ICS 35.040
CCS L 71

团　体　标　准

T/AI 115.3—2024

信息技术 神经网络表示与模型压缩 第3部分：图神经网络

Information technology - Neural network representation and model compression — Part 3: Graph neural network

2024－12－30 发布　　2024－12－30 实施

中关村视听产业技术创新联盟　发布

目　　次

前　言

本文件按照GB/T 1.1—2020《标准化工作导则 第1部分：标准化文件的结构和起草规则》的规定起草。

本文件是T/AI 115《信息技术　神经网络表示与模型压缩》的第2部分。T/AI 115已经发布了以下部分：

——第1部分：卷积神经网络；

——第3部分：图神经网络。

本文件由新一代人工智能产业技术创新战略联盟AI标准工作组提出。

本文件由中关村视听产业技术创新联盟归口。

本文件起草单位：北京大学、北京邮电大学、华为技术有限公司、蚂蚁金服（杭州）网络技术有限公司、北京百度网讯科技有限公司、鹏城实验室、吉林大学、西安交通大学、上海图灵天问智能科技有限公司、中国电子技术标准化研究院、阿里巴巴（中国）有限公司、北京海致星图科技有限公司、北京图数创智科技有限公司、杭州海康威视数字技术股份有限公司、上海上湖信息技术有限公司、中国科学院自动化研究所、中科南京人工智能创新研究院、中移动信息技术有限公司、东北大学、深圳市腾讯计算机系统有限公司、北京交通大学、清华大学、中国联合网络通信有限公司、腾讯科技(深圳)有限公司。

本文件主要起草人：石川、田永鸿、黄海、任宇翔、张志强、冯仕堃、陈光耀、杨博、裴红斌、刘忠雨、范深、沈游人、杨成、岳大帅、刘曜齐、耿赛赛、陈均泽、郝健翔、杨晋豫、文思源、张学仓、崔佳旭、王平辉、李慧园、陈泰乐、曾菊香、何径舟、黄正杰、黄世维、方泽阳、陈泽裕、于佃海、李伟彬、周俊、胡斌斌、韦绍玮、梁磊、敬斌、张翀、倪铭坚、彭佩玺、张文涛、盛则昂、黎洋、沈彧、杨智、崔斌、孙铭蔚、王鹏、刘洪、杨帆、邹磊、吴伟、周光煜、方超、陈诚、姜伟浩、王扬、王春平、陈磊、张一帆、赵士云、程健、陈卓、尚晶、李珍、张岩峰、蒋杰、陶阳宇、欧阳文、孙瑞鸿、耿阳李敖、李清勇、王鑫、朱文武、胡浩基、王化良、胡欣怡、杨雨泽、鲍薇、郑若琳、沈芷月、杨斌、程新洲、张伟民、赵海英、黄铁军、高文。

引　言

图神经网络已经在推荐系统、知识图谱、智能交通等领域得到广泛应用。基于图神经网络的深度学习算法及其衍生应用系统在许多图任务上取得了突破性的进展，但是其具体的实现仍然依赖于不同的算法平台。由于目前没有统一的图神经网络的表示标准，不同的算法平台采用了不同的图神经网络的表示和存储标准；不同图神经网络框架之间不能互操作与协同工作；图数据和图神经网络模型不能在不同平台上直接迁移使用；不同图神经网络框架对于图神经网络粒度定义不同，妨碍了硬件厂商对于图神经网络的加速和优化；对于新出现的算子，框架都需要进行重定义。本文件旨在提供图数据和图神经网络统一的表示，以及模型算子接口的统一规范参考，提升用户对图神经网络模型的复用效果。对于本标准规定的表示方法不要求平台原生支持，可以通过转换、工具包等形式进行支持。本文件的定义可转化为与特定计算设备、框架匹配的形式和实现。T/AI 115旨在确立适用于不同种类神经网络的表示方法与模型压缩的规范，拟由三个部分组成：

——第 1 部分：卷积神经网络。目的在于确立适用千卷积神经网络的表示与模型压缩标准。

——第 2 部分：大规模预训练模型。目的在于确立适应多种推理平台和计算要求的大规模预训练模型的基本表示方法与加速压缩过程。

——第3部分：图神经网络。目的在于确立适应多种计算要求的高效图神经网络模型的基本表示方法与压缩加速过程。

本文件的发布机构提请注意，声明符合本文件时，可能涉及到 7.2.1.5 与《一种基于移位图卷积神经网络骨骼点行为识别系统及其识别方法》（专利号：202010419839.4）、《一种自适应时序移位神经网络时序行为识别方法》（专利号：202010419814.4）、《异质图神经网络生成方法、装置、电子设备及存储介质》（专利号：201910349275.9）；7.3.2 与《一种用于短文本的分类方法及装置》（专利号：201910945503.9）、《基于多通路图卷积神经网络的对象分类方法及装置》（专利号：202010555093.X）、《一种基于图卷积网络模型的分类方法及装置》（专利号：202011604370.8）；7.3.3 与《一种信息预测方法、装置、存储介质及计算机设备》（专利号：202110552598.5）、《一种套现用户检测方法、装置及设备》（专利号：201910052269.7）、《一种基于异质图神经网络的节点处理方法、装置及设备》（专利号：202010750181.5）、《基于统一优化目标框架图神经网络的数据分类方法及装置》《专利号：202110023447.0》；7.4.2 与《一种基于异质信息网络表示的推荐方法及装置》（专利号：201711239629.1）、《一种基于元路径引导嵌入的查询推荐方法及装置》（专利号：201910342766.0）；7.4.4 与《图节点关系表征生成和图节点业务关系预测方法及装置》（专利号：202111003074.7）；9.2.2 与《一种图数据采样方法和系统》（专利号：202011038681.2）相关的专利的使用。

本文件的发布机构对于该专利的真实性、有效性和范围无任何立场。

该专利持有人已向本文件的发布机构承诺，他愿意同任何申请人在合理且无歧视的条款和条件下，就专利授权许可进行谈判。该专利持有人的声明已在本文件的发布机构备案，相关信息可以通过以下联系方式获得：

专利持有人：北京大学、北京邮电大学、蚂蚁集团、深圳市腾讯计算机系统有限公司、中国科学院自动化研究所、中科南京人工智能创新研究院

地址：北京市海淀区颐和园路 5 号、北京市海淀区西土城路 10 号、浙江省杭州市西湖区西溪路 569 号蚂蚁 A 空间、广东省深圳市南山区腾讯滨海大厦、北京市海淀区中关村东路 95 号、江苏省南京市创研路 266 号麒麟人工智能产业园 3 号楼 3 楼

联 系 人：黄铁军

通讯地址：北京大学理科 2 号楼 2641 室

邮政编码：100871

电子邮件：tjhuang@pku.edu.cn

电话：+8610-62756172

传真：+8610-62751638
网址：http://www.aitisa.org.cn

请注意除上述专利外，本文件的某些内容仍可能涉及专利。本文件的发布机构不承担识别这些专利的责任。

信息技术 神经网络表示与模型压缩 第3部分：图神经网络

1 范围

本文件规定了适应多种计算要求的高效图神经网络模型的基本表示方法与压缩加速过程。

本文件适用于各种图神经网络模型的研制、开发、测试评估过程，以及在端云领域的高效应用。

注：对于本文件规定的表示与模型压缩方法不要求机器学习框架原生支持，可以通过转换、工具包等形式支持。

2 规范性引用文件

下列文件中的内容通过文中的规范性引用而构成本文件必不可少的条款。其中，注日期的引用文件，仅注日期对应的版本适用于本文件；不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

GB/T 5271.34-2006 信息技术 词汇 第34部分：人工智能 神经网络

GB/T 42382.1-2023 信息技术 神经网络表示与模型压缩 第1部分：卷积神经网络

3 术语和定义

下列术语和定义适用于本文件。

3.1

图神经网络 graph neural network

用于处理由图数据结构表示数据的神经网络。

注：是人工智能应用中代表性的深度学习算法。

3.2

模型 model

对应完成单个或多个任务的神经网络。

3.3

层 layer

神经网络中的分级结构。

注：每个网络层包含多个算子，例如输入层，卷积层，全连接层。

［来源：GB/T 42382.1-2023，3.2］

3.4

权重 weight

权重张量 weight tensor

不同单元之间的连接强度。

注：权重是神经网络的固有参数之一。

［来源：GB/T 42382.1-2023，3.21，有修改］

3.5

图 graph

一个三元组，包含节点集、边集、关联函数。

3.6

节点 node
图结构数据中的节点。

3.7

边 edge
图结构数据中的边。

3.8

特征 feature
数据集、神经网络计算过程中，数据的属性向量。

3.9

图基本运算 graph fundamental operation
对图结构信息的基本运算，例如求解节点的度等。

3.10

图基本任务 graph fundamental task
基于图数据实现某种功能，例如节点分类、链路预测、图分类任务等。

3.11

基础算子 basic operator
应用在图神经网络模型上的基本运算操作定义。

3.12

点级模型 node-level task model
应用于节点级任务的图神经网络模型。

3.13

边级模型 edge-level task model
应用于边级任务的图神经网络模型。

3.14

图级模型 graph-level task model
应用于图级任务的图神经网络模型。

3.15

模型加速 model acceleration
降低神经网络模型推理时间，提高图神经网络模型运行和传输效率的方法。

3.16

模型压缩 model compression
减小神经网络模型规模，提高神经网络模型运行和传输效率的方法。

3.17

采样 sampling

采样是指用部分图数据或图特征来代替完整图数据或图特征的方法。

3.18

量化 quantization
将输入值从一个大集合映射到一个较小集合的过程。

3.19

剪枝 pruning
将深度学习模型去掉一些影响性能较小的单元的方法。

3.20

结构化矩阵 structured matrix
一类特殊的矩阵，可以通过使用较少的数据以及一定的排列规律，构成完整的矩阵。
［来源：GB/T 42382.1-2023，3.14］

3.21

分块结构化矩阵 block structured matrix
指可以分为多个块，且每个分块均按照某种规律排列的矩阵。
［来源：GB/T 42382.1-2023，3.6］

3.22

随机游走 random walk
遍历图中节点的方式，通过随机选择节点的相邻节点来生成节点序列，以捕捉图信息并用于图的表示学习。

3.23

点采样器 node sampler
根据起始节点，对输入的图数据进行采样的采样器。

3.24

层采样器 layer sampler
根据起始节点，对输入的图数据进行层级采样的采样器。

3.25

子图采样器 subgraph sampler
根据输入图数据进行子图采样的采样器。

4 缩略语

下列缩略语适用于本文件。
MLP：多层感知机（Multilayer Perceptron）
GNN：图神经网络（Graph Neural Network）
GCN：图卷积网络（Graph Convolutional Networks）
GAT：图注意力网络（Graph Attention Networks）
GraphSAGE：图采样与图聚合（Graph Sample and Aggregation）
GIN：图同构网络（Graph Isomorphism Network）
GGNN：门控图神经网络（Gated Graph Neural Network）

LSTM：长短期记忆网络（Long Short Term Memory）
ResGatedgraph：残差门控图神经网络（Residual Gated Graph Neural Network）
GINE：带边特征的图同构网络（Graph Isomorphism Network with Edge features）
GaAN：图注意力网络（Graph Attention Network）
AM-GCN：自适应多通道图卷积网络（Adaptive Multi-channel Graph Convolutional Network）
FAGCN：频率自适应图卷积网络（Frequency Adaptation Graph Convolutional Networks）
GeomGCN：几何图卷积网络（Geometric Graph Convolutional Networks）
HGAT：异质图注意力网络（Heterogeneous Graph Attention Network）
GCNII：简单深度图卷积网络（Simple and Deep Graph Convolutional Networks）
SGC：简化图卷积网络（Simple Graph Convolution）
APPNP：近似个性化神经预测传播（Approximate Personalized Propagation of Neural Predictions）
GPRGNN：广义网页排名（以下简称“PageRank”）图神经网络（Generalized PageRank Graph Neural Network）
ChebNet：切比雪夫谱图卷积网络（Chebyshev Spectral Graph Convolutional Network）
JKnet：跳跃知识网络（Jumping Knowledge Networks）
DCNN：深度卷积神经网络（Deep Convolutional Neural Networks）
Line：大规模信息网络嵌入（Large-scale Information Network Embedding）
HPN：异质图传播网络（Heterogeneous Graph Propagation Network）
GAMLP：图注意力多层感知器（Graph Attention Multi-Layer Perceptron）
GNN-LF/HF：低频/高频图神经网络（Graph Neural Network with Low Frequency/High Frequency）
HeCo：具有协同对比学习的自监督异质图神经网络（Self-Supervised Heterogeneous Graph Neural Network with Co-Contrastive Learning）
HACUD：基于属性异质信息网络和分层注意机制的套现用户检测模型（Cash-Out User Detection based on Attributed Heterogeneous Information Network with a Hierarchical Attention Mechanism）
GTN：图转换器（以下简称“transformer”）网络（Graph Transformer Network）
SAT：简单实例自适应自训练半监督文本分类（Improving Semi-Supervised Text Classification with Simple Instance-Adaptive Self-Training）
GraphGPS：基于图的渐进式传播与搜索（Graph-based Progressive Propagation and Searching）
GloGNN：全局节点信息图神经网络（Graph Neural Network with Global Information）
R-GCN：关系图卷积网络（Relational graph convolutional network）
HIN：异质信息网络（Heterogeneous Information Network）
BERT：双向编码器表征法（Bidirectional Encoder Representations from Transformers）
MCCF：多组件图卷积协同过滤模型（Multi-Component Graph Convolutional Collaborative Filtering）
MEIRec：用于意图推荐的元路径引导嵌入方法（Metapath-guided Embedding method for Intent Recommendation）
HERec：基于异质网络嵌入的推荐方法（Heterogeneous Information Network Embedding for Recommendation）
SEAL：从子图、嵌入和属性学习进行链接预测（Learning from Subgraphs, Embeddings and Attributes for Link Prediction）
GraIL：图归纳学习（Graph Inductive Learning）
NBFNet：神经贝尔曼-福特网络（Neural Bellman-Ford Network）
PAGNN：路径感知图神经网络（Path-aware Graph Neural Network）
Cluster-GCN：聚类图卷积网络（Cluster-Graph Convolution Network）
VQ-GNN：矢量量化图神经网络（Vector Quantization-Graph Neural Network）
LADIES：层相关重要性采样（Layer-Dependent Importance Sampling）
HGT：异质图转换器（Heterogeneous Graph Transformer）
GraphSaint：基于图采样的归纳学习（Graph Sampling Based Inductive Learning）
EXACT：极限激活压缩（Scalable Graph Neural Networks Training via Extreme Activation Compression）
SGC：简单图卷积网络（Simplifying Graph Convolutional Networks）

PPNP：基于个性化 PageRank 算法的图神经网络预测与传播（Predict then Propagate: Graph Neural Networks meet Personalized PageRank）

LSP：局部敏感剪枝（Locality-Sensitive Pruning）

LTD：元蒸馏（Learning to Distill）

GRACED：基于自定义知识蒸馏的图增强多层感知机（Graph Augmented MLPs via Customized Knowledge Distillation）

ROD：接收感知在线蒸馏（Reception-aware Online Distillation for Sparse Graphs）

GNN-SD：图神经网络自蒸馏（GNN Self-Distillation）

IGSD：迭代图自蒸馏（Iterative Graph Self-Distillation）

Node2Vec：一种学习网络中节点连续特征表示的算法框架（Node to Vector）

DiffPool：可微图池化（differentiable graph pooling）

MPSN：动作感知伪暹罗网络（Motion-aware Pseudo Siamese Network）

MPNN：消息传递图神经网络（Message Passing Neural Networks）

D-MPNN：有向消息传递神经网络（Directed Message Passing Neural Networks）

DmoN：深度模块化网络（Deep Modularity Networks）

GMN：图匹配网络（Graph Matching Networks）

FGNN：因子图神经网络（Factor Graph Neural Network）

molGAN：分子生成对抗网络（Molecular Generative Adversarial Network）

GRAN：图循环注意网络（Graph Recurrent Attention Networks）

BNS：边界节点采样（Boundary Node Sampling）

SSP：失效同步并行（stale synchronous parallel）

5 图神经网络表示与模型压缩概述

本标准为图神经网络表示和模型压缩提供了统一参考规范，其内容总体架构如图1所示。该架构规范化了图神经网络的表示格式，图神经网络压缩和加速方法，以及图神经网络计算框架，有利于提高图神经网络模型在各类设备上的开发与运行效率。

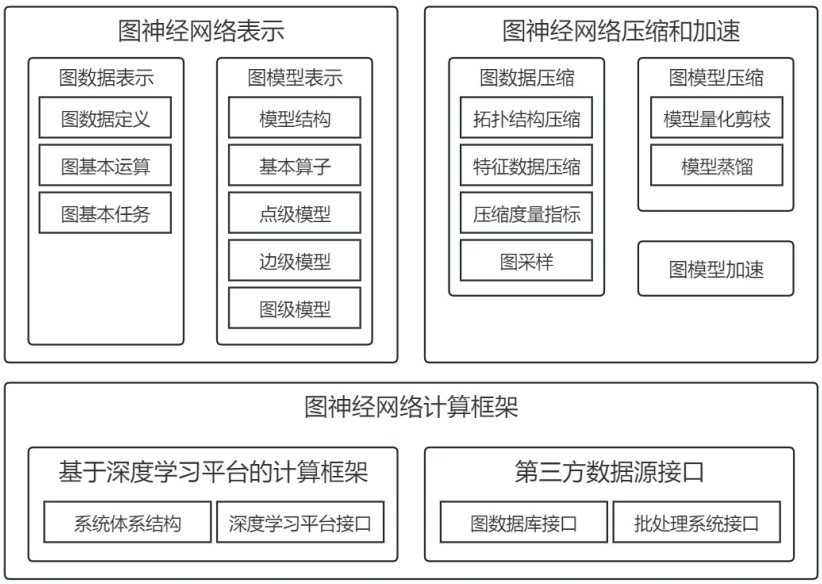

图 1 图神经网络表示与模型压缩总体架构

总体架构包括以下三个部分。

图神经网络表示：规范化了图神经网络的表示方法，包括图数据的表示方法和图神经网络模型的表示方法。图数据表示中涵盖基本数据类型定义、图数据类型定义、图基本运算定义以及图基本任务定义。图神经网络模型表示中涵盖模型结构和基础算子定义，并从任务划分的角度规范了图神经网络点级模型、边级模型和图级模型的算子接口。本部分具体对应标准的第6、第7章节。

图神经网络压缩和加速：规范了图神经网络压缩方法和模型加速策略及其算子接口。压缩方法包括图数据的压缩以及图模型的压缩两个部分。图数据的压缩包括图拓扑结构压缩、图特征数据压缩以及图采样。图模型的压缩包括模型量化剪枝和模型蒸馏。对于图数据和模型的压缩方法，均定义了用于度量性能的指标。图模型加速包括并行加速策略、迭代加速策略、图划分策略以及通信加速策略。本部分具体对应标准的第8章节。

图神经网络计算框架：规范了基于深度学习平台的图神经网络计算框架，以及该计算框架与第三方数据源的接口定义。基于深度学习平台的图神经网络计算框架定义了系统逻辑体系结构，以及与深度学习平台的接口规范。与第三方数据源接口定义包括图数据库接口和批处理系统接口。本部分具体对应标准的第9章节。

6 图数据表示

6.1 基本定义

6.1.1 概述

本标准定义了数据结构以及各类基本数据类型。

6.1.2 数据结构定义

本标准定义消息（message）为数据结构的基本单元，字段（field）为构成消息的基本元素，如图 2 所示。

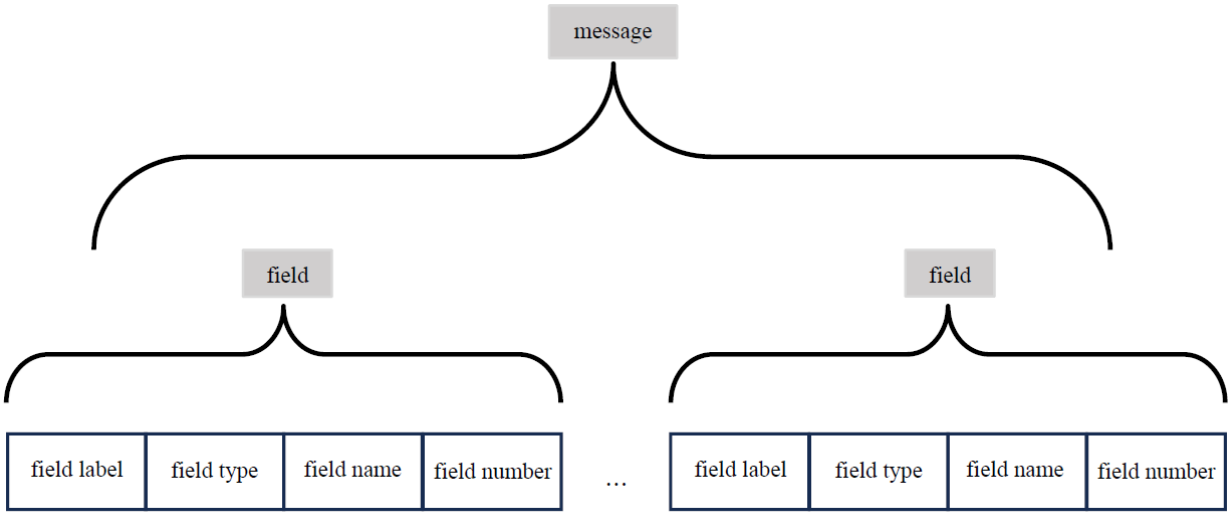

图 2 数据结构定义规范

消息由一系列具有特定标签（field label）、类型（field type）、名称（field name）和标号（field number）的字段组成。可使用消息本身作为字段的类型，以实现数据结构的嵌套。

a) 字段标签（field label）：字段标签用于定义字段的属性和使用规则。默认情况下，所有字段均被视为可选（"optional"），无需显式声明。此做法旨在简化编程语言的使用，并确保向后兼容

性。当字段需要表示多个值时，应使用"repeated"关键字进行标注，以明确该字段可以包含一个以上的值。

b) 字段类型（field type）：字段类型定义了字段可以存储的数据类型。该类型包括 int32、float、double、bool 和 string 等标准数据类型，以及枚举（enum）、映射（map）、消息（message）等复合类型。

c) 字段名称（field name）：字段名称是在消息定义中为字段指定的唯一名称，用于标识和访问消息的具体部分。

d) 字段编号（field number）：字段编号是一个正整数，用于在消息的二进制格式中唯一标识字段。该编号是消息序列化和反序列化过程中的关键，确保了即使字段名称在不同语言间有所不同，数据的结构仍能被正确理解。

6.1.3 基本数据类型定义

6.1.3.1 整数定义

整数(int)定义见表 1。

表 1 整数定义

字段	类型	定义
value	One of {uint32, uint64, int32, int64, sint32, sint64, fixed32, fixed64, sfixed32, sfixed64}	一组可选的整数类型，确保在不同情况下使用适当的数值范围和编码效率

6.1.3.2 浮点数定义

浮点数（Float）定义见表 2。

表 2 浮点数定义

字段	类型	定义
value	One of {float, double}	一组可选的浮点数类型，确保在不同情况下使用适当的数值范围和编码效率

6.1.3.3 元组定义

元组（Tuple）定义见表 3。

表 3 元组定义

字段	类型	定义
element-1	DataType	元组的第一个元素
element-2	DataType	元组的第二个元素
…	…	…
element-n	DataType	元组的第 n 个元素

6.1.3.4 列表定义

列表（List）定义见表 4。

表 4　列表定义

字段	类型	定义
elements	DataType [repeated]	列表的各元素

6.1.3.5　字典定义

字典（Dict）定义见表 5。

表 5　字典定义

字段	类型	定义
entries	map<DataType, DataType>	字典的键值对 <Key, value> 映射，其中浮点数、消息类型和枚举类型不被允许作为键出现。

6.1.3.6　张量定义

张量（tensor）定义见表 6。

表 6　张量定义

字段	类型	定义
dtype	DataType	张量的数据类型,通常是数值类型，例如 int、float、double 等
shape	int64 [repeated]	张量的形状
data	DataType [repeated]	张量存储的数据
require_grad	bool	张量是否需要计算梯度
device	string	张量位于的设备

6.1.3.7　稀疏张量定义

稀疏（SparseTensor）张量定义见表 7。

表 7　稀疏张量定义

字段	类型	定义
dtype	DataType	稀疏张量的数据类型,通常是数值类型，例如 int、float、double 等
dense_shape	int64 [repeated]	稀疏张量稠密表示下的形状
format	enum {"coo", "csc", "csr"}	稀疏张量的存储格式
data	DataType [repeated]	稀疏张量非零元素的数据
index_format	oneof {COOIndex, CSCIndex, CSRIndex}	稀疏张量索引的格式
require_grad	bool	稀疏张量是否需要计算梯度
device	string	稀疏张量位于的设备

6.1.3.8　坐标格式索引

坐标格式索引（COOIndex）定义见表 8。

表 8　坐标格式索引

字段	类型	定义
row_indices	int64 [repeated]	COO 格式的行索引
col_indices	int64 [repeated]	COO 格式的列索引

6.1.3.9　压缩稀疏行索引

压缩稀疏行索引（CSRIndex）定义见表 9。

表 9　压缩稀疏行索引定义

字段	类型	定义
rowptr	int64 [repeated]	CSR 格式的行指针
col_indices	int64 [repeated]	CSR 格式的列索引

6.1.3.10　压缩稀疏列索引

压缩稀疏列索引（CSCIndex）定义见表 10。

表 10　压缩稀疏列索列定义

字段	类型	定义
row_indices	int64 [repeated]	CSC 格式的行索引
colptr	int64 [repeated]	CSC 格式的列指针

6.2　图数据类型定义

6.2.1　同质图定义

同质图（Graph）定义见表 11。

表 11　同质图定义

字段	类型	定义
X	tensor	可选，节点特征矩阵
edge_index	oneof {tensor, SparseTensor}	可选，边索引
edge_weight	tensor	可选，边特征矩阵
Y	tensor	可选，节点或图标签
pos	tensor	可选，节点位置

6.2.2　异质图定义

异质图（HeteroGraph）定义见表 12。

表 12　异质图定义

字段	类型	定义
X_dict	Dict [string, tensor]	可选，节点特征矩阵字典
edge_index_dict	oneof {Dict [string, tensor], Dict [string, SparseTensor]}	可选，边索引字典
edge_weight_dict	Dict [string, tensor]	可选，边特征矩阵字典
Y_dict	Dict [string, tensor]	可选，节点或图标签字典
pos_dict	Dict [string, tensor]	可选，节点位置字典

6.2.3 批量图定义

批量图（BatchGraph）定义见表 13。

表 13 批量图定义

字段	类型	定义
X	tensor	可选，节点特征矩阵
edge_index	oneof {tensor, SparseTensor}	可选，边索引
edge_weight	tensor	可选，边特征矩阵
Y	tensor	可选，图的标签
pos	tensor	可选，节点位置矩阵
batch	tensor	可选，指示每个节点属于哪个图
ptr	tensor	可选，用于指示每个图在批量图中的范围，便于还原原图

6.2.4 动态图定义

动态图（DynamicGraph）定义见表 14。

表 14 动态图定义

字段	类型	定义
src	tensor	可选，源节点列表
dst	tensor	可选，目标节点列表
t	tensor	可选，事件时间戳列表
msg	tensor	可选，消息特征矩阵

6.2.5 坐标图定义

坐标图（CoordinateGraph）定义见表 15。

表 15 坐标图定义

字段	类型	定义
R	tensor	可选，节点坐标
X	tensor	可选，节点特征
X_coord	tensor	可选，节点矢量特征
edge_index	oneof {tensor, SparseTensor}	可选，边索引
edge_attr	tensor	可选，边特征矩阵
Y	tensor	可选，节点或图标签
Y_coord	tensor	可选，节点矢量标签
dis_coord	tensor	可选，坐标系中距离类型

6.3 图基本运算

图基本运算的基本运算定义见表 16~表 48。
degree 运算操作定义见表 16。

表 16　degree 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
degree	计算给定的一维索引张量对应图的（未加权）度	Input	index	索引张量	tensor
			dtype	可选，返回张量的数据类型	string
			edge_type	可选，边的类型	string
			num_nodes	可选，节点的数量	int
		Output	Y	输出张量	tensor

dropEdge 运算操作定义见表 17。

表 17　dropEdge 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
dropEdge	根据伯努利分布，以概率 p 从图中随机丢弃边	Input	edge_index	边索引	tensor SparseTensor
			edge_weight	可选，边权重矩阵或者多维边特征矩阵	tensor
			p	可选，丢弃概率	float
			force_undirected	可选，如果设置为 True，则将丢弃或保留无向边的两个边。	bool
			num_nodes	可选，节点的数量	Tuple [int, int]
			training	可选，如果设置为 False，则该操作将无作用	bool
		Output	Y	输出张量	tensor

sort_edge_index 运算操作定义见表 18。

表 18　sort_edge_index 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
sort_edge_index	根据节点 ID 对边索引进行排序	Input	edge_index	边索引	tensor SparseTensor
			edge_weight	可选，边权重矩阵或者多维边特征矩阵	tensor
			num_nodes	可选，节点的数量	int
			sort_by_row	可选，如果设置为 True，则按源节点 ID 对边索引排序，否则将按目标节点 ID 排序	bool
		Output	Y	输出张量	tensor

add_self_loops 运算操作定义见表 19。

表 19　add_self_loops 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
add_self_loops	为每个节点添加自环，如果是加权图，则将添加自环并且根据 fill_value 添加边权重	Input	edge_index	边索引，和 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵或者多维边特征矩阵；如关键字包含 g，则删除该项	tensor
			g	输入图，和 edge_index 二选一	Graph
			fill_value	可选，在 edge_weight 不为空的情况下，该值会用来填充边权重	tensor float string
			num_nodes	可选，节点数量	int
			allow_duplicate	可选，如果设置为 False，则会先去除所有自环以避免重复的自环	bool
		Output	edge_index	输出边索引，和 g 二选一	tensor SparseTensor
			edge_weight	输出边权重，如输出 g，则删除该项	tensor
			g	输出图，和 edge_index 二选一	Graph

remove_self_loops 运算操作定义见表 20。

表 20　remove_self_loops 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
remove_self_loops	删除图中的每个自环	Input	edge_index	边索引，和 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵或者多维边特征矩阵；如关键字包含 g，则删除该项	tensor
			g	输入图，和 edge_index 二选一	Graph
		Output	edge_index	输出边索引，和 g 二选一	tensor SparseTensor

