体地，LoRA 将权重更新分解为两个低秩矩阵的乘积：

$$\boldsymbol{W}=\boldsymbol{AB} \tag{6}$$

其中，$\boldsymbol{A}\in\mathbb{R}^{d_1\times r}$，$\boldsymbol{B}\in\mathbb{R}^{r\times d_2}$，$r\ll d_1$，$r\ll d_2$。在前向传播过程中，LoRA 的计算式为：$h=\boldsymbol{W}x+\boldsymbol{AB}x$，其中 $\boldsymbol{W}$ 保持冻结状态，新添加的 $\boldsymbol{AB}$ 在适应过程中进行训练。因此，即使在资源受限的环境中，LoRA 仍能展现良好的性能。

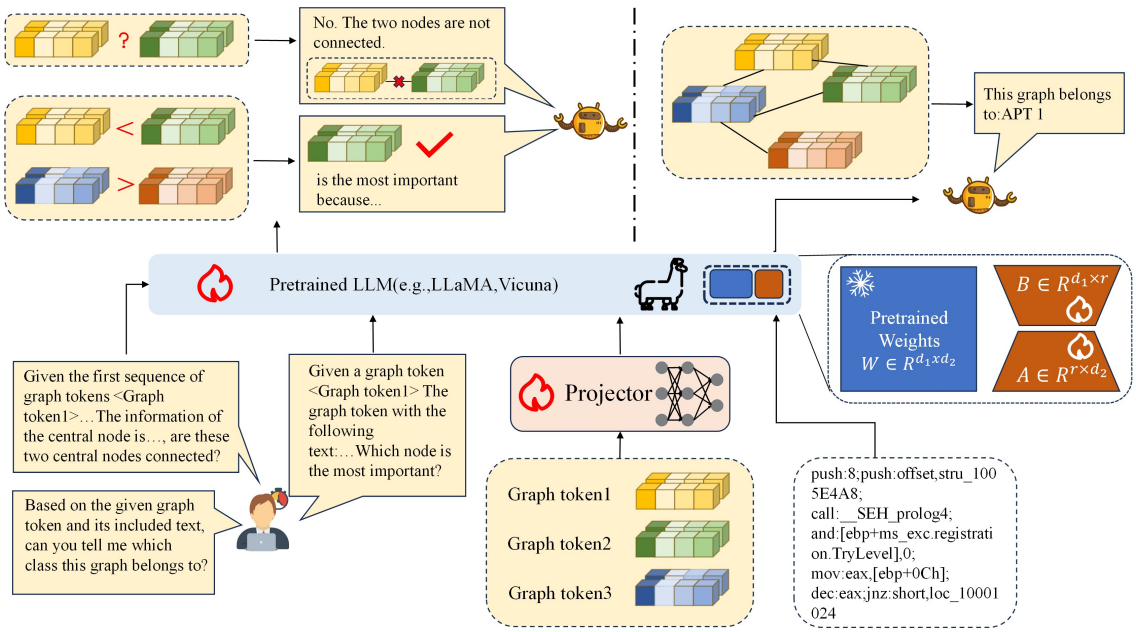

图 4 特定任务的指令微调

Fig. 4 Task-specific instruction fine-tuning

4 实验

4.1 数据集

本文用于评估的数据集 APTMalware[1)]，包含 3 500 多个与 12 个 APT 组相关的恶意软件样本，这些 APT 组由 5 个不同的国家赞助。所有样本均根据其 SHA-256 哈希值命名，并按 APT 组进行分组。在零样本任务中，使用 Microsoft Malware Classification Challenge(BIG2015) 来评估模型图分类和识别子图任务上的性能。在这种设置下，模型在 APTMalware 数据集上进行训练，但在 BIG2015 数据集上进行测试，且不进行额外的训练。

