# Plexus v2: Extreme-scale GNN Training with Communication-free Sampling and 4D Hybrid Parallelism

**Cunyang Wei[1], Siddharth Singh[2], Aishwarya Sarkar[3], Daniel Nichols[4], Aditya K. Ranjan[5], Sayan Ghosh[6], Ali Jannesari[3], Nathan R. Tallent[6], Abhinav Bhatele[1]**

[1]University of Maryland [2]NVIDIA, Inc. [3]Iowa State University [4]Lawrence Livermore National Laboratory [5]Google, Inc. [6]Pacific Northwest National Laboratory

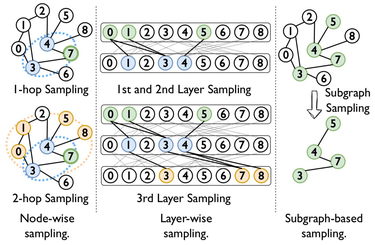
## Abstract

GNNs are widely used for learning on graph datasets derived from various real-world scenarios. Learning from extremely large graphs requires distributed training, and mini-batch training is a popular approach for parallelizing GNN training. Existing distributed mini-batch approaches have significant performance bottlenecks due to expensive sampling methods and limited scaling when using data parallelism. We present Plexus-v2, a four-dimensional hybrid parallel framework for scalable mini-batch GNN training that combines communication-free distributed sampling, three-dimensional parallel matrix multiplication, and data parallelism. Plexus-v2 introduces a uniform vertex sampling algorithm, enabling each process to construct its local shard of subgraph without any inter-process communication. Combining 3D PMM with data parallelism enables Plexus-v2 to scale to much larger GPU counts than vanilla data parallelism with significantly lower communication overheads. We evaluate Plexus-v2 on five graph datasets and scale to 2048 GPUs/GCDs. On Perlmutter, Plexus-v2 achieves 3.5x end-to-end training speedup over the best-performing state-of-the-art frameworks on ogbn-products.

## Background & Motivation

Mini-batch GNN training samples a small subgraph every step; existing sampling algorithms fall into three families:

- **Node-wise (GraphSAGE):** recursively samples a fixed number of neighbors per vertex and layer — neighborhood explosion and remote feature fetches
- **Layer-wise (FastGCN, LADIES)**: samples a vertex set per layer — avoids explosion but gives sparse inter-layer connectivity
- **Subgraph-based (Cluster-GCN, GraphSAINT)**: trains all layers on one sampled subgraph — needs normalization statistics derived from global graph structure




### Bottlenecks in most SOTA baselines

1) CPU-side sampling with neighbor & feature communication dominates epoch time

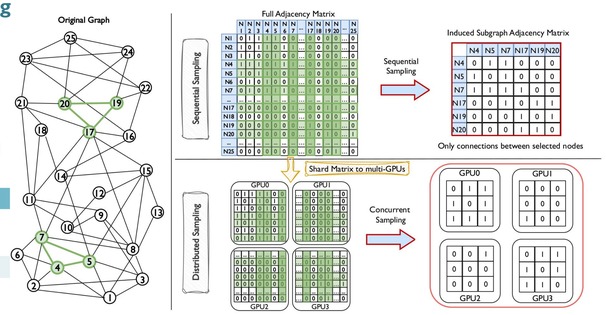
2) Vanilla data parallelism raises throughput, but not end-to-end training time

## Methodology

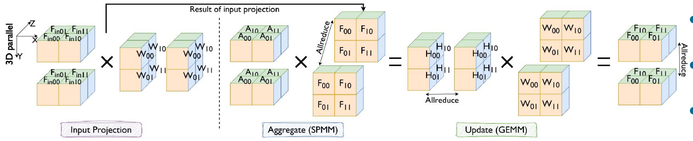
### Communication-Free Uniform Vertex Sampling

- Sample B vertices uniformly with a shared random seed — every GPU derives the identical sample independently
- Build the vertex-induced subgraph; rescale off-diagonal edge weights by 1/p with p = (B−1)/(N−1) → unbiased aggregation
- Zero sampling communication: each GPU extracts its local shard from its local CSR partition (binary search + vectorized gather)

| Test accuracy (%) | Reddit | ogbn-products |
|---|---|---|
| GraphSAINT (node) | 96.2 | 80.2 |
| GraphSAGE | 95.4 | 79.6 |
| Plexus-v2 (ours) | 96.3 | 81.3 |