表 20　remove_self_loops 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
remove_self_loops	删除图中的每个自环	Output	edge_weight	输出边权重或者多维边特征矩阵，如输出 g，则删除该项	Tensor
			g	输出图，和 edge_index 二选一	Graph

segregate_self_loops 运算操作定义见表 21。

表 21　segregate_self_loops 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
segregate_self_loops	从图中分离出自环	Input	edge_index	边索引	tensor SparseTensor
			edge_weight	可选，边权重矩阵或者多维边特征矩阵	tensor
		Output	edge_index	无自环边索引	tensor SparseTensor
			edge_weight	无自环边特征矩阵	tensor
			loop_edge_index	自环边索引	tensor SparseTensor
			loop_edge_weight	自环边特征矩阵	tensor

add_remain_self_loops 运算操作定义见表 22。

表 22　add_remain_self_loops 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
add_remain_self_loops	将自环添加到没有自环的节点上，如果图被加权，则将添加自环的同时根据 fill_value 添加自环的权重	Input	edge_index	边索引	tensor SparseTensor
			edge_weight	可选，边权重矩阵或者多维边特征矩阵	tensor
			fill_value	可选，在 edge_weight 不为空的情况下，该值会用来填充边权重	float tensor string
			num_nodes	可选，节点数量	int
		Output	edge_index	边索引输出	tensor SparseTensor
			edge_weight	边权重矩阵或者多维边特征矩阵输出	tensor

contains_isolated_nodes 运算操作定义见表 23。

表 23　contains_isolated_nodes 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
contains_isolated_nodes	判断图中是否含有孤立的节点	Input	edge_index	边索引	tensor SparseTensor
			num_nodes	可选，节点数量	int
		Output	has	是否含有孤立的节点	bool

remove_isolated_nodes 运算操作定义见表 24。

表 24　remove_isolated_nodes 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
remove_isolated_nodes	删除图中孤立的节点	Input	edge_index	边索引	tensor SparseTensor
			edge_weight	可选，边权重或者边特征矩阵	tensor
			num_nodes	可选，节点数量	int
		Output	edge_index	邻接矩阵输出	tensor SparseTensor
			edge_weight	边权重或者边特征矩阵输出	tensor
			mask	孤立节点的掩码	tensor

subgraph 运算操作定义见表 25。

表 25　subgraph 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
subgraph	提取节点编号为 subset 的子图	Input	subset	保留的节点编号	tensor List [int]
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵或者多维边特征矩阵；如关键字包含 g，则删除该项	tensor
			g	输入图，和 edge_index 二选一	Graph

表 25　subgraph 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
subgraph	提取节点编号为 subset 的子图	Input	relabel_nodes	可选，如果设置为 True，则得到的子图节点编号将变为从零开始的连续索引	Bool
			num_nodes	可选，节点数量	int
			return_edge_mask	可选，如果设置为 True，将返回用于过滤多余的边特征的掩码	bool
		Output	edge_index	输出边索引，和 g 二选一	tensor SparseTensor
			edge_weight	输出边权重或者多维边特征矩阵，如输出 g，则删除该项	tensor
			g	输出图，和 edge_index 二选一	Graph
			edge_mask	可选，输出边特征的掩码，如输出 g，则删除该项	tensor

k_hop_subgraph 运算操作定义见表 26。

表 26　k_hop_subgraph 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
k_hop_subgraph	计算 k 跳子图	Input	node_idx	中心节点编号	tensor List [int] int
			num_hops	跳数	int
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			relabel_nodes	可选，如果设置为 True，则得到的子图节点编号将变为从零开始的连续索引	bool
			g	输入图，和 edge_index 二选一	Graph
			num_nodes	可选，节点数量	int

表 26　k_hop_subgraph 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
k_hop_subgraph	计算 k 跳子图	Input	flow	可选，k 跳聚合的流向，可选"source_to_target"或"target_to_source"	string
			directed	可选，若设置为 False，则会包括被采样到节点间所有的有向边	bool
		Output	subset	子图中涉及的节点编号，如输出 g，则删除该项	tensor
			edge_index	输出边索引，和 g 二选一	tensor SparseTensor
			mapping	从节点索引到其新位置的映射	tensor
			g	输出图，和 edge_index 二选一	Graph
			edge_mask	边掩码，指示边的保留情况，如输出 g，则删除该项	tensor

get_laplacian 运算操作定义见表 27。

表 27　get_laplacian 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
get_laplacian	计算出图的拉普拉斯矩阵	Input	edge_index	边索引	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			normalization	可选，拉普拉斯归一化方法	string
			num_nodes	可选，节点数量	int
		Output	edge_index	拉普拉斯矩阵对应的边索引	tensor SparseTensor
			edge_weight	拉普拉斯矩阵边权重矩阵输出	tensor

to_dense_batch 运算操作定义见表 28。

表 28　to_dense_batch 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
to_dense_batch	将批量图上的节点特征转换为各个子图上稠密的节点特征	Input	X	节点特征矩阵	tensor
			batch	可选，每一个节点所属批次	tensor
			fill_value	可选，填充输出节点缺失的特征	float
			max_num_nodes	可选，各个批次最大节点数目	Int
			batch_size	可选，批量大小	int
		Output	Y	稠密节点特征矩阵	tensor
			mask	节点掩码，指示哪些节点存在	tensor

to_dense_adj 运算操作定义见表 29。

表 29　to_dense_adj 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
to_dense_adj	将稀疏批量边索引张量转换为稠密张量	Input	edge_index	边索引	tensor SparseTensor
			batch	可选，批向量，将每个节点分配一个批量	tensor
			edge_weight	可选，边权重或者边特征矩阵	tensor
			max_num_nodes	可选，各个批次最大节点数目	int
			batch_size	可选，批量大小	int
		Output	adj	稠密的批量邻接矩阵输出	tensor

to_sparse 运算操作定义见表 30。

表 30　to_sparse 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
to_sparse	将稠密的批量邻接矩阵转换为稀疏表示	Input	adj	稠密邻接矩阵	tensor
			mask	可选，描述节点的保留情况	tensor
		Output	edge_index	稀疏边索引输出	tensor SparseTensor
			edge_weight	边权重或者边特征矩阵输出	tensor

normalized_cut 运算操作定义见表 31。

表 31　normalized_cut 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
normalized_cut	计算加权图的归一化分割	Input	edge_index	边索引	tensor SparseTensor
			edge_weight	边权重或者边特征矩阵	tensor
			num_nodes	可选，节点的数量	int
		Output	cut	图的归一化分割	tensor SparseTensor

grid 运算操作定义见表 32。

表 32　grid 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
grid	返回给定宽高的网格图的边索引以及节点位置矩阵	Input	height	网格图的高	int
			width	网格图的宽	int
		Output	edge_index	网格图边索引	tensor SparseTensor
			pos	网格图节点位置矩阵	tensor

softmax_nodes 运算操作定义见表 33。

表 33　softmax_nodes 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
softmax_nodes	对节点特征执行 softmax 计算	Input	g	输入图，和 X 二选一	Graph HeteroGraph
			X	输入的节点特征，和 graph 二选一	tensor
			node_type	可选，节点类型	string
		Output	Y	归一化后的节点特征	tensor

softmax_edges 运算操作定义见表 34。

表 34　softmax_edges 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
softmax_edges	对边特征执行 softmax 计算	Input	g	输入图，和 X 二选一	Graph HeteroGraph
			edge_weight	输入的边特征，和 graph 二选一	tensor
			edge_type	可选，边类型	string

表 34　softmax_edges 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
softmax_edges	对边特征执行 softmax 计算	Output	edge_weight	归一化后的边特征	Tensor

broadcast_nodes 运算操作定义见表 35。

表 35　broadcast_nodes 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
broadcast_nodes	将图级特征广播到节点级表示	Input	g	输入图	Graph HeteroGraph
			feature	输入的图级表示	tensor
			node_type	节点类型	string
		Output	Y	节点特征	tensor

broadcast_edges 运算操作定义见表 36。

表 36　broadcast_edges 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
broadcast_edges	将图级特征广播到边级表示	Input	g	输入图	Graph HeteroGraph
			feature	输入的图级表示	tensor
			edge_type	边类型	string
		Output	edge_weight	边特征	tensor

graph_partition 运算操作定义见表 37。

表 37　graph_partition 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
graph_partition	将图划分成若干分片	Input	g	输入图数据	Graph HeteroGraph
			num_part	分片个数	int
			part_method	分片策略	string
		Output	Y	输出图数据的列表	Dict [Graph] Dict [HeteroGraph]

fps 运算操作定义见表 38。

表 38　fps 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
fps	一种用于点云数据采样的运算符	Input	X	节点特征矩阵	tensor
			batch	可选，每一个样本点所属批次	tensor

表 38　fps 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
fps	一种用于点云数据采样的运算符	Input	ratio	可选，采样率	Float
			random_start	可选，是否采用第一个节点作为起始节点	bool
			batch_size	可选，批量大小	int
		Output	Y	输出张量	tensor

graclus 运算操作定义见表 39。

表 39　graclus 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
graclus	一种贪婪聚类运算符，该算法选择未标记的节点并将其与未标记的节点之一匹配	Input	edge_index	边索引，和 g 二选一	tensor SparseTensor
			g	输入图，和 edge_index 二选一	Graph
			relabel_idx	如果设置为 True，将节点编号重新映射成连续的编号	bool
			edge_weight	可选，边权重矩阵，如关键字包含 g，则删除该项	tensor
			num_nodes	可选，节点数目，如输出 g，则删除该项	int
		Output	Y	输出节点编号	tensor

knn 运算操作定义见表 40。

表 40　knn 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
knn	用于为 y 中的每个元素查找 x 中的 k 个最近邻的运算符	Input	X	测试样本点坐标的集合	tensor
			Y	训练样本点坐标的集合	tensor
			k	选取训练样本点的数目	int
			batch_x	可选，X 样本点的所属批次	tensor
			batch_y	可选，Y 样本点的所属批次	tensor
			cosine	可选，是否采用余弦距离	bool
			num_workers	可选，并发数目	int
			batch_size	可选，批量大小	int

表 40　knn 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
knn	用于为 y 中的每个元素查找 x 中的 k 个最近邻的运算符	Output	Z	输出张量，表示选出的最近邻	tensor

knn_graph 运算操作定义见表 41。

表 41　knn_graph 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
knn_graph	用于构建样本点满足 k 近邻的图	Input	X	点坐标集合	tensor
			k	近邻大小	int
			batch	可选，批量向量，用于将每个节点分配给一个特定样本	tensor
			dist	可选，采用的距离函数	string
			batch_size	可选，批量大小	int
			algorithm	可选，采用的算法	string
		Output	g	满足要求的 k 近邻图	Graph

nearest 运算操作定义见表 42。

表 42　nearest 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
nearest	用于为 y 中每个元素，在 x 中查找距离最近的元素的运算符	Input	X	测试样本点坐标的集合	tensor
			Y	训练样本点坐标的集合	tensor
			batch_x	可选，X 样本点的所属批次	tensor
			batch_y	可选，Y 样本点的所属批次	tensor
		Output	Z	输出张量，表示聚类结果	tensor

radius 运算操作定义见表 43。

表 43　radius 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
radius	用于为 y 中的每个元素找到 x 中距离不超过 r 的所有点的搜索运算符	Input	X	测试样本点坐标的集合	tensor
			Y	训练样本点坐标的集合	tensor

表 43　radius 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
radius	用于为 y 中的每个元素找到 x 中距离不超过 r 的所有点的搜索运算符	Input	r	距离半径	Float
			batch_x	可选，X 样本点的所属批次	tensor
			batch_y	可选，Y 样本点的所属批次	tensor
			max_num_neighbors	可选，y 中的每个元素返回的最大邻居数	int
			num_workers	可选，并发数目	int
			batch_size	可选，批量大小	int
		Output	Z	输出张量，表示半径内搜索结果	tensor

radius_graph 运算操作定义见表 44。

表 44　radius_graph 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
radius_graph	构建距离在一定半径内的图	Input	X	点坐标集合	tensor
			r	距离大小	float
			p	可选，距离计算范数	float
			self_loop	可选，构建的图是否包含自环	bool
			compute_mode	可选，计算模式	string
			get_distances	可选，是否返回构建的图对应边的距离	bool
		Output	g	满足距离半径在 r 之内的图	Graph

voxel_grid 运算操作定义见表 45。

表 45　voxel_grid 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
voxel_grid	利用在点云上覆盖规则网格并进行聚类的体素网格池化运算符	Input	pos	输入张量，节点位置矩阵	tensor
			batch	每一个样本点所属批次	tensor
			size	每个体素（voxel）的尺寸	float, tensor List [float]
			start	可选，每个维度中网格的起始坐标	float, tensor List [float]

表 45　voxel_grid 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
voxel_grid	利用在点云上覆盖规则网格并进行聚类的体素网格池化运算符	Input	end	可选，每个维度中网格的结束坐标	float, tensor List [float]
		Output	Y	输出张量	tensor

dropNode 运算操作定义见表 46。

表 46　dropNode 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
dropNode	使用来自伯努利分布的样本，以概率 p 从图节点中随机丢弃某些节点出发的所有边	Input	edge_index	边索引	SparseTensor
			X	节点特征	tensor
			p	可选，丢弃概率	float
			drop_num_nodes	可选，丢弃的节点个数	int
		Output	edge_index	边索引输出	SparseTensor
			Y	节点特征输出	tensor

dropFeature 运算操作定义见表 47。

表 47　dropFeature 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
dropFeature	使用来自伯努利分布的样本，以概率 p 从图节点中随机丢弃所有节点某些维度的特征元素	Input	edge_index	边索引	SparseTensor
			X	节点特征	tensor
			p	可选，丢弃概率	float
			drop_num_elements	可选，丢弃节点特征元素的列数	int
		Output	edge_index	边索引输出	SparseTensor
			Y	节点特征输出	tensor

dropMessage 运算操作定义见表 48。

表 48　dropMessage 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
dropMessage	使用来自伯努利分布的样本，以概率 p 从图节点中随机丢弃节点发出消息中的元素	Input	edge_index	边索引	SparseTensor
			X	节点特征	tensor
			edge_attr	可选，边权重矩阵或者多维边特征矩阵	tensor
			p	可选，丢弃概率	float
			drop_num	可选，丢弃节点发出的消息元素的个数	int
		Output	edge_index	边索引输出	SparseTensor
			Y	节点特征输出	tensor

dropGraph 运算操作定义见表 49。

表 49　dropGraph 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
dropGraph	使用来自伯努利分布的样本，以概率 p 从图节点中随机丢弃某些节点及其 Q 阶邻居节点，并引入注意力机制，让被网络关注的区域具有更大的被丢弃的概率	Input	Edge_index	边索引	SparseTensor
			X	节点特征	tensor
			Keep_prob	保持节点不被丢弃的概率	float
			Q	邻居节点范围	int
		Output	Y	节点特征输出	tensor

6.4　图基本任务

图神经网络的下游任务按任务形式与监督形式进行划分。

a)　按任务形式划分

按任务形式，图神经网络的下游任务划分为节点级别任务、边级别任务和图级别任务。

节点级别任务分为：节点分类、节点回归、节点聚类等。节点分类是指对于给定的图中的节点，将其划分为某类别或标签；节点回归是指对给定图中的节点进行数值预测，用于预测节点的属性值；节点聚类是指将图中的节点划分为不同的群组或簇。

边级别任务分为：边分类、边回归、链接预测等。边分类是指对于给定的图中的边，将其划分为某类别或标签；边回归是指对给定图中的边进行数值预测，以预测边上的属性值或边的权重等信息；链接预测是指预测图中尚未出现但潜在存在的边。

图级别分为：图分类、图回归、图匹配、图生成等。图分类是指将给定的图划分为某类别或标签；图回归是指预测图的某些连续性目标变量；图匹配是指对给定的一对图进行相似度评估或匹配；图生成是指模型生成图对象实例，使其满足符合要求的某些特定性质。

b)　按监督形式划分

按监督形式，图神经网络的下游任务分为：监督训练、半监督训练、无监督训练三种形式。

监督训练是指每一个参与训练的样本对象都有对应的标记标签。

半监督训练是指仅有部分参与训练的样本对象有对应的标记标签。在测试阶段，直推式学习配置方式让图神经网络预测已经在训练阶段给出的未标记样本的标签值；归纳式学习配置方式让图神经网络预测未在训练阶段见过的全新样本。

无监督训练是指参与训练的样本没有来自外部的标记标签，模型寻找样本对象的内在模式，将其转化为监督信号来指导模型训练。预训练策略是使用图神经网络来学习样本对象的向量化表示，向量化表示应表征样本对象在图上的内在性质。

7 图神经网络模型

7.1 模型结构

7.1.1 计算图定义

计算图（Computational Graph）定义见表 50。

表 50 计算图定义

字段	类型	定义
node	NodeDef [repeated]	计算图中所有操作节点的列表
versions	Version	计算图的兼容性版本信息
library	FunctionDefLibrary	提供用户定义函数的库
debug_info	GraphDebugInfo	包含计算图节点的调试信息

7.1.1.1 操作节点定义

操作节点定义（NodeDef）见表 51。

表 51 操作节点定义

字段	类型	定义
name	string	操作节点的唯一标识名
op	string	操作节点的操作类型
input	string [repeated]	操作节点的输入列表
device	string	操作节点所在的设备
attr	map<string, DataType>	操作节点属性的键值对

7.1.1.2 版本定义

版本定义（Version）见表 52。

表 52 版本定义

字段	类型	定义
producer	int32	生成这个数据的代码的版本
min_consumer	int32	任何低于此版本的消费者不允许使用这个数据
bad_consumers	int32 [repeated]	不允许使用这个数据的特定消费者版本（通常由于 bug）

7.1.1.3 函数库定义

函数定义库（FunctionDefLibrary）见表 53。

表 53　函数库定义

字段	类型	定义
function	FunctionDef [repeated]	函数定义的集合
gradient	GradientDef [repeated]	每个函数的梯度函数定义
registered_gradients	RegisteredGradient [repeated]	注册在函数库中使用的梯度函数

函数定义（FunctionDef）见表 54。

表 54　函数定义

字段	类型	定义
signature	OpDef	函数的签名，包括函数名称、参数、返回值、属性等信息
attr	map<string, DataType>	特定于此函数定义的属性
arg_attr	map<uint32, map<string, DataType>>	函数参数的属性
resource_arg_unique_id	map<uint32, uint32>	每个资源参数的唯一标识符
node_def	NodeDef [repeated]	函数体中的节点定义
ret	map<string, string>	函数的返回值与函数体中节点输出的映射
control_ret	map<string, string>	函数的控制输出与函数体中节点名称的映射

函数签操作定义（OpDef）见表 55。

表 55　函数签名操作定义

字段	类型	定义
name	string	操作的名称，以大写字母开头，符合特定的正则表达式规则
input_arg	string [repeated]	输入参数的描述列表
output_arg	string [repeated]	输出参数的描述列表
control_output	string [repeated]	命名控制输出的列表，用于复合操作
attr	string [repeated]	图构建时配置的描述，即 NodeDef 中将指定的属性字段
summary	string	操作的简略可读描述
description	string	操作的详细可读描述
is_commutative	bool	指示操作是否是可交换的
is_aggregate	bool	指示操作是否是聚合操作
is_stateful	bool	指示操作是否是有状态的
allows_uninitialized_input	bool	指示操作是否允许未初始化输入
is_distributed_communication	bool	指示操作实现是否使用分布式通信

梯度定义（GradientDef）见表 56。

表 56　梯度定义

字段	类型	定义
function_name	string	定义梯度函数的函数名称
gradient_func	string	对应的梯度函数名称

注册梯度定义（RegisteredGradient）见表 57。

表 57　注册梯度定义

字段	类型	定义
gradient_func	string	梯度函数的名称
registered_op_type	string	梯度函数注册的操作类型

7.1.1.4　计算图调试信息

计算图调试信息定义（GraphDebugInfo）见表 58。

表 58　计算图调试信息定义

字段	类型	定义
files	string [repeated]	所有源代码文件的名称列表，可通过 file_index 索引
frames_by_id	map<fixed64, FileLineCol>	按唯一标识符索引的帧到文件行列信息的映射
traces_by_id	map<fixed64, StackTrace>	按唯一标识符索引的栈追踪信息的映射
name_to_trace_id	map<string, fixed64>	将节点名称映射到 traces_by_id 中包含的跟踪 ID

文件行列定义（FileLineCol）见表 59。

表 59　文件行列定义

字段	类型	定义
file_index	int32	文件名的索引，用于从 files 字符串列表中检索具体文件名
line	int32	在文件中的行号
col	int32	在文件中的列号
func	string	包含该文件行的函数名称
code	string	该文件行包含的源代码

栈追踪定义（StackTrace）见表 60。

表 60　栈追踪定义

字段	类型	定义
frame_id	fixed64 [repeated]	栈帧的唯一标识符，用于标识栈追踪中的每个帧的 ID

7.1.2　图神经网络模型结构定义

图神经网络模型结构定义（Model）见表 61。

表 61　图神经网络模型结构定义

字段	类型	定义
version	Version	模型定义的版本号
contributors	string [repeated]	贡献者列表
framework_name	string	框架名称，如 TensorFlow、PyTorch

表 61　图神经网络模型结构定义（续）

字段	类型	定义
framework_version	string	使用的框架版本
model_name	string	模型名称
model_version	string	模型版本号
graph	Computational Graph [repeated]	描述模型计算图的结构
attribute	map<string, DataType>	模型属性的键值对，描述模型的特性
signature	OpDef	模型的输入输出等参数信息的签名定义
description	string	模型的详细描述

7.2　基础算子

7.2.1　概述

基础算子是图神经网络中传递和聚合消息的基本组件，分为消息传递算子（Message Passing Operator）、池化算子（Pooling Operator）、和归一化算子（Normalization Operator）。

7.2.2　消息传递算子

7.2.2.1　概述

消息传递算子是图神经网络中的核心操作，它模拟了消息在图结构中的传播和交互过程。其计算过程包括三个子算子：消息算子（message operator）、聚合算子（aggregate operator）、更新算子（update operator）。基于消息传递的图神经网络卷积算子，可通过消息算子、聚合算子和更新算子的组合方式实现。

7.2.2.2　消息算子

消息算子是定义在每条边上，通过将边的特征与其附带节点的特征相结合来生成信息的操作。send_message 运算操作定义见表 62。

表 62　send_message 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
send_message	自定义消息发送函数	Input	src_feat	源节点特征	tensor Dict [string, tensor]
			dst_feat	可选，目标节点特征	tensor Dict [string, tensor]
			edge_feat	可选，边特征	tensor Dict [string, tensor]
			edge_index	边索引	tensor SparseTenosor Dict [string, tensor] Dict [string, SparseTensor]
		Output	Y	输出张量字典	tensor Dict [string, tensor]
		Attributes	message_func	消息产生模块	Model

7.2.2.3 聚合算子

聚合算子是消息传递过程中的关键组件，用于从节点的邻居聚合消息。
具体定义见表 63~表 67。
recv_message 运算操作定义见表 63。

表 63 recv_message 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
recv_message	自定义消息接收函数	Input	X	输入消息	tensor Dict [string, tensor]
			edge_index	边索引	tensor SparseTenosor Dict [string, tensor] Dict [string, SparseTensor]
		Output	Y	输出结果	tensor Dict [string, tensor]
		Attributes	reduce_func	消息接收模块	Model

sum 聚合操作定义见表 64。

表 64 sum 聚合操作

运算操作	描述	字段	关键字	定义	数据类型
sum	聚合来自邻居节点的特征，通过聚合函数“和”生成节点级表示	Input	X	接收的消息	tensor Dict [string, tensor]
			edge_index	边索引	tensor SparseTenosor Dict [string, tensor] Dict [string, SparseTensor]
			num_nodes	可选，节点个数	int
		Output	Y	输出聚合结果	tensor Dict [string, tensor]

max 聚合操作定义见表 65。

表 65 max 聚合操作

运算操作	描述	字段	关键字	定义	数据类型
max	聚合来自邻居节点的特征，通过聚合函数“最大”生成节点级表示	Input	X	接收的消息	tensor Dict [string, tensor]

表 65　max 聚合操作（续）

运算操作	描述	字段	关键字	定义	数据类型
max	聚合来自邻居节点的特征，通过聚合函数“最大”生成节点级表示	Input	edge_index	边索引	tensor SparseTensor Dict [string, tensor] Dict [string, SparseTensor]
			num_nodes	可选，节点个数	int
		Output	Y	输出聚合结果	tensor Dict [string, tensor]

min 聚合操作定义见表 66。

表 66　min 聚合操作

运算操作	描述	字段	关键字	定义	数据类型
min	聚合来自邻居节点的特征，通过聚合函数“最小”生成节点级表示	Input	X	接收的消息	tensor Dict [string, tensor]
			edge_index	边索引	tensor SparseTensor Dict [string, tensor] Dict [string, SparseTensor]
			num_nodes	可选，节点个数	int
		Output	Y	输出聚合结果	tensor Dict [string, tensor]

mean 聚合操作定义见表 67。

表 67　mean 聚合操作

运算操作	描述	字段	关键字	定义	数据类型
mean	聚合来自邻居节点的特征，通过聚合函数“平均”生成节点级表示	Input	X	生成的消息	tensor Dict [string, tensor]
			edge_index	边索引	tensor SparseTensor Dict [string, tensor] Dict [string, SparseTensor]
			num_nodes	可选，节点个数	int
		Output	Y	输出表示张量	tensor Dict [string, tensor]

7.2.2.4 更新算子

更新算子是消息传递过程中的关键组件，定义在每个节点上，把聚合的消息或自身的特征进行处理以更新节点特征。

update 更新操作定义见表 68。

表 68 update 更新操作

运算操作	描述	字段	关键字	定义	数据类型
update	在每个节点上定义，用于处理聚合的消息或自身的特征来更新节点特征	Input	H	聚合后的消息	tensor Dict [string, tensor]
			X	自身特征	tensor Dict [string, tensor]
			num_nodes	可选，节点个数	int
		Output	Y	更新后的节点表征	tensor Dict [string, tensor]
		Attributes	update_rule	更新规则，如使用 ReLU，线性层等	Module

7.2.2.5 卷积算子

本标准定义了各种经典的图卷积算子，具体定义见表 69~表 119。

GCNConv 运算操作定义见表 69。

表 69 GCNConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
GCNConv	构建谱域图卷积，使用特征张量和邻接矩阵，输出表示张量	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int Tuple [int, int]
			out_channels	输出特征的维度	int
			cached	可选，是否缓存第一次执行的对称归一化邻接矩阵的结果	bool
			add_self_loops	可选，是否给输入图添加自环	bool

表 69　GCNConv 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
GCNConv	构建谱域图卷积，使用特征张量和邻接矩阵，输出表示张量	Attributes	improved	可选，是否在添加自环操作时，添加两个自环	bool
			normalize	可选，是否进行归一化	bool
			bias	可选，是否添加偏置项	bool
			activation	可选，激活函数名称	string
			allow_zero_in_degree	可选，是否允许入度为 0 的节点出现	bool

ChebConv 运算操作定义见表 70。

表 70　ChebConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
ChebConv	构建 Chebyshev 谱图卷积运算符，使用特征张量和邻接矩阵，输出表示张量	Input	X	节点特征矩阵	tensor
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			g	输入图，和 edge_index 二选一	Graph
			edge_weight	可选，边权重矩阵	tensor
			batch	可选，每个节点的批次所属	tensor
			lambda_max	可选，拉普拉斯正则化矩阵的最大特征值	tensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			k	切比雪夫滤波器大小	int
			normalize	可选，归一化方法	string
			bias	可选，是否添加偏置项	bool
			activation	可选，激活函数名称	string

SAGEConv 运算操作定义见表 71。

表 71　SAGEConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
SAGEConv	通过对图中的节点进行采样和聚合操作来学习节点的表示，从而能够支持对图数据的有效的学习和推理	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			size	可选，邻接矩阵大小	Tuple [int, int]
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int Tuple [int, int]
			out_channels	输出特征的维度	int
			root_weight	可选，是否将转换后的根节点特征添加到输出中	bool
			normalize	可选，是否进行归一化	bool
			aggr	可选，聚合方法	string List [string]
			feat_drop	可选，特征的丢弃率	float
			activation	可选，激活函数名称	string
			bias	可选，是否添加偏置项	bool

GraphConv 运算操作定义见表 72。

表 72　GraphConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
GraphConv	构建高阶图神经网络，使用特征张量和邻接矩阵，输出特征张量	Input	X	节点特征矩阵	int Tuple [int, int]
			edge_index	边索引	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			size	可选，邻接矩阵大小	Tuple [int, int]
		Output	Y	输出特征表示	tensor

表 72　GraphConv 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
GraphConv	构建高阶图神经网络，使用特征张量和邻接矩阵，输出特征张量	Attributes	in_channels	输入特征的维度	int Tuple [int, int]
			out_channels	输出特征的维度	int
			aggr	可选，要使用的聚合方法	string
			bias	可选，是否添加偏置项	bool

GravNetConv 运算操作定义见表 73。

表 73　GravNetConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
GravNetConv	使用最近邻居动态构建图。邻居通过特征空间的可学习的低维投影来构造。然后，使用距离权重将输入特征空间的第二个投影从相邻点传播到每个顶点，该距离权重是通过对距离应用高斯函数得出的	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			batch	每一个节点的批次所属	tensor Tuple [tensor, tensor]
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			space_dimesions	用于构造邻居的空间的维数	int
			k	最近邻居的数量	int

GatedgraphConv 运算操作定义见表 74。

表 74　GatedgraphConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
GatedgraphConv	构建门控图卷积运算符，使用特征张量和邻接矩阵，输出特征张量	Input	X	节点特征矩阵	tensor
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			edge_type	可选，边的类型	tensor
			edge_weight	可选，边权重矩阵	tensor
		Output	Y	输出特征表示	tensor
		Attributes	out_channels	输出特征的维度	int
			num_steps	可选，序列长度	int

表 74　GatedgraphConv 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
GatedgraphConv	构建门控图卷积运算符，使用特征张量和邻接矩阵，输出特征张量	Attributes	aggr	可选，要使用的聚合方法	string
			n_etypes	可选，边类型数目	int
			bias	可选，是否添加偏置项	bool

ResGatedgraphConv 运算操作定义见表 75。

表 75　ResGatedgraphConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
ResGatedgraphConv	构建残差门控图卷积运算符，使用特征张量和邻接矩阵，输出特征张量	Input	X	节点特征矩阵	tensor
			edge_index	边索引	tensor SparseTensor
			edge_weight	可选，边的权重矩阵	tensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			edge_dim	可选，边的特征维度	int
			act	可选，激活函数名称	string
			bias	可选，是否添加偏置项	bool
			root_weight	可选，是否将转换后的根节点特征添加到输出中	bool

