GENTI: GPU-powered Walk-based Subgraph Extraction for Scalable Representation Learning on Dynamic Graphs

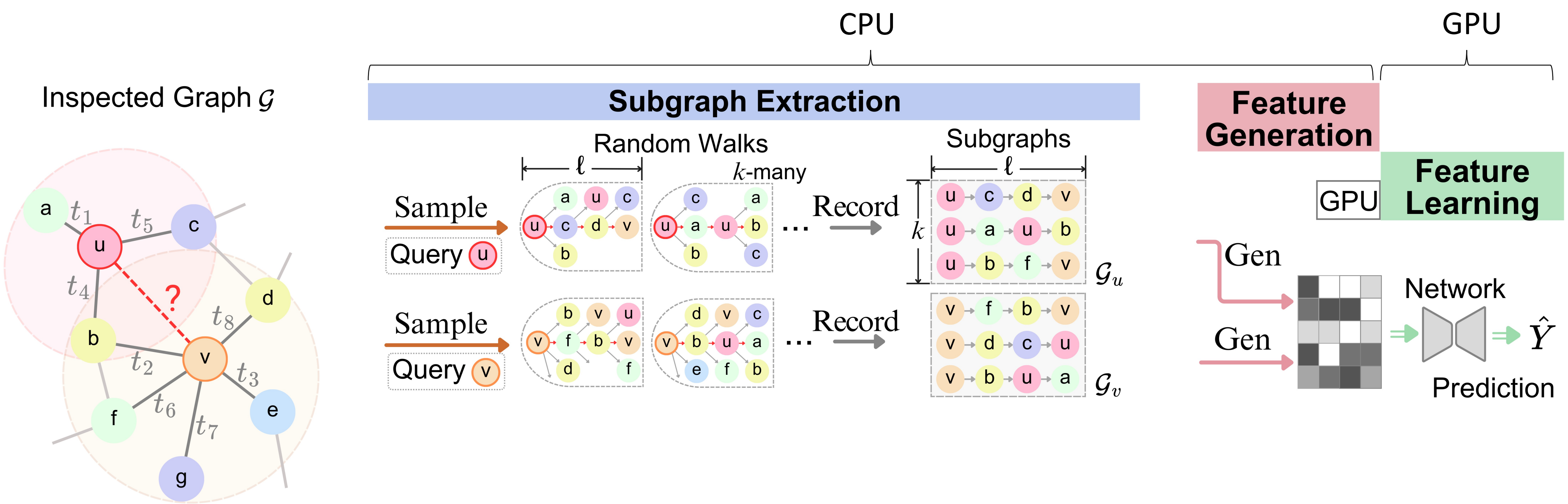
VLDB 2024

Zihao Yu, Ningyi Liao, Siqiang Luo

Presented by: Zihao Yu

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Subgraph-based Graph Learning: Pipeline



① Link prediction (u, v)

② Extract subgraphs for u and v by
i) Random Walk
ii) Metrics (e.g., Personalized PageRank)

③ Generate Feature
④ Feature Learning
i) Encoder (Attention, LSTM)
ii) Decoder (MLP)

Motivations

- Subgraph-based Graph Representation Learning (SGRL) methods achieve strong performance on many graph understanding tasks due to their ability to capture complex graph motifs.
- Subgraph extraction stage limits the **scalability** of SGRL.
  - CAW[1] uses 7h to finish a training epoch on Wiki-Talk (1M nodes, 7M edges).
  - During a single prediction process of CAW [1], the CPU computation accounts for **81%** of the time, while the GPU accounts for only **19%**, leading to imbalanced workloads and poor parallelism.
  - Moving the subgraph extraction stage to the GPU is challenging because large-scale graphs are hard to store on the GPU.

[1] Y. Wang, Y.-Y. Chang, Y. Liu, J. Leskovec, and P. Li, “Inductive representation learning in temporal networks via causal anonymous walks,” International Conference on Learning Representations, 2021.