4.2 模型设置

1)图编码器选择：使用 5 层跳跃知识的图同构网络(GIN-JK)作为控制流图编码器 f_g，隐藏维度设为 128。

2)预训练模型初始化：基于 MoleculeSTM 初始化 GIN 模型的图编码器，通过控制流图和汇编指令进行对比学习预训练。

3)大语言模型初始权重：使用 vicuna-7b-v1.5-16k 模型(通过对 LLaMA 进行监督指令微调生成的版本)作为初始权重。

4.3 训练设置

4.3.1 第一阶段训练

1)数据集：使用 APTMalware 数据集，约 2.4×10^5 个样本的训练集划分组成。

2)批量大小与迭代次数：批量大小设置为 16，训练迭代次数为 3。

3)优化器设置：采用 AdamW 优化器，参数 $\beta=(0.9, 0.999)$，学习率设置为 2×10^{-3}；不进行权重衰减。

4)学习率调度：在总训练步骤的 3% 进行预热，之后采用余弦衰减调度。

5)模型输入设置：LLMs 的最大输入长度设置为 2 048。

4.3.2 第二阶段训练：

1)具体场景训练：训练时保持批量大小为 16，学习率设为 5×10^{-5}。

2)LoRA 设置：在 LLMs 中使用线性层，LoRA 秩为 64，缩放值 α 为 16。

4.3.3 实验环境

所有实验均在 4×RTX A800(80GB) GPU 上进行。

4.4 恶意软件分类任务

1)实验设置

为了全面评估模型性能，将其与多种基线模型进行比较。数据集按 7∶3 的比例划分为训练集与测试集，评估指标包括图分类的准确率和 F1 分数。为了处理控制流图的复杂性和图数据的高维信息，特别在图的长度较长时，引入了滑动窗口技术，窗口重叠率设置为 0.3，确保能够有效保留上下文信息，避免超出模型的输入长度限制。该方法充分发挥了大型语言模型在处理上下文信息方面的优势，从而提升了对图数据的整体理解。

2)基线模型

第一类基线方法采用多层感知器进行节点表示。该方法结构简单，但在某些情况下可能会忽视图结构的复杂性。第二类方法采用多种经典图神经网络编码器，包括 GraphSAGE[27]，GCN[28] 和 GAT[29]。这些方法能够在图数据上有效学习节点之间的关系，适用于大多数图分类任务。第三类方法采用了自监督学习技术，如深度图嵌入方法 DGI[30]。DGI 通过无监督方式学习图结构特征，对于缺少标签数据的任务具有重要意义。第四类基线方法则探讨了结合知识蒸馏技术的图神经网络(例如 GLNN[31])。通过蒸馏技术，模型能够从预训练的模型中汲取知识，提高图学习的效率和精度。最后，还考虑了基于开源大型语言模型(如 Qwen2-7B 等)的基线方法，作为理解文本属性图数据的一种方式。这类模型在处理文本和图数据之间的关系时，展现出较强的能力。

3)结果

表 1 列出了各类方法在图分类任务中的整体性能。在所有基线方法中，基于 Instruct-Malware 的框架在准确率和 F1 分数指标上均表现出色。通过结合滑动窗口机制，该方法在图数据的理解和分类上表现出较强的上下文连贯性和泛化能力。

表 1 在图分类任务上各种方法的性能比较

Table 1 Performance comparison of various methods on graph classification tasks

Models	APTMalware	
	Accuracy	*F1-score*
MLP	0.8314	0.7371
GCN	0.8953	0.8964
GAT	0.9149	0.9129
GraphSAGE	0.9084	0.9027
DGI	0.8442	0.8424
GLNN	0.8783	0.8783
Qwen2-7B	0.8691	0.8682
Llama3.1-8B	0.8822	0.8799
Instruct-Malware-7B	0.9398	0.9395

4.5 恶意软件识别子图任务

在恶意软件分析中，CFGExplainer 通过对基本块(节点)的重要性进行评分来识别对分类决策最为关键的子图，因为在控制流图中，节点(代码块)对于分析人员来说尤为重要。因此，将子图识别任务分为两个子任务：节点重要性判断和链路预测。

4.5.1 节点重要性判断任务

1)实验设置

节点重要性判断任务的目标是，根据得分从一组节点中识别出最重要的节点，并评估它们对分类决策的贡献。准确率用于衡量在所有被模型预测为关键节点的样本中，实际确实为关键节点的比例，从而有效评估模型在节点重要性评估中的精确度。为了提升节点重要性判断的准确性，本文引入了 COT(Chain of Thought)[32] 方法，将控制流图信息和任务描述作为输入，使用 GPT-4o 大语言模型进行推理。通过顺序推理，模型不仅能生成对重要节点的预测，还提供了每个预测的详细解释。这种透明的推理过程增强了决策的可理解性，

1) https://github.com/cyber-research/APTMalware

确保用户可以清楚了解模型判断节点重要性时的思考逻辑。

2)基线模型

由于节点重要性判断任务是一个相对较新的研究方向，因此传统的 GNN 方法可能不完全适用。考虑到这一点，选择了同样为大型语言模型的 Qwen2-7B 和 Llama3.1-8B 作为基线模型。