GATConv 运算操作定义见表 76。

表 76　GATConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
GATConv	构建图注意力运算符，使用特征张量和邻接矩阵，输出特征张量	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			size	可选，邻接矩阵大小	Tuple [int, int]
			get_attention	可选，是否返回注意力权重	bool

表 76　GATConv 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
GATConv	构建图注意力运算符，使用特征张量和邻接矩阵，输出特征张量	Output	Y	输出特征表示	tensor Tuple [tensor, tensor]
			attention_weight	可选，注意力权重	tensor
		Attributes	in_channels	输入特征的维度	int Tuple [int, int]
			out_channels	输出特征的维度	int
			heads	可选，多头注意力的数量	int
			concat	可选，如果是 False，对多头注意力进行 average 操作，如果是 True，则进行 concat 操作	bool
			negative_slope	可选，负斜率的 LeakyReLU 角度	float
			feat_drop	可选，特征的丢弃率	float
			attn_drop	可选，在训练过程中，将每个节点暴露于随机采样的邻域的归一化注意力系数的丢弃概率	float
			bias	可选，是否添加偏置项	bool
			add_self_loops	可选，是否给输入图添加自环	bool
			edge_dim	可选，边特征维度	int
			fill_value	可选，用于生成自环的边特征	float tensor string
			allow_zero_in_degree	可选，是否允许入度为 0 的节点出现	bool
			residual	可选，是否使用残差连接	bool
			activation	可选，激活函数名称	string

TransformerConv 运算操作定义见表 77。

表 77　TransformerConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
TransformerConv	构建图 transformer 运算符，使用特征张量、邻接矩阵和边属性张量，输出特征张量	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			edge_index	边索引	tensor SparseTensor
			edge_weight	可选，边特征矩阵	tensor
			get_attention	可选，是否返回注意力权重	bool
		Output	Y	输出张量	tensor
			attention_weight	注意力权重	tensor
		Attributes	in_channels	输入特征的维度	int Tuple [int, int]
			out_channels	输出特征的维度	int
			heads	可选，多头注意力的数量	int
			concat	可选，如果是 False，对多头注意力进行 average 操作，如果是 True，则进行 concat 操作	bool
			beta	可选，通过 beta 加权聚合信息和跳过信息	bool
			dropout	可选，在训练过程中，将每个节点暴露于随机采样的邻域的归一化注意力系数的丢弃概率	float
			edge_dim	可选，边特征的维度	int
			root_weight	可选，是否将转换后的根节点特征添加到输出中	bool
			bias	可选，是否添加偏置项	bool

AGNNConv 运算操作定义见表 78。

表 78　AGNNConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
AGNNConv	通过引入注意力机制来加权邻居节点的贡献，使模型能够更加关注于重要的邻居节点，从而提高图神经网络处理图数据的能力	Input	X	节点特征矩阵	tensor
			edge_index	边索引	tensor SparseTensor
		Output	Y	输出张量	tensor
		Attributes	requires_grad	可选，参数 beta 是否可学习	bool
			add_self_loops	可选，是否给输入图添加自环	bool
			beta	可选，参数 beta 的初始值	float
			allow_zero_in_degree	可选，是否允许入度为 0 的节点出现	bool

TAGConv 运算操作定义见表 79。

表 79　TAGConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
TAGConv	基于节点特征自适应的图卷积方法，它通过考虑节点自身特征和邻居节点特征之间的关系来更新节点的表示	Input	X	节点特征矩阵	tensor
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			k	可选，跳数 k	int
			activation	可选，激活函数名称	string
			normalize	可选，是否进行归一化	bool
			bias	可选，是否添加偏置项	bool

GINConv 运算操作定义见表 80。

表 80　GINConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
GINConv	基于图同构的卷积操作。不考虑节点之间的顺序关系，而是专注于节点之间的结构关系	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			edge_weight	可选，边的权重	tensor
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			size	邻接矩阵大小	Tuple [int, int]
		Output	Y	输出特征表示	tensor
		Attributes	nn	可选，将输入节点特征映射到输出维度的神经网络方法	Model
			eps	可选，初始自环缩放参数	float
			train_eps	可选，初始自环缩放参数是否可学习	bool
			activation	可选，激活函数名称	string
			aggregation_type	可选，聚合类型	string

GINEConv 运算操作定义见表 81。

表 81　GINEConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
GINEConv	改进的图同构运算符，使用一种新的策略和自监督方法来预训练图神经网络	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			g	输入图，和 edge_index 二选一	Graph
			edge_weight	可选，边特征矩阵	tensor
			size	邻接矩阵大小	Tuple [int, int]
		Output	Y	输出特征表示	tensor
		Attributes	nn	将输入节点特征映射到输出维度的神经网络方法	Model
			edge_dim	可选，边特征维度	int
			eps	可选，初始自环缩放参数	float
			train_eps	可选，初始自环缩放参数是否可学习	bool

ARMAConv 运算操作定义见表 82。

表 82　ARMAConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
ARMAConv	构建 ARMA 图卷积运算符，通过结合自回归和移动平均模型的思想，对图上的节点进行特征更新	Input	X	节点特征矩阵	tensor
			edge_index	边索引	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			num_stacks	可选，并行栈数	int
			num_layers	可选，层数	int
			act	可选，激活函数名称	string
			shared_weights	可选，每个栈是否共享相同的参数	bool
			dropout	可选，丢弃率	float
			bias	可选，是否添加偏置项	bool

SGConv 运算操作定义见表 83。

表 83　SGConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
SGConv	简化传统图卷积网络中的计算复杂度，并提高模型的效率的图卷积算子 简化传统图卷积网络中的计算复杂度，并提高模型的效率的图卷积算子	Input	X	节点特征矩阵	tensor
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			k	可选，跳数	int
			cached	可选，是否缓存第一次执行特征聚合的结果	bool
			add_self_loops	可选，是否添加自环	bool
			normalize	可选，归一化函数名称	string
			activation	可选，激活函数名称	string
			bias	可选，是否添加偏置项	bool
			allow_zero_in_degree	可选，是否允许入度为 0 的节点出现	bool

APPNPConv 运算操作定义见表 84。

表 84　APPNPConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
APPNPConv	构建神经预测层的近似个性化预测传播运算符，使用特征张量、邻接矩阵和边权重矩阵（可选），输出特征张量	Input	X	节点特征矩阵	tensor
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
		Output	Y	输出特征表示	tensor
		Attributes	k	可选，跳数	int
			alpha	传送概率参数	float
			edge_drop	可选，节点接收的消息丢弃率	Float

表 84 APPNPConv 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
APPNPConv	构建神经预测层的近似个性化预测传播运算符，使用特征张量、邻接矩阵和边权重矩阵（可选），输出特征张量	Attributes			
			cached	可选，是否缓存第一次执行的对称归一化邻接矩阵的结果	bool
			add_self_loops	可选，是否添加自环	bool
			normalize	可选，是否进行归一化	bool

MFConv 运算操作定义见表 85。

表 85　MFConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
MFConv	通过对分子图进行卷积操作，捕捉分子的结构特征，生成分子指纹，用于分子属性预测、药物筛选等应用的卷积算子 通过对分子图进行卷积操作，捕捉分子的结构特征，生成分子指纹，用于分子属性预测、药物筛选等应用的卷积算子	Input	X	节点特征矩阵	Tuple [tensor, tensor]
			edge_index	边索引	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			size	邻接矩阵大小	Tuple [int, int]
		Output	Y	输出特征表示	Tensor
		Attributes	in_channels	输入特征的维度	Tuple [tensor, tensor]
			out_channels	输出特征的维度	int
			max_dgree	可选，更新权重时要考虑的最大节点度	int
			bias	可选，是否添加偏置项	bool

SignedConv 运算操作定义见表 86。

表 86　SignedConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
SignedConv	SignedConv 利用了正负边的信息来调整节点表示，使得节点能够更好地捕捉图中正负边的影响	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			pos_edge_index	正向邻接矩阵	tensor SparseTensor
			neg_edge_index	负向邻接矩阵	tensor SparseTensor
		Output	Y	输出特征表示	tensor

表 86　SignedConv 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
SignedConv	SignedConv 利用了正负边的信息来调整节点表示，使得节点能够更好地捕捉图中正负边的影响	Attributes	in_channels	输入特征的维度	Int Tuple [int, int]
			out_channels	输出特征的维度	int
			first_aggr	可选，是否进行额外一次聚合	bool
			bias	可选，是否添加偏置项	bool

DNAConv 运算操作定义见表 87。

表 87　DNAConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
DNAConv	在节点更新时，根据节点自身特征和邻居节点的特征动态地确定邻域大小和邻域节点的重要性，从而实现对节点表示的动态聚合 在节点更新时，根据节点自身特征和邻居节点的特征动态地确定邻域大小和邻域节点的重要性，从而实现对节点表示的动态聚合	Input	X	节点特征矩阵	tensor
			edge_index	边索引	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
		Output	Y	输出特征表示	tensor
		Attributes	channels	输入或输出特征维度	int Tuple [int, int]
			groups	可选，进行线性映射的组数	Int
			heads	可选，多头注意力的数量	int
			dropout	可选，注意系数的丢弃概率	float
			cached	可选，是否缓存第一次执行的对称归一化邻接矩阵的结果	bool
			normalize	可选，是否进行归一化	bool
			add_self_loops	可选，是否添加自环	bool
			bias	可选，是否添加偏置项	bool
此处 x 输入的节点特征的形状为[num_nodes, num_layers, channels]。					

PointConv 运算操作定义见表 88。

表 88　PointConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
PointConv	专用于点云数据，在计算节点表示时，不再依赖于节点之间的连接关系，而是仅基于点的坐标和属性信息来进行卷积操作	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			edge_index	边索引	tensor SparseTensor
			pos	点坐标矩阵	tensor Tuple [tensor, tensor]
		Output	Y	输出特征表示	tensor
		Attributes	local_nn	可选，映射节点特征和相对空间坐标的神经网络	Model
			global_nn	可选，映射聚合后的节点特征的神经网络	Model
			add_self_loops	可选，是否添加自环	bool

GMMConv 运算操作定义见表 89。

表 89　GMMConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
GMMConv	利用高斯混合模型学习生成每个节点的自适应卷积核。这些卷积核允许网络根据节点的局部邻域结构和特征动态地调整，从而更好地捕捉图中的信息	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			pseudo	可选，伪坐标张量大小	tensor
			g	边索引，和 g 二选一	Graph
			edge_index	输入图，和 edge_index 二选一	tensor SparseTensor
			size	可选，邻接矩阵大小	Tuple [int, int]
			edge_weight	可选，边特征矩阵	tensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int Tuple [int, int]
			out_channels	输出特征的维度	int
			dim	伪坐标维度	int
			k	可选，核的数量	int
			separate_gaussians	可选，为每对输入和输出通道学习单独的 GMM	bool
			aggr	可选，要使用的聚合方法	string
			root_weight	可选，是否将转换后的根节点特征添加到输出中	bool
			coord_dim	伪坐标的维数	int
			residual	可选，是否使用残差连接	bool
			bias	可选，是否添加偏置项	bool
			allow_zero_in_degree	可选，是否允许入度为 0 的节点出现	bool

SplineConv 运算操作定义见表 90。

表 90　SplineConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
SplineConv	使用 B-spline 基函数来建模图上的卷积操作。从而捕捉图中节点之间的局部结构和关系，允许网络学习节点特征之间更复杂的关联 使用 B-spline 基函数来建模图上的卷积操作。从而捕捉图中节点之间的局部结构和关系，允许网络学习节点特征之间更复杂的关联	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			edge_index	边索引	tensor SparseTensor
			size	可选，邻接矩阵大小	Tuple [int, int]
			edge_weight	可选，边特征矩阵	tensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int Tuple [int, int]
			out_channels	输出特征的维度	int
			dim	伪坐标维度	int
			kernel_size	卷积核的大小	int List [int]
			is_open_spline	可选，是否在此维度上使用封闭的 B 样条曲线	bool
			degree	可选，B 样条基度	int
			aggr	可选，要使用的聚合方法	string
			root_weight	可选，是否将转换后的根节点特征添加到输出中	bool
			bias	可选，是否添加偏置项	bool

NNConv 运算操作定义见表 91。

表 91　NNConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
NNConv	基于自适应卷积核的卷积操作，通过神经网络动态地学习每个节点的卷积核	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			size	可选，邻接矩阵大小	Tuple [int, int]
			edge_weight	可选，边特征矩阵	tensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int Tuple [int, int]
			out_channels	输出特征的维度	Int
			nn	可选，映射边特征的神经网络	Model
			aggr	可选，要使用的聚合方法	string
			root_weight	可选，是否将转换后的根节点特征添加到输出中	bool
			residual	可选，是否使用残差连接	bool
			bias	可选，是否添加偏置项	bool

CGConv 运算操作定义见表 92。

表 92　CGConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
CGConv	构建晶体图卷积运算符，根据晶体图中原子的邻接关系以及原子的特征向量，利用图卷积操作来聚合每个原子的特征	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			edge_index	边索引	tensor SparseTensor
			edge_weight	可选，边特征矩阵	tensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int Tuple [int, int]
			edge_dim	可选，边特征维度	Int
			aggr	可选，要使用的聚合方法	string
			batch_norm	可选，是否使用批量归一化	bool
			bias	可选，是否添加偏置项	bool

EdgeConv 运算操作定义见表 93。

表 93　EdgeConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
EdgeConv	一种处理点云数据的动态图卷积算子，通过构建动态图来描述点之间的连接关系，利用这些边来聚合每个点的邻域信息，并更新点的特征表示	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor SparseTensor
		Output	Y	输出特征表示	tensor Tuple [tensor, tensor]
		Attributes	nn	可选，映射成对串联节点特征的神经网络	Model
			in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			batch_norm	可选，是否使用批归一化函数	bool
			bias	可选，是否添加偏置项	bool
			aggr	可选，要使用的聚合方法	string
			allow_zero_in_degree	可选，是否允许入度为 0 的节点出现	bool

DynamicEdgeConv 运算操作定义见表 94。

表 94　DynamicEdgeConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
DynamicEdgeConv	一种处理点云数据的动态图卷积算子，通过构建动态图来描述点之间的连接关系，利用这些边来聚合每个点的邻域信息，并更新点的特征表示。图是使用特征空间中的最近邻居动态构建的	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			batch	可选，批向量，将每个节点分配一个批量	tensor
		Output	Y	输出特征表示	tensor
		Attributes	nn	映射成对节点串联特征。	Model
			k	最近邻居数量	int
			aggr	可选，要使用的聚合方法	string
			num_workers	可选，k-NN 计算的进程数	int

XConv 运算操作定义见表 95。

表 95　XConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
XConv	构建 X 变换点上的卷积运算符，对点云中的每个点进行变换，将其转换到局部坐标系中。然后，它利用这些局部坐标系来构建卷积核，从而捕捉点云中点之间的局部关系	Input	X	节点特征矩阵	tensor
			pos	点坐标矩阵	tensor
			batch	可选，每一个节点的批次所属	tensor Tuple [tensor, tensor]
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			kernel_size	卷积核的大小，即包括自循环在内的邻居数	int
			hidden_channel	可选，隐含层点特征维度	int
			dilation	可选，邻居扩散因子	int
			bias	可选，是否添加偏置项	bool
			num_workers	可选，k-NN 计算的进程数	int

PPFConv 运算操作定义见表 96。

表 96　PPFConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
PPFConv	一种有效且鲁棒的方式来处理 3D 点云数据，使得神经网络能够更好地理解和利用点云中的局部信息，并在点云匹配任务中取得更好的性能	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			pos	点坐标矩阵	tensor Tuple [tensor, tensor]
			edge_index	边索引	tensor SparseTensor
			normal	法线矩阵	tensor Tuple [tensor, tensor]
		Output	Y	输出特征表示	tensor
		Attributes	local_nn	可选，映射节点特征和相对空间坐标的神经网络	Model
			global_nn	可选，映射聚合后的节点特征的神经网络	Model
			add_self_loops	可选，是否添加自环	bool

FeatStConv 运算操作定义见表 97。

表 97　FeaStConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
FeaStConv	根据输入的特征信息来动态调整卷积核的权重。这些权重可以根据输入特征的情况来调整，从而使网络能够更好地适应不同形状的特征分布	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			edge_index	边索引	tensor SparseTensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int Tuple [int, int]
			out_channels	输出特征的维度	int
			heads	可选，注意力头的数量	int
			add_self_loops	可选，是否添加自环	bool
			bias	可选，是否添加偏置项	bool

HypergraphConv 运算操作定义见表 98。

表 98　HypergraphConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
HypergraphConv	通过引入超边权重，用于控制超边对节点特征的影响程度，从而更好地捕捉超图中节点之间的关系	Input	X	节点特征矩阵	tensor
			hyperedge_index	超边索引	tensor
			hyperedge_weight	可选，超边特征矩阵	tensor
			num_edges	可选，超边数量	int
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			use_attention	可选，是否使用注意力机制	bool
			attention_mode	可选，计算注意力机制的模式	string
			heads	可选，注意力头的数量	int
			concat	可选，多头注意力机制进行串联，否则进行平均	int
			negative_slope	负斜率的 LeakyReLU 角度	float
			dropout	可选，丢弃率	float
			bias	可选，是否添加偏置项	bool

LEConv运算操作定义见表99。

表 99　LEConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
LEConv	在每个节点上对局部邻域进行卷积操作，并通过自适应的结构感知机制加强图的层次表示	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			edge_index	边索引	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			bias	可选，是否添加偏置项	bool

PNAConv 运算操作定义见表 100。

表 100　PNAConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
PNAConv	通过将节点特征进行多项式变换来捕捉节点自身的信息。利用聚合函数将变换后的节点特征与邻居节点的特征进行结合，以获得节点的新特征表示	Input	X	节点特征矩阵	tensor
			edge_index	边索引	tensor SparseTensor
			edge_weight	可选，边属性矩阵	tensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			aggr	聚合函数的标识符集合	List [string]
			scalers	缩放功能标识符集合	List [string]
			deg	可选，训练集中的节点的度数直方图，供缩放器用来进行归一化	tensor
			delta	可选，度正则化有关的因子	float
			edge_dim	可选，边特征维度	int
			act	可选，激活函数名称	string
			tower	可选，塔数	int
			pre_layers	可选，聚合之前的转换层数	int
			post_layers	可选，聚合之后的转换层数	int
			train_norm	可选，正则化参数是否可训练	bool
			residual	可选，是否添加残差连接	bool
			dropout	可选，丢弃率	float
			divided_Input	可选，输入特征是否应在塔之间划分	bool

ClusterGCNConv 运算操作定义见表 101。

表 101　ClusterGCNConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
ClusterGCNConv	将图数据分解成多个子图（cluster），然后在每个子图上进行独立的 GCN 训练，从而减少计算和内存需求	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			edge_index	边索引	tensor SparseTensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			diag_lambda	可选，对角增强值	Float
			add_self_loops	可选，是否添加自环	bool
			bias	可选，是否添加偏置项	bool

GENConv 运算操作定义见表 102。

表 102　GENConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
GENConv	使用一系列可学习的参数化函数来对节点之间的消息传递进行建模。这些参数化函数可以动态地调整以适应不同的图结构和特征分布	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			edge_index	边索引	tensor SparseTensor
			size	可选，邻接矩阵大小	Tuple [int, int]
			edge_weight	可选，边特征矩阵	tensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			aggr	可选，要使用的聚合方法	string
			t	可选，softmax 聚集的初始反向温度。	float
			learn_t	可选，softmax 的反向温度是否可学习	bool
			expansion	可选，MLP 中隐藏层的扩散因子	int
			p	可选，功率均值聚合的初始功率值	float
			learn_p	可选，功率均值聚合是否为动态可学习	bool
			msg_norm	可选，是否使用消息归一化	bool
			learn_msg_scale	可选，消息归一化的比例因子是否可学习	bool
			normalize	多层感知器的归一化方式	string
			bias	可选，是否添加偏置项	bool
			edge_dim	可选，边特征维度	int
			num_layers	可选，多层感知器的层数	int
			eps	可选，消息构造函数的 epsilon 值	float

GCNIIConv 运算操作定义见表 103。

表 103　GCNIIConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
GCNIIConv	通过利用节点的自身特征和邻居节点的特征，结合多项式参数化函数来进行信息传递和聚合。从而学习节点特征表示	Input	X	节点特征矩阵	tensor
			X0	初始化特征矩阵	tensor
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			g	输入图，和 edge_index二选一	Graph
			edge_weight	可选，边权重矩阵	tensor

表 103　GCNIIConv 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
GCNIIConv	通过利用节点的自身特征和邻居节点的特征，结合多项式参数化函数来进行信息传递和聚合。从而学习节点特征表示	Output	Y	输出特征表示	Tensor
		Attributes	channels	输入/输出特征的维度	int
			alpha	初始残差连接的强度	float
			theta	可选，计算身份映射的强度的超参	float
			layer	可选，在该模块中执行的层数	int
			shared_weights	可选，共享平滑表示和初始残差连接的权重矩阵	bool
			activation	可选，激活函数名称	string
			bias	可选，是否添加偏置项	bool
			allow_zero_in_degree	可选，是否允许入度为 0 的节点出现	bool
			cached	可选，是否缓存第一次执行的对称归一化邻接矩阵的结果	bool
			normalize	可选，是否添加自环并应用对称归一化	bool
			add_self_loops	可选，是否添加自环	bool

WLConv 运算操作定义见表 104。

表 104　WLConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
WLConv	构建 Weisfeiler Lehman 运算符，对每个节点及其邻居节点的特征使用 WL 图同构测试对编码后的图进行处理，最后通过卷积操作将编码后的节点特征进行聚合和更新	Input	X	节点特征矩阵	tensor
			edge_index	边索引	tensor SparseTensor
		Output	Y	输出特征表示	tensor

FiLMConv 运算操作定义见表 105。

表 105　FiLMConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
FiLMConv	在传统的图卷积操作中引入了FiLM 机制，通过对每个节点的特征进行线性调制，以增强或减弱特定维度的特征表示，从而提高网络对图数据的建模能力和泛化能力	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			edge_index	边索引	tensor SparseTensor
			edge_type	可选，边种类矩阵	tensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			num_relation	可选，关系的数量	int
			nn	可选，节点特征映射的神经网络	Model
			act	可选，激活函数	Model
			aggr	可选，要使用的聚合方法	string

SuperGATConv 运算操作定义见表 106。

表 106　SuperGATConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
SuperGATConv	利用图注意力机制对节点之间的关系进行建模，以确定每个节点与其邻居节点之间的重要性。然后，利用自监督学习的方法来引导网络学习更有意义的节点表示，使得节点表示能够更好地捕捉图的结构和特征	Input	X	节点特征矩阵	tensor
			edge_index	边索引	tensor SparseTensor
			neg_edge_index	负向边索引	tensor SparseTensor
			batch	可选，每一个节点的批次所属	tensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			heads	可选，多头注意力的数量	int
			concat	可选，如果是 False，对多头注意力进行 average 操作，如果是 True，则进行 concat 操作	bool
			negative_slope	可选，负斜率的 LeakyReLU 角度	float
			dropout	可选，丢弃率	float
			bias	可选，是否添加偏置项	bool
			add_self_loops	可选，是否添加自环	bool
			attention_type	可选，注意力机制的种类	string
			neg_sample_ratio	可选，采样的负边数与正边数之比	float

表 106　SuperGATConv 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
SuperGATConv	利用图注意力机制对节点之间的关系进行建模，以确定每个节点与其邻居节点之间的重要性。然后，利用自监督学习的方法来引导网络学习更有意义的节点表示，使得节点表示能够更好地捕捉图的结构和特征	Attributes	edge_sample_ratio	可选，训练边数中用于训练的样本比例	Float
			is_undirected	可选，输入图是否是无向的。如果未给出，则在执行负采样时将使用输入图自动计算	bool

CFConv 运算操作定义见见表 107。

表 107　CFConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
CFConv	使用可学习的滤波器函数来对原子之间的相互作用进行建模，将原子间的相对位置和特征映射到一个连续的函数空间中，这些滤波后的特征表示会根据原子之间的相对位置进行聚合，以捕获分子的局部结构和化学环境	Input	X	节点特征矩阵	tensor
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			edge_weight	输入的边特征	tensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入节点特征的维度	int
			edge_dim	边特征维度	int
			hidden_channels	隐藏表示的维度	int
			out_channels	输出表示的维度	int

DOTGATConv 操作定义见表 108。

表 108　DOTGATConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
DOTGATConv	构建 GAT 中应用点积方法计算注意力权重系数的图卷积层	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			get_attention	可选，是否返回注意力值	bool
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int Tuple [int, int]
			out_channels	输出表示的维度	int
			num_heads	多头注意中的头数	int
			bias	是否带偏置	bool
			allow_zero_in_degree	可选，是否允许入度为 0 的节点出现	bool

RGCNConv 运算操作定义见表 109。

表 109　RGCNConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
RGCNConv	利用每种关系类型的参数化权重来进行节点特征的聚合和更新，从而更好地捕捉图中不同类型的关系信息	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			edge_type	可选，一维边类型	tensor
			norm	可选，表示边规范值的一维张量	tensor
			is_sorted	可选，表示输入图的边是否已经按照其类型排序	bool
		Output	Y	输出张量	tensor
		Attributes	in_channels	输入节点特征的维度	int
			out_channels	输出节点表示的维度	int
			num_relations	边关系的数量	int
			num_bases	可选，基数分解正则化的基数，None 为不使用基数正则化方案	int
			num_blocks	可选，块对角分解正则化的块数，None 为不使用块对角正则化方案	int
			aggr	可选，要使用的聚合方法	string
			root_weight	可选，是否将转换后的根节点特征添加到输出中	bool
			bias	可选，是否添加偏置项	bool
			activation	可选，激活函数名称	string
			add_self_loops	可选，是否添加自环	bool
			dropout	可选，丢弃率	float
			layer_norm	可选，是否进行归一化	bool

HANConv 运算操作定义见表 110。

表 110　HANConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
HANConv	使用注意力机制和元路径来动态地计算节点之间的关系权重，并利用这些权重对节点特征进行聚合和更新	Input	X_dict	节点特征字典，用于存储每一种节点类型的节点特征信息	Dict [string, tensor]
			g	输入图，至少和 edge_index_dict 二选一	Graph
			edge_index_dict	边索引字典，至少和 g 二选一	Dict [Tuple [string, string, string], tensor] Dict [Tuple [string, string, string], SparseTensor]

表 110　HANConv 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
HANConv	使用注意力机制和元路径来动态地计算节点之间的关系权重，并利用这些权重对节点特征进行聚合和更新	Input	get_attention	可选，是否额外返回每种目标节点类型的语义级注意力权重	Bool
		Output	Y	输出张量	Dict [string, tensor]
		Attributes	in_channels	输入特征的维度	int Dict [string, int]
			out_channels	输出特征的维度	int
			metadata	异质图的元数据，即由一个字符串列表给出的节点类型和由一个字符串三元组列表给出的边类型	Tuple [List [string]，List [Tuple [string, string, string]]]
			heads	可选，多头注意力的数量	int
			negative_slope	可选，负斜率的 LeakyReLU 角度	float
			dropout	可选，丢弃率	float

HGTConv 运算操作定义见表 111。

表 111　HGTConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
HGTConv	构建异质图 transformer 运算符，自注意力机制来动态地计算节点之间的关系权重，并利用这些权重对节点特征进行聚合和更新	Input	X	节点特征矩阵	Dict [string, tensor]
			g	输入图，和 edge_index_dict 二选一	Graph
			edge_index_dict	边索引字典，和 g 二选一	Dict [Tuple [string, string, string], tensor] Dict [Tuple [string, string, string], SparseTensor]
			node_type	可选，一维节点类型索引	tensor
			edge_type	可选，一维边类型索引	tensor
			is_sorted	可选，表示输入图的边是否已经按照其类型排序	bool
		Output	Y	输出张量	Dict [string, tensor]
		Attributes	in_channels	输入特征的维度	int Dict [string, int]
			out_channels	输出特征的维度	int
			metadata	可选，异质图的元数据，即由一个字符串列表给出的节点类型和由一个字符串三元组列表给出的边类型	Tuple [List [string], List [Tuple [string, string, string]]]
			heads	可选，多头注意力的数量	int

表 111　HGTConv 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
HGTConv	构建异质图 transformer 运算符，自注意力机制来动态地计算节点之间的关系权重，并利用这些权重对节点特征进行聚合和更新	Attributes	group	可选，用来对不同关系产生的节点嵌入分组的聚合方式	string
			num_ntypes	可选，节点类型的数量	int
			num_etypes	可选，边类型的数量	int
			dropout	可选，丢弃率	float
			layer_norm	可选，是否进行归一化	bool

MetaPath2Vec 运算操作定义见表 112。

表 112　MetaPath2Vec 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
MetaPath2Vec	构建元路径随机游走嵌入学习运算符，利用这些关系模式来生成节点序列。然后，它使用基于深度学习的技术，如 Word2Vec，来学习节点的嵌入，以保持嵌入空间中相似节点的相似性	Input	node_type	想要获得嵌入的节点类型	string
			batch	可选，想要获取嵌入的节点的索引	tensor
		Output	Y	输出张量（节点嵌入）	tensor
		Attributes	edge_index_dict	边索引字典，至少和 g 二选一	Dict [Tuple [string, string, string], tensor] Dict [Tuple [string, string, string], SparseTensor]
			g	输入图，和 edge_index_dict 二选一	Graph
			embedding_dim	嵌入向量的大小	int
			metapath	元路径	List [Tuple [string, string, string]]
			walk_length	游走的长度	int
			context_size	正例的上下文窗口大小	int
			walks_per_node	可选，对于每个节点进行游走的次数	int
			num_negative_samples	可选，对于每个正例，负采样的数量	int

表 112　MetaPath2Vec 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
MetaPath2Vec	构建元路径随机游走嵌入学习运算符，利用这些关系模式来生成节点序列。然后，它使用基于深度学习的技术，如Word2Vec，来学习节点的嵌入，以保持嵌入空间中相似节点的相似性	Attributes	num_nodes_dict	可选，节点数量字典，保存每种节点类型的节点数量	Dict [string, int]
			sparse	可选，进行反向传播时，权重矩阵的梯度是否设置为稀疏矩阵	bool

HERec 运算操作定义见表 113。

表 113　HERec 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
HERec	利用一个交互式表示学习框架，将用户和商品的隐含表示进行交叉融合，以产生更加准确的用户-商品匹配表示。最后，HERec利用这些匹配表示来进行推荐，以提高推荐系统的性能和准确度	Input	X	节点特征矩阵	Dict [string, tensor]
			edge_index_dict	边索引	Dict [Tuple [string, string, string], tensor] Dict [Tuple [string, string, string], SparseTensor]
			g	输入图，和edge_index_dict 二选一	Graph
			node_type	节点类型	tensor
			edge_type	边类型	tensor
		Output	Y	输出特征表示	tensor
		Attributes	embedding_node_type	元路径上抽取某一的节点类型	string
			embedding_dim	嵌入向量的大小	int
			metapath	元路径	List [Tuple [string, string, string]]
			walk_length	游走的长度	int
			context_size	正例的上下文窗口大小	int
			walks_per_node	对于每个节点进行游走的次数	int
			num_negative_samples	对于每个正例，负采样的数量	int