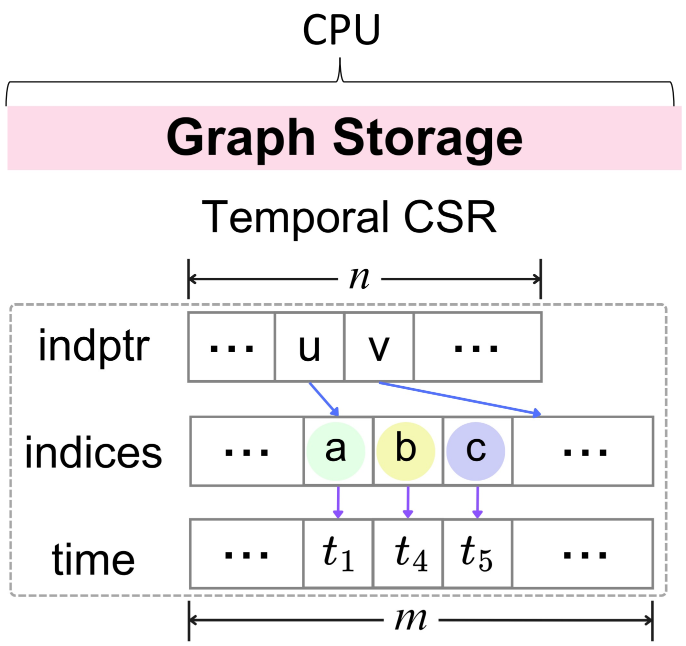
Highlights & Contributions

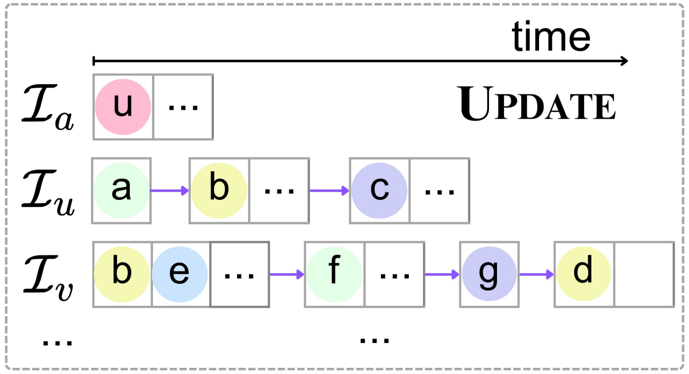
- **Algorithm**: We decouple the resource-intensive subgraph extraction stage to be separately conducted on CPUs and GPUs, which enables full GPU utilization and offers improved efficiency.
- **System**: We design the data structure for maintaining subgraphs on GPU with improved memory complexity and fast batch processing ability. We also enhance the graph storage to render it applicable for scalable and dynamic updating and sampling.
- **Performance Evaluation**: up to 30× faster subgraph extraction, and up to 26× acceleration in overall learning time.

GENTI: Graph Storage

- Vallina storage scheme: temporal compressed sparse row (CSR)
  - Indptr[u] points to where the neighbors of u are stored in the indices and time array.
  - CSR data structure is precomputed and static.