3)结果

表 2 列出了 Instruct-Malware 在节点重要性判断任务上的零样本能力，训练是在 APTMalware 中的一个类别 APT 1 上进行的，而评估分别在 APT 1 以及 BIG2015 的 Ramnit 和 Lollipop 类别上进行。从实验结果可以看出，Instruct-Malware-7B 在所有类别中均表现出色，特别是在 APT 1 类别中，准确率达到了 0.8199，显著高于 Qwen2-7B 和 Llama3.1-8B 模型；在 Ramnit 和 Lollipop 上的零样本设置下，仍优于其他基线模型，表现出较强的泛化能力。

表 2 监督学习与零样本设置下节点重要性评估方法比较

Table 2 Comparing node importance evaluation under supervised and zero-shot settings

Model	*Accuracy*		
	APT 1-APT 1	APT 1-Ramnit	APT 1-Lollipop
Qwen2-7B	0.7377	0.4819	0.5228
Llama3.1-8B	0.7308	0.4853	0.5460
Instruct-Malware-7B	0.8199	0.4957	0.5523

4.5.2 链路预测任务

1)实验设置

链路预测任务旨在评估模型在预测图中潜在节点连接方面的能力。通过分析图中的边，可以预测哪些节点之间可能存在潜在的连接关系。在链路预测任务中，采用 AUC(Area Under the ROC Curve)和 AP(Average Precision)作为主要评价指标。AUC 衡量模型在区分正负样本方面的能力；而 AP 则综合考虑了模型在不同阈值下的表现，适用于不均衡数据集的评估。

2)基线模型

链路预测任务基线模型包括以下几种：1) GCN 模型，利用图卷积网络聚合节点特征进行链路预测；2) GAT 模型，通过动态调整邻居节点的影响力进行链路预测；3) GraphSAGE 模型，通过邻居采样在大规模图中高效学习；4) 例如 Qwen2-7B 和 Llama3.1-8B 这样的预训练大型语言模型，利用文本信息进行链路预测任务。

3)结果

实验结果如表 3 所列。

表 3 监督学习与零样本设置下链接预测方法性能比较

Table 3 Performance comparison of link prediction methods under supervised and zero-shot settings

Models	supervision on APTMalware		zero shot on BIG2015	
	AUC	*AP*	*AUC*	*AP*
GCN	0.8816	0.8694	0.6590	0.6558
GAT	0.8665	0.8764	0.6379	0.6402
GraphSAGE	0.8392	0.8459	0.5818	0.5632
Qwen2-7B	0.9228	0.9186	0.6636	0.5105
Llama3.1-8B	0.9465	0.9161	0.6623	0.5083
Instruct-Malware-7B	**0.9736**	**0.9561**	**0.7828**	**0.6664**

从中可以看出，Instruct-Malware-7B 在所有实验设置中都表现出色，尤其在 APTMalware 数据集上的监督学习中，AUC 和 AP 均为最高，分别达到了 0.9736 和 0.9561。在 BIG2015 数据集上的零样本学习中，Instruct-Malware-7B 也显著优于其他模型，AUC 为 0.7828，AP 为 0.6664，其表现出强大的泛化能力。相比之下，传统的 GNN 方法(如 GCN，GAT 和 GraphSAGE)在监督学习设置下表现较好，但在零样本设置下，性能有所下降。Qwen2-7B 和 Llama3.1-8B 在零样本学习中的表现优于 GNN 方法，但仍不如 Instruct-Malware-7B。

4.6 消融实验

本节进行了消融实验，以探究所提框架的架构和训练方案设计的有效性，实验结果如表 4 所列。实验从多个角度探讨了不同的变化，并在图分类任务上进行了验证。

1)实验对比了使用 LoRA 针对特定任务微调大语言模型与仅训练投影器两种方法的效果。实验结果显示，仅训练投影器的方法虽然在处理广泛的任务时表现出一定的通用性，但在特定任务上的表现却不及使用 LoRA 进行微调的方法。

2)对比了使用线性投影层和 MLP 作为投影器的效果[33]。实验结果表明，线性投影层在图分类任务上的表现优于 MLP。这可能是因为对于特定任务，MLP 的复杂性未能有效提高性能。在通用模型中，仅训练投影器时，MLP 的表现可能会优于线性层。

3)将图编码器 f_g 替换为与原编码器参数规模相同的单模态模型，如 GIN-JK 模型。实验结果表明，GIN-JK 的表现不如预先与文本模态对齐的编码器。单模态模型虽然能学习图数据表示，但未能有效捕捉跨模态的深层联系。相比之下，与文本模态对齐的编码器加强了图与文本间的联系，提升了模态对齐的有效性，使模型能够同步学习文本和图结构，在复杂任务(如代码分析)中提供了更丰富的表示，增强了理解与推理能力。

