《机器学习》课程项目二

1. 项目简介

图结构数据在现实世界中无处不在，如社交网络、生物信息学和学术引用网络。图机器学习，特别是图神经网络（GNNs），已成为从这些图数据中学习表征的强大工具。

本项目的目标是深入探讨节点属性预测（Node Property Prediction）这一图学习任务，你需要使用 Open Graph Benchmark (OGB) 提供的三个数据集，实现、训练和评估多种代表性的图学习方法。

项目要求在指定的数据集上，对比基线模型、图嵌入方法（Embedding）和图神经网络（GNN）的性能。此外，对于 `ogbn-arxiv` 数据集，由于其提供了每篇论文的原始文本，你可以自由探索大语言模型（LLM）的应用方法。

2. 项目要求

2.1. 数据集 (Datasets)

需要使用 OGB 提供的如下三个数据集进行实验：

- **ogbn-products**：这是一个亚马逊商品共同购买网络图。图中的节点代表亚马逊上销售的商品，边连接了被一起购买的商品，预测任务是进行一个多分类，预测每件商品所属的类别。
- **ogbn-proteins**：这是一个蛋白质-蛋白质相互作用（PPI）网络。图中的节点是蛋白质，边表示蛋白质之间不同类型的生物学关联，预测任务是进行一个多标签二分类，预测每个蛋白质节点存在哪些功能。
- **ogbn-arxiv**：这是一个 ArXiv 论文引用网络图。图中的节点是计算机科学（CS）领域的论文，有向边表示一篇论文引用了另一篇论文，预测任务是进行一个多分类，预测每篇论文所属的主题领域（例如 `cs.AI` 或 `cs.LG`）。

更多关于数据集的详细信息，请查阅 OGB 官方文档：https://ogb.stanford.edu/docs/nodeprop/

[注]：OGB 数据集不需要手动下载，请使用 **OGB** 库来加载数据，当通过代码第一次请求 `ogbn-products`、`ogbn-proteins` 或 `ogbn-arxiv` 数据集时，**OGB** 库会自动从官方源下载数据、进行预处理，并加载标准的数据划分（Train/Valid/Test Split）。

2.2. 模型实现

需要实现的机器学习模型包含以下四种类别：基线模型，图嵌入，GNN 和 LLM

对于基线模型、图嵌入和 GNN 类别的模型，需要从下方提供的选项中各选择一种方法，并在 `ogbn-products`、`ogbn-proteins` 和 `ogbn-arxiv` 所有三个数据集上都进行实验；对于 LLM 类别，仅需要在 `ogbn-arxiv` 数据集上进行实验。

[注]：每种方法对应的原始论文已统一放置在`5.1. 模型论文` 中，作为实现方法的参考；此外，为帮助评估计算资源和时间开销，`5.2. 参考实现` 中提供了两个模型（GNN 和 LLM）的具体配置和运行时间，以供参考。

(a). 基线模型

简介： 这类方法将每个节点视作独立的数据点，仅使用节点自身的特征作为输入，而完全忽略图的连接关系，最后直接输出节点标签。

选项： MLP、Logistic Regression、SVM、XGBoost

(b). 图嵌入方法

简介： 图嵌入方法的核心是为图中的每个节点学习一个低维向量，学习的准则是若两个节点在原始图中相似，那么它们在嵌入空间中的向量也应该相似。相比起基线模型，这类方法通常会利用节点之间的连接关系，即图的拓扑结构。

选项： Node2vec、Label Propagation (LP)、LINE (Large-scale Information Network Embedding)

(c). 图神经网络

简介： GNN 同时利用节点特征和图的连接关系，其核心思想是消息传递 (Message Passing)，每个节点将其自身的特征向量，与通过边关系从其邻居节点聚合而来的信息相结合，以此来更新自己的特征表示。

选项： GCN (Graph Convolutional Network)、GraphSage、AGDN (Attentive Graph Disentanglement Network)

(d). 大语言模型

简介： 使用任何一种 LLM 进行多分类任务，仅要求在 `ogbn-arxiv` 数据集上进行实验，因为其提供了每篇论文的原始文本（标题和摘要）。

要求： 此部分不限制具体方法，允许自由发挥，旨在鼓励探索 LLM 处理相关任务的潜力，你可以使用任意 LLM 的 API 或开源模型，以任何方式完成实验。

一些可供选择的方法：

- **LLM 作为独立分类器：**设计有效的提示词（Prompt），例如 Zero-shot 或 Few-shot 分类，将论文的标题和摘要作为输入，直接让 LLM 输出预测结果。也可以采用思维链（CoT）技术，一步步引导 LLM 输出最后的分类结果
- **LLM 作为特征提取器：**利用预训练的开源 LLM 将文本转换为高质量的 Embedding 向量，将该向量作为新的节点特征，输入到的 MLP 或 GNN 模型中，进行训练和预测。

