VLDB 2025

WELCOME TO LONDON

Machine Learning for Graph Data Management and Query Processing

Contributors: Hanchen Wang, Ying Zhang and Wenjie Zhang

UTS UNIVERSITY OF TECHNOLOGY SYDNEY

ZHEJIANG GONGSHANG UNIVERSITY 1911 浙江工商大学

UNSW THE UNIVERSITY OF NEW SOUTH WALES

Contributors

Hanchen Wang is a Lecturer and ARC DECRA Fellow at Australian Artificial Intelligence Institute, University of Technology Sydney.

Ying Zhang is a Professor at the School of Computer Science and Technology, Zhejiang Gongshang University.

Wenjie Zhang is a full Professor and ARC Future Fellow in the School of Computer Science and Engineering, the University of New South Wales, Australia.

Acknowledgments

Runze Li is a PhD student at the School of Computer Science and Engineering, the University of New South Wales, Australia.

Qiuyu Guo is a PhD student in the School of Computer Science and Engineering at the University of New South Wales, Australia.

Wei Huang is currently a PhD student at the School of Computer Science and Engineering, the University of New South Wales, Australia. He received his bachelor's degree in Economics from the University of New South Wales.

Yifan Zhu is a Mphil student in the School of Computer Science and Engineering at the University of New South Wales, Australia.

VLDB 2025

WELCOME TO LONDON

UTS UNIVERSITY OF TECHNOLOGY SYDNEY

ZHEJIANG GONGSHANG UNIVERSITY

UNSW THE UNIVERSITY OF NEW SOUTH WALES

Machine Learning for Graph Data Management and Query Processing

Introduction

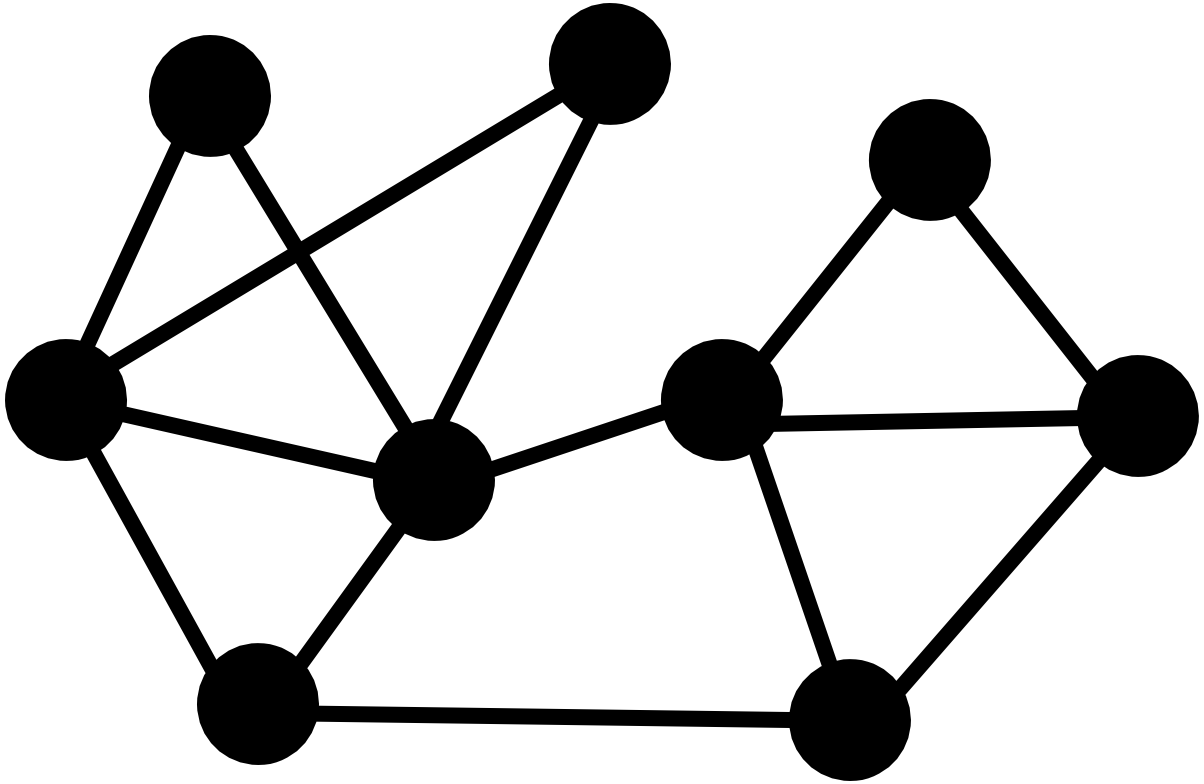
Speaker:
Hanchen Wang

Lecturer & ARC DECRA Fellow
Australian Artificial Intelligence Institute,
University of Technology Sydney

Contributors: Hanchen Wang, Ying Zhang and Wenjie Zhang

Graphs

Graphs are a general language for describing and analyzing **entities with relations/interactions**.



Many types of data are graphs

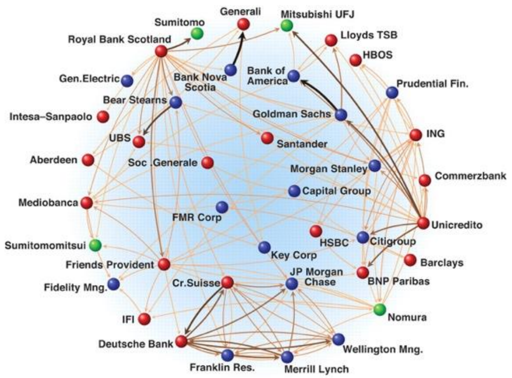
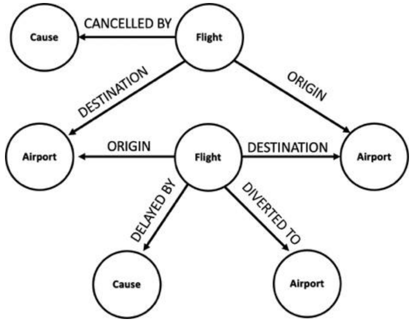


Image credit: Science

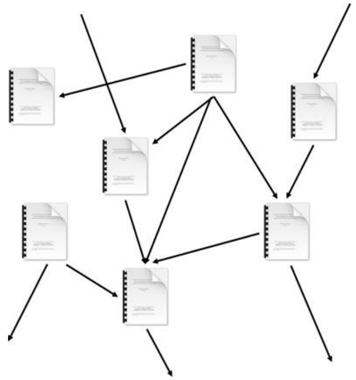
Economic Networks

Image credit: Lumen Learning

Communication Networks



Event Graphs



Citation Networks

Image credit: Missoula Current News

Internet

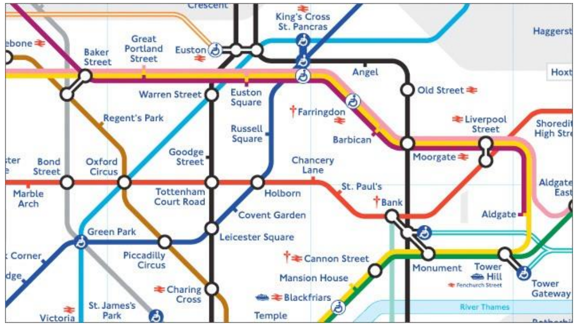


Image