4)在第二阶段训练中，选择不更新 LLM 的权重，导致模型在收敛和任务推理方面出现问题。由于权重未更新，模型难以适应特定任务，表现出推理不准确及结果不一致等问题。这表明，在指令微调阶段更新 LLM 权重对于任务适配至关重要。

表 4 消融模型架构与训练方案设计实验

Table 4 Ablation study on model architecture and training scheme design

METHODS	*Accuracy*	*F1-score*
Instruct-Malware	**0.9398**	**0.9395**
+2-MLP	0.9228(−1.7%)	0.9226(−1.69%)
+GIN-JK	0.9280(−1.18%)	0.9258(−1.37%)
Only Projector	0.8560(−8.3%)	0.8563(−8.32%)
Freeze LLM	−	−

4.7 结论

本文提出了一种将控制流图与大型语言模型相结合的方法，提升了恶意软件分类和子图识别任务的性能。通过两阶段训练策略，该方法不仅取得了较高的分类准确率和 F1 分数，还增强了模型的可解释性和鲁棒性。实验结果表明，在图分类任务上，Instruct-Malware-7B 在 APTMalware 数据集上

的 Accuracy 达到 93.98%，F1-score 达到 93.95%，相较于结果最好的方法 GAT，分别提升了 2.49%和 2.66%。在链路预测任务中，Instruct-Malware-7B 在 APTMalware 数据集上的 AUC 达到了 0.973 6，AP 为 0.956 1，相较于基线模型 Qwen2-7B 和 Llama3.1-8B 分别提升了 5.5%和 2.9%。在节点重要性判断任务中，Instruct-Malware-7B 的准确率在 APT 1 数据集上分别达到了 81.99%，49.57%和 55.23%，相比于 Qwen2-7B 和 Llama3.1-8B 有明显提升。该方法在处理大规模、多样化的恶意软件样本时表现出较强的泛化能力，为构建高效且可靠的恶意软件分析系统奠定了坚实的基础。

结束语 本文提出了一种名为 Instruct-Malware 的创新框架，该框架将控制流图与大型语言模型相结合，以提升恶意软件分类和子图识别任务的性能。通过引入轻量级图-文本对齐投影和双阶段指令优化策略，显著增强了恶意软件分析的灵活性、鲁棒性和解释能力。实验结果表明，在多个任务中，包括恶意软件分类、节点重要性判断以及链路预测任务，Instruct-Malware 均展现了优越的性能，超越了现有的主流方法。未来可以进一步探索将更多模态的信息（如网络流量、系统调用等）纳入模型，以提升恶意软件分析的全面性和准确性。

参 考 文 献

[1] Av-Test. Malware Statistics & Trends Report by Av-Test[EB/OL]. https://www.av-test.org/en/statistics/malware.

[2] SHARMA O, SHARMA A, KALIA A. Windows and IoT Malware Visualization and Classification with Deep CNN and Xception CNN Using Markov Images[J]. Journal of Intelligent Information Systems, 2023, 60(2): 349-375.

[3] ALZUBI O A, ALZUBI J A, ALZUBI T M, et al. Quantum Mayfly Optimization with Encoder-Decoder Driven LSTM Networks for Malware Detection and Classification Model[J]. Mobile Networks and Applications, 2023, 28(2): 795-807.

[4] XIAO G Q, LI X Q, CHEN Y D, et al. A Review of Large-Scale Graph Neural Networks[J]. Journal of Computer Science, 2024, 47(1): 148-171.

[5] YAN J, YAN G, JIN D. Classifying Malware Represented as Control Flow Graphs Using Deep Graph Convolutional Neural Network[C]// 2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN). 2019: 52-63.

[6] WU B, XU Y, ZOU F. Malware Classification by Learning Semantic and Structural Features of Control Flow Graphs[C]// 2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications. 2021: 540-547.

[7] YING Z, BOURGEOIS D, YOU J, et al. Gnnexplainer: Generating Explanations for Graph Neural Networks[C]// Advances in Neural Information Processing Systems. 2019: 9240-9251.

[8] YUAN H, YU H, WANG J, et al. On Explainability of Graph Neural Networks Via Subgraph Explorations[C]// International Conference on Machine Learning. 2021: 12241-12252.

[9] HERATH J D, WAKODIKAR P P, YANG P, et al. Cfgexplainer: Explaining Graph Neural Network-Based Malware Classification From Control Flow Graphs[C]// 2022 52nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN). 2022: 172-184.