SetTransformerEncoder 运算操作定义见表 114。

表 114　SetTransformerEncoder 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
SetTransformerEncoder	基于注意力机制的置换不变神经网络的编码运算符	Input	g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor
			X	输入特征向量	tensor
		Output	Y	输出特征向量	tensor
		Attributes	d_model	模型的隐藏维度	int
			n_heads	注意力头的数目	int
			d_heads	每个注意力头的隐藏大小	int
			d_ff	FFN 层的核大小	int
			n_layers	可选，模型层数	int
			block_type	可选，包含 SAB 模块或 ISAB 模块	string
			m	可选，Induced 向量的数目（如果包含 ISAB 模块）	int
			dropout	可选，每个子层的 dropout 率	float
			dropout_head	可选，注意力头的 dropout 率	float

SetTransformerDecoder 运算操作定义见表 115。

表 115　SetTransformerDecoder 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
SetTransformerDecoder	基于注意力机制的置换不变神经网络的解码运算符	Input	g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor
			X	输入特征向量	tensor
		Output	Y	输出特征向量	tensor
		Attributes	d_model	模型的隐藏维度	int
			num_heads	注意力头的数目	int
			d_head	每个注意力头的隐藏大小	int
			d_ff	FFN 层的核大小	int
			n_layers	可选，模型层数	int
			k	PMA 层中种子向量的数目	int
			dropout	可选，每个子层的 dropout 率	float
			dropout_head	可选，注意力头的 dropout 率	float

GeomConv 运算操作定义见表 116。

表 116　GeomConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
GeomConv	几何图卷积神经网络，利用图表征空间中的几何信息，提取或“重构”传统聚合算子丢失的信息	Input	X	节点特征矩阵	tensor
			g	输入图	Graph
		Output	Y	输出张量（节点特征）	tensor
		Attributes	in_channels	可选，输入特征数量	int
			out_channels	可选，输出类别	int
			hidden_channels	可选，隐藏层数量	int
			num_divisions	子图数量	int
			dropout	dropout 的比率	float
			num_heads_layer_one	第一层 GAT 注意力头数	int
			num_heads_layer_two	第二层 GAT 注意力头数	int
			layer_one_ggcn_merge	对多子图的聚合操作	string
			layer_one_channel_merge	对多注意力头的聚合操作	string
			layer_two_ggcn_merge	对多子图的聚合操作	string
			layer_two_channel_merge	对多注意力头的聚合操作	string

EGNNConv 运算操作定义见表 117。

表 117　EGNNConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
EGNNConv	构建等变图神经网络运算符，使用特征张量、邻接矩阵、节点坐标特征矩阵和边特征矩阵（可选），输出特征张量和节点坐标特征张量	Input	X	节点特征矩阵	tensor
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			coord_feat	节点的坐标特征矩阵	tensor
			edge_weight	可选，边特征矩阵	tensor
		Output	Y	输出特征表示	tensor
			coord_feat_out	节点的坐标特征输出	tensor
		Attributes	in_channels	输入特征的维度	int
			hidden_channels	隐藏层特征维度	int
			out_channels	输出特征的维度	int
			edge_dim	可选，边特征维度	int

TWIRLSConv 运算操作定义见表 118。

表 118　TWIRLSConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
TWIRLSConv	构建基于迭代重加权最小二乘的图卷积运算符，使用特征张量和邻接矩阵，输出特征张量	Input	X	节点特征矩阵	tensor
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor SparseTensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	隐藏层特征维度	int
			prop_step	传播步数	int
			num_mlp_before	传播前的多层感知机层数	int
			num_mlp_after	传播后的多层感知机层数	int
			normalize	多层感知机内的规范化类型	string
			precond	是否使用预处理和归一化拉普拉斯矩阵	bool
			alp	梯度迭代步数超参数	float
			lam	权衡全局信息和局部信息的超参数	float
			attention	是否在传播中添加注意力层	bool
			tau	能量函数参数	float
			T	能量函数参数	float
			p	能量函数参数	float
			use_eta	是否在注意力的每个维度上添加可学习权重	bool
			attn_bef	是否在传播之前添加注意力层	bool
			dropout	多层感知机的丢弃率	float
			attn_dropout	注意力值的丢弃率	float
			inp_dropout	输入特征的丢弃率	float

GDO 运算操作定义见表 119。

表 119　GDO 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
GDO	采用分离提纯的思想，从节点的原始特征中抽离共有特征获得蒸馏增强特征，使用特征张量和邻接矩阵，输出特征张量（节点增强表示嵌入）	Input	X	节点原始特征矩阵	tensor Tuple [tensor, tensor]
			g	输入图，和 edge index 二选一	Graph
			edge_index	边索引，和g二选一	tensor
			edge_weight	可选，边权重矩阵	tensor
		Output	Y	输出蒸馏后增强特征表示	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			cached	可选，是否缓存第一次执行的对称归一化邻接矩阵的结果	bool
			add_self_loops	可选，是否给输入图添加自环	bool

Shift-GCNConv 运算操作定义见表 120。

表 120　Shift-GCNConv 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
Shift-GCNConv	使用移位方法+1×1 卷积进行图卷积运算，使用特征张量和邻接矩阵，输出表示张量	Input	X	节点特征矩阵	tensor Tuple [tensor, tensor]
			A	输入图的邻接矩阵	tensor
			edge_index	边索引，和 A 二选一	tensor SparseTensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int Tuple [int, int]
			out_channels	输出特征的维度	int
			global	可选，是否进行全局移位操作	bool
			bias	可选，是否添加偏置项	bool

Shift-GCNConv 算法伪代码见表 121。

表 121　Shift-GCNConv 算法伪代码

序号	Shift-GCNConv 算法	注释
1.	输入：in_channels, out_channels, X, A,global，bias	
2.	输出：Y	
3.	if in_channels != out_channels:	
4.	self.down = nn.Sequential(nn.Conv2d(in_channels, out_channels, kernel_size=1, W_init=nn.initializers.XavierNormal(), b_init='zeros', data_format='channels_first'), nn.BatchNorm2d(out_channels, gamma_init=nn.initializers.Constant(1), data_format='channels_first'))	如果输入特征维度和输出特征维度不相等，进行降采样（1×1 卷积运算）

表 121　Shift-GCNConv 算法伪代码（续）

序号	Shift-GCNConv 算法	注释
5.	self.down = nn.Sequential(nn.Conv2d(in_channels, out_channels, kernel_size=1, W_init=nn.initializers.XavierNormal(), b_init='zeros', data_format='channels_first'), nn.BatchNorm2d(out_channels, gamma_init=nn.initializers.Constant(1), data_format='channels_first'))	如果输入特征维度和输出特征维度不相等，进行降采样（1×1 卷积运算）
6.	else:	
7.	self.down = lambda x: x	
8.	index_array_in=np.empty(in_channels).astype(np.int)	定义特征索引向量
9.	index_array_out=np.empty(out_channels).astype(np.int)	定义特征索引向量
10.	if global:	进行全局移位时，计算移位后的特征索引向量
11.	for j in range(in_channels):	
12.	index_array_in[in_channels + j] = (in_channels + j + j*in_channels)%(in_channels)	
13.	for j in range(out_channels):	
14.	index_array_out[out_channels + j] = (out_channels + j - j*out_channels)%(out_channels)	进行全局移位时，计算移位后的特征索引向量
15.	self.shift_in = nn.Parameter(torch.from_numpy(index_array),requires_grad=False)	
16.	self.shift_out = nn.Parameter(torch.from_numpy(index_array),requires_grad=False)	
17.		
18.	else:	进行局部移位时，计算移位后的特征索引向量
19.	A[A==0] = 0	进行局部移位时，计算移位后的特征索引向量 对特征向量进行第一次移位操作
20.	A[A!=0] = 1	
21.	partition = np.sum(A[i])	
22.	channel_per_partition_in = in_channels//partition	
23.	channel_per_partition_out = out_channels//partition	
24.	current_A = A[0]	
25.	current_A[0] = 0	
26.	neighbors = np.nonzero(current_A)[0]	
27.	for j in range(int(partition)):	
28.	if j == 0:	
29.	index_array_in[int(channels):int(channels + channel_per_partition_in)] = np.arange(channels,channels + channel_per_partition_in).astype(np.int)	
30.	index_array_out[int(channels):int(channels + channel_per_partition_out)] = np.arange(channels,channels + channel_per_partition_out).astype(np.int)	
31.	else:	
32.	index_array_in[int(channels + j*channel_per_partition_in) : int(channels + (j+1)*channel_per_partition_in)] = np.arange(neighbors[j-1]*channels + j*channel_per_partition_in, neighbors[j-1]*channels + (j+1)*channel_per_partition_in).astype(np.int)	
33.	index_array_out[int(channels + j*channel_per_partition_out) : int(channels + (j+1)*channel_per_partition_out)] = np.arange(neighbors[j-1]*channels + j*channel_per_partition_out, neighbors[j-1]*channels + (j+1)*channel_per_partition_out).astype(np.int)	
34.	self.shift_in = nn.Parameter(torch.from_numpy(index_array_in),requires_grad=False)	

表 121　Shift-GCNConv 算法伪代码（续）

序号	Shift-GCNConv 算法	注释
35.	self.shift_out = nn.Parameter(torch.from_numpy(index_array_out),requires_grad=False)	
36.	X = torch.index_select(X, 1, self.shift_in)	
37.		
38.	if bias:	添加偏差
39.	self.linear_bias = nn.Parameter(torch.zeros(1, 1, out_channels, requires_grad=True, device='cuda'), requires_grad=True)	添加偏差 进行第二次移位操作
40.	X = X+self.linear_bias	
41.	X = torch.index_select(X, 1, self.shift_out)	
42.	X = X + down(X)	对特征向量进行降采样
43.	Y = relu(X)	
44.	return Y	

7.2.3　池化算子

图池化算子允许图神经网络处理可变大小的图，并能够在图级别进行有效的信息汇总，用于图分类、图嵌入、图回归等任务。图池化主要分为局部池化和全局池化两种形式。

a)　局部池化：局部池化层通过下采样来粗化图，类似于在卷积神经网络中的池化层。

b)　全局池化：全局池化层，也称为读出层，提供整个图的固定大小表示。全局池化层是排列不变的，使图节点和边的排列顺序的变化不会改变最终输出。

池化算子运算操作定义具体见表 122 ~表 143。

ASA_pool 运算操作定义见表 122。

表 122　ASA_pool 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
ASA_pool	自适应结构感知池化运算符	Input	X	节点特征矩阵	tensor
			g	输入图，和 edge_index 二选一	Graph BatchGraph
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			batch	可选，每一个节点的批次所属	tensor
		Output	Y	池化后的表示	tensor
			edge_index	粗化后的边索引	tensor
			edge_weight	可选，粗化后的边权重矩阵	tensor
			batch	粗化后的批次向量	tensor
			perm	池化后保留的节点的前 Topk 节点索引	tensor
		Attributes	in_channels	每个输入节点类型的特征维度	int
			ratio	图池化比率	float
			GNN	可选，使用聚类内部属性的图神经网络层	Model
			dropout	可选，归一化注意力系数的丢弃概率	float
			negative_slope	可选，负斜率的 LeakyReLU 角	float
			add_self_loops	可选，是否添加自环	bool

edge_pool 运算操作定义见表 123。

表 123　edge_pool 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
edge_pool	利用计算边分数进行池化的边池化运算符	Input	X	节点特征矩阵	tensor
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor
			batch	批量向量，用于将每个节点分配给一个特定批量	tensor
		Output	Y	池化后的表示	tensor
			edge_index	粗化后的边索引	tensor
			batch	批量向量，用于将每个节点分配给一个特定批量	tensor
		Attributes	in_channels	输入特征的维度	int
			edge_score_method	可选，用于从原始边分数计算边分数的函数方法	string
			dropout	可选，丢弃率	float
			add_to_edge_score	可选，被添加到边分数上的一个值	float

SAG_pool 运算操作定义见表 124。

表 124　SAG_pool 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
SAG_pool	通过边分数进行计算的边收缩运算符	Input	X	节点特征矩阵	tensor
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor
			edge_weight	可选，边特征矩阵	tensor
			batch	可选，每一个节点的批次所属	tensor
			attn	可选，当这个参数被提供时，该矩阵会被用来计算注意力分数，而不是使用节点特征矩阵	tensor
		Output	Y	池化后的表示	tensor
			edge_index	粗化后的边索引	tensor
			edge_weight	粗化后的边权重矩阵	tensor

表 124　SAG_pool 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
SAG_pool	通过边分数进行计算的边收缩运算符	Output	Batch	粗化后的批次向量	Tensor
			perm	池化后保留的节点的前 Topk 节点索引	tensor
			score[perm]	Topk 索引对应分数	tensor
		Attributes	in_channels	每个输入节点类型的特征维度	float
			ratio	图池比率	float
			GNN	用于计算投影得分的图神经网络层	Model
			min_score	可选，最小节点得分	float
			multiplier	可选，在池化后特征被乘以的系数	float
			nonlinearity	可选，使用的非线性函数	string

TopK_pool 运算操作定义见表 125。

表 125　TopK_pool 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
TopK_pool	Top-k 池化运算符	Input	X	节点特征矩阵	tensor
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor
			edge_weight	可选，边特征矩阵	tensor
			batch	可选，每一个节点的批次所属	tensor
			attn	可选，当这个参数被提供时，该矩阵会被用来计算注意力分数，而不是使用节点特征矩阵	tensor
		Output	Y	池化后的节点嵌入	tensor
			edge_index	粗化后的边索引	tensor
			edge_weight	粗化后的边权重矩阵	tensor
			batch	粗化后的批次向量	tensor

表 125　TopK_pool 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
TopK_pool	Top-k 池化运算符	Output	perm	池化后保留的节点的前 Topk 节点索引	Tensor
			score	Topk 索引对应分数	tensor
		Attributes	in_channels	每个输入节点类型的特征维度	float
			ratio	图池比率	float
			GNN	用于计算投影得分的图神经网络层	Model
			min_score	可选，最小节点得分	float
			multiplier	可选，在池化后特征被乘以的系数	float
			nonlinearity	可选，使用的非线性函数名称	string

average_pool 运算操作定义见表 126。

表 126　average_pool 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
average_pool	根据聚类定义的平均节点特征池化运算符	Input	X	节点特征矩阵	tensor
			g	输入图	Graph
			cluster	可选，每一个节点的所属集群	tensor
		Output	g	输出池化后的图	Graph

max_pool 运算操作定义见表 1267。

表 127　max_pool 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
max_pool	对节点做最大池化的运算，最终节点特征由同一聚类内所有节点的最大特征定义，节点位置被平均	Input	X	节点特征矩阵	tensor
			g	输入图	Graph
			cluster	可选，每一个节点的所属集群	tensor
		Output	g	输出池化后的图	Graph

min_pool 运算操作定义见表 128。

表 128　min_pool 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
min_pool	对节点做最小池化的运算，最终节点特征由同一聚类内所有节点的最小特征定义，节点位置被平均	Input	X	节点特征矩阵	tensor
			g	输入图	Graph
			cluster	可选，每一个节点的所属集群	tensor
		Output	g	输出池化后的图	Graph

sum_edges 聚合操作定义见表 129。

表 129　sum_edges 聚合操作

运算操作	描述	字段	关键字	定义	数据类型
sum_edges	聚合边特征，并按聚合类型“和”生成图级表示	Input	X	边特征	tensor
		Output	graph_attr	图特征	tensor

max_edges 聚合操作定义见表 130。

表 130　max_edges 聚合操作

运算操作	描述	字段	关键字	定义	数据类型
max_edges	聚合边特征，并按聚合类型“最大”生成图级表示	Input	X	边特征	tensor
		Output	graph_attr	图特征	tensor

avg_edges 聚合操作定义见表 131。

表 131　avg_edges 聚合操作

运算操作	描述	字段	关键字	定义	数据类型
avg_edges	聚合边特征，并按聚合类型“平均”生成图级表示	Input	X	边特征	tensor
		Output	graph_attr	图特征	tensor

节点特征 topk_nodes 操作定义见表 132。

表 132　节点特征 topk_nodes 操作

运算操作	描述	字段	关键字	定义	数据类型
topk_nodes	通过节点特征上的图 top-k 返回图级表示	Input	X	输入的节点特征	tensor
			k	topk 的节点数量	int
			sortby	所依照的排序基础。如果为 null 或-1，则按所有特征独立排序	int
		Output	graph_attr	图特征	tensor

边特征 topk_edges 操作定义见表 133。

表 133　边特征 topk_edges 操作

运算操作	描述	字段	关键字	定义	数据类型
topk_edges	通过边特征上的图 top-k 返回图级表示	Input	X	输入的边特征	tensor
			k	topk 的边数量	int
			sortby	所依照的排序基础。如果为 null 或-1，则按所有特征独立排序	int
		Output	graph_attr	图特征	tensor

global_add_pool 运算操作定义见表 134。

表 134　global_add_pool 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
global_add_pool	在整个节点维度上添加节点特征的运算符	Input	X	节点特征矩阵	tensor
			g	可选，输入图	Graph
			batch	可选，每一个节点的批次所属	tensor
			size	可选，批次数目	int
		Output	Y	输出表示	tensor

global_max_pool 运算操作定义见表 135。

表 135　global_max_pool 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
global_max_pool	通过获取整个节点维度上的通道级最大值进行最大池化的运算符	Input	X	节点特征矩阵	tensor
			g	可选，输入图	Graph
			batch	可选，每一个节点的批次所属	tensor
			size	可选，批次数目	int
		Output	Y	输出表示	tensor

global_mean_pool 运算操作定义见表 136。

表 136　global_mean_pool 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
global_mean_pool	通过对整个节点维度上的节点特征求平均进行均值池化的运算符	Input	X	节点特征矩阵	tensor
			g	可选，输入图	Graph
			batch	可选，每一个节点的批次所属	tensor
			size	可选，批次数目	int
		Output	Y	输出表示	tensor

global_sort_pool 运算操作定义见表 137。

表 137　global_sort_pool 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
global_sort_pool	通过对节点特征进行排序进行全局池化的运算符	Input	X	节点特征矩阵	tensor
			g	输入图，和 edge_index 二选一	Graph

表 137　global_sort_pool 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
global_sort_pool	通过对节点特征进行排序进行全局池化的运算符	Input	edge_index	边索引，和 g 二选一	Tensor
			batch	可选，每一个节点的批次所属	tensor
		Output	Y	输出表示	tensor
		Attributes	k	每个图要保留的节点数	int

global_min_pool 运算操作定义见表 138。

表 138　global_min_pool 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
global_min_pool	通过获取整个节点维度上的通道级最小值进行最小池化的运算符	Input	X	节点特征矩阵	tensor
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，至少和 g 二选一	tensor
			batch	可选，每一个节点的批次所属	tensor
		Output	Y	输出表示	tensor

global_attention 运算操作定义见 139。

表 139　global_attention 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
global_attention_pool	全局 soft attention 池化运算符	Input	X	输入节点特征	tensor
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor
			get_attention	可选，是否获取注意力值	bool
		Output	Y	输出表示	tensor
		Attributes	gate_nn	通过对节点特征进行映射来计算注意力得分的神经网络	Model
			feat_nn	将特征结合之前对每个特征进行处理的神经网络	Model

Set2Set 运算操作定义见表 140。

表 140　Set2Set 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
Set2Set	一种基于内容的迭代注意力机制下的全局池化运算符	Input	X	节点特征矩阵	tensor
			index	可选，用于应用聚合操作的元素的索引，index 和 ptr 必须至少定义一个	tensor
			ptr	可选，如果提供，将基于 CSR 表示的排序输入来计算聚合。index 和 ptr 必须至少定义一个。	tensor
			dim_size	输出节点类型的特征维度	int
			dim	可选，在哪个维度上进行聚合	int
			max_num_elements	可选，单个聚合组内的最大元素数	int
			g	可选，输入图	Graph
		Output	Y	输出表示	tensor
		Attributes	in_channels	输入节点类型的特征维度	int
			num_iters	迭代次数	int
			num_layers	可选，模型层数	int

WeightAndSum 运算操作定义见表 141。

表 141　WeightAndSum 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
WeightAndSum	用于计算原子的重要性权重并执行加权总和的运算符	Input	X	节点特征矩阵	tensor
			g	用于并行处理多个分子的图对象	Graph
		Output	Y	各分子的特征表示	tensor
		Attributes	in_channels	输入特征的维度	int

dense_diff_pool 运算操作定义见表 142。

表 142　dense_diff_pool 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
dense_diff_pool	可微分池化运算符	Input	X	节点特征矩阵	tensor
			adj	邻接矩阵	tensor
			s	分配张量	tensor
			mask	可选，mask 矩阵	tensor
			normalize	可选，链接预测损失是否会被 adj.numel() 除	bool
		Output	Y	输出节点特征矩阵	tensor
			Y_adj	输出邻接矩阵	tensor
			link_loss	节点预测损失	tensor
			ent_loss	正交损失	tensor

dense_mincut_pool 运算操作定义见表 143。

表 143　dense_mincut_pool 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
dense_mincut_pool	最小切割池化运算符	Input	X	输入张量	tensor
			adj	邻接矩阵	tensor
			s	分配张量	tensor
			mask	可选，mask 矩阵	tensor
			temp	可选，softmax 的温度参数	float
		Output	Y	输出节点特征矩阵	tensor
			Y_adj	输出邻接矩阵	tensor
			link_loss	Mincut 损失	tensor
			ent_loss	正交损失	tensor

7.2.4　归一化算子

归一化算子用以调整和规范化数据的分布，提高模型的训练效率、稳定性和性能。主要包含节点特征归一化和消息归一化。节点特征归一化确保不同节点特征的值在相似的范围内；消息归一化平衡不同节点传递的信息量，提高学习效率和模型稳定性。

具体定义见表 144 ~表 151。

HeteroBatchNorm 运算操作定义见表 144。

表 144　HeteroBatchNorm 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
HeteroBatchNorm	按照批归一化的方式对异质图上的特征进行归一化，即每种点或边类型单独归一化	Input	X	节点特征矩阵	tensor
			type_vec	节点类型矩阵	tensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int
			num_types	类型的数量	int
			eps	可选，加到归一化分母上的值，用于保证稳定性	float
			momentum	可选，用于控制均值和方差更新速度	float
			affine	可选，是否加入可学习参数	bool
			track_running_stats	可选，是否记录运行中的均值与方差	bool

HeteroLayerNorm 运算操作定义见表 145。

表 145　HeteroLayerNorm 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
HeteroLayerNorm	按照层归一化的方式对异质图上的特征进行归一化，即每种点或边类型单独归一化	Input	X	节点特征矩阵	tensor
			type_vec	可选，节点类型矩阵	tensor
			type_ptr	定义节点类型的边界	tensor List [int]
		Output	Y	输出特征表示	tensor

表 145　HeteroLayerNorm 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
HeteroLayerNorm	按照层归一化的方式对异质图上的特征进行归一化，即每种点或边类型单独归一化	Attributes	in_channels	输入特征的维度	Int
			num_types	类型的数量	int
			eps	可选，加到归一化分母上的值，用于保证稳定性	float
			affine	可选，是否加入可学习参数	bool
			mode	可选，默认为“node”：每个节点特征被当做一个元素；若选择“graph”:每张图被当做一个元素	string

GraphSizeNorm 运算操作定义见表 146。

表 146　GraphSizeNorm 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
GraphSizeNorm	按照 Graph Size Normalization 的方式，根据节点数对每张图中的特征进行归一化	Input	X	节点特征矩阵	tensor
			batch	可选，描述每个节点从属的 batch	tensor
			batch_size	可选，batch 的大小	int
		Output	Y	输出特征表示	tensor

PairNorm 运算操作定义见表 147。

表 147　PairNorm 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
PairNorm	按照 Pair Normalization 的方式，对节点特征进行归一化	Input	X	节点特征矩阵	tensor
			batch	可选，描述每个节点从属的 batch	tensor
			batch_size	可选，batch 的大小	int
		Output	Y	输出特征表示	tensor
		Attributes	scale	可选，用于归一化的规模参数	float
			scale_individually	可选，是否独立计算归一化的 scale	bool
			eps	可选，加到归一化分母上的值，用于保证稳定性	float

MessageNorm 运算操作定义见表 148。

表 148　MessageNorm 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
MessageNorm	按照 Message Normalization 的方式，对聚合的消息进行归一化	Input	X	节点特征矩阵	tensor
			msg	消息矩阵	tensor
			p	用于归一化的归一化常量	float
		Output	Y	输出特征表示	tensor

表 148　MessageNorm 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
MessageNorm	按照 Message Normalization 的方式，对聚合的消息进行归一化	Attributes	learn_scale	可选，默认为 False；若设置为 True，则规模常量会调整为可学习参数	Bool

DiffGroupNorm 运算操作定义见表 149。

表 149　DiffGroupNorm 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
DiffGroupNorm	按照 Diff Group Normalization 的方式，通过一个分组为基础的软聚类节点划分来归一化节点特征	Input	X	节点特征矩阵	tensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int
			groups	分组的数量	int
			lamda	可选，用于平衡输入表示和归一化表示的参数	float
			eps	可选，加到归一化分母上的值，用于保证稳定性	float
			momentum	可选，用于控制均值和方差更新速度	float
			affine	可选，是否加入可学习参数	bool
			track_running_stats	可选，是否记录运行中的均值与方差	bool

MeanSubtractionNorm 运算操作定义见表 150。

表 150　MeanSubtractionNorm 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
MeanSubtractionNorm	结合层归一化的方式，以减去邻居节点特征均值的方式对特征进行归一化	Input	X	节点特征矩阵	tensor
			batch	可选，描述每个节点从属的 batch	tensor
			dim_size	可选，batch 的大小	int
		Output	Y	输出特征表示	tensor

GraphNorm 运算操作定义见表 151。

表 151　GraphNorm 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
GraphNorm	按照 Graph Normalization 的方式，对每张图中的特征进行归一化	Input	X	节点特征矩阵	tensor
			batch	可选，描述每个节点从属的 batch	tensor
			batch_size	可选，batch 的大小	int

表 151　GraphNorm 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
GraphNorm	按照 Graph Normalization 的方式，对每张图中的特征进行归一化	Output	Y	输出特征表示	Tensor
		Attributes	in_channels	输入特征的维度	int
			eps	可选，加到归一化分母上的值，用于保证稳定性	float

7.3　点级模型

7.3.1　概述

点级模型指的是应用于点级任务的模型。点级图神经网络模型主要利用图的连通关系以及特征矩阵，将存在关系的节点特征进行聚合并得到新的节点表征。根据其节点聚合的范围，将图神经网络分为一阶聚合、多阶聚合以及任意聚合三种类型。

7.3.2　一阶聚合

一阶聚合是指在聚合过程中仅仅包含一阶邻居，将与节点直接相连的其他节点的表征作为信息聚合，而后与节点的原始表征进行组合得到新的节点表征。

一阶聚合模型定义见表 152 ~表 169。

GCN 模型对原始的邻接矩阵添加自环，而后计算其拉普拉斯矩阵并进行归一化，最后通过得到的矩阵进行消息传递。模型定义见表 152。

表 152　GCN 模型定义

模型	描述	字段	关键字	定义	数据类型
GCN	根据添加自环以及归一化后的拉普拉斯矩阵进行消息传递	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int

GAT 模型在对节点信息进行聚合时，首先计算其各邻居节点对其的注意力分数，然后根据得到的注意力分数加权求和以进行信息的聚合。模型定义见表 153。

表 153　GAT 模型定义

模型	描述	字段	关键字	定义	数据类型
GAT	利用注意力机制进行带权重的消息传递	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
			attention_weights	可选，注意力权重	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			heads	注意力头数量	int

GraphSAGE 模型对节点的邻居进行采样，根据采样得到的节点进行消息传递。模型定义见表 154。

表 154　GraphSAGE 模型定义

模型	描述	字段	关键字	定义	数据类型
GraphSAGE	采样部分一阶邻居进行聚合操作	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			size	采样邻接矩阵大小	int

GIN 模型通过构造单射函数，使 GNN 获得了与 WL-test 相当的性能。它通过串联（CONCAT）的方式，将节点在所有迭代步骤中的表示聚合起来，以此来计算节点的最终表征。模型定义见表 155。

表 155　GIN 模型定义

模型	描述	字段	关键字	定义	数据类型
GIN	设计单射函数以及聚合过程使其达到 WL-test 的表达能力	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor

表 155　GIN 模型定义（续）

模型	描述	字段	关键字	定义	数据类型
GIN	设计单射函数以及聚合过程使其达到 WL-test 的表达能力	Input	edge_weight	可选，边权重矩阵	Tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			aggregation_type	聚合方法	string

GGNN 模型借鉴类似于 LSTM 的门控机制，以实现图神经网络中节点与邻居特征的聚合。模型定义见表 156。

表 156　GGNN 模型定义

模型	描述	字段	关键字	定义	数据类型
GGNN	构建门控图卷积模型，使用特征矩阵和邻接矩阵，输出节点表征	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			aetype	可选，边的类型	string

ResGatedgraph 模型将残差连接引入到门控图卷积神经网络中，以实现性能的增益。定义见表 157。

表 157　ResGatedgraph 模型定义

模型	描述	字段	关键字	定义	数据类型
ResGatedgraph	构建残差门控图卷积模型，使用特征矩阵和邻接矩阵，输出特征表征	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor

表 157　ResGatedgraph 模型定义（续）

模型	描述	字段	关键字	定义	数据类型
ResGatedgraph	构建残差门控图卷积模型，使用特征矩阵和邻接矩阵，输出特征表征	Attributes	in_channels	输入特征的维度	Int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int

GINE 模型将边的特征与残差连接引入到图卷积神经网络中，以实现性能的增益。定义见表 158。

表 158　GINE 模型定义

模型	描述	字段	关键字	定义	数据类型
GINE	改进的图同构模型，使用特征张量和邻接矩阵，输出特征张量	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	可选，边特征矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			size	邻接矩阵的大小	Tuple [int, int]

GaAN 模型在 GAT 的基础上，对不同的注意力头添加不同的权重参数以获取更优的性能。定义见表 159。

表 159　GaAN 模型定义

模型	描述	字段	关键字	定义	数据类型
GaAN	对不同的注意力头添加不同的权重系数	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			heads	注意力头数量	int