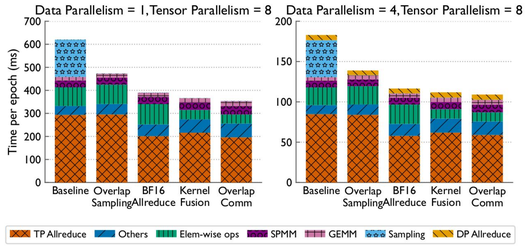



### 4D Hybrid Parallelism: Data Parallelism × 3D PMM




- GPUs form a Gd × Gx × Gy × Gz grid; each tensor-parallel (TP) group holds a full model copy
- 3D PMM parallelizes SpMM + GEMM with all-reduces along grid axes
- DP groups train independent mini-batches; only gradients are synchronized

## Performance Optimizations



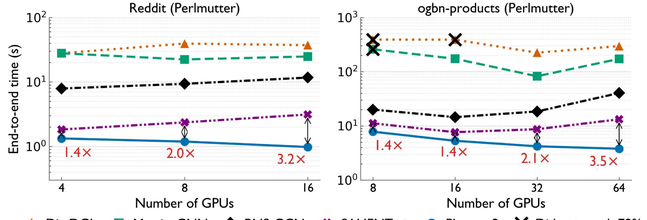

- **Overlap sampling with training** on a dedicated CUDA stream: −24% epoch time
- **BF16 collectives for 3D PMM**; numerically sensitive reductions stay FP32: −17%
- **Kernel fusion** of element-wise operators via torch.compile: −6%
- **Overlap communication with computation** in the backward pass: −3%
- Combined: **1.75×** faster epochs over the base implementation

*Read more*

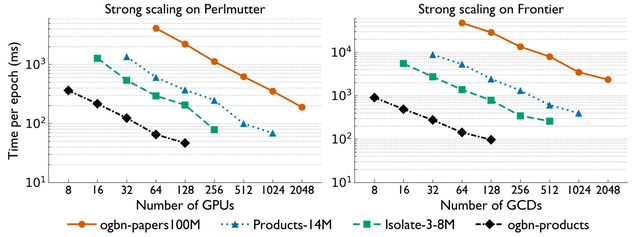
**Acknowledgements**

This material is based upon work supported in part by the National Science Foundation under Grant No.~2047120. This research used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S.~Department of Energy (DOE) Office of Science User Facility, operated under Contract No.~DE-AC02-05CH11231 using NERSC awards DDR-ERCAP0034262 and ALCC-ERCAP0034775, and that of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. DOE under Contract No. DE-AC05-00OR22725. This work was performed in part under the auspices of the U.S.~Department of Energy (DOE) by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 (LLNL-CONF-2017867).

## Results




- End-to-end training time to target accuracy on Perlmutter. On ogbn-products at 64 GPUs, Plexus-v2 is 3.5x faster than SALIENT++ and 78.2x faster than DistDGL.




- Strong scaling on ogbn-papers100M: 21.7× speedup from 64 → 2048 GPUs on Perlmutter (epoch time 4.1 s → 189 ms); 20.3× on Frontier (→ 2048 GCDs)

## Conclusion

- Plexus-v2 combines communication-free uniform vertex sampling, 3D PMM, and data parallelism for extreme-scale mini-batch GNN training
- Sampling preserves model quality: 81.3% on ogbn-products vs. 80.2% (GraphSAINT) and 79.6% (GraphSAGE)
- 3.5× end-to-end speedup over the best SOTA framework, with strong scaling on three leadership-class systems

**3.5×**
faster end-to-end than best SOTA

**2048**
GPUs / GCDs strong scaling

**References**

[1] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. ICLR 2017.
[2] W. L. Hamilton et al. Inductive representation learning on large graphs. NeurIPS 2017.
[3] H. Zeng et al. GraphSAINT: Graph sampling based inductive learning method. ICLR 2020.
[4] R. C. Agarwal et al. A three-dimensional approach to parallel matrix multiplication. IBM J. Res. Dev. 1995.
[5] A. K. Ranjan et al. Plexus: Taming billion-edge graphs with 3D parallel full-graph GNN training. SC 2025.
[6] D. Zheng et al. DistDGL: Distributed GNN training for billion-scale graphs. IA3 2020.