Multi-View Graph Aggregation Latency Overhead in Code Generation Models

Assignee Research

June 1, 2026

Abstract

This report synthesises findings from 13 peer-reviewed papers addressing the following research question: What is the inference latency overhead of integrating multi-view graph aggregation versus single-view representations in code generation models. Graph Neural Networks (GNNs) have gained significant attention in recent years due to their ability to learn representations of graph-structured data. Two common methods for training GNNs are mini-batch training and full-graph training. 16 claims were extracted from source literature; 3 were independently verified against retrieved documents. An automated multi-reviewer quality assessment produced a score of 5.1/10. This report is a machine-generated literature synthesis and does not constitute original research.

1 Introduction

This paper examines: Graph Neural Network Training Systems: A Performance Comparison of Full-Graph and Mini-Batch. Research question: What is the inference latency overhead of integrating multi-view graph aggregation versus single-view representations in code generation models?.

2 Methodology

Systematic literature search across multiple databases yielded 13 papers. Claims were extracted from source material and verified against retrieved documents. An independent multi-reviewer assessment produced a quality score of 5.1/10.

3 Results

13 papers retrieved. 16 claims extracted; 3 independently verified. Quality review score: 5.1/10.

4 Limitations

This report is a machine-generated literature synthesis and does not constitute original research. Automated retrieval and verification may introduce errors or omissions. Review scores reflect automated assessment, not human peer review. Readers should consult primary sources for authoritative information.

5 Extracted Claims

Claim	Verified	Confidence
Mini-batch training systems are consistently faster than full-graph training counterparts in reaching the same target ac	✓	0.24
Mini-batch training typically requires fewer epochs to converge because it updates the model parameters multiple times p	×	0.11
Mini-batch training systems perform more work per epoch than full-graph training ones to run multiple iterations, but th	✓	0.24
With proper hyperparameter tuning, mini-batch training can reach a similar accuracy as, if not higher accuracy than, ful	✓	0.19
Hyperparameters that yield high accuracy for one training method do not work as well for the other.	×	0.09
Filtering information during graph aggregation is not necessarily detrimental to accuracy.	×	0.10
PipeGCN, BNS-GCN, and AdaQP are considered for full-graph training.	×	0.05
DGL, DistDGL, and Quiver are considered for mini-batch training.	×	0.10
Neighborhood Sampling, ClusterGCN, and GraphSaint are considered sampling algorithms for mini-batch training.	×	0.10
Time-to-accuracy is a better metric for comparison than epoch time.	×	0.12
The datasets considered include Pubmed, Arxiv, Reddit, Products, Orkut, and Papers100M.	×	0.01
The number of nodes in the datasets ranges from 19k to 111M.	×	0.03
The number of edges in the datasets ranges from 108k to 1.6B.	×	0.03
The number of classes in the datasets ranges from 3 to 172.	×	0.04
The number of features in the datasets ranges from 500 to 128.	×	0.02
The train/val/test splits for the datasets are provided.	×	0.02

References

- http://arxiv.org/abs/2203.07969v1
- http://arxiv.org/abs/2406.00552v4
- http://arxiv.org/abs/2302.02408v2