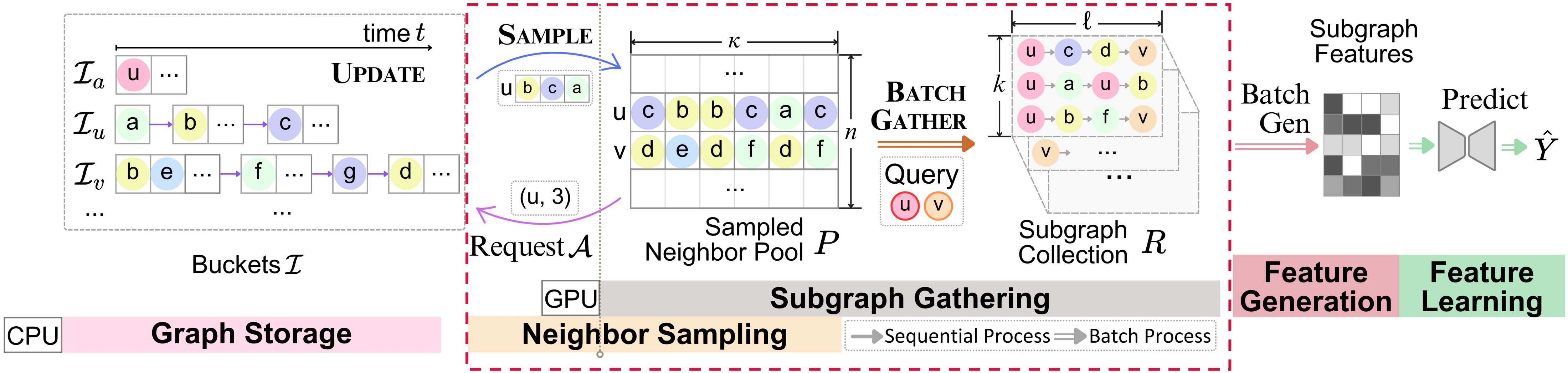
- Backet-based dynamic graph storage
  - Edge insertion and deletion: $O(1)$
  - Sample k neighbors of u: $O(k + log_{deg_u})$





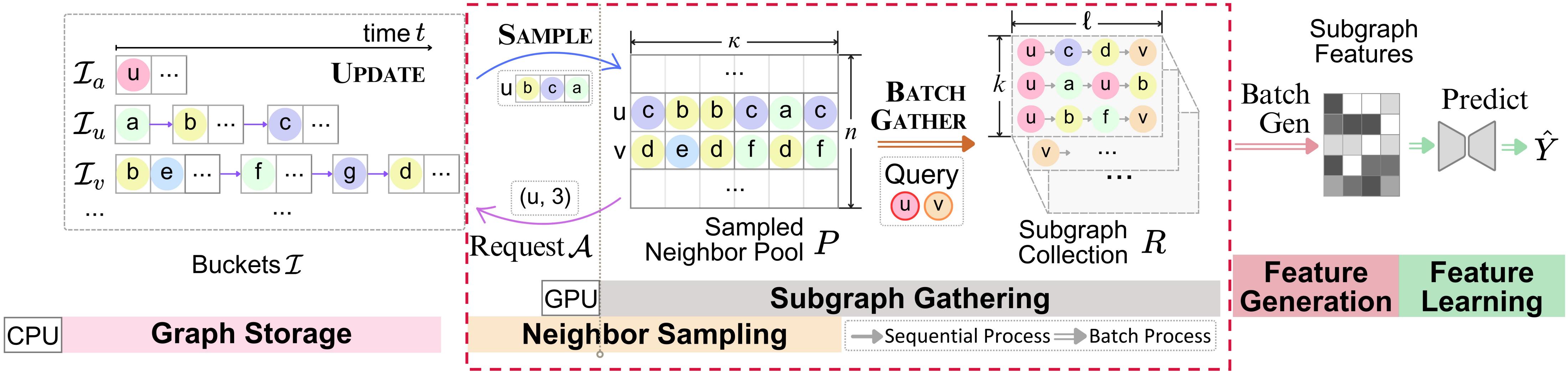
Buckets $\mathcal{I}$

GENTI: Subgraph Extraction



- Sampled Neighbor Pool (SNP) is an $N \times \kappa$ ($\kappa = O(k\sqrt{l})$) matrix stored on GPU, which is the key design to decouple the subgraph extraction into two parallel stages.
- SNP is dynamically and parallelly updated by the Neighbor Sampling operation, and its items are frequently consumed by the subsequent Subgraph Gathering and Feature Learning stages.