[10] LUO D, CHENG W, XU D, et al. Parameterized Explainer for Graph Neural Network[C]// Advances in Neural Information Processing Systems. 2020: 19620-19631.

[11] ZENG A, LIU X, DU Z, et al. Glm-130B: An Open Bilingual Pre-Trained Model[C]// Proceedings of the International Conference on Learning Representations. 2023: 1-56.

[12] Openai. Chatgpt: A Language Model for Conversational AI[EB/OL]. https://chatgpt.com/.

[13] LIU H, LI C, WU Q, et al. Visual Instruction Tuning[C]// Advances in Neural Information Processing Systems. 2024: 34892-34916.

[14] ZHU D, CHEN J, SHEN X, et al. Minigpt-4: Enhancing Vision-Language Understanding with Advanced Large Language Models[C]// Proceedings of the International Conference on Learning Representations. 2024: 1-17.

[15] YE Q, XU H, XU G, et al. Mplug-Owl: Modularization Empowers Large Language Models with Multimodality[J]. arXiv: 2304.14178, 2023.

[16] WEN Z, FANG Y. Prompt Tuning On Graph-Augmented Low-Resource Text Classification[J]. IEEE Transactions on Knowledge and Data Engineering, 2024, 36(12): 9080-9095.

[17] ZHANG H, LI X, BING L. Video-Llama: An Instruction-Tuned Audio-Visual Language Model for Video Understanding[C]// Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations. 2023: 543-553.

[18] TANG J, YANG Y, WEI W, et al. Graphgpt: Graph Instruction Tuning for Large Language Models[C]// Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2024: 491-500.

[19] WANG Y, KORDI Y, MISHRA S, et al. Self-Instruct: Aligning Language Models with Self-Generated Instructions[C]// Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics. 2023: 13484-13508.

[20] VASWANI A. Attention is All You Need[C]// Advances in Neural Information Processing Systems. 2017: 5998-6008.

[21] RADFORD A, KIM J W, HALLACY C, et al. Learning Transferable Visual Models From Natural Language Supervision[C]// International Conference on Machine Learning. 2021: 8748-8763.

[22] WEN Z, FANG Y. Augmenting Low-Resource Text Classification with Graph-Grounded Pre-Training and Prompting[C]// Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2023: 506-516.

[23] LIU S, NIE W, WANG C, et al. Multi-Modal Molecule Structure-Text Model for Text-Based Retrieval and Editing[J]. Nature Machine Intelligence, 2023, 5(12): 1447-1457.

[24] LU Y, PENG J, ZHU Y, et al. Pre-Training Molecular Graph Representations with Motif-Enhanced Message Passing[C]// 2024 International Joint Conference on Neural Networks (IJC-

NN). 2024:1-8.
[25] OORD A V D, LI Y, VINYALS O. Representation Learning with Contrastive Predictive Coding[J]. arXiv: 1807. 03748, 2018.
[26] HU E J, SHEN Y, WALLIS P, et al. Lora: Low-Rank Adaptation of Large Language Models[C]// Proceedings of the International Conference on Learning Representations. 2022:1-13.
[27] HAMILTON W, YING Z, LESKOVEC J. Inductive Representation Learning On Large Graphs[C]// Advances in Neural Information Processing Systems. 2017:1024-1034.
[28] KIPF T N, WELLING M. Semi-Supervised Classification with Graph Convolutional Networks[C]// Proceedings of the International Conference on Learning Representations. 2017:1-14.
[29] VELIČKOVIĆ P, CUCURULL G, CASANOVA A, et al. Graph Attention Networks[C]// Proceedings of the International Conference on Learning Representations. 2018:1-12.
[30] VELIČKOVIĆ P, FEDUS W, HAMILTON W L, et al. Deep Graph Infomax[C]// Proceedings of the International Conference on Learning Representations. 2018:1-17.
[31] ZHANG S, LIU Y, SUN Y, et al. Graph-Less Neural Networks: Teaching Old Mlps New Tricks Via Distillation[C]// Proceedings of the International Conference on Learning Representations. 2022:1-21.
[32] WEI J, WANG X, SCHUURMANS D, et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models[J]. Advances in Neural Information Processing Systems, 2022, 35: 24824-24837.
[33] LIU H, LI C, LI Y, et al. Improved Baselines with Visual Instruction Tuning[C]// Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2024:26296-26306.

ZHOU Yuchen, born in 2000, postgraduate. His main research interests include malicious code analysis and multimodal large language models.

LI Peng, born in 1979, Ph.D, professor, Ph.D supervisor, is a member of CCF (No. 48573M). His main research interests include computer communication networks, cloud computing and information security.

(责任编辑:柯颖)