AM-GCN 模型根据节点特征构建一个 K-Nearest Neighbor（KNN）图。然后通过两个特有图卷积网络，分别在原始拓扑图和构建属性图上提取各自节点特征。再通过一个共同图卷积网络，在原始拓扑图

和构建属性图上分别提取节点特征求平均得到共同节点特征。最后采用注意力机制对三种节点特征进行自适应聚合，从而达到自适应提取网络拓扑信息以及节点特征信息的目的。模型定义见表 160。

表 160　AM-GCN 模型定义

模型	描述	字段	关键字	定义	数据类型
AM-GCN	自适应多通道图卷积网络	Input	X	节点特征矩阵	tensor
			edge_index_s	结构图边索引	tensor SparseTensor
			edge_index_f	特征图边索引	tensor SparseTensor
		Output	Y	输出张量	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			dropout	特征丢弃率	float
			hidden_channels1	第一隐藏层特征的维度	int
			hidden_channels2	第二隐藏层特征的维度	int

AM-GCN 算法伪代码见表 161。

表 161　AM-GCN 算法伪代码

序号	AM-GCN 算法	注释
1.	输入：in_channels, out_channels, hidden_channels1, hidden_channels2, dropout, X, edge_index_s, edge_index_f	
2.	输出：Y	
3.	SGCN1 = Sequential([GCNConv(in_channels,hidden_channels1), Dropout(dropout), GCNConv(hidden_channels2,hidden_channels1)])	独特图卷积网络 1，在拓扑图上进行卷积运算
4.	SGCN2 = Sequential([GCNConv(in_channels,hidden_channels1), Dropout(dropout), GCNConv(hidden_channels2,hidden_channels1)])	独特图卷积网络 2，在特征图上进行卷积运算
5.	CGCN = Sequential([GCNConv(in_channels,hidden_channels1), Dropout(dropout), GCNConv(hidden_channels2,hidden_channels1)])	共同图卷积网络，在拓扑图和特征图上进行卷积运算
6.	attention = Attention(hidden_channels2)	注意力模块，用于融合不同图卷积网络的表示
7.	MLP = Sequential([Linear(hidden_channels2, out_channels), LogSoftmax(dim=1)])	多层感知机，根据节点表示分类
8.	emb1 = SGCN1(X, edge_index_s)	使用独特图卷积网络 1，根据特征和拓扑图得到节点表示
9.	emb2 = SGCN2(X, edge_index_f)	使用独特图卷积网络 2，根据特征和特征图得到节点表示
10.	com1 = CGCN(X, edge_index_s)	使用共同图卷积网络，根据特征和拓扑图得到节点表示
11.	com2 = CGCN(X, edge_index_f)	使用共同图卷积网络，根据特征和特征图得到节点表示
12.	Xcom = (com1 + com2) / 2	将共同图卷积网络在拓扑图和特征图上分别得到的节点表示求平均得到共同节点表示

表 161　AM-GCN 算法伪代码（续）

序号	AM-GCN 算法	注释
13.	emb = stack([emb1, emb2, Xcom], dim=1)	将三种节点表示拼接
14.	emb = attention(emb)	对拼接后的三种节点表示求注意力分数，并融合得到最终表示
15.	Y = MLP(emb)	使用多层感知机计算得到输出
16.	return Y	

FAGCN 模型使用一个低通滤波器以及一个高通滤波器，用于分离节点特征中的低频信号以及高频信号。使用注意力机制来学习低频信号以及高频信号的比例并实现频率自适应地聚合低频信号以及高频信号。模型定义见表 162。

表 162　FAGCN 模型定义

模型	描述	字段	关键字	定义	数据类型
FAGCN	频率自适应图卷积网络模型	Input	X	节点特征矩阵	tensor
			edge_index	边索引	tensor SparseTensor
		Output	Y	输出张量	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			dropout	特征丢弃率	float
			hidden_channels	隐藏层特征的维度	int
			layer_num	FAConv 的层数	int
			eps	FAConv 的 epsilon 值	float

FAGCN 算法伪代码见表 163。

表 163　FAGCN 算法伪代码

序号	FAGCN 算法	注释
1.	输入：in_channels, out_channels, hidden_channels, dropout, X, edge_index, layer_num	
2.	输出：Y	
3.	layers = []	
4.	lin1 = Linear(in_channels, hidden_channels)	线性变换层 1
5.	for i in range(layer_num):	
6.	layers.append(FAConv(hidden_channels, eps, dropout))	频率自适应卷积层
7.	lin2 = Linear(hidden_channels, out_channels)	线性变换层 2
8.	h = dropout(X, p=dropout)	
9.	h = relu(lin1(h))	预线性变换
10.	h = dropout(X, p=dropout)	
11.	raw = h	存储预变换后的节点表示
12.	for i in range(num_layers):	
13.	h = layers[i](h)	使用频率自适应卷积层聚合节点特征
14.	h = eps*raw + h	
15.	Y = log_softmax(lin2(h), dim=1)	计算输出结果
16.	return Y	

GeomGCN 以集合神经网络获取图表征，保证连续空间的平稳性、局部性和组合型。其运算操作定义见表 164。

表 164　GeomGCN 模型定义

模型	描述	字段	关键字	定义	数据类型
GeomGCN	几何图卷积神经网络，利用图表征空间中的几何信息，提取或“重构”传统聚合算子丢失的信息。输入特征张量和邻接矩阵，输出特征张量	Input	X	节点特征矩阵	tensor
			g	图	Graph
		Output	Y	输出张量（节点特征）	tensor
		Attributes	num_input_features	可选，输入特征数量	int
			num_output_classes	可选，输出类别	int
			num_hidden	可选，隐藏层数量	int
			num_divisions	子图数量	int
			dropout_rate	dropout 的比率	float
			num_heads_layer_one	第一层 GAT 注意力头数	int
			num_heads_layer_two	第二层 GAT 注意力头数	int
			layer_one_ggcn_merge	对多子图的聚合操作	string
			layer_one_channel_merge	对多注意力头的聚合操作	string
			layer_two_ggcn_merge	对多子图的聚合操作	string
			layer_two_channel_merge	对多注意力头的聚合操作	string

HGAT 基于一种 HIN 框架的异质图注意力网络模型来对短文本进行建模。其具体运算操作见表 165。

表 165　HGAT 模型定义

模型	描述	字段	关键字	定义	数据类型
HGAT	基于双层注意力机制的半监督短文本分类模型	Input	X_dict	节点特征字典，用于存储每一种节点类型的节点特征信息。	Dict [string, tensor]
			edge_index_dict	边索引字典，至少和 g 二选一	Dict [Tuple [string, string, string], tensor] Dict [Tuple [string, string, string], SparseTensor]
			num_nodes_dict	节点数量字典，用于储存每个类型的节点的数量	Dict [string, int]
		Output	out_dict	输出概率值	Dict [string, tensor]
		Attributes	in_channels	输入特征的维度	int Dict [string, int]
			out_channels	输出特征的维度	int
			metadata	异质图的元数据，即由一个字符串列表给出的节点类型和由一个字符串三元组列表给出的边类型	Tuple [List [string]，List [Tuple [string, string, string]]]

HGAT 算法伪代码见表 166。

表 166　HGAT 算法伪代码

序号	HGAT 算法	注释
1.	输入：x_dict, edge_index_dict, num_nodes_dict	
2.	输出：out_dict	
3.	Type_Attention_Value_dict = {}	特征注意力
4.	for edge_type, edge_index in edge_index_dict.items():	获取元路径邻居节点
5.	Type_Attention_Value_dict[edge_type] = []	
6.	for edge_type, edge_index in edge_index_dict.items():	
7.	src_type, _, dst_type = edge_type	
8.	src = edge_index[0,:]	
9.	dst = edge_index[1,:]	
10.	h_l = self.Linear_dict_l[src_type](x_dict[src_type])	
11.	h_r = self.Linear_dict_r[dst_type](x_dict[dst_type])	
12.	h_l = tlx.gather(h_l,src)	
13.	h_r = tlx.gather(h_r,dst)	
14.	Type_Attention_Value = h_l + h_r	
15.	Type_Attention_Value = self.leakyReLu(Type_Attention_Value)	
16.	Type_Attention_Value_dict[edge_type].append(Type_Attention_Value)	
17.	x_value, edge_index, edge_value = to_homograph(x_dict,edge_index_dict,num_nodes_dict, Type_Attention_Value_dict)	异质图转同质图
18.	alpha = segment_softmax(edge_value,edge_index[1,:],num_segments=None)	
19.	alpha = self.dropout(alpha)	
20.	src = edge_index[0,:]	
21.	dst = edge_index[1,:]	
22.	h_l = self.Linear_l(x_value)	
23.	h_r = self.Linear_r(x_value)	
24.	h_l = tlx.gather(h_l,src)	
25.	h_r = tlx.gather(h_r,dst)	
26.	Node_Attention_Value = (h_l + h_r)*alpha	节点注意力
27.	beta = segment_softmax(Node_Attention_Value,edge_index[1,:],num_segments=None)	
28.	out = unsorted_segment_sum(beta*tlx.gather(x_value,edge_index[0,:]),edge_index[0,:])	
29.	out_dict = to_heterograph(self.dropout(out),edge_index,num_nodes_dict)	同质图转异质图
30.	return out_dict	

GCNII 在原有的图卷积神经网络上增加初始连接和恒等变换两个模块以缓解过平滑现象。模型定义见表 167。

表 167　GCNII 模型定义

模型	描述	字段	关键字	定义	数据类型
GCNII	GCNII 利用 增加初始连接和恒等变换两个模块以缓解 GCN 中的过平滑现象	Input	X	节点特征矩阵	tensor
			edge_index	边索引	tensor SparseTensor
		Output	Y	输出张量	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			dropout	特征丢弃率	float
			hidden_channels	隐藏层特征的维度	int
			num_layers	GCNIIConv 的层数	int

表 167　GCNII 模型定义（续）

模型	描述	字段	关键字	定义	数据类型
GCNII	GCNII 利用 增加初始连接和恒等变换两个模块以缓解 GCN 中的过平滑现象	Attributes	alpha	GCNIIConv 中初始连接比例	Float
			lamda	GCNIIConv 中恒等映射比例	float

GNNII 算法伪代码见表 168。

表 168　GCNII 算法伪代码

序号	GCNII 算法	注释
1.	输入：in_channels, out_channels, hidden_channels, dropout, X, edge_index, nlayer, alpha, lamda	
2.	输出：Y	
3.	layers = []	
4.	lin1 = Linear(in_channels, hidden_channels)	线性变换层 1
5.	for i in range(layer_num): layers.append(GCNIIConv(hidden_channels, hidden_channels))	GCNII 卷积层
6.	lin2 = Linear(hidden_channels, out_channels)	线性变换层 2
7.	h = dropout(X, p=dropout)	
8.	h = relu(lin1(h))	
9.	h0 = h	存储初始特征
10.	for i in range(nlayer): 　　h = dropout(X, p=dropout) 　　beta = log(lamda / (i+1) + 1) 　　h = layers[i](X, edge_index, alpha, h0, beta) 　　h = relu(h)	使用 GCNII 卷积层聚合节点特征
11.	h = dropout(h, p=dropout)	
12.	Y = log_softmax(self.lin2(h), dim=1)	计算输出结果
13.	return Y	

7.3.3　多阶聚合

多阶聚合指的是在聚合过程中不只包含一阶邻居，根据其拓扑结构，选择更高阶的邻居进行聚合。

多阶聚合模型定义见表 169 ~表 184。

SGC 模型将邻接矩阵进行 K 次幂，获取了节点的 K 阶邻居信息，而后进行消息传递。模型定义见表 169。

表 169　SGC 模型定义

模型	描述	字段	关键字	定义	数据类型
SGC	去除 GCN 中的非线性层，仅保留 k 层消息传递模型以进行信息聚合	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			K	跳数（即取到 K 阶邻居）	int

APPNP 模型使用 personalized PageRank 算法进行随机游走，得到节点的邻域信息以进行消息传递以及信息聚合。模型定义见表 170。

表 170　APPNP 模型定义

模型	描述	字段	关键字	定义	数据类型
APPNP	利用随机游走得到节点邻域以进行消息传递	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			K	随机游走迭代次数	int
			alpha	重启传送概率参数	float

GPRGNN 模型使用 generalized PageRank 方法使图神经网络适用于异配图，先计算 K 轮消息传递，而后使用 K 层特征的线性组合作为表征，线性组合的参数通过学习获得。模型定义见表 171。

表 171　GPRGNN 模型定义

模型	描述	字段	关键字	定义	数据类型
GPRGNN	使用 K 轮消息传递表征的线性组合得到最终表征	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			K	消息传递的轮数	int

ChebNet 模型中使用切比雪夫多项式代替谱域的卷积核，谱域的卷积核的取值是与特征值相关的函数，以避免显式的特征值分解，然后来用切比雪夫多项式来逼近这个函数。模型定义见表 172。

表 172　ChebNet 模型定义

模型	描述	字段	关键字	定义	数据类型
ChebNet	使用切比雪夫多项式近似卷积核	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			K	切比雪夫滤波器大小	int

JKnet 模型使用节点进行聚合时，极大受到所在子结构影响（若为中心节点如微博大 V，则很少几步就可以得到大量邻居信息），不同的节点应该具有不同的聚合步数。该模型先计算聚合 1-K 步的所有结果，而后自适应的选择需要的步数进行组合。模型定义见表 173。

表 173　JKnet 模型定义

模型	描述	字段	关键字	定义	数据类型
JKnet	不同节点采用自适应的聚合步数	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			K	最大聚合步数	int

DCNN 模型使用 diffusion 对每一个节点采用 H 个 hop 的矩阵进行表示，每一个 hop 表示其邻近信息，以更好得获取局部信息。模型定义见表 174。

表 174　DCNN 模型定义

模型	描述	字段	关键字	定义	数据类型
DCNN	利用不同 hop 的信息进行聚合得到节点表征	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor

表 174　DCNN 模型定义（续）

模型	描述	字段	关键字	定义	数据类型
DCNN	利用不同 hop 的信息进行聚合得到节点表征	Input	edge_weight	可选，边权重矩阵	Tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			K	Hop 数	int

Line 模型使用节点与其共同邻居的相似性表示两节点的相似性，以构造二阶的相邻关系。模型定义见表 175。

表 175　Line 模型定义

模型	描述	字段	关键字	定义	数据类型
Line	利用节点与其共同邻居的相似性表示两节点的相似性	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int

HPN 模型通过语义传播机制和语义融合机制来缓解语义混淆。语义传播机制通过赋予节点的局部语义适当权重，改善节点层级聚合过程，使节点嵌入即使在更深的 HeteroGNN 架构中也能保持区分度。语义融合机制致力于学习元路径的重要性，并对其进行合理融合。模型定义见表 176。

表 176　HPN 模型定义

模型	描述	字段	关键字	定义	数据类型
HPN	异质图传播网络，改善元路径和注意力机制在消息传递过程中的聚合过程，以获得更鲁棒的表示	Input	X_dict	节点特征字典，用于存储每一种节点类型的节点特征信息	Dict [string, tensor]
			g	输入图，至少和 edge_index_dict 二选一	HeteroGraph

表 176　HPN 模型定义（续）

模型	描述	字段	关键字	定义	数据类型
HPN	异质图传播网络，改善元路径和注意力机制在消息传递过程中的聚合过程，以获得更鲁棒的表示	Input	edge_index_dict	边索引字典，至少和 g 二选一	Dict [Tuple [string, string, string], tensor] Dict [Tuple [string, string, string], SparseTensor]
		Output	Y	输出张量	Dict [string, tensor]
		Attributes	in_channels	输入特征的维度	int Dict [string, int]
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			heads	可选，多头注意力的数量	int

HPN 模型伪代码见表 177。

表 177　HPN 算法伪代码

序号	HPN 算法	注释
1.	输入：in_channels, out_channels, metadata, iter_K, alpha, negative_slope, drop_rate, x_dict, edge_index_dict, num_nodes_dict	
2.	输出：out_dict	
3.	for edge_type in metadata[1]:	
4.	src_type, _, dst_type = edge_type	
5.	edge_type = '__'.join(edge_type)	
6.	self.appnp_dict[edge_type] = APPNPConv(in_channels[src_type], out_channels, iter_K, alpha, drop_rate)	
7.	out_dict = {}	
8.	for node_type, _ in x_dict.items():	
9.	out_dict[node_type] = []	
10.	for edge_type, edge_index in edge_index_dict.items():	
11.	src_type, _, dst_type = edge_type	
12.	edge_type = '__'.join(edge_type)	
13.	out = self.appnp_dict[edge_type](x_dict[src_type], edge_index, num_nodes_dict[dst_type])	
14.	out = relu(out)	
15.	out_dict[dst_type].append(out)	
16.	for node_type, outs in out_dict.items()	
17.	outs = stack(outs)	
18.	w = mean(linear(out_channels, 1)(outs, dim=1))	
19.	beta = softmax(w, dim=0)	
20.	beta = beta.unsqueeze(-1)	
21.	out_dict[node_type] = sum(beta * outs, dim=0)	
22.	return out_dict	

GAMLP 模型通过 recursive attention 或 JK attention，以学习的方式为各个节点提供个性化的针对 K 轮消息传递结果的聚合权重。模型定义见表 178。

表 178　GAMLP 模型定义

模型	描述	字段	关键字	定义	数据类型
GAMLP	为不同节点学习个性化的线性组合 K 轮消息传递结果的权重	Input	X	节点特征矩阵	tensor
			y_train	独热编码的节点标签矩阵，仅含训练集节点信息	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			K	消息传递的轮数	int
			β	节点标签分支的权重	float
			attention_type	可选，0 表示使用 Recursive，1 表示使用 JK	bool

GNN-LF/HF 是以统一框架理解和分析不同 GNN，开发两个考虑可调节图核的新颖目标函数，分别显示低通或高通滤波能力。模型定义见表 179。

表 179　GNN-LF/HF 模型定义

模型	描述	字段	关键字	定义	数据类型
GNN_LF/HF	多组件图的卷积协同过滤模型	Input	X	节点特征矩阵	tensor
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
		Output	X	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	Dict [string, int]
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int

GNN-LF/HF 伪代码见表 180。

表 180　GNN-LF/HF 算法伪代码

序号	GNN-LF/HF 算法	注释
1.	输入：in_channels, out_channels, g, alpha, beta, mu, x, niter	
2.	输出：Y	输出特征表示
3.	layer_inner = self.act_fn(self.fcs[0](self.dropout(x)))	
4.	for fc in self.fcs[1:-1]:	
5.	layer_inner = self.act_fn(fc(layer_inner))	
6.	local_logits = self.fcs[-1](dropout(layer_inner))	
7.	propagation = LFExact(graph.adj_matrix, alpha, mu)	GNN-LF closed
8.	propagation = LFPowerIteration(g.adj_matrix, alpha, mu)	GNN-LF itera
9.	propagation = HFExact(g.adj_matrix, alpha, beta)	GNN-HF closed
10.	propagation = HFPowerIteration(g.adj_matrix, alpha, beta)	GNN-HF itera
11.	final_logits = self.propagation(local_logits, idx)	
12.	Return log_softmax(final_logits, dim=-1)	

HeCo 是利用对比学习的自监督方式训练异质图神经网络，采用交叉视图对比机制，从两个视图中提取正、负嵌入，使两个视图能够协作监督，学习更高级别的节点嵌入。模型定义见表 181。

表 181　HeCo 模型定义

模型	描述	字段	关键字	定义	数据类型
HeCo	融合对比学习的自监督异质图神经网络	Input	X	存储每一种节点类型的节点特征信息。	List [tensor]
			pos	正样本	tensor
			mps	元路径邻接矩阵	List [tensor]
			nei_index	邻居索引列表	List [tensor]
		Output	loss	输出损失值	tensor
		Attributes	hidden_channels	隐藏层维度	int
			feats_dim_list	不同类型特征的维度	List [int]
			feat_drop	特征丢弃率	float
			sample_rate	采样率	float
			nei_num	邻居类型数量	int
			tau	温度值	float
			lam	lamda 值	float
			attn_drop	attention 聚合丢弃率	float
			p	元路径数量	int

HeCo 算法伪代码见表 182。

表 182　HeCo 算法伪代码

序号	HeCo 算法	注释
1.	输入：feats, pos, mps, nei_index	输入特征、正样本、原路径、邻居索引
2.	输出：loss	
3.	h_all = []	
4.	for i in range(len(feats)):	
5.	h_all.append(F.elu(self.feat_drop(self.fc_list[i](feats[i]))))	

表 182　HeCo 算法伪代码（续）

序号	HeCo 算法	注释
6.	z_mp = self.mp(h_all[0], mps)	获取元路径级嵌入
7.	z_sc = self.sc(h_all, nei_index)	获取模式级嵌入
8.	loss = self.contrast(z_mp, z_sc, pos)	两级融合对比学习
9.	return loss	
10.	embeds = []	self.mp(h_all[0], mps)展开介绍
11.	for i in range(self.P):	
12.	mps[i] = mps[i]	
13.	embeds.append(self.node_level[i](h, mps[i]))	GCN 聚合同元路径节点特征
14.	z_mp = self.att(embeds)	attention 聚合不同元路径节点特征
15.	return z_mp	
16.	embeds = []	self.sc(h_all, nei_index)展开介绍
17.	for i in range(self.nei_num):	
18.	sele_nei = []	选择邻居节点
19.	sample_num = self.sample_rate[i]	
20.	for per_node_nei in nei_index[i]:	
21.	if len(per_node_nei) >= sample_num:	
22.	select_one = torch.tensor(np.random.choice(per_node_nei, sample_num, replace=False))[np.newaxis]	
23.	else:	
24.	select_one = torch.tensor(np.random.choice(per_node_nei, sample_num, replace=True))[np.newaxis]	
25.	sele_nei.append(select_one)	
26.	sele_nei = torch.cat(sele_nei, dim=0).cuda()	
27.	one_type_emb = F.elu(self.intra[i](sele_nei, nei_h[i + 1], nei_h[0]))	GAT 聚合邻居节点
28.	embeds.append(one_type_emb)	
29.	z_mc = self.inter(embeds)	self-attention 聚合
30.	return z_mc	
31.	z_proj_mp = self.proj(z_mp)	self.contrast(z_mp, z_sc, pos)展开介绍
32.	z_proj_sc = self.proj(z_sc)	特征投影
33.	matrix_mp2sc = self.sim(z_proj_mp, z_proj_sc)	计算相似度
34.	matrix_sc2mp = matrix_mp2sc.t()	
35.	matrix_mp2sc = matrix_mp2sc/(torch.sum(matrix_mp2sc, dim=1).view(-1, 1) + 1e-8)	计算损失
36.	lori_mp = -torch.log(matrix_mp2sc.mul(pos).sum(dim=-1)).mean()	
37.	matrix_sc2mp = matrix_sc2mp / (torch.sum(matrix_sc2mp, dim=1).view(-1, 1) + 1e-8)	
38.	lori_sc = -torch.log(matrix_sc2mp.mul(pos).sum(dim=-1)).mean()	
39.	return self.lam * lori_mp + (1 - self.lam) * lori_sc	

HACUD是引入基于层次注意的套现用户检测模型，以解决套现欺诈问题。模型定义见表183。

表 183　HACUD 模型定义

模型	描述	字段	关键字	定义	数据类型
HACUD	基于分层注意力机制的 Cash-out 用户检测模型	Input	X_dict	节点特征字典，用于存储每一种节点类型的节点特征信息	Dict [string, tensor]

表 183　HACUD 模型定义（续）

模型	描述	字段	关键字	定义	数据类型
HACUD	基于分层注意力机制的 Cash-out 用户检测模型	Input	g	输入图，至少和 edge_index_dict 二选一	HeteroGraph
			edge_index_dict	边索引字典，至少和 g 二选一	Dict [Tuple [string, string, string], tensor] Dict [Tuple [string, string, string], SparseTensor]
			metadata	异质图的元数据，即由一个字符串列表给出的节点类型和由一个字符串三元组列表给出的边类型	Tuple [List [string], List [Tuple [string, string, string]]]
		Output	p	输出概率值	tensor
		Attributes	in_channels	输入特征的维度	int Dict [string, int]
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int

HACUD 算法伪代码见表 184。

表 184　HACUD 算法伪代码

序号	HACUD 算法	注释
1.	输入：hidden_channels, metadata, node_type, node_id , x_dict, edge_index_dict	
2.	输出：e_u	
3.	aggregated_features = {}	
4.	for meta_path, edge_index in edge_index_dict.items():	获取元路径邻居节点
5.	_, edge_types = metadata	
6.	for edge_type in edge_types:	
7.	src_type, _, dst_type = edge_type	
8.	if src_type == node_type:	
9.	if edge_type in edge_index_dict:	
10.	edge_indices = edge_index_dict[edge_type]	
11.	node_edge_indices = (edge_indices[0] == node_id).nonzero(as_tuple=True)[0]	
12.	if len(node_edge_indices) == 0:	
13.	continue	
14.	neighbors = edge_indices[1][node_edge_indices]	
15.	neighbor_features = x_dict[dst_type][neighbors]	
16.	aggregated_features[edge_type] = neighbor_features.sum(dim=0)	
17.	x_features = x_dict[node_type][node_id]	
18.	h_u = Linear(x_features.shape(0), hidden_channels)(x_features) + Parameter(hidden_channels)	
19.	fusion_features = {}	特征融合
20.	for type, feature in aggregated_features:	
21.	h_rou = Linear(feature.shape(0), hidden_channels)(feature) + Parameter(hidden_channels)	
22.	aggregated_features[type] = h_rou	
23.	cat_feature = cat(h_u, h_rou)	

表 184　HACUD 算法伪代码（续）

序号	HACUD 算法	注释
24.	fusion_feature = relu(Linear(2*hidden_channels, hidden_channels)(cat_feature) + Parameter(hidden_channels))	
25.	fusion_features[type] = fusion_feature	
26.	att_features = {}	特征注意力
27.	for type, feature in fusion_features:	
28.	cat_feature = cat(h_u, feature)	
29.	v_rou = relu(Linear(hidden_channels, hidden_channels)(cat_feature) + Parameter(hidden_channels))	
30.	alpha_rou = relu(Linear(hidden_channels, hidden_channels)(v_rou) + Parameter(hidden_channels))	
31.	alpha_rou = softmax(alpha_rou, dim=0)	
32.	att_features[type] = alpha_rou * feature	
33.	attention_vectors = Parameter((len(att_features), hidden_channels))	元路径注意力
34.	f_concat = torch.cat(list(att_features.values()), dim=1)	
35.	attention_scores = torch.matmul(attention_vectors, f_concat)	
36.	beta = F.softmax(attention_scores, dim=1)	
37.	e_u = stack([beta[p] * f_pu_tilde_dict[meta_path] for p, meta_path in enumerate(f_pu_tilde_dict)]).sum(dim=0)	
38.	return e_u	

7.3.4　任意聚合

任意聚合指的是无视图的连通关系，任意的节点之间可进行信息的传递，根据节点本身的特性进行聚合操作，无视拓扑结构或者仅使用拓扑结构进行辅助。

任意聚合模型定义见表 185 ~表 190。

GTN 模型指的是用 transformer 进行全局聚合，以拉普拉斯特征向量进行位置编码。模型定义见表 185。

表 185　GTN 模型定义

模型	描述	字段	关键字	定义	数据类型
GTN	利用 transformer 进行消息聚合，并采用拉普拉斯作为特征编码	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 而选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
			attention_weights	可选，注意力权重	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			heads	注意力头数量	int

Graphormer 模型指的是用 transformer 进行全局的消息传递以及聚合，在计算注意力分数时，加入两个节点之间的一些信息如最短路径、边特征等，以进行空间位置编码来影响注意力权重。模型定义见表 186。

表 186　Graphormer 模型定义

模型	描述	字段	关键字	定义	数据类型
Graphormer	利用 transformer 进行消息聚合，并采用几种不同的空间编码，如最短路径等	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
			attention_weights	可选，注意力权重	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			heads	注意力头数量	int
			position_encoding	位置编码方式	string

GraphTrans 模型指的是在标准 GNN 层上加入 transformer，用 Bert 中的[CLS]token，引入 readout 机制，抛弃位置编码以保持图的置换不变性。模型定义见表 187。

表 187　GraphTrans 模型定义

模型	描述	字段	关键字	定义	数据类型
GraphTrans	在标准 GNN 层后添加 transformer 并利用[CLS]进行全局聚合	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
			attention_weights	可选，注意力权重	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			heads	注意力头数量	int

SAT 模型，指的是使用结构提取器专门用于抓取子图的表征以计算注意力分数。允许模型在特征聚合过程中融入图结构的信息，增强模型对图结构的敏感度和表征能力。模型定义见表 188。

表 188　SAT 模型定义

模型	描述	字段	关键字	定义	数据类型
SAT	Structure-Aware Transformer 通过在每个节点提取以其为中心的子图表示，然后计算注意力，从而将结构信息整合到原始自注意力中	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
			attention_weights	可选，注意力权重	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	Int
			hidden_channels	可选，隐藏层特征的维度	int
			heads	注意力头数量	int
			k_hop	抽取子图的 Hop 数	int
			struct_extractor	提取子图表征的方式	string

GraphGPS 模型指的是广泛使用，且适用不同规模大小的图的 Graph Transformer 框架。模型定义见表 189。

表 189　GraphGPS 模型定义

模型	描述	字段	关键字	定义	数据类型
GraphGPS	构建可以广泛使用，且适用不同规模大小的图的 Graph Transformer 框架	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
			attention_weights	可选，注意力权重	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，可选，隐藏层特征的维度	int
			heads	注意力头数量	int
			type_mpnn	局部注意力计算方式	string
			type_transformer	全局注意力计算方式	string
			position_encoding	位置编码方式	string

GloGNN 模型指的是以聚合图中全局节点的信息生成节点的嵌入，用优化问题的闭式解求出全局聚合中发挥作用的系数矩阵。模型定义见表 190。

表 190　GloGNN 模型定义

模型	描述	字段	关键字	定义	数据类型
GloGNN	利用系数矩阵对全局节点的信息进行聚合	Input	X	节点特征矩阵	tensor

表 190　GloGNN 模型定义（续）

模型	描述	字段	关键字	定义	数据类型
GloGNN	利用系数矩阵对全局节点的信息进行聚合	Input	edge_index	边索引，与 g 二选一	tensor Sparse Tensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，可选，隐藏层特征的维度	int
			k_hop	系数矩阵需要近似的邻接矩阵的阶数	int

7.4　边级模型

7.4.1　概述

边级模型指的是以边为主体对图进行建模和处理的模型。根据边表示信息的编码方式，将边级模型分为 3 类：基于节点的边级模型、基于子图的边级模型和基于路径的边级模型。

7.4.2　基于节点的边级模型

基于节点表征的边级模型指的是通过聚合目标边的首尾节点表征作为输入，构建出边级表征的模型。模型首先通过图神经网络对节点的领域信息做聚合，获取节点的向量表征，再将边上的首尾节点的表征聚合，输入到链接预测器中，得到边级的表征模型。