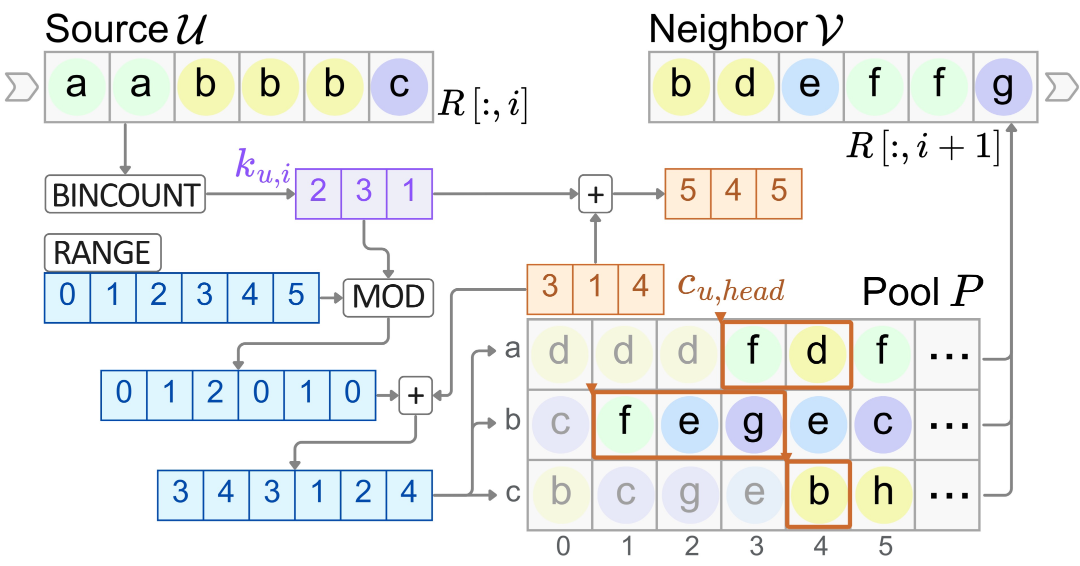
GENTI: Subgraph Extraction



- Neighbor Sampling: requests are sent from GPU to CPU to update the corresponding positions of the pool, which is the only host-device communication within the whole pipeline.
- Subgraph Gathering: k-many ℓ-length random walks can be formed by multiple GPU gather operators.

GENTI: BSGather Operation

- **BSGather** is a GPU operator that can be regarded as a batch of source nodes randomly moving to their neighbors.
- k ℓ-length random walks can be generated by calling BSGather ℓ times with k source nodes.
- Key steps of BSGather:
  - Calculate the index used to be gather items from SNP.
  - Update the pointers that maintain the SNP.
  - Record pool usage and generate requests to be sent to the CPU.



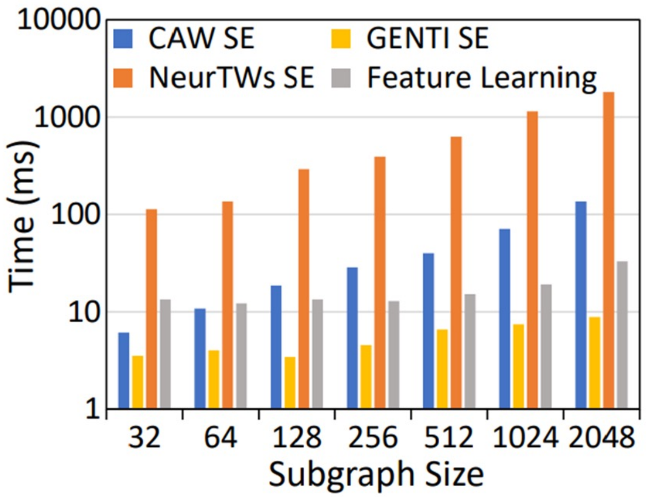
Experimental Evaluation

- **Time Efficiency**: 3-26× speedup in overall training time across all datasets compared to other SGRL methods.
- **Effectiveness**: similar or better average precision.