鼓励提出并尝试任何其他合理的方法，方法和形式不限，只需要在预测任务中使用 LLM 即可。

3. 提交内容

请提交一个 `.zip` 压缩包，包含以下内容：

1. **实验报告：**一份 `PDF` 格式的完整报告。
2. **源代码：**所有实现、训练和评估模型的代码。
    - 禁止直接复用他人的代码，所有核心模型逻辑必须自己实现。
    - 根目录应包含 `README.md` 文件，简单说明如何配置环境和复现结果。

4. 实验报告要求

报告是评估你的工作的核心依据，其中需要展现你的实验结果和相应的调参过程，以及对实验结果的分析，具体要求如下：

[注]：`5.3. 参考 Benchmark 论文`中给出了 3 篇 benchmark 论文，作为实验报告撰写的参考。

格式要求： 报告正文（不包括参考文献）不得超过 8 页，每超过一面（Page），总分扣除 5%

内容要求： 报告应包含至少如下五个部分：

一．引言 (Introduction)：简要介绍本项目的背景和目标

二．方法介绍 (Methods)：

- 陈述本次实验选择的具体模型，并简单介绍选择模型的基本工作原理。
- 清晰解释你所使用的 LLM 方法，包括具体使用的模型以及技术原理。

三．实验设置 (Experiments Setup)

- **实现框架：** 简要说明你所使用的主要库或者框架（如使用 PyG 来实现 GNN）。
- **超参数调优：** 需要清晰地展示你是如何通过确定模型的超参数的，具体形式不限，可以采用表格展示（列出表格展示不同参数组合及其对应性能）、搜索说明（简述所使用的系统性调参方法、如网格搜索）、过程叙述（以文字形式叙述调参思路和过程）中任意一种方法来呈现调参过程
- 对于调用 LLM 进行 prompt 的方法，则需要解释选择 or 调整 prompt 的理由

四．结果与分析 (Results and Analysis)

- **主结果表：** 需要包含一个清晰的表格或者直方图，展示所有模型在 3 个数据集上的测试集性能，使用 OGB 官方评估指标，LLM 仅需报告在 `ogbn-arxiv` 上的结果。（可以参考`5.3. 参考 Benchmark 论文`中参考文献 [1] 的图 6 或者表 9 & 表 10）
- **学习曲线：**对于选择的需要训练的模型，提供其相应的学习曲线。比如展示模型在训练过程中的 Loss 和 Accuracy 随 Epoch 变化的曲线图（可以参考`5.2. 参考实现`给出的学习曲线图），请至少在一个数据集上给出所有模型完整的学习曲线图。
- **实验分析：**请基于你们超参数调优、实验结果和学习曲线，进行分析和讨论。具体的分析角度不限，但需要体现对整个项目实验的思考。以下是一些供选择的角度
  i. **模型间的对比：** GNN/Embedding 相比 MLP 有提升吗，在哪个数据集上提升最明显，为什么？
  ii. **数据集间的对比：** 选择的不同方法在三个数据集上的表现一致吗（比如相互之间的性能差异对比），为什么会有这个现象？
  iii. **LLM 分析：** 使用的 LLM 方法在 `ogbn-arxiv` 上的表现如何，和 GNN/MLP 相比，你认为它的优势和局限性在哪里？
  iv. **异常结果分析：** 实验中是否出现了任何反常或者不符合预期的结果，你是如何分析和解释这些现象的？（可以参考`5.2. 参考实现` 中的异常分析）

五．总结 (Conclusion)：总结你的主要发现、遇到的挑战以及对图学习方法的思考。

5. 补充材料

5.1. 模型论文

以下供选择的模型对应的原始论文，作为在实现模型时的参考：

- **Node2vec:** Grover, A., & Leskovec, J. (2016). node2vec: Scalable feature learning for networks.
- **Label Propagation (LP):** Zhu, X., & Ghahramani, Z. (2002). Learning from labeled and unlabeled data with label propagation.
- **LINE:** Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., & Mei, Q. (2015). LINE: Large-scale information network embedding.
- **GCN:** Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks.
- **GraphSage:** Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Inductive representation learning on large graphs.
- **AGDN:** Sun, C., Hu, J., Gu, H., Chen, J., & Yang, M. (2020). Adaptive Graph Diffusion Networks.
- **LLM - Chain of Thought (CoT):** Wei, J., Wang, X., et al. (2022). Chain-of-Thought Prompting Elicits Reasoning in Large Language Models.

- **LLM + GNN:** Chen, Z., et al. (2024). Exploring the Potential of Large Language Models (LLMs) in Learning on Graphs.

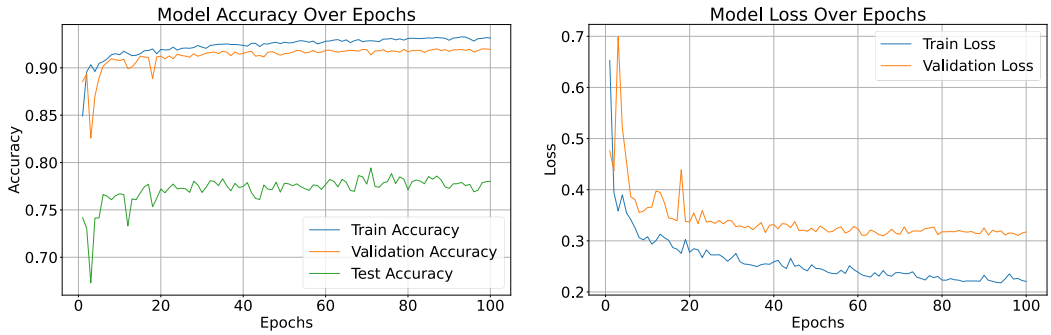
5.2. 参考实现

为帮助评估计算资源、时间开销和实现难度，这里提供了 GIN 和 LLM（部署开源模型 Qwen3-8b，直接 Zero-shot 提问让 LLM 进行分类） 两种方法的实现，完整代码请分别参阅项目附件中的 `GNN` 和 `LLM` 文件夹

[!!] 重要限制： 上述提供的这两种方法（GIN 和 Zero-shot LLM）仅作为性能参考。 你们在本次项目中，不能选择 GIN 作为 GNN 方法，也不能选择部署 LLM 直接进行 Zero-shot 分类作为你们的 LLM 方法（即直接复现给出的参考实现）。但欢迎使用其他方法，例如调用 LLM API（即便同样使用 Zero-shot 提示词），或者使用 Few-shot、CoT 等方法。

参考实现 1：GIN (Graph Isomorphism Network) on `ogbn-products`

- 硬件配置为 NVIDIA RTX 3090 显卡(24GB 显存)，环境配置为 torch == 2.1.2+cu121，torch-geometric == 2.7.0，ogb == 1.3.6
- 总耗时大约 7.1 小时，最终测试集准确率为 78.55%
- **学习曲线：**



- **异常分析：**模型在训练集和验证集上的准确率较高，但是测试集的准确率却低很多，官方文档说明 ogbn-products 是按照销量排名划分的，因此猜测数据分布偏移（Distribution Shift）现象严重，即验证集和训练集分布类似，但是测试集分布却截然不同。为了验证这一猜想，我们提取所有数据的特征，降维后观察训练集、验证集和测试集的分布，若模式差异较大，则说明我们的猜测成立。

参考实现 2：LLM (Zero-shot Direct Classification) on `ogbn-arxiv`

- 硬件配置为 NVIDIA RTX 3090 显卡(24GB 显存)，环境配置为 torch == 2.9.0+cu121，torch-geometric == 2.7.0，transformers == 4.57.1，accelerate == 1.11.0，ogb == 1.3.6
- 总耗时大约 2.6 小时，最终测试集准确率为 5.86%

5.3. 参考 Benchmark 论文

在撰写实验报告时，报告的分析、图表和表格可以参考以下论文的组织形式。

[注]：以下论文内容不一定与本项目直接相关，它们是作为如何撰写一份好的实验报告的范例

1. Zhenqian Shen, Mingyang Zhou, Yongqi Zhang, Quanming Yao. (2025). Benchmarking Drug-drug Interaction Prediction Methods: A perspective of distribution changes. Bioinformatics.
2. Hu, W., Fey, M., et al. (2020). Open Graph Benchmark: Datasets for Machine Learning on Graphs.
3. Dwivedi, V. P., Joshi, C. K., et al. (2020). Benchmarking graph neural networks.