基于节点的边级模型定义见表 191~表 198。

GraphSage Link Predictor 用于同质图的链路预测，基于 GraphSage 模型计算节点表征，根据点积（以下简称“dot”）或 mlp 融合头尾节点表征，得到边概率。模型定义见表 191。

表 191　GraphSage Link Predictor 模型定义

运算操作	描述	字段	关键字	定义	数据类型
GraphSage Link Predictor	基于 GraphSage 计算节点表征，根据 dot 或 mlp 融合头尾节点表征，得到边概率	Input	X	节点特征	tensor
			g	输入图	Graph
			query_edge	待预测边列表	List [string, string]
		Output	edge_prob	边概率	tensor
			edge_label	可选，边标签	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			link_combinator	可选(dot/mlp)，默认 dot	string

R-GCN Link Predictor 指的是接受异质边作为输入，基于 R-GCN 模型计算节点表征，根据 dot 或 mlp 融合头尾节点表征，得到边概率。模型定义见表 192。

表 192　R-GCN Link Predictor 模型定义

运算操作	描述	字段	关键字	定义	数据类型
R-GCN Link Predictor	基于 R-GCN 计算节点表征，根据 dot 或 mlp 融合头尾节点表征，得到边概率	Input	node_feature	节点特征	tensor
			g	输入图	Graph
			query_edge	待预测边列表	List [string, string]
		Output	edge_prob	边概率	tensor
			edge_label	可选，边标签	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			link_combinator	可选(dot/mlp)，默认 dot	string

MCCF模型指的是多组件图卷积协同过滤方法，包含两个模块：分解器和组合器。前者首先分解“用户-物品”图中的边，以识别可能引起购买关系的潜在组件；后者自动重新组合这些潜在组件，从而获得用于预测“用户-物品”的边级表征。模型定义见表193。

表 193　MCCF 模型定义

模型	描述	字段	关键字	定义	数据类型
MCCF	多组件图的卷积协同过滤模型	Input	X	节点特征矩阵	tensor
			g	输入图，和 edge_index 二选一	Graph
			edge_index	边索引，和 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
		Output	Y	输出特征表示	tensor
		Attributes	in_channels	输入特征的维度	Dict [string, int]
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			heads	可选，多头注意力的数量	int

MCCF 算法伪代码见表 194。

表 194　MCCF 算法伪代码

序号	MCCF 算法	注释
1.	输入：in_channels, out_channels, nodes_u, nodes_i, droprate	
2.	输出：Y	
3.	nodes_u_embed = self.u_embed(nodes_u, nodes_i)	
4.	nodes_i_embed = self.i_embed(nodes_u, nodes_i)	
5.	x_u = relu(self.u_bn(self.u_layer1(nodes_u_embed)), inplace = True)	

表 194　MCCF 算法伪代码（续）

序号	MCCF 算法	注释
6.	x_u = dropout(x_u, p = droprate)	
7.	x_u = self.u_layer2(x_u)	
8.	x_i = relu(self.i_bn(self.i_layer1(nodes_i_embed)), inplace = True)	
9.	x_i = dropout(x_i, p = droprate)	
10.	x_i = self.i_layer2(x_i)	
11.	x_ui = cat((x_u, x_i), dim = 1)	
12.	x = relu(self.ui_bn(self.ui_layer1(x_ui)), inplace = True)	
13.	Y = dropout(x, p = droprate)	

MEIRec 指的是利用元路径来指导不同步骤邻居的选择，设计一个异质的 GNN，以获得“用户-查询”的边级表征。模型定义见表 195。

表 195　MEIRec 模型定义

模型	描述	字段	关键字	定义	数据类型
MEIRec	基于元路径的异质图神经意图推荐模型	Input	X_dict	节点特征字典，用于存储每一种节点类型的节点特征信息	Dict [string, tensor]
			g	输入图，至少和 edge_index_dict 二选一	HeteroGraph
			edge_index_dict	边索引字典，至少和 g 二选一	Dict [Tuple [string, string, string], tensor] Dict [Tuple [string, string, string], SparseTensor]
		Output	Y	输出张量	Dict [string, tensor]
		Attributes	in_channels	输入特征的维度	int Dict [string, int]
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int

MEIRec 算法伪代码见表 196。

表 196　MEIRec 算法伪代码

序号	MEIRec 算法	注释
1.	输入：in_channels, out_channels, g, num_nodes_dict, metapath	
2.	输出：out_dict	
3.	out = {}	
4.	one_step_dict = g.sample_neighbors(num_nodes_dict, -1)	
5.	one_step_dict = one_step_dict.edges()[0]	
6.	two_step_dict = g.sample_neighbors(one_step_dict , -1)	
7.	two_step_dict = two_step_dict.edges()[0]	
8.	for meta_path in metapath:	
9.	out[meta_path] = mean(two_step_dict[meta_path])	
10.	out[meta_path] = LSTM(in_channels, out_channels, out[meta_path])	
11.	out_dict = Aggregate(out)	
12.	return out_dict	

HERec 指的是一种基于异质网络嵌入的基于 HIN 的推荐方法，通过基于元路径的随机游走策略，以生成有意义的节点序列用于网络嵌入。学习到的节点表征首先通过一组融合函数进行变换，然后集成到扩展矩阵分解（MF）模型中。模型定义见表 197。

表 197　HERec 模型定义

模型	描述	字段	关键字	定义	数据类型
HERec	基于元路径的异质图推荐模型	Input	X_dict	节点特征字典，用于存储每一种节点类型的节点特征信息	Dict [string, tensor]
			g	输入图，至少和 edge_index_dict 二选一	HeteroGraph
			edge_index_dict	边索引字典，至少和 g 二选一	Dict [Tuple [string, string, string], tensor] Dict [Tuple [string, string, string], SparseTensor]
		Output	Y	输出张量	Dict [string, tensor]
		Attributes	in_channels	输入特征的维度	int Dict [string, int]
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int

HERec 算法伪代码见表 198。

表 198　HERec 算法伪代码

序号	HERec 算法	注释
1.	输入：in_channels, out_channels, metadata, x_dict, edge_index_dict, edge_type, w, M, b	
2.	输出：out_dict	
3.	outs = {}	
4.	for edge_type in metadata:	
5.	node_sequence = RandomWalk(x_dict, edge_index_dict, edge_type)	
6.	outs[edge_type] = Node2Vec(node_sequence)	
7.	for edge_type, out in outs:	
8.	out_dict = sum(w[edge_type] * M * out + b, dim=0)	
9.	return out_dict	

7.4.3　基于子图的边级模型

基于子图的边级模型，指的是通过构造子图、提取特征、子图嵌入、链接推理和预测等步骤，利用图形中节点和边之间的局部结构信息，预测节点之间的关系。

基于子图的边级模型见表 199 ~表 200。

SEAL 指的是应用于同质图的边级模型，提取每个目标边(u,v)周围 K 阶范围内的封闭子图，应用图级 GNN 来分类判断子图标签对应于边存在或类别。模型定义见表 199。

表 199　SEAL 模型定义

模型	描述	字段	关键字	定义	数据类型
SEAL	基于 K 阶子图表征的同质图边级模型	Input	X_dict	节点特征字典，用于存储每一种节点类型的节点特征信息	Dict [string, tensor]
			g	输入图，至少和 edge_index_dict 二选一	HeteroGraph
			edge_index_dict	边索引字典，至少和 g 二选一	Dict [Tuple [string, string, string], tensor] Dict [Tuple [string, string, string], SparseTensor]
			query_edge	目标边	List [string, string]
		Output	Y	目标边的概率	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			gcn_type	可选，GCN 编码器(GCN/GraphSage)	string
			k-hop	可选，子图抽取的阶数，默认为 2	int
			nbr_num	可选，各阶聚合的邻居数量	string
			sampler	可选，邻居采样方式	string

GraIL 指的是针对异质图/知识图谱的边级模型，提取目标边(u,v)周围 K 阶范围的子图，应用 R-GCN 作为图级 GNN 判断边的存在或类别。GraIL 的封闭子图不包括只是一个目标节点的邻居但不是另一个目标节点的邻居的节点。模型定义见表 200。

表 200　GraIL 模型定义

模型	描述	字段	关键字	定义	数据类型
GraIL	基于子图推理的异质图边级模型	Input	X_dict	节点特征字典，用于存储每一种节点类型的节点特征信息。	Dict [string, tensor]
			g	输入图，至少和 edge_index_dict 二选一	HeteroGraph
			edge_index_dict	边索引字典，至少和 g 二选一	Dict [Tuple [string, string, string], tensor] Dict [Tuple [string, string, string], SparseTensor]
			query_edge	目标边	List [string, string]
		Output	Y	目标边的概率	tensor

表 200　GraIL 模型定义（续）

模型	描述	字段	关键字	定义	数据类型
GraIL	基于子图推理的异质图边级模型	Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			k-hop	可选，子图抽取的阶数，默认为 2	int
			nbr_num	可选，各阶聚合的邻居数量	string
			sampler	可选，邻居采样方式	string

7.4.4　基于路径的边级模型

基于路径的边级模型指的是利用目标边首尾节点间的路径作为信息编码对象，通过路径传播的信息聚合方式，利用路径上的所有节点和边信息进行高效编码后学习其表征。

基于路径的边级模型定义见表 201~表 203。

NBFNet 指的是将 Bellman-Ford 算法神经网络化，结合路径传播与图神经网络算子，应用于同质图和异质图链路预测。模型定义见表 201。

表 201　NBFNet 模型定义

运算操作	描述	字段	关键字	定义	数据类型
NBFNet Link Predictor	将 Bellman-Ford 算法神经网络化，结合路径传播与图神经网络算子，可应用于同质图和异质图链路预测	Input	X	节点特征	tensor
			g	输入图	Graph
			query_edge	待预测边列表	List [string, string]
			query_edge_type	待预测边类型	int
		Output	edge_prob	边概率	tensor
			edge_label	可选，边标签	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			num_relation	可选，默认 10	int
			message_func	可选，默认 distmult	string
			aggregate_func	可选，默认 pna	string
			symmetric	是否计算对称边，默认 False	bool

PAGNN 指的是通过利用广播和聚合操作，对目标边的两个关联节点之间的所有交互（即路径）和邻域信息进行建模，并基于这些信息生成边级的表征用于链路预测。模型定义见表 202。

表 202　PAGNN 模型定义

模型	描述	字段	关键字	定义	数据类型
PAGNN	根据 pagnn 算法进行边级别子图预测	Input	g	输入图数据	Graph
			egde_index	待预测的边集合	tensor

表 202　PAGNN 模型定义（续）

模型	描述	字段	关键字	定义	数据类型
PAGNN	根据 pagnn 算法进行边级别子图预测	Input	batch_size	可选，每一批次目标节点数目	Int
		Output	Y	边属于各个类别标签的概率	tensor
		Attribute	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			layer_num	pagnn 层数	int

PAGNN 算法伪代码见表 203。

表 203　PAGNN 算法伪代码

序号	PAGNN 算法	注释
1.	输入：in_channels, out_channels, hidden_channels, X, edge_index, layer_num	
2.	输出：Y	
3.	G'= enclosing_subgraph_extract(edge_index)	每个结点抽取标上的封闭子图
4.	h = Linear(in_channels, hidden_channels)(X)	线性变换层 1
5.	N = {u}	
6.	for i in range(layer_num):	广播操作
7.	N' = {p | (q, p) ∈ G', q ∈ N}	
8.	for p in N':	
9.	h = LSTM(ATT_OP(h, N'))	
10.	N = N'	
11.	h' = Linear(in_channels, hidden_channels)(X)	
12.	r = [h',h]	
13.	for i in range(layer_num):	聚合操作
14.	r = ATT_OP(r, N')	
15.	s = concat(r, edge_index)	生成边表征
16.	Y = log_softmax(Linear(hidden_channels, out_channels)(h), dim=1)	计算输出结果
17.	return Y	

7.5　图级模型

7.5.1　概述

图级模型指的是用于图级任务的模型。图级模型根据任务方式划分为 6 类，分别为图分类、图回归、图聚类、图匹配和图生成模型。

7.5.2　图分类

图神经网络的图分类指的是将多图数据中的每一个单图分配到预定义的类别中，单图所对应类别可以是一个或是多个。图级模型通过在图上进行消息传递来学习节点表示，并根据节点特征通过池化等方式学习到图表示，基于学习到的图表示来计算单图的分类概率。

图分类模型见表 204~表 205。

DiffPool 的是通过提供一个可微模块对图节点进行分层池化，构建深度、多层的 GNN 模型。模型定义见表 204。

表 204　DiffPool 模型定义

模型	描述	字段	关键字	定义	数据类型
DiffPool	能够对图节点进行分层池化的可微模块 K 轮消息传递结果	Input	X	节点特征矩阵	tensor
			y_train	独热编码的节点标签矩阵，仅含训练集节点信息	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	图级标签	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			K	消息传递的轮数	int

MPSN 模型指的是基于代数拓扑中的单纯复形概念扩展图神经网络中的邻域特征聚合概念。该模型将原始图提升为 2-复形（包含图中的所有三角结构），基于单纯形的边界邻接、联合边界邻接、下邻接、上邻接这四种单纯复形中的“邻域”概念进行特征聚合，通过读出函数获得最终结果。模型定义见表 205。

表 205　MPSN 模型定义

模型	描述	字段	关键字	定义	数据类型
MPSN	基于代数拓扑中的单纯复形概念扩展图神经网络中的邻域特征聚合概念。首先将原始图提升为 2-复形（包含图中的所有三角结构），接着基于单纯形的边界邻接、联合边界邻接、下邻接、上邻接这四种单纯复形中的“邻域”概念进行特征聚合，最后通过读出函数获得最终结果	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	边特征矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	图级标签	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	Int
			n_layers	隐藏层层数	int
			read_out	读出函数选择	string

7.5.3　图回归

图神经网络中的图回归指的是通过学习图结构中包含节点的特征、相邻节点的信息以及边的信息的图表征，预测该图所对应的连续性目标值或属性。与图分类不同，图回归的目标是预测数值型的图属性，而不是离散的类别。

图回归模型见表 206 ~表 208。

MPNN 模型指的是消息传递和读出两阶段模型框架。对于图中任意节点，消息传递考虑所有与该点相连的节点，基于节点特征与边特征共同计算消息向量，求和后与原有特征向量结合得出新的节点特征向量，通过读出函数得出整个图的特征表示。MPNN 为图级一阶模型提供了统一的抽象形式。模型定义见表 206。

表 206　MPNN 模型定义

模型	描述	字段	关键字	定义	数据类型
MPNN	通过消息传递和读出两阶段实现图级任务学习。为图级一阶模型提供了统一的抽象形式	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	边特征矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	图级表示	tensor
			in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			n_layers	隐藏层层数	int
			message_size	信息函数输出维度	int
			type	任务类型	string

D-MPNN 模型通过一个有向图卷积网络对分子性质进行预测。模型定义见表 207。

表 207　D-MPNN 模型定义

模型	描述	字段	关键字	定义	数据类型
D-MPNN	一个对分子性质进行预测的有向图卷积网络	Input	X	节点特征矩阵	tensor
			edge_weight	边特征矩阵	tensor
			y_train	分子性质的标签矩阵，仅含训练集节点信息	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	分子性质的预测值	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			K	消息传递的轮数	int
			dropout_r	丢弃比率	float

PPGN 模型指的是一种简单的 2 阶张量网络架构，基于边特征变换、矩阵乘法、堆叠构建基础网络模块，每层包含两个特征变换操作，对图同构问题具有 3-WL 判别能力。模型定义见表 208。

表 208　PPGN 模型定义

模型	描述	字段	关键字	定义	数据类型
PPGN	基于边特征变换、矩阵乘法、堆叠构建基础网络模块，每层包含两个特征变换操作。对图同构问题具有 3-WL 判别能力	Input	edge_feature	边特征张量	tensor
		Output	Y	图级表示	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels_1	隐藏层特征的维度 1	int
			hidden_channels_2	隐藏层特征的维度 2	int
			n_layers	隐藏层层数	int
			type	任务类型	string

7.5.4　图聚类

图神经网络中的图聚类是指通过识别具有相似拓扑结构、相似节点属性分布或其他关联性的图数据，从而将每个图划分到预先定义的类别中的任务。图神经网络利用节点的属性、图的拓扑结构，通过学习图的表征并用于计算属于不同类别的概率或分数。

图聚类模型见表 209。

DMoN 指的是通过聚类质量模块度度量搭配坍缩正则化使得优化目标更易实现，应对现实图数据中的聚类结构恢复问题的监督池化方法。模型定义见表 209。

表 209　DMoN 模型定义

模型	描述	字段	关键字	定义	数据类型
DMoN	学习一种启发性无监督池化算法	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	F	聚类表征	tensor
			C	聚类分配矩阵	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			n_clusters	模型内的聚类个数	int
			collapse_regularization	坍塌正则化权重	float
			dropout	丢弃率	float
			do_unpooling	可选，1 表示使用反池化，0 表示不使用反池化	bool

7.5.5　图匹配

图神经网络中的图匹配是指通过图神经网络对图中的节点和边进行表征学习，基于节点和边的表征同时考虑节点相似性和边相似性，在两个或多个图之间建立节点有意义的结构对应关系。

图匹配模型见跟 210 ~表 211。

GMN 的是以两张图作为输入，通过交叉图注意力匹配机制，联合计算两张图之间的相似性得分。与。模型定义见表 210。

表 210　GMN 模型定义

模型	描述	字段	关键字	定义	数据类型
GMN	基于交叉图注意力匹配机制的图匹配模块 K 轮消息传递结果	Input	X	节点特征矩阵	tensor
			y_train	独热编码的节点标签矩阵，仅含训练集节点信息	tensor
			edge_index_1	边索引，与 g 二选一	tensor SparseTensor
			edge_weight_1	可选，边权重矩阵	tensor
			g_1	输入图，与二 edge_index 选一	Graph
			edge_index_2	边索引，与 g 二选一	tensor SparseTensor
			edge_weight_2	可选，边权重矩阵	tensor
			g_2	输入图，与二 edge_index 选一	Graph
		Output	Y	图匹配度	float
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			K	消息传递的轮数	int
			similarity	相似度计算方式，可选 dot_product，euclidean_distance 等	string

FGNN 模型指的是使用张量结构对图进行表示，引入 folklore graph layer 操作实现对图特征的置换等变性变换，在最后一层加入置换不变性函数。模型定义见表 211。

表 211　FGNN 模型定义

模型	描述	字段	关键字	定义	数据类型
FGNN	引入 folklore graph layer 操作构建具有置换不变性的图神经网络，在图同构判别问题上具有比 MPNN 更强的表达能力	Input	X	节点特征矩阵	tensor
			edge_index_1	边索引，与 g 二选一	tensor SparseTensor
			g_1	输入图，与二 edge_index 选一	Graph
			edge_index_2	边索引，与 g 二选一	tensor SparseTensor
			g_2	输入图，与二 edge_index 选一	Graph
		Output	Y	图匹配度	float
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			n_layers	隐藏层层数	int

7.5.6　图生成

图神经网络中的图生成指的是基于给定的任意数量节点和边的图样本，图神经网络从中学习图样本的分布，并从此分布中抽取新的图。根据生成图的大小，图生成任务可以被分成两类：图节点数量固定的图生成和图节点数量可变的图生成。

图生成模型见表 212~表 213。

MolGAN 模型指的是基于生成对抗思想的分子图生成框架。随机产生一个高斯噪声后送入一个生成器产生一个邻近张量 A 和节点特征矩阵 X。其中 A 的形状为(N,N,Y)，其中 Y 表示边的类型数；X 的形状为(N,T)，其中 T 表示每个节点（原子）的类型数。(A,X)共同刻画了一个分子图结构，将其送入由 R-GCN 构成的判别器中进行监督训练。模型定义见表 212。

表 212　MolGAN 模型定义

模型	描述	字段	关键字	定义	数据类型
MolGAN	基于生成对抗思想的分子图生成网络架构，判别器采用了 Relational-GCN	Input	z	随机噪声向量	tensor
		Output	A	图邻接张量	tensor
			X	图节点特征矩阵	tensor
		Attributes	in_channels	输入特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			n_layers	隐藏层层数	int
			n_nodes	输出图节点个数	int
			edge_dim	边特征维度	int

GRANs 模型将图生成过程划分成多个时间块，采用注意力机制计算新加入图结构的节点与已存在节点间产生连边的概率。模型定义见表 213。

表 213　GRANs 模型定义

模型	描述	字段	关键字	定义	数据类型
GRANs	该模型将图生成过程划分成多个时间块，采用注意力机制计算新加入图结构的节点与已存在节点间产生连边的概率	Input	X	节点特征矩阵	tensor
			y_train	节点之间的连接情况的标签，仅含训练集节点信息	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			g1	输入图，与 edge_index 二选一	Graph
		Output	g2	基于 g1 生成的图	Graph
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			B	每次图生成的块大小	int
			N	允许的最大图尺寸	int
			π	节点生成顺序	List [int]

8　图神经网络压缩和加速

8.1　图数据压缩

8.1.1 概述

图数据压缩指的是对图形数据进行压缩，可以减少图数据存储和传输开销，提高图数据处理的效率和可扩展性。图数据压缩可以归类为图拓扑结构压缩和图特征数据压缩。具体定义见表 217~表 220。

8.1.2 图拓扑结构压缩

图拓扑结构压缩指的是将图的拓扑结构表示方式转化为更紧凑、更高效的形式。对图的拓扑结构进行压缩，可以减小图拓扑数据的存储空间占用并提高处理效率。

Cluster-GCN 是一种图拓扑结构压缩的方法，Cluster-GCN 利用图聚类结构改进训练算法，可以降低内存占用并提高计算效率。模型定义见表 214。

表 214 Cluster-GCN 模型定义

模型	描述	字段	关键字	定义	数据类型
Cluster-GCN	利用图聚类结构改进训练算法，从而改善内存使用和提高计算效率	Input	X	节点特征矩阵	tensor
			Y	节点标签矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			g	输入图，与 edge_index 二选一	Graph
			Module	输入初始化模型	Model
		Output	Y	类（标签）矩阵	tensor
			H	预测矩阵及传送集	tensor
			Module	输出训练后的模型	Model
		Attributes	dropout_r	丢弃比率	float
			clustering-method	聚类方法	string
			cluster-number	聚类数量	int

WebGraph 是一种图拓扑结构压缩的方法，通过充分利用图拓扑数据的相似性和局部性特点改进图拓扑结构存储方式，可以降低存储占用并提高计算效率。模型定义见表 215。

表 215 WebGraph 模型定义

模型	描述	字段	关键字	定义	数据类型
WebGraph	利用图拓扑数据的相似性和局部性特点改进图拓扑结构存储，从而降低存储占用并提高计算效率	Input	is_compress	True 时表示压缩操作；Flase 时表示解压操作；	bool
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			g	输入图，与 edge_index 二选一	Graph
		Output	edge_index	边索引，与g二选一	tensor SparseTensor
			g	输入图，与 edge_index 二选一	Graph

8.1.3 特征数据压缩

图特征数据压缩是指对图数据中节点、边或全图的特征数据进行压缩，以减小特征数据的空间占用并提高图神经网络处理特征数据的计算效率的方法。图数据的节点和边特征能够包含多个数值，如节点的特征向量、边的特征向量等。特征数据压缩通过使用合适的压缩方法和算法，将向量或数组的属性表示方式转化为更紧凑的形式，以减少数据的存储空间。

VQ-GNN 是一个图特征数据压缩框架，VQ-GNN 扩展了基于卷积的 GNN，使用矢量量化(VQ)压缩图特征数据。模型定义见表 216。

表 216　VQ-GNN 模型定义

模型	描述	字段	关键字	定义	数据类型
VQ-GNN	为不同节点学习个性化的线性组合 K 轮消息传递结果的权重	Input	X	节点特征矩阵	tensor
			R	码字分配矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			edge_weight	可选，边权重矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			K	消息传递的轮数	int
			C	卷积矩阵	tensor

VQ-GNN 算法伪代码见表 217。

表 217　VQ-GNN 算法伪代码

序号	VQ-GNN 算法	注释
1.	输入：X	
2.	输出：Y	
3.	For l in range(L):	
4.	initialize GNN learnable parameters $W^{(l)}$ and $\theta^{(l)}$ and $\tilde{V}^{(l)} = \tilde{X}^{(l)}\|\|\tilde{G}^{(l+1)}$	初始化
5.	For indices i_b sampled from $\{1,\ldots,n\}$:	
6.	Load the mini-batch features $X_B = X_{<i_b>}$, labels Y_B	
7.	For $l = 0, \ldots\ldots, L-1$	前向传递
8.	Compute the approximate message passing weight matrix using C_B, C_B^T, $R_B^{(l)}$	
9.	Estimate next layer's features $X_B^{(l+1)}$ with $X_B^{(l)}$ and feature codewords $\tilde{X}^{(l)}$	
10.	For $l = L-1, \ldots\ldots, 0$	反向传播
11.	Estimate lower layer's gradients $G_B^{(l)}$ and $\nabla_{W^{(l)}}l$	
12.	For $l = 0, \ldots\ldots, L-1$	VQ 更新
13.	Update the concatenated codewords $\tilde{V}^{(l)} = \tilde{X}^{(l)}\|\|\tilde{G}^{(l+1)}$	

8.1.4　图采样

8.1.4.1　概述

根据图采样的层次不同，图采样方法分为三类：

a) 节点采样：节点采样对单个节点的局部邻居采样。节点采样为图中的每个节点选择固定数量的邻居以计算该节点的特征。此类方法允许模型在考虑每个节点的局部结构的同时，降低由于考虑全部邻居带来的计算负担。
b) 层级采样：层级采样在 GNN 的每一层对多个节点的邻居进行采样。与节点采样相比，此类方法能够在保证采样效率的同时避免邻居数量的指数级增长问题。层级采样的节点数量与层数成线性关系，从而降低整体的内存开销，提高整体计算速度，可以在计算资源受限的情况下，有效地处理深层的 GNN 模型。
c) 子图采样：子图采样在原图中采样得到一个包含选定节点和边的子图，并在这个子图上进行全图式 GNN 的训练。此类方法可以将大规模图分割成多个小的、更易于处理的子图，可以减少计算负载，同时保持图的全局特性。

8.1.4.2 节点采样

节点采样指的是对目标节点的邻居进行采样以减少输入图数据量，可以降低内存消耗，并加速模型训练。

此类方法实现流程和细节为：

a) 输入：包括节点特征、邻接矩阵、边特征等信息的图数据，选定若干目标节点；
b) 使用采样策略对邻居进行采样；
c) 输出：采样后的图数据，各节点在原图中的节点序号，目标节点在输出图中的序号。

具体定义见表 218 ~表 220。

node_sampler 运算操作定义见表 218。

表 218 node_sampler 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
node_sampler	根据起始节点，对输入的图数据进行采样	Input	g	输入图数据	Graph HeteroGraph
			target_nodes	目标节点的编号	tensor
			num_neighbors	每层邻居节点数	List [int] Dict [Tuple [string, string, string], List [int]]
			batch_size	可选，每一批次目标节点数目	int
			shuffle	可选，是否每轮采样采用随机洗牌策略	bool
			drop_last	可选，是否丢弃最后一个不完整的批次	bool
		Output	blocks	输出子图列表	List [Graph] List [HeteroGraph]
			sample_index	输出图在原图的节点编号	tensor
			node_index	起始节点在输出中的编号	tensor

graphsage_sampler 运算操作定义见表 219。

表 219　graphsage_sampler 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
graphsage_sampler	按照 graphsage 的方式进行邻居采样	Input	g	输入图数据	Graph
			target_nodes	目标节点的编号	tensor
			num_neighbors	每层邻居节点数	List [int]
			batch_size	可选，每一批次目标节点数目	int
		Output	blocks	输出图列表	List [Graph]
			sample_index	输出图在原图的节点序号	tensor
			node_index	起始节点在输出中的序号	tensor
			neighbors	每层邻居数	List [int]

pinsage_sampler 运算操作定义见表 220。

表 220　pinsage_sampler 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
pinsage_sampler	根据起始节点，依据 pinsage 算法对输入的图数据进行采样	Input	g	输入图数据	HeteroGraph
			target_nodes	目标节点的编号	tensor
			num_neighbors	每层邻居节点数	List [int] Dict [Tuple [string, string, string], List [int]]
			node_type	目标节点的类型	string
			other_type	其他节点类型	string
			num_traversals	单次进行基于元路径随机游走的最大遍历次数	int
			termination_prob	每次基于元路径遍历后的终止概率	float
			num_random_walks	对每一个给定节点的随机游走次数	int
			batch_size	可选，每一批次目标节点数目	int
			shuffle	可选，是否每轮采样采用随机洗牌策略	bool
			drop_last	可选，是否丢弃最后一个不完整的批次	bool

表 220　pinsage_sampler 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
pinsage_sampler	根据起始节点，依据 pinsage 算法对输入的图数据进行采样	Output	block	输出图列表	List [HeteroGraph]
			sample_index	输出图在原图的节点编号	tensor
			node_index	起始节点在输出中的编号	tensor

8.1.4.3　层级采样

层级采样指的是在 GNN 的每一层采样固定数量的邻居节点。层级采样可以解决随着网络深度增加，邻居数量指数式地增长的问题，降低内存消耗并加速模型训练。

四种具体层级采样运算操作见表 221 ~表 224。

layer_sampler 运算操作定义见表 221。

表 221　layer_sampler 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
layer_sampler	根据起始节点，对输入的图数据进行层级采样	Input	g	输入图数据	Graph HeteroGraph
			target_nodes	目标节点的编号	tensor
			num_samples	每层采样节点数	List [int] Dict [Tuple [string, string, string], List [int]]
			batch_size	可选，每一批次目标节点数目	int
			shuffle	可选，是否每轮采样采用随机洗牌策略	bool
			drop_last	可选，是否丢弃最后一个不完整的批次	bool
		Output	block	输出图	List [Graph] List [HeteroGraph]
			sample_index	输出图在原图的节点编号	tensor
			node_index	起始节点在输出中的编号	tensor

hgt_sampler 运算操作定义见表 222。

表 222　hgt_sampler 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
hgt_sampler	异质图按层级采样的方式进行邻居采样	Input	g	输入图数据	HeteroGraph
			target_nodes	目标节点的编号	tensor
			num_samples	每层采样节点数	Dict [Tuple [string, string, string], List [int]]
			batch_size	可选，每一批次目标节点数目	int