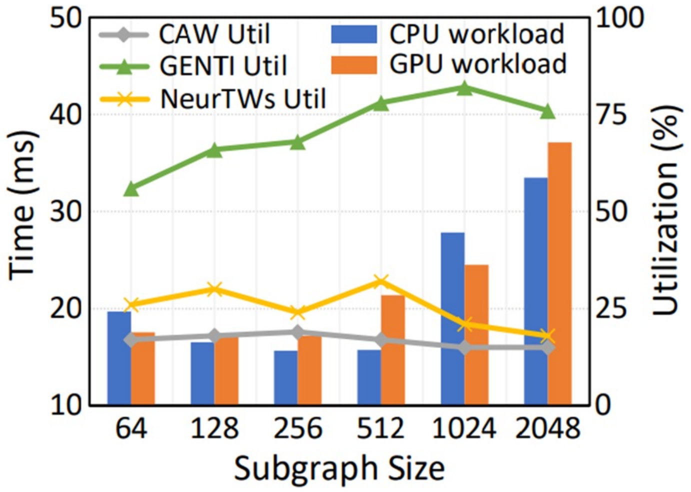
Model	UCI-MSG			Wikipedia			Reddit		
	Trans AP	Induct AP	Time (#)	Trans AP	Induct AP	Time (#)	Trans AP	Induct AP	Time (#)
JODIE	80.27 ± 0.1	71.64 ± 0.6	431(12)	95.16 ± 0.4	93.13 ± 0.5	1985(18)	95.83 ± 0.3	93.20 ± 0.4	8320.4(12)
TGAT	60.25 ± 0.3	75.27 ± 2.3	689(25)	94.26 ± 0.1	92.88 ± 0.3	2428(29)	97.80 ± 0.2	96.08 ± 0.3	11138(24)
TGN	78.91 ± 0.1	75.47 ± 0.1	507(21)	98.58 ± 0.1	98.05 ± 0.1	1839(26)	98.66 ± 0.1	97.55 ± 0.1	8152(26)
APAN	84.02 ± 0.3	83.14 ± 0.5	266(25)	96.41 ± 0.5	96.06 ± 0.4	1352(21)	98.50 ± 0.2	97.62 ± 0.7	7728(9)
Zebra	92.74 ± 0.2	91.16 ± 0.3	483(31)	98.63 ± 0.1	98.65 ± 0.1	1329(32)	98.73 ± 0.1	98.42 ± 0.1	6207(25)
D-DGNN	90.41 ± 0.1	89.72 ± 0.1	14467(30)	99.16 ± 0.3	98.54 ± 0.2	15173(30)	98.93 ± 0.2	98.56 ± 0.1	50342(30)
CAW	95.33 ± 0.3	95.19 ± 0.2	1488(8)	99.18 ± 0.1	99.34 ± 0.1	3720(5)	98.80 ± 0.1	98.99 ± 0.1	30912(8)
NeurTWs	95.46 ± 0.3	95.70 ± 0.2	44064(12)	99.17 ± 0.1	99.32 ± 0.1	65448(9)	98.32 ± 0.2	98.05 ± 0.1	TLE
GENTI	95.36 ± 0.3	95.82 ± 0.3	394(8)	99.18 ± 0.1	99.37 ± 0.1	739(8)	98.87 ± 0.1	99.18 ± 0.1	5890(8)

Experimental Evaluation

- Achieves up to 30× faster subgraph extraction with balanced and concurrent workloads, even as subgraph size increases.
- Average device utilization reaches 80% when extracting large subgraphs in batches.



(a) Subgraph Extraction (SE)



(b) Device Usage

Summary and Future Work

- **GPU-powered Training Pipeline**: our algorithm design allows graph learning on GPUs to proceed without delays caused by CPU stages.
- **System Design**: we enhance the data structure used for graph storage, enabling efficient support for frequent graph updates and sampling.
- **Performance Evaluation**: 3-26× speedup in overall learning time, up to 30× faster subgraph extraction.
- **Future Work**: expand GENTI to distributed setting, and further optimize the system's implementation.

THANK YOU

Acknowledgments
This research is supported by Singapore MOE AcRF Tier-2 funding (MOE-T2EP20122-0003) and NTU NAP startup grant (022029-00001). Ningyi Liao is supported by the Joint NTU-WeBank Research Centre on FinTech, Nanyang Technological University, Singapore.

NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE