Task-Conditioned Graph Embeddings for Enhanced Graph Reasoning via Large Language Models

Ethan Reinhart

May 7, 2025

1 Introduction

Graph computation tasks, particularly NP-complete and polynomial-time problems, pose significant challenges for current large language models (LLMs). Recent benchmarks, such as GraphArena, highlight limitations in using LLMs directly on structured graph data, particularly regarding scalability and optimality. This research proposes a novel approach that integrates Graph Neural Networks (GNNs) and LLMs using task-conditioned graph embeddings to effectively address these limitations.

2 Proposed Methodology

2.1 Overall Architecture

Our proposed architecture comprises four key stages:

1. **Task-conditioned Graph Neural Network (GNN)**
2. **Hierarchical Pooling**
3. **Linear Projection**
4. **Integration with a Large Language Model (LLM) via prefix-tuning**

This pipeline allows structured graph data to be effectively encoded and utilized by powerful language models.

2.2 Detailed Implementation

Task-Conditioned GNN: We employ a GraphSAGE or Graph Attention Network (GAT) backbone. Each node's embedding is conditioned explicitly on the task type (e.g., shortest path, traveling salesman problem) via a learned embedding concatenated to the node features. This allows the network to specialize without requiring multiple separate GNN models.

Hierarchical Pooling: To manage computational complexity and effectively summarize large graphs, we cluster nodes using METIS and apply mean-pooling within each cluster. The output is a fixed number of cluster embeddings suitable for LLM input.

Linear Projection: A linear projector maps the pooled GNN embeddings into the embedding space compatible with the LLM. This lightweight module ensures computational efficiency.

LLM Integration via Prefix Tuning: The graph embeddings are translated into soft prefix tokens injected directly into the input sequence of a pre-trained LLM. We use prefix tuning initially, with additional LoRA fine-tuning to optimize the LLM's interpretability of graph embeddings.

2.3 Training Strategy

We propose a two-phase training strategy:

- **Phase 1 (Pre-training)**: GNN pre-training using self-supervised contrastive learning tasks to ensure efficient downstream adaptation.
- **Phase 2 (Fine-tuning)**: Jointly fine-tune the GNN, linear projector, and prefix tokens with a frozen LLM to adapt effectively to task-specific graph reasoning. Optional LoRA fine-tuning is explored to enhance performance further.

3 Experiments and Evaluation

We will utilize benchmark datasets from GraphArena, supplemented by synthetic graphs generated via NetworkX. Performance will be evaluated on both NP-complete and polynomial-time tasks across four metrics:

1. Exact accuracy
2. Optimality gap
3. Hallucination rate
4. Runtime

We will compare our approach against baseline models, including pure prompt-based LLMs and standalone GNN-based solvers.

4 Expected Contributions

This research aims to:

- Introduce a novel methodology of task-conditioned embeddings for GNN-LLM integration.
- Demonstrate the effectiveness of hierarchical pooling in graph-to-text conversion.
- Evaluate and benchmark the approach comprehensively across multiple graph problem classes.

5 Computational Resources

Initial prototyping will leverage an RTX 4090 GPU (24 GB VRAM) utilizing 8-bit quantization. Should resource constraints arise during advanced experimentation (e.g., extensive fine-tuning), I may request cluster access.

6 Timeline

Timeframe	Task
Weeks 6–7	Dataset preparation, Implementation of task-conditioned GNN, and hierarchical pooling
Week 8	Integration with prefix-tuned LLM and initial evaluations
Week 9	Fine-tuning experiments and optimization
Week 10	Final evaluations, ablations, analysis and paper writing