表 222　hgt_sampler 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
hgt_sampler	异质图按层级采样的方式进行邻居采样	Output	graphs	输出图	List [HeteroGraph]
			sample_index	输出图在原图的节点编号	tensor
			node_index	起始节点在输出中的编号	tensor

fastgcn_sampler 运算操作定义见表 223。

表 223　fastgcn_sampler 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
fastgcn_sampler	按照 FastGCN 的方式进行邻居采样	Input	g	输入图数据	Graph
			target_nodes	目标节点的编号	tensor
			num_samples	每层采样节点数	List [int]
			batch_size	可选，每一批次目标节点数目	int
		Output	block	输出图列表	List [Graph]
			sample_index	输出图在原图的节点编号	tensor
			node_index	起始节点在输出中的编号	tensor

ladies_sampler 运算操作定义见表 224。

表 224　ladies_sampler 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
ladies_sampler	按照 LADIES 的方式进行邻居采样	Input	g	输入图数据	Graph
			target_nodes	目标节点的编号	tensor
			num_samples	每层采样节点数	List [int]
			batch_size	可选，每一批次目标节点数目	int
		Output	block	输出图列表	List [Graph]
			sample_index	输出图在原图的节点编号	tensor
			node_index	起始节点在输出中的编号	tensor

8.1.4.4　子图采样

子图采样指的是对整张图数据进行采样得到若干子图，可以降低内存消耗并加速模型训练。此类方法实现流程为：

a)　输入节点特征、邻接矩阵、边特征等信息的图数据，子图数量、子图节点最大数量；
b)　使用随机游走等方式采样子图，并根据子图数量、子图节点最大数量等限制子图的规模；
c)　输出子图集合，子图各节点在原图中的节点序号。

五种具体子图采样运算操作见表 225~表 229。

子图采样运算操作定义见表 225。

表 225　subgraph_sampler 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
subgraph_sampler	根据输入图数据进行子图采样	Input	X	输入图数据	Graph
			target_nodes	目标节点的编号	tensor
			num_neighbors	每层邻居节点数	List [int] Dict [Tuple [string, string, string], List [int]]
			batch_size	可选，每一批次目标节点数目	Int
			shuffle	可选，是否每轮采样采用随机洗牌策略	bool
			drop_last	可选，是否丢弃最后一个不完整的批次	bool
		Output	block	输出图列表	List [Graph]
			sample_index	输出图在原图的节点编号	tensor
			node_index	起始节点在输出中的编号	tensor

clustergcn_sampler 运算操作定义见表 226。

表 226　clustergcn_sampler 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
clustergcn _sampler	根根 clustergcn 算法进行子图采样	Input	g	输入图数据	Graph
			num_cluster	聚类个数	int
			clustering_method	聚类算法	string
			batch_size	可选，每一批次目标节点数目	int
		Output	block	输出图列表	List [Graph]
			sample_index	输出图在原图的节点编号	tensor
			node_index	起始节点在输出中的编号	tensor

saint_sampler 运算操作定义见表 227。

表 227　saint_sampler 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
saint_sampler	根据 SAINT 算法进行子图采样	Input	g	输入图数据	Graph
			saint_method	采样算法	string
			batch_size	可选，每一批次目标节点数目	int
		Output	block	输出图列表	List [Graph]

表 227　saint_sampler 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
saint_sampler	根据 SAINT 算法进行子图采样	Output	sample_index	输出图在原图的节点编号	Tensor
			node_index	起始节点在输出中的编号	tensor

random_walk_sampler 运算操作定义见表 228。

表 228　random_walk_sampler 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
random_walk_sampler	从起始节点生成随机游走序列而采样子图的接口	Input	g	图	Graph HeteroGraph
			nodes	起始节点	tensor
			metapath	可选，元路径	List [Tuple [string, string, string]]
			prob	存储每条边的目标节点的转移概率	tensor
			restart_prob	结束本次随机游走的概率	float
			length	可选，随机游走路径长度	int
			return_eids	可选，是否返回边编号	bool
		Output	traces	随机游走序列的节点编号	tensor
			eids	随机游走序列的边编号	tensor
			types	随机游走序列的节点类型编号	tensor

node2vec_sampler 运算操作定义见表 229。

表 229　node2vec_sampler 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
node2vec_sampler	从起始节点生成基于 node2vec 随机游走序列的采样接口	Input	g	图	Graph
			nodes	起始节点编号	tensor
			prob	存储每条边的目标节点的转移概率	tensor
			p	在随机游走过程中节点转移控制参数	float
			q	在随机游走过程中节点转移控制参数	float
			walk_len	随机游走长度	int
			return_eids	可选，是否返回边编号	bool
		Output	traces	随机游走节点编号轨迹张量	tensor
			eids	随机游走边编号轨迹张量	tensor

8.1.5 图数据压缩度量指标

图数据压缩度量指标是用来评估图数据压缩算法的效果和性能的指标。图数据压缩算法的效果主要体现在压缩后的空间占用、数据的表征能力以及可恢复性，图数据压缩算法的的性能主要体现在压缩算法的时间开销和规模可扩展。常用的图数据压缩的度量指标有图编辑距离、压缩比、信息熵和压缩时间。

8.2 模型量化与剪枝

8.2.1 量化

图模型的量化指的是将模型中的参数转换为定点数或浮点数。量化可以减少模型的存储空间和计算复杂度，提高模型的效率和推理速度。

EXACT 是一种使用量化方法结合随机投影压缩节点嵌入的 GNN 内存优化技术，可以显著降低在大型图上训练 GNN 的内存需求而近乎不损失精度。模型定义见表 230。

表 230 EXACT 模型定义

模型	描述	字段	关键字	定义	数据类型
EXACT	一个使用压缩激活训练图神经网络的内存框架	Input	X	节点特征矩阵	tensor
			edge_weight	边特征矩阵	tensor
			y_train	独热编码的节点标签矩阵，仅含训练集节点信息	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	节点属于各个类别标签的概率	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			activation_compression_bits	压缩激活比特	int
			dropout_r	丢弃率	float

EXACT 算法伪代码见表 231。

表 231 EXACT 算法伪代码

序号	EXACT 算法	注释
1.	输入：$H^{(l)}, \theta^{(l)}$, layers L	GCN 层
2.	输出：$H^{(l+1)}$	
3.	$ctx^{(l)} \leftarrow \{\}$	为反向传播保存张量
4.	$J^{(l)} \leftarrow MM(H^{(l)}, \theta^{(l)})$	
5.	Add $H^{(l)}$ and $\theta^{(l)}$ to $ctx^{(l)}$	MM.backward
6.	$H^{(l+1)} \leftarrow SPMM(\hat{A}, J^{(l)})$	
7.	Add $\hat{A}$ (in CSR format) and $J^{(l)}$ to $ctx^{(l)}$	SPMM.backward
8.	If $l \neq L-1$:	ReLU.backward
9.	Add $1_{\{H^{(l+1)}>0\}}$ to $ctx^{(l)}$	
10.	$H^{(l+1)} = ReLU(H^{(l+1)})$	
11.	return $H^{(l+1)}$	

量化方法有较强的兼容性，可以与其他压缩方法联合使用。

a) 量化方法和剪枝压缩联合使用

量化方法可以应用于剪枝后的模型以实现进一步的优化。结构化剪枝后，可以直接量化精简后的模型。非结构化剪枝后，需要量化未被剪枝掉的权重。

b) 量化方法和结构化矩阵压缩联合使用

量化方法可以应用于结构化矩阵压缩之前。先进行量化操作，简化矩阵数据中的内容，再按照结构化矩阵的方法进行处理。

8.2.2 剪枝

剪枝指的是减少对图模型的性能没有显著影响的参数，可以提高模型的效率和推理速度。图模型剪枝技术可根据其在图神经网络的不同功能层的应用分类，分为信息传递层剪枝和信息聚合层剪枝。

8.2.2.1 信息传递层剪枝

信息传递层剪枝指的是在图神经网络中稀疏化信息传递层的参数。具体方法包括：基于敏感性的剪枝、基于权重值的剪枝、稀疏剪枝等。此类方法可以根据具体的信息传递层和应用场景单独应用或结合使用。

SGC 模型把剪枝应用在传统 GNN 信息传递方式上，移除了连续层之间的非线性和折叠权重矩阵，可以去除非必要的模型复杂性和冗余计算。模型定义见表 232。

表 232 SGC 模型定义

模型	描述	字段	关键字	定义	数据类型
SGC	通过连续移除连续层之间的非线性和折叠权重矩阵,对传统 GNN 信息传递方式进行剪枝，以去除不必要的复杂性和冗余计算	Input	X	节点特征矩阵	tensor
			S	带有自环边的对称归一化邻接矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	类（标签）矩阵	tensor
			H	预测矩阵及传送集	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			K	消息传递的轮数	int
			dropout_r	丢弃率	float

8.2.2.2 信息聚合层剪枝

信息聚合层剪枝是指稀疏化图神经网络中信息聚合步骤中的参数，具体方法包括执行部分邻居信息聚合、激活操作剪枝、维度变换简化等。

PPNP 结合了 GCN 和 PageRank 算法来更新消息传递的步骤，可以聚合大范围内的邻居信息。模型定义见表 233。

表 233　PPNP 模型定义

模型	描述	字段	关键字	定义	数据类型
PPNP	通过结合 GCN 和 PageRank 算法对传统 GNN 信息聚会方式进行剪枝，以达到聚合大范围内邻居信息的目的	Input	X	节点特征矩阵	tensor
			A	带有自环边的对称归一化邻接矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			g	输入图，与 edge_index 二选一	Graph
		Output	Y	类（标签）矩阵	tensor
			H	预测矩阵及传送集	Tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			K	消息传递的轮数	int
			dropout_r	丢弃率	float

8.2.2.3　剪枝评估标准

剪枝评估标准是用于衡量剪枝算法效果的标准。图模型常用剪枝评估标准包括：

a) 精度准确度：剪枝后模型在验证集或测试集上的精度准确度以及能否保持与原始模型相近的准确度。
b) 参数数量：剪枝后模型的参数数量是否减少。
c) 计算量：剪枝后模型的计算量是否减少。
d) 网络稀疏度：剪枝后模型的网络稀疏度是否增加。

8.3　模型蒸馏

8.3.1　概述

模型蒸馏指的是用一个小型学生模型学习一个大型、训练好的教师模型的知识来提高模型性能。过程中教师模型的综合知识可以转化为更精简、更有效的表示。模型蒸馏可以保持较高预测性能的同时，极大地降低模型的复杂性和计算资源需求，实现模型的轻量化和高效化。图神经网络的模型蒸馏可以分为离线蒸馏、在线蒸馏以及自蒸馏三种类型。

8.3.2　离线蒸馏

离线蒸馏采用单向知识转移和两阶段的训练过程。第一阶段为教师模型的预训练，教师模型可以充分学习训练数据来捕获和编码丰富的知识；第二阶段为知识迁移，学生模型能够通过有效的知识传递机制从教师模型中学习到关键信息。例如 LSP、TinyGNN、LTD、GRACED 为离线蒸馏模型。

离线蒸馏模型见表 234~表 239。

LSP 在图神经网络领域进行离线蒸馏，使用局部结构保留模块，可以显式传递拓扑差异信息，实现拓扑感知的知识转移。LSP 适用于动态图，能在图结构变化时有效蒸馏知识。模型定义见表 234。

表 234　LSP 模型定义

模型	描述	字段	关键字	定义	数据类型
LSP	LSP 是首次针对 GNN 进行知识蒸馏的专门方法,通过提出的局部结构保留模块，可以显式地衡量并传递拓扑差异信息，从而实现拓扑感知的知识转移	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			g	输入图，与 edge_index 二选一	Graph
			teacher_logits	教师模型的输出，即软标签	tensor
		Output	Y	节点标签	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			heads	可选，多头注意力的数量，默认为 1	int
			hidden_channels	可选，隐藏层特征的维度	Int
			beta	可选，通过 beta 加权知识蒸馏中的硬损失和软损失，默认为 0.1	float
			dropout	可选，在训练过程中，神经网络每个节点的丢弃概率，默认为 0.2	float

TinyGNN 使用邻居蒸馏策略和对等节点信息弥补了小型 GNN 与深层 GNN 间的信息差距，可以在维持高性能的同时加速 GNN 推理。模型定义见表 235。

表 235　TinyGNN 模型定义

模型	描述	字段	关键字	定义	数据类型
TinyGNN	TinyGNN 是为了在保持高性能的同时，实现 GNN 的快速推理而提出的模型。该模型通过利用对等节点信息和采用邻居蒸馏策略，来弥补小型 GNN 与深层 GNN 之间的邻居信息差距	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			g	输入图，与 edge_index 二选一	Graph
			teacher_logits	教师模型的输出，即软标签	tensor
		Output	Y	节点标签	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			hidden_channels	可选，隐藏层特征的维度	int
			dropout	可选，在训练过程中，神经网络每个节点的丢弃概率，默认为 0.2	float

LTD 是一个适用于多种预训练图神经网络模型的知识蒸馏框架，LTD 通过学习节点特定的蒸馏温度来提升性能，可以提高蒸馏模型的性能。模型定义见表 236。

表 236　LTD 模型定义

模型	描述	字段	关键字	定义	数据类型
LTD	提出一个通用的知识蒸馏框架，可以应用于任何预训练的图神经网络模型以进一步提高它们的性能。为了解决分离问题，LTD 提出学习节点特定的蒸馏温度，以提高蒸馏模型的性能	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			g	输入图，与 edge_index 二选一	Graph
			teacher_logits	教师模型的输出，即软标签	tensor
		Output	Y	节点标签	tensor
		Attributes	in_channels	每个输入样本的大小	int
			out_channels	每个输出样本的大小	int
			heads	可选，多头注意力的数量，默认为 1	Int
			hidden_dim	可选，隐藏层维度，默认为 64	int
			beta	可选，通过 beta 加权知识蒸馏中的硬损失和软损失，默认为 0.1	float
			dropout	可选，在训练过程中，神经网络每个节点的丢弃概率，默认为 0.2	float

LTD 算法伪代码见表 237。

表 237　LTD 算法伪代码

序号	LTD 算法	注释
1.	输入：X, g, teacher_logits, y	
2.	输出：Y	
3.	While warmup do:	
4.	student_logits = self.gnn_student(g, X)	
5.	loss_distill = self.ce_loss(student_logits, teacher_logits, y)	
6.	self.gnn_student.update(loss_distill)	
7.	student_logits = self.gnn_student(g, X)	
8.	While not converge do:	
9.	temperature = LTDTemp(student_logits, teacher_logits)	
10.	loss_distill = self.ce_loss(student_logits, teacher_logits, y)	
11.	self.gnn_student.update(loss_distill)	
12.	student_logits = self.gnn_student(g, X)	
13.	loss_temp = self.ce_loss(student_logits, y)	
14.	self.LTDTemp.update(loss_temp)	
15.	final_logits = self.gnn_student(g, X)	
16.	Return log_softmax(final_logits, dim=-1)	

GRACED 模型是一种基于预聚合的图神经网络蒸馏推理加速模型。GRACED 通过定制化知识蒸馏策略来适应性地修改从教师图神经网络模型继承的知识，可以减轻图异配性对蒸馏的不良影响。模型定义见表 238。

表 238　GRACED 模型定义

模型	描述	字段	关键字	定义	数据类型
GRACED	提出一种基于预聚合的图神经网络蒸馏推理加速方案	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			g	输入图，与 edge_index 二选一	Graph
			teacher_logits	教师模型的输出，即软标签	tensor
		Output	Y	节点标签	tensor
		Attributes	in_channels	每个输入样本的大小	int
			out_channels	每个输出样本的大小	int
			hidden_dim	可选，隐藏层维度，默认为 64	Int
			dropout	可选，在训练过程中，神经网络每个节点的丢弃概率，默认为 0.2	float

GRACED 算法伪代码见表 239。

表 239　GRACED 算法伪代码

序号	GRACED 算法	注释
1.	输入：X, g, teacher_logits, y	
2.	输出：final_loss	
3.	While not converge do:	
4.	GA_Input = self.GA_Func(X,g)	
5.	student_logits = slef.GAMLP(GA_Input)	
6.	loss_student_pred = self.ce_loss(student_logits, y)	
7.	loss_teacher_pred = self.ce_loss(teacher_logits, y)	
8.	distll_weight = self.weight_func(loss_student_pred, loss_teacher_pred)	
9.	loss_distill = self.ce_loss(student_logits, teacher_logits)	
10.	final_loss = distll_weight * loss_distill + (1- distll_weight) * loss_student_pred	
11.	final_loss.backward()	
12.	Return final_loss	

8.3.3　在线蒸馏

在线蒸馏同时更新教师模型和学生模型，整个框架端到端可训练，可以应对无法有效获得参数量大且高性能的教师模型的情况。

例如 ROD、FreeKD 为在线蒸馏模型，见表 240 ~表 241。

ROD 使用同行教学和接收感知多尺度知识的在线蒸馏，优化稀疏图学习中学生模型的互补性与多样性，可以高效地传递知识。模型定义见表 240。

表 240　ROD 模型定义

运算操作	描述	字段	关键字	定义	数据类型
ROD	一种新颖的面向稀疏图学习的接收感知在线知识蒸馏方法，设计了多尺度接收感知图知识、基于任务的监督和丰富的蒸馏知识三种监督信号，输出多个学生模型在多个不同下游任务上的预测结果	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			g	输入图，与 edge_index 二选一	Graph
		Output	prediction	输出张量	tensor
		Attributes	hidden_units	隐藏层维度	int
			k	学生数量	int
			learning_rate	优化器的学习率	float
			alpha	损失函数中多任务间的平衡系数	float
			beta	损失函数中多任务间的平衡系数	float

FreeKD 是一种结合了强化学习的在线知识蒸馏框架，FreeKD 使用分层策略协同构建两个较浅的 GNN 进行有效知识交换，可以实现自由方向的知识蒸馏。模型定义见表 241。

表 241　FreeKD 模型定义

运算操作	描述	字段	关键字	定义	数据类型
FreeKD	可以动态调整知识传递方向的在线图神经网络知识蒸馏方法，使用特征张量、标签张量、邻接矩阵，输出两个模型预测的节点标签	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			g	输入图，与 edge_index 二选一	Graph
			L	训练轮次	int
		Output	prediction	输出张量	tensor
		Attributes	gamma	Reward 中的超参数	float
			alpha	损失函数中的平衡系数	float
			beta	损失函数中的平衡系数	float

8.3.4　自蒸馏

自蒸馏指的是教师模型和学生模型使用相同的网络结构的知识蒸馏方法。

例如 GNN-SD、IGSD 为自蒸馏模型，见表 242 ~表 243

GNN-SD 是通过邻域差异率和自适应差异率保留正则化的自蒸馏方法，GNN-SD 无需外部教师模型即可在 GNN 层间有效传递关键知识。模型定义见表 242。

表 242　GNN-SD 模型定义

运算操作	描述	字段	关键字	定义	数据类型
GNN-SD	可以通过在 GNN 不同层之间传递知识来缓解过平滑问题的自蒸馏方法，使用特征张量、标签张量、邻接矩阵，输出学生模型在不同下游任务上的预测结果	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			g	输入图，与 edge_index 二选一	Graph

表 242　GNN-SD 模型定义（续）

运算操作	描述	字段	关键字	定义	数据类型
GNN-SD	可以通过在 GNN 不同层之间传递知识来缓解过平滑问题的自蒸馏方法，使用特征张量、标签张量、邻接矩阵，输出学生模型在不同下游任务上的预测结果	Output	prediction	输出张量	Tensor
		Attributes	hidden_dim	隐藏层维度	int
			dropout	神经网络每个节点的丢弃概率	float
			learning_rate	优化器的学习率	float
			alpha	损失函数中的平衡系数	float
			beta	损失函数中的平衡系数	float

IGSD 是一种迭代图自蒸馏方法。IGSD 使用自监督对比学习和图扩散增强，无监督学习图级表示，在半监督场景中通过结合有监督和自监督损失进行优化，可以提升图表示的性能和区分度。模型定义见表 243。

表 243　IGSD 模型定义

运算操作	描述	字段	关键字	定义	数据类型
IGSD	用于图表示学习的图神经网络自蒸馏方法，通过对图实例的增强视图的实例鉴别来迭代执行蒸馏过程，使用特征张量、邻接矩阵，输出学习到的图表示（自监督场景）或图预测情况（半监督场景）	Input	X	节点特征矩阵	tensor
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			g	输入图，与 edge_index 二选一	Graph
		Output	prediction	输出张量	tensor
		Attributes	proj_hidden_size	映射隐藏层维度	int
			proj_size	映射层维度	int
			learning_rate	优化器的学习率	float
			label_thre	伪标签阈值	float
			Y	可选，图标签	tensor

8.4　模型加速

8.4.1　并行加速策略

8.4.1.1　流水线并行

流水线并行是将模型按层或模块划分为多个部分，并分配给不同的设备进行并行处理。每一个设备都执行前向和反向的计算，最后在其他设备上进行更新参数。常用的小批量流水线并行将数据切分为不同的批次，第一个批次的数据首先在第一个设备上进行前向传播，然后将中间结果传递给下一个设备。同时，第二个批次的数据可以在第一个设备上进行前向传播，不同的设备可以同时进行计算，等所有数据计算完成后进行反向计算。每个设备根据计算出的梯度更新其部分的模型参数。重复上述步骤，直到完成所有数据的训练批次。模型定义见表244。

表 244　流水线并行模型定义

模型	描述	字段	关键字	定义	数据类型
PipelineParallel	不同的 GNN 层被分配给不同的设备。每个设备负责处理一层 GNN 层的数据。数据从一个设备流向下一个设备，每个设备在其负责的层上进行计算	Input	module	模型	Model
			device_num	设备数量	int
			g	图拓扑	Graph
			feature	顶点特征	tensor
			feature_dim	顶点特征维度	int
		Output	output_device	输出设备的 id	int

8.4.1.2　数据并行

数据并行模型根据设备数量对数据进行划分，把每份数据分配给不同的计算设备。每个计算设备上都有完整的模型副本。在训练过程中，每个设备独立地进行前向和反向传播，计算梯度。最后汇总所有设备上的梯度，更新模型参数。重复上述过程，直至训练完成。模型定义见表 245。

表 245　数据并行模型定义

模型	描述	字段	关键字	定义	数据类型
DataParallel	数据集划分到不同的设备上，每个设备都具有完整的模型副本。设备之间各自完成前向和后向计算	Input	module	模型	Model
			device_num	设备数量	int
			g	图拓扑	Graph
			feature	顶点特征	tensor
			batch_size	批量大小	int
		Output	output_device	输出设备的 id	int

8.4.1.3　张量并行

张量并行是指将单个张量，例如顶点特征和顶点嵌入，按照维度切分到多个设备上，每一个设备处理数据部分维度的并行计算方法。该方法首先将顶点特征或嵌入张量按特定维度切分，每个子张量分配到不同设备上。每个设备独立进行前向传播，处理其负责的子张量部分，同时在计算过程中各设备间进行必要的信息传递，以确保非线性变换和其他依赖完整张量信息的操作能够正确执行。在使用完整张量进行下游任务与损失函数计算后，得到的梯度张量继续按照特定维度切分到不同设备上进行反向传播。最后汇总所有设备上的模型参数梯度，更新模型参数。模型定义见表246。

表 246　张量并行模型定义

模型	描述	字段	关键字	定义	数据类型
TensorParallel	根据设备数量切分张量，每个设备计算部分数据	Input	module	模型	Model
			device_num	设备数量	int
			g	图拓扑	Graph
			feature	顶点特征	tensor
		Output	Module	并行化的模块	Model

8.4.2　迭代加速策略

迭代加速可以在大规模图数据的处理过程中提高图神经网络的训练和推理效率。分布式训练过程中，迭代机制分为同步机制，异步机制，以及同步异步混合形式。同步机制要求所有计算节点在进入下一计算周期前达成一致，能够保持全局参数的一致性；异步机制中各节点独立更新其局部参数，不必等

待其他节点，能够提高训练的并行性和效率；同步异步混合模式可以优化训练过程中的效率与模型准确性之间的权衡。该模式综合运用了同步和异步两种基本的训练机制，能够适应不同的网络状况和计算需求，训练性能更高。

SSP 是一种混合训练模型。该模型以异步的方式执行训练，每隔固定迭代次数强制进行同步更新。SSP 能够根据训练的具体情况灵活调整同步和异步操作，提高模型训练的效率，保证模型的收敛性并适应不同的硬件和网络环境。模型定义见表 247。

表 247　SPP 模型定义

模型	描述	字段	关键字	定义	数据类型
SSP	使用有界陈旧性的同步异步混合模式并行训练	Input	X	节点特征矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			ssp_flag	是否使用同步异步混合模式	bool
			staleness_bound	有界陈旧值	int
		Output	Y	节点标签	tensor
		Attributes	in_channels	输入特征的维度	int
			out_channels	输出特征的维度	int
			Module	可选，隐藏层特征的维度	int
			n_layers	隐藏层层数	int

8.4.3　图划分策略

应用在分布式图神经网络系统中的图划分方法有四种：哈希、Metis、Metis-extend和流式划分。

a) 哈希划分通过定义不同的映射规则如顶点或边的哈希值，随机分配顶点以平衡计算和通信负载。
b) Metis 可以快速高效地将大型稀疏图划分为多个大小几乎相等的子图，同时最小化割边。
c) Metis-extend 方法使用聚类算法和添加各种约束，如顶点掩码和顶点度数，优化传统的 Metis 算法。该方法能够使带标签顶点的邻居集中，平衡顶点和边的数量，减小计算和通信负载。
d) 流式划分采用动态划分策略，通过调整评分函数灵活应对不同的划分任务。

partitiong_graph结合了不同的图划分方法，可以优化图的划分效果，适应不同的GNN系统需求和运行环境。partitiong_graph运算操作定义见表248。

表 248　partitiong_graph 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
partitiong_graph	对原始图拓扑进行图划分	Input	X	节点特征矩阵	tensor
			g	输入图，与 edge_index 二选一	Graph
			edge_index	边索引，与 g 二选一	tensor SparseTensor
			num_parts	图分区数量	int
			part_method	分区方法	string

表 248　partitiong_graph 运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
partitiong_graph	对原始图拓扑进行图划分	Output	partition_g	分区图	List [Graph]
			partition_X	分区图节点特征	List [tensor]
		Attributes	balance_ntypes	Mestis 可选，根据节点类型平衡划分图	bool
			balance_nodes	Mestis 可选，根据节点数量平衡划分图	bool
			balance_edges	Mestis 可选，根据边数量平衡划分图	bool
			balance_weight	Mestis 可选，根据节点或边权重平衡划分图	bool
			function_eval	流式划分可选，应用评分函数评价划分图	bool

8.4.4　通信加速策略

8.4.4.1　无损通信优化

无损通信优化是图神经网络分布式训练中的关键技术，可以提高图神经网络的通信效率，同时保证模型的准确率不受影响。无损通信优化主要通过缓存实现，分为三种类型：优先级缓存、全量缓存和部分缓存。没有缓存则称为DepComm。

优先级缓存可以分为度数大优先缓存和采样率高优先缓存。度数大优先缓存优先存储度数较大的，对训练影响较大的节点数据。采样率高优先缓存则优先存储被高频采样的节点数据，以减少频繁通信带来的开销。

全量缓存（DepCache）是将所有节点的数据全部缓存，以避免重复的通信操作。

部分缓存根据实际需求和缓存资源，选择性地缓存部分节点数据，可以在通信和存储资源之间找到平衡，是DepComm和DepCache的结合。

度数大优先缓存以更高的概率采样高出度节点进行缓存。度数大优先缓存运算操作定义见表 249。

表 249　度数大优先缓存运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
度数大优先缓存	选择高出度顶点来填充缓存	Input	g	输入图	Graph
			embed_names	特征名称列表	List [string]
			sort_nid	按照出度排序的顶点 ID	tensor
			node_num	顶点数量	int
			dims	特征维度	int
			nid_map	本地顶点 ID 到全局 ID 的映射	tensor
			data_frame	包含特征名称和对应特征数据的字典	Dict [string, float]

表 249　度数大优先缓存运算操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
度数大优先缓存	选择高出度顶点来填充缓存	Output	cached_feature	缓存的特征	Tensor
		Attributes	peak_allocated_mem	GPU 上分配的最大内存	int
			peak_cached_mem	GPU 上缓存的最大内存	int

采样率高优先通过 GPU 内存使用的一次性采样来决定缓存大小。采样率高优先缓存运算操作定义见表 250。

表 250　采样率高优先缓存运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
采样率高优先缓存	在第一个 mini-batch 的训练周期结束后，检查空闲 GPU 内存的大小，并相应地分配可用的 GPU 内存来缓存图数据	Input	g	输入图	Graph
			embed_names	特征名称列表	List [string]
			node_num	顶点数量	int
			dims	特征维度	Int
			nid_map	本地顶点 ID 到全局 ID 的映射	tensor
			data_frame	包含特征名称和对应特征数据的字典	Dict [string, float]
		Output	cached_feature	缓存的特征	tensor
		Attributes	peak_allocated_mem	GPU 上分配的最大内存	int
			peak_cached_mem	GPU 上缓存的最大内存	int

DepCache 将所有节点的数据全部缓存。DepCache 运算操作定义见表 251。

表 251　DepCache 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
DepCache	每个 worker 在训练开始前，让依赖它们的邻居在本地做好准备	Input	g	输入图	Graph
			L	模型层数	int
			label	顶点标签	tensor
			h	初始节点特征	tensor
			W	初始参数	tensor
		Output	W	更新后的参数	tensor

部分缓存是指在内存有限，特别是 GPU 内存有限的情况下，只缓存图的一部分节点及其特征数据。部分缓存运算操作定义见表 252。

表 252　部分缓存运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
部分缓存	使用贪婪的启发式方法来划分 DepCache 和 DepComm 的依赖关系	Input	vertices_subset	顶点子集	set
			edge_subset	边子集	set
			remote_dependent_neighbors	远程依赖邻居	List [set]
		Output	partitions_of_dependencies	依赖关系的划分集合	List [set]
		Attributes	S	内存限制	int

8.4.4.2 通信压缩与有损通信优化

8.4.4.2.1 概述

图神经网络的通信压缩是指在分布式GNN训练和推理过程中减少节点之间的数据传输量，降低通信开销，提高整体计算效率的过程。通信压缩可以在确保GNN模型准确性和性能的前提下，显著降低分布式计算中的通信负载，提高分布式GNN训练和推理的效率。通信压缩和有损通信优化可以分为边界的随机采样，数据随机丢弃和通信数据量化。

a) 边界的随机采样随机选取部分远程邻居顶点以减少传输数据量。
b) 数据随机丢弃将要传输的数据进行随机丢弃以减少传输数据量。
c) 通信数据量化将浮点数数据转换为低精度格式以减少传输数据量。

