NeurIPS
2026
MAIN TRACK

NGDB-Zoo

Towards Efficient and Scalable Neural Graph Databases Training

Zhongwei Xie[1] · **Jiaxin Bai**[2] · **Shujie Liu**[3] · **Haoyu Huang**[1] · **Yufei Li**[1] · **Yisen Gao**[1] · **Hong Ting Tsang**[1] · **Yangqiu Song**[1,✉]

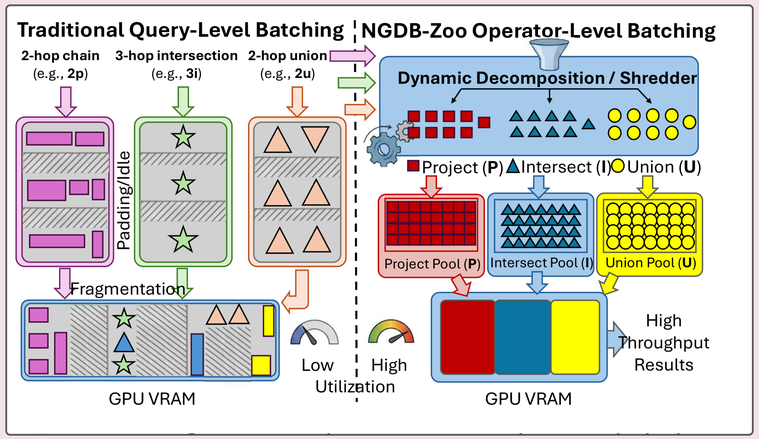
[1]HKUST · [2]Hong Kong Baptist University · [3]Microsoft Research Asia

TRAINING SYSTEM **NGDB-Zoo** executes logical operators across heterogeneous query DAGs using dynamic scheduling, while decoupling text-embedding computation from the training loop.

1.8× throughput vs SMORE	**6.8×** throughput vs KGReasoning
4–9× cached vs online encoding	**4M+** entities in ATLAS-Wiki

1 Why change the batching unit?

Traditional NGDB training groups queries by **identical topology**. Mixed workloads split into many small batches, fragmenting GPU work and wasting memory bandwidth.

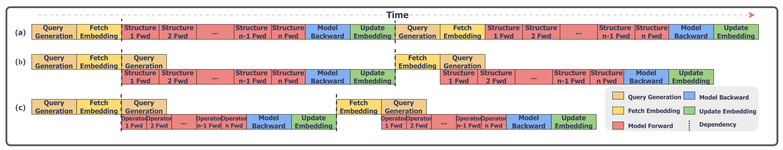


From queries to operators. Decompose heterogeneous DAGs, then pool Project / Intersect / Union operators across queries (paper Fig. 2).

Core idea: Batch like operators across heterogeneous query shapes.

2 Overlap work across training steps

Asynchronous query generation, embedding fetches, operator execution, and backward passes reduce idle time between successive batches.

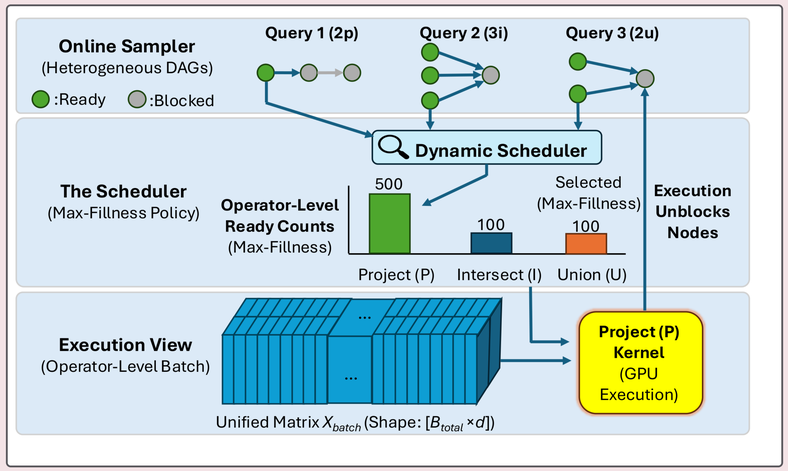


Execution pipeline. CPU preparation and GPU computation overlap while respecting dependencies (paper Fig. 4).

CUDA streams overlap independent operators.

3 Dynamic operator-level scheduler

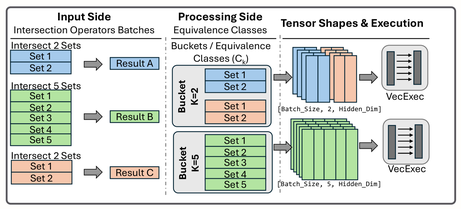
Every query is a dependency DAG. As operators become ready, NGDB-Zoo pools them by type and schedules the **most-filled** ready pool for GPU execution.



Max-Fillness policy. Dispatch the operator type with the highest ready-count / batch-capacity ratio, then unblock successors (paper Fig. 3).

Autograd-safe: Respect dependencies and gradients; reclaim intermediate tensors early.

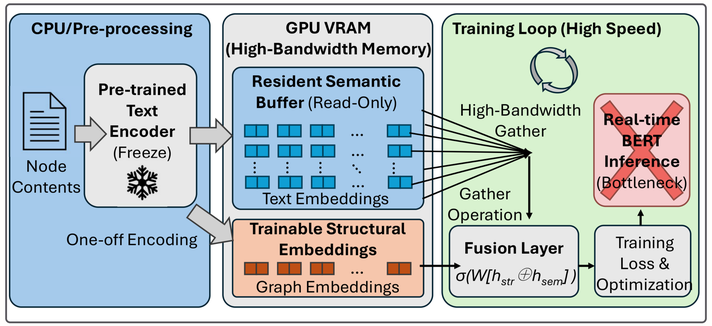
4 Vectorized set operators



Shape-aware batches. Group same-arity intersections for dense execution (paper Fig. 5).

5 Decoupled structural + semantic reasoning

Compute frozen text embeddings once, cache them, and fuse with trainable graph embeddings.



Offline: frozen encoding. **Online:** cached gather, fusion and optimization (paper Fig. 6).

Cache placement adapts to available GPU memory, avoiding repeated text-encoder inference during training.

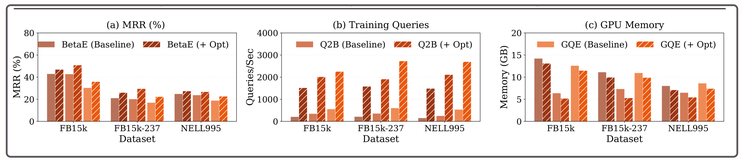
6 Measured training throughput

Average over BetaE, Q2B and GQE on FB15k, FB15k-237 and NELL995 (paper Table 4).

Training framework	Training queries/s ↑	GPU memory (GB) ↓	MRR (%) ↑
KGReasoning	748	1.71	25.67
SMORE	2,802	1.08	25.65
SQE	1,298	16.36	25.72
NGDB-Zoo	**5,086**	8.12	26.90

Key finding: NGDB-Zoo achieves the highest training throughput while maintaining competitive reasoning accuracy.

7 Semantic augmentation in practice



Qwen3-Embedding-0.6B: MRR, throughput and memory (Fig. 8).

Result: Cached access improves throughput and lowers peak GPU memory (Sec. 5.4).