8.4.4.2.2 边界的随机采样/数据随机丢弃

边界节点采样BNS是用于GNN训练的方法。该方法能够有效处理大规模图数据，提升训练效率，同时保持模型的性能。

BNS选择性地关注图中的边界节点，只采样和处理具有较大的信息量和代表性的一部分节点及其邻居，能够减少计算开销，实现高效的训练过程。BNS的运算操作定义见表253。

表 253 BNS 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
BNS	用于图神经网络(GNN) 的分布式训练方法，其通过在每个训练迭代中随机采样边界顶点，可以减少通信开销，提高训练效率	Input	X	顶点特征矩阵	tensor
			g	输入图，与edge_index 二选一	Graph
			edge_index	边索引，与 g 二选一	tensor
		Output	New_X	通信后拥有部分边界邻居信息的节点特征矩阵	tensor
		Attributes	Fanout	边界点采样率	float

8.4.4.2.3 通信数据量化

通信数据量化是在分布式GNN训练中减少通信开销和加速训练过程的技术。

通信数据量化将高精度的浮点数表示简化为低精度表示，如8位或16位整数，从而减少数据传输量。尽管量化会引入一定的误差，但适当的量化策略可以在保持模型性能的前提下，大幅减少通信开销。通信数据量化的运算操作定义见表254。

表 254 通信数据量化运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
Comm-quantization	将节点通信的数据转换为精度更低的定点数或浮点数，减少数据传输量。/用更少的比特位量化传出数据，以减少分布式通信量	Input	X	顶点特征矩阵	tensor
			g	输入图，与edge_index 二选一	Graph
			edge_index	边索引，与 g 二选一	tensor
		Output	Quantization_data	量化后要传输的嵌入矩阵	tensor
		Attributes	n_bits	量化后的数据比特位数	int

9 图神经网络计算框架

9.1 基于深度学习平台的图神经网络计算框架

9.1.1 系统体系结构

基于深度学习平台的图神经网络计算框架由多个组件构成，这些组件协同工作以支持图数据的表示学习、模型训练和推理等任务。框架的组成和相关组件之间的关系如下：

a) 系统的整体组件逻辑结构

- 图数据预处理层：提供全图加载和批量加载的图数据导入方式，负责图数据的加载、格式化和预处理，以便后续的模型训练和推理。该层包括图的构建、特征的归一化、标签的准备等。
- 图模型构建层：通过图卷积网络（GCN）或其他类型的 GNN 模型来学习节点、边或图的表示。这一层是 GNN 的核心，负责从图数据中提取有用的信息。
- 模型训练层：包含优化算法和反向传播机制，用于调整模型参数以最小化损失函数。这一层与深度学习平台紧密集成，以利用高效的计算资源。整个训练过程一般包括前向传播、损失计算、反向传播和参数更新等。
- 模型推理和评测层：在训练完成后，用于在新数据上执行预测和分类等任务，评测模型性能，通常包括准确性、F1 分数、AUC 和其它度量标准。

b) 各个组件的职责、交互方式和关系

- 数据预处理层与图模型构建层：数据预处理层输出格式化的图数据，供图模型构建层使用。这两层之间的交互是通过数据流进行的，预处理后的数据作为模型输入。
- 图模型构建层与模型训练层：图模型构建层的输出（节点或图的嵌入）被用于模型训练层中的损失计算。这两层通过梯度下降等优化算法进行交互，以改进模型性能。
- 模型训练层与推理层：训练好的模型参数被保存并部署，以进行模型推理。

框架图如图 3 所示。

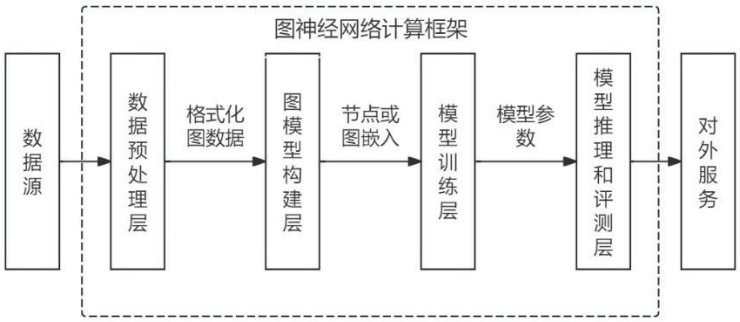

图 3 基于深度学习平台的图神经网络计算框架系统结构示意图

9.1.2 与深度学习平台的接口规范

9.1.2.1 张量计算接口

张量计算接口是用于执行张量基本运算的接口，支持包括随机张量的生成、张量的四则运算以及逻辑运算在内的多种操作。其中，部分接口定义信息见 GB/T 42382.1-2023，具体如下：

abs：返回张量中每一个元素的绝对值。abs 运算操作定义见 GB/T 42382.1-2023 表 18。

add：用于两个张量相加，如果张量形状不同，会进行广播。add 运算操作定义见 GB/T 42382.1-2023 表 20。

arrange：用于创建一个在指定区间内等间隔排列的一系列数值。arrange 运算操作定义见 GB/T 42382.1-2023 表 93。

argmax：返回张量中最大值的索引。argmax 运算操作定义见 GB/T 42382.1-2023 表 23。

argmin：返回张量中最小值的索引。argmin 运算操作定义见 GB/T 42382.1-2023 表 24。
cast：修改张量的数据类型。cast 运算操作定义见 GB/T 42382.1-2023 表 32。
ceil：对张量中每一个元素向上取整。ceil 运算操作定义见 GB/T 42382.1-2023 表 33。
concat：沿指定维度对几个张量进行拼接。concat 运算操作定义见 GB/T 42382.1-2023 表 34。
cos：计算张量中每个元素的余弦值。cos 运算操作定义见 GB/T 42382.1-2023 表 41。
diag：从一个一维张量创建一个对角矩阵或从矩阵提取对角元素。diag 运算操作定义见 GB/T 42382.1-2023 表 49。
divide：张量逐元素相除。divide 运算操作定义见 GB/T 42382.1-2023 表 50。
equal：判断两个张量是否逐元素相等。equal 运算操作定义见 GB/T 42382.1-2023 表 57。
exp：计算张量中每个元素的 exp 值。exp 运算操作定义见 GB/T 42382.1-2023 表 58。
gather：根据索引从输入张量抽取值。gather 运算操作定义见 GB/T 42382.1-2023 表 62。
logical_and：对两个张量进行逐元素逻辑与。logical_and 运算操作定义见 GB/T 42382.1-2023 表 22。
logical_not：对两个张量进行逐元素逻辑非。logical_not 运算操作定义见 GB/T 42382.1-2023 表 83。
logical_or：对两个张量进行逐元素逻辑或。logical_or 运算操作定义见 GB/T 42382.1-2023 表 85。
logical_xor：对两个张量进行逐元素逻辑异或。logical_xor 运算操作定义见 GB/T 42382.1-2023 表 125。
matmul：对两个满足要求的张量进行矩阵乘。matmul 运算操作定义见 GB/T 42382.1-2023 表 78。
multiply：两个张量的逐元素乘。multiply 运算操作定义见 GB/T 42382.1-2023 表 81。
pow：对张量中每个元素进行指数运算。pow 运算操作定义见 GB/T 42382.1-2023 表 90。
reduce_mean：计算指定维度上张量的均值。reduce_mean 运算操作定义见 GB/T 42382.1-2023 表 96。
reduce_max：计算指定维度上张量的最大值。reduce_max 运算操作定义见 GB/T 42382.1-2023 表 97。
reduce_min：计算指定维度上张量的最小值。reduce_min 运算操作定义见 GB/T 42382.1-2023 表 98。
reduce_sum：计算指定维度上张量的数值和。reduce_sum 运算操作定义见 GB/T 42382.1-2023 表 100。
reshape：用于改变张量的形状。reshape 运算操作定义见 GB/T 42382.1-2023 表 104。
shape：用于获取张量的形状。shape 运算操作定义见 GB/T 42382.1-2023 表 108。
sin：计算张量中每个元素的正弦值。sin 运算操作定义见 GB/T 42382.1-2023 表 110。
sqrt：计算张量中每个元素的平方根。sqrt 运算操作定义见 GB/T 42382.1-2023 表 115。
stack：沿一个新维度堆叠一系列张量。stack 运算操作定义见 GB/T 42382.1-2023 表 117。
substract：张量逐元素相减。substract 运算操作定义见 GB/T 42382.1-2023 表 118。
squeeze：移除张量中所有长度为 1 的维度。squeeze 运算操作定义见 GB/T 42382.1-2023 表 121。
transpose：用于交换张量的两个维度。transpose 运算操作定义见 GB/T 42382.1-2023 表 122。
unsqueeze：在张量中添加一个长度为 1 的维度。unsqueeze 运算操作定义见 GB/T 42382.1-2023 表 123。
zeros：生成形状一定，数值为 0 的张量。zeros 运算操作定义见 GB/T 42382.1-2023 表 129。
本标准还包括一些额外的张量计算接口，具体见表 255 ~表 266。
atan 运算操作定义见表 255。

表 255　atan 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
atan	计算张量中每一个元素的反正切值	Input	X	输入张量	tensor
		Output	Y	输出张量	tensor

cumsum 运算操作定义见表 256。

表 256　cumsum 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
cumsum	计算张量中沿指定维度的累积和	Input	X	输入张量	tensor
			dim	可选，求和维度	int
		Output	Y	输出张量	tensor

eye 运算操作定义见表 257。

表 257　eye 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
eye	创建一个二维的单位矩阵	Input	m	行数	int
			n	可选，列数	Int
			dtype	可选，生成张量的数据类型	string
		Output	Y	输出张量	tensor

mask_select 运算操作定义见表 258。

表 258　mask_select 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
mask_select	根据一个布尔掩码张量筛选输入张量元素	Input	X	输入张量	tensor
			mask	掩码张量	tensor
			axis	可选，选择维度	int
		Output	Y	输出张量	tensor

ones 运算操作定义见表 259。

表 259　ones 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
ones	生成形状一定，数值为 1 的张量	Input	shape	输出张量的维度	List [int]
			dtype	可选，输出张量的数据类型	string
		Output	Y	输出张量	tensor

ones_like 运算操作定义见表 260。

表 260　ones_like 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
ones_like	创建一个形状与输入张量相同、所有元素为 1 的张量	Input	X	输入张量	tensor
			dtype	可选，输出张量的数据类型	string
		Output	Y	输出张量	tensor

tile 运算操作定义见表 261。

表 261　tile 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
tile	用于沿各个维度生成重复的张量	Input	X	输入张量	tensor
			multiples	重复次数	tensor Tuple [int, int] List [int]
		Output	Y	输出张量	tensor

tan 运算操作定义见表 262。

表 262　tan 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
tan	计算张量中每个元素的正切值	Input	X	输入张量	tensor
		Output	Y	输出张量	tensor

to_tensor 运算操作定义见表 263。

表 263　to_tensor 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
to_tensor	将数据转换为张量	Input	X	输入张量	tensor
			dtype	可选，输出张量的数据类型	string
		Output	Y	输出张量	tensor

to_numpy 运算操作定义见表 264。

表 264　to_numpy 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
to_numpy	将张量转换为 numpy.array	Input	X	输入张量	tensor
		Output	Y	输出张量	tensor

where 运算操作定义见表 265。

表 265　where 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
where	根据一个条件张量筛选两个张量的元素	Input	X	输入张量 1	tensor
			Y	输入张量 2	tensor
			condition	条件	tensor
		Output	Z	输出张量	tensor

zeros_like 运算操作定义见表 266。

表 266　zeros_like 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
zeros_like	创建一个形状与输入张量相同、所有元素为 0 的张量	Input	X	输入张量	tensor
			dtype	输出张量的数据类型	string
		Output	Y	输出张量	tensor

9.1.2.2　数据加载与预处理接口

数据加载与预处理接口是用于加载和预处理数据的接口，支持文本或序列数据的加载与处理。包括数据加载接口，dataloader 的定义等，具体见表 267。

表 267　dataloader 操作定义

运算操作	描述	字段	关键字	定义	数据类型
dataloader	加载和预处理数据集，共神经网络模型使用	Input	dataset_path	输入数据集路径	string
			batch_size	可选，批量大小	int
			shuffle	可选，是否打乱	bool
			drop_last	可选，是否丢弃最后一组数据	bool
		Output	Y	一组划分好的数据	List [tensor]

9.1.2.3　模型构建接口

模型构建接口是用于构建神经网络模型的接口，包括模型结构的定义、参数的初始化等。其中，部分接口定义信息见 GB/T 42382.1-2023，具体如下：

Linear：线性层，全连接层。Linear 运算操作定义见 GB/T 42382.1-2023 表 130。

random_unifrom：根据均匀分布初始化网络权重。random_unifrom 运算操作定义见 GB/T 42382.1-2023 表 92。

random_normal：根据正态分布初始化网络权重。random_normal 运算操作定义见 GB/T 42382.1-2023 表 91。

本标准还包括一些额外的模型构建接口，具体见表 268 ~表 273。

Module 定义见表 268

表 268　Module 操作定义

运算操作	描述	字段	关键字	定义	数据类型
Module	神经网络模块的基类，自定义的模型应当继承这个类	Input	X	输入数据	tensor
		Output	Y	输出张量	tensor
		Attributes	params	模型参数	map<string , tensor>
			layers	神经网络层	map<string , Module> map<string , None>
			params_status	模型参数是否为可学习参数	map<string , bool>
			trainable_weights	可学习参数列表	List [tensor]

表 268　Module 操作定义（续）

运算操作	描述	字段	关键字	定义	数据类型
Module	神经网络模块的基类，自定义的模型应当继承这个类	Attributes	nontrainable_weights	不可学习参数列表	List [tensor]
			all_weights	所有参数列表	List [tensor]
			is_train	模型是否为训练模式	bool

ModuleList 定义见表 269。

表 269　ModuleList 操作定义

运算操作	描述	字段	关键字	定义	数据类型
ModuleList	专门存储神经网络模块的列表	Input	X	输入数据	tensor
		Output	Y	输出张量	tensor
		Attributes	layer_list	神经网络列表	List [Module]

ModuleDict 定义见表 270。

表 270　ModuleDict 操作定义

运算操作	描述	字段	关键字	定义	数据类型
ModuleDict	专门存储神经网络模块的字典	Input	X	输入数据	tensor
		Output	Y	输出张量	tensor
		Attributes	layer_dict	神经网络列表	map<string , Module>

truncated_normal 运算操作定义见表 271。

表 271　truncated_normal 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
truncated_normal	根据截断正态分布中抽取的样本，初始化网络权重	Input	shape	输出张量的形状	Tuple [int, int]
			mean	可选，正态分布的期望	float
			std	可选，正态分布的标准差	float
			dtype	可选，输出张量的数据类型	string
			seed	可选，随机数种子	int
		Output	Y	输出张量	tensor

xavier_uniform 运算操作定义见表 272。

表 272　xavier_uniform 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
xavier_uniform	从均匀分布中初始化网络权重，分布范围根据输入和输出神经元数量自动调整	Input	shape	输出张量的形状	Tuple [int, int]
			gain	可选，均匀分布参数	float
			dtype	可选，输出张量的数据类型	string
			seed	可选，随机数种子	int
		Output	Y	输出张量	tensor

xavier_normal 运算操作定义见表 273。

表 273　xavier_normal 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
xavier_normal	从正态分布中初始化网络权重，分布范围根据输入和输出神经元数量自动调整	Input	shape	输出张量的形状	Tuple [int, int]
			gain	可选，正态分布参数	float
			dtype	可选，输出张量的数据类型	string
			seed	可选，随机数种子	int
		Output	Y	输出张量	tensor

9.1.2.4　激活函数接口

激活函数接口是调用非线性变换函数的接口，包括 ReLU、Sigmoid 等。其中，部分接口定义信息见 GB/T 42382.1-2023，具体如下：

ELU：为负输入提供负值缓解梯度消失问题。ELU 运算操作定义见 GB/T 42382.1-2023 表 134。

ReLU：最常用的激活函数之一。ReLU 运算操作定义见 GB/T 42382.1-2023 表 102。

ReLU6：ReLU 的变体，通过设置上限为 6 以提供输出数值的范围限制。ReLU6 运算操作定义见 GB/T 42382.1-2023 表 103。

LeakyReLU：类似于 PReLU，但斜率是预先设定，不可学习。LeakyReLU 运算操作定义见 GB/T 42382.1-2023 表 140。

PReLU：ReLU 的一种变体，负数部分的斜率是可学习的参数，不同的输入特征，斜率可以不同。PReLU 运算操作定义见 GB/T 42382.1-2023 表 142。

Sigmoid：将输入值压缩到[0, 1]范围内，通常用于二分类问题的输出层。Sigmoid 运算操作定义见 GB/T 42382.1-2023 表 143。

Softmax：归一化指数函数，通常用于多分类问题的输出层，将一组值转换为概率分布。Softmax 运算操作定义见 GB/T 42382.1-2023 表 145。

Softplus：ReLU 的平滑版本。Softplus 运算操作定义见 GB/T 42382.1-2023 表 146。

Tanh：双曲正切激活函数。Tanh 运算操作定义见 GB/T 42382.1-2023 表 148。

本标准还包括额外的激活函数接口，GeLU 运算操作定义等具体见表 274。

表 274　GeLU 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
GeLU	该激活函数常见于 Transformer 模型，可以看作 ReLU 和 ELU 的一种折中选择	Input	X	输入张量	tensor
		Output	Y	输出张量	tensor

9.1.2.5　损失函数接口

损失函数接口时用于计算预测值和真实值之间的差距的接口。其中，部分接口定义信息见 GB/T 42382.1-2023，具体如下：

cross_entropy：适用于多分类任务的损失函数，对于每个样本，该函数会计算正确类别概率的负对数。cross_entropy 运算操作定义见 GB/T 42382.1-2023 表 43。

simoid_cross_entropy：适用于二分类任务，会通过 sigmoid 函数将输出压缩到[0, 1]范围，随后计算交叉熵损失函数。simoid_cross_entropy 运算操作定义见 GB/T 42382.1-2023 表 144。

本标准还包括一些额外的损失函数接口，具体见表 275~表 276。

binary_cross_entropy 运算操作定义见表 275。

表 275　binary_cross_entropy 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
binary_cross_entropy	用于二元分类任务的损失函数，计算真实标签和预测标签之间的交叉熵	Input	output	输入张量	tensor
			target	标签	tensor
			reduction	可选，计算模式，可以选择“mean”、“sum”、“none”	string
		Output	Y	输出张量	tensor

mean_squared_error 运算操作定义见表 276。

表 276　mean_squared_error 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
mean_squared_error	适用于回归任务的损失函数，计算预测值和真实值之间的均方误差	Input	output	输入张量	tensor
			target	标签	tensor
			reduction	可选，计算模式，可以选择“mean”、“sum”、“none”	string
		Output	Y	输出张量	tensor

9.1.2.6　优化器接口

优化器接口是用于模型训练中参数优化的接口。以下优化器部分定义及描述信息见 GB/T 42382.1-2023 表 154 归一化表达以及后续接口支持情况说明。

SGD：最传统的优化方法，旨在有效地调整网络参数以最小化损失函数。

Adagrad：适用于处理稀疏数据，频繁更新的参数学习率会降低，较少更新的参数学习率会增加。

Adadelta：主要解决 Adagrad 训练后期学习率急剧下降的问题。

Adam：计算梯度的一阶和二阶矩估计，并根据这些估计调整学习率。

Adamax：Adam 的一个变体，主要基于无穷范数，提供一种更稳定的优化方法，特别是在非稳定参数更新的情况。

RMSprop：通过维持一个平均值来调整每一个参数的学习率，避免 Adagrad 学习率持续下降的问题。

9.1.2.7 正则化接口

正则化接口是用于提高模型的泛化能力并防止模型过拟合的接口。其中，部分接口定义信息见 GB/T 42382.1-2023，具体如下：

batchnorm：将输入规范化，使其均值为 0，方差为 1。batchnorm 运算操作定义见 GB/T 42382.1-2023 表 132。

dropout：在训练过程中随机丢弃一些神经元，使得网络不依赖于某一特定特征，提升其泛化能力。dropout 运算操作定义见 GB/T 42382.1-2023 表 51。

layernorm：类似于 batchnorm，但归一化是在单个样本层面而非整个批次。layernorm 运算操作定义见 GB/T 42382.1-2023 表 72。

l2_normalize：将输入特征向量除以其 L2 范数，通常用于约束网络权重和特征向量，使其不会变得过大。l2_normalize 运算操作定义见 GB/T 42382.1-2023 表 76。

本标准还包括额外的正则化接口，具体见表 277。

gaussian_noise 运算操作定义见表 277。

表 277 gaussian_noise 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
gaussian_noise	在输入添加服从高斯分布的随机噪声，可以增加模型对输入噪声的鲁棒性，提高泛化能力	Input	mean	可选，噪声均值	float
			std	可选，噪声方差	float
			is_always	可选，是否在训练和测试中都添加噪声	bool
			seed	可选，随机数种子	int
		Output	Y	输出张量	tensor

9.1.2.8 下游评测接口

下游评测接口是用于模型的评估和测试的接口。其中，部分接口定义信息见 GB/T 42382.1-2023，具体如下：

accuracy：表示模型预测正确的样本占总样本的比例。accuracy 运算操作定义见 GB/T 42382.1-2023 表 19。

auc：衡量模型在不同阈值设置下的性能，AUC 越高，代表模型区分不同类别的能力越强。auc 运算操作定义见 GB/T 42382.1-2023 表 29。

本标准还包括一些额外的下游评测接口，具体见表 278 ~表 279。

precision 运算操作定义见表 278。

表 278　precision 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
precision	衡量模型预测为正类的样本有多少是真正的正类	Input	pred	预测结果	tensor
			label	标签	tensor
		Output	Y	输出张量	tensor

recall 运算操作定义见表 279。

表 279　recall 运算操作定义

运算操作	描述	字段	关键字	定义	数据类型
recall	衡量模型正确识别正类样本的能力	Input	pred	预测结果	tensor
			label	标签	tensor
		Output	Y	输出张量	tensor

9.1.2.9　其他接口

其他深度学习接口，包括日志、模型保存、可视化功能等。

9.2　图神经网络计算框架与第三方数据源接口

9.2.1　图数据库接口规范

图神经网络计算框架与图数据库的接口规范定义了图神经网络框架与图数据库之间进行数据交互的方式和规则。将图数据库作为外部数据源接入图神经网络框架的基本过程包括：

a)　图神经网络框架连接数据库；
b)　从图数据库中拉取数据，并构造图神经网络框架支持的图数据类型；
c)　使用图神经网络框架的采样、消息传递等模块进行训练/推理；
d)　将得到的嵌入/预测结果写回图数据库。

同时，考虑到图神经网络框架获取部分子图的需求，还应该提供图数据库的查询接口。

据此，图数据库接口规范应该主要包含以下模块：

1)　图数据库连接接口：实现图神经网络计算框架与图数据库的连接，使计算框架能够与图数据库通过接口进行交互。
2)　数据查询接口：用于向图数据库查询并获取图数据。这些接口可能包括指定节点或边的查询，按条件过滤数据，获取节点/边的属性信息等并输入至图神经网络框架。
3)　数据获取接口：用于将数据从图数据库导入到计算框架中，并进行图神经网络模型的训练和推断。这些接口可以将从图数据库中获取的数据转化为规定的中间标准数据形式。
4)　数据写回接口：在图神经网络计算过后，可能需要对图数据库中的数据进行写回操作，如将训练结果写回图数据库中。

9.2.1.1　图数据库连接接口

图数据库连接接口（GraphDBConnection）是用于维护图数据库连接的接口，图数据库的连接信息定义见表 280。

表 280　图数据库的连接信息定义

字段	类型	定义
graph_address	string	图数据库地址
user_name	string	用户名
password	string	密码

9.2.1.2 获取图数据接口

获取图数据接口是用于从图数据获取图数据的接口。先规定点、边的导出配置，然后通过导出配置来控制图数据的输出，可以是整图也可以是规定了点类型或者边类型的子图。具体定义见表 281~表 283。

点导出配置（NodeExportConfig）决定了如何从图数据库拉取节点数据。点导出配置定义见表 281。除了节点本身外，节点上可能带有特征和标签的信息。表中 x_property_names 定义了需要从图数据库中获取哪些属性用来构建特征信息，而 y_property_names 表示标签对应的属性名。

表 281 点导出配置定义

字段	类型	定义
label_name	string	节点类型名
x_property_names	List [string]	可选，节点特征对应的属性名
y_property_names	List [string]	可选，节点标签对应的属性名

类似地，边导出配置（EdgeExportConfig）定义见表 282。

表 282 边导出配置定义

字段	类型	定义
label_name	string	边类型名
src_dst_label	Tuple [string, string]	源节点与目的节点类型
x_property_names	List [string]	可选，边特征对应的属性名
y_property_names	List [string]	可选，边标签对应的属性名

通过点、边的导出配置，可从图数据库中导出图神经网络框架需要的图数据结构。同时可以通过点、边的不同导出配置，规定点或边的类型来实现获取相应子图。图数据的详细定义参见本标准第 5.2 章节中的图数据类型定义。本标准使用一个统一的列表来表示点、边导出配置，当图数据为同质图时，列表元素只有一个，而为异质图时则可以有多个元素。获取图数据接口定义见表 283。

表 283 获取图数据接口定义

运算操作	描述	字段	关键字	定义	数据类型
get_graph	从图数据库中导出异质图	Input	conn	图数据库连接信息	GraphDBConnection
			graph_name	图名	string
			node_export_config	点导出配置	List [NodeExportConfig]
			edge_ export_config	边导出配置	List [EdgeExportConfig]
		Output	graph	图神经网络框架的图数据	HeteroGraph

9.2.1.3 查询接口

查询接口是用于向图数据库查询并获取图数据信息的接口。这些接口可以按类型、属性值等条件过滤数据以获取具体的节点/边，同时返回符合条件的点或边的全部信息。具体接口的定义见表 284。

表 284　查询接口定义

运算操作	描述	字段	关键字	定义	数据类型
match	在图数据库中查询	Input	graph_name	图名	string
			label_name	点或边的类型	List [string]
			src_dst_label	对于边要限制源节点与目的节点类型	Tuple [string, string]
			x_property_names	点或边的属性及其值	Dict [string, value]
			y_property_names	点或边的标签及其值	Dict [string, value]
		Output	node_all 或 edge_all	符合条件的点或边的全部信息	Dict[string, List]

9.2.1.4　写回接口

写回接口是用于向图数据库写回属性的接口，写回的属性包括训练得到的节点嵌入，以及节点或边的预测标签等。写回接口定义见表 285。

表 285　写回接口定义

运算操作	描述	字段	关键字	定义	数据类型
write_property	向图数据库写入属性	Input	label_name	类型名	string
			src_dst_label	对于边要限制源节点与目的节点类型	Tuple [string, string]
			property_name	写入属性名	string
			property_values	写入值	List [value]
		Output	success	写入成功标志	bool

9.2.2　批处理系统接口规范

9.2.2.1　批处理输入

图数据接口需要分别定义点、边数据格式，同时，还需要定义标签数据格式，用于描述需要对哪些点、边生成图样本。具体见表 286~表 291。

点数据格式定义见表 286。

表 286　点数据格式

字段名	字段名（中文）	说明
id	点编号	用于唯一标识图中的一个点
feature	点属性	用于描述节点的属性信息

feature 字段用于存储类型信息和特征信息，类型信息用于表示点类型，特征信息用于表示各种类型的特征，定义见表 287。

表 287　点数据 feature 字段定义

字段名	字段名（中文）	说明
type	类型名	表示点类型，如果是同质图，可以为空
sparseKey	稀疏键	表示一组特征 ID，如：1，2，3
sparseKV	稀疏键值对	表示一组特征 ID-Value 对，如：1:0.1，2:0.2，4:0.01
dense	稠密数组	表示一个特征向量，如：0.1，0.2，0.002，0.4
raw	原始数据	表示一个字符串

边数据格式定义见表 288。

表 288　边数据格式定义

字段名	字段名（中文）	说明
src_id	起点编号	用于标识一条边的起点
dst_id	终点编号	用于标识一条边的终点
feature	边属性	用于描述边的属性信息

相较于点上的 feature，边上的 feature 字段在类型信息上有所区别，具体定义见表 289。

表 289　边数据 feature 字段定义

字段名	字段名（中文）	说明
src_type	起点类型名	表示起点类型，如果是同质图，可以为空
edge_type	边类型名	表示边类型，如果是同质图，可以为空
dst_type	终点类型名	表示终点类型，如果是同质图，可以为空
sparseKey	稀疏键	表示一组特征 ID，如：1，2，3
sparseKV	稀疏键值对	表示一组特征 ID-Value 对，如：1:0.1，2:0.2，4:0.01
dense	稠密数组	表示一个特征向量，如：0.1，0.2，0.002，0.4
raw	原始数据	表示一个字符串

点标签数据格式定义见表 290。

表 290　点标签数据格式定义

字段名	字段名（中文）	说明
id	起点编号	用于标识一条输入样本的点
label	标签	用于标识一条输入样本的类型

边标签数据格式定义见表 291。

表 291　边标签数据格式定义

字段名	字段名（中文）	说明
src_id	起点编号	用于标识一条样本边的起点
dst_id	终点编号	用于标识一条样本边的终点
label	标签	用于标识一条样本边的标签

9.2.2.2　批处理输出

批处理系统的输出可以描述为输入+子图，具体见表 292~表 294。
点样本数据格式定义见表 292。

表 292　点样本数据格式定义

字段名	字段名（中文）	说明
id	起点编号	用于标识一条输入样本的点
subgraph	子图	用于表示点 id 的子图，按照本标准第 5.2 节所示结构组织数据
label	标签	用于标识一条输入样本的标签，对于 inference 样本，可以为空

边样本数据格式定义见表 293。

表 293　边样本数据格式定义

字段名	字段名（中文）	说明
src_id	起点编号	用于标识一条样本边的起点
src_subgraph	起点子图	用于表示起点 src_id 的子图，按照本标准第 5.2 节所示结构组织数据
dst_id	终点编号	用于标识一条样本边的终点
dst_subgraph	终点子图	用于表示终点 dst_id 的子图，按照本标准第 5.2 节所示结构组织数据
label	标签	用于标识一条样本边的标签，对于 Inference 样本，可以为空

批处理接口的输入主要分为两部分，即输入图数据和采样参数，前者是加工对象，后者描述加工约束，详细定义见表 294。

表 294　批处理接口定义

运算操作	描述	字段	关键字	定义	数据类型
BatchProcess	批处理接口	Input	nodes	输入点数据	-
			edges	输入边数据	-
			labels	输入标签数据，可以是边标签，也可以是点标签	-
			processArgs	处理参数，如邻居跳数、每跳邻居数、出入边、采样策略等	-
		Output	success	边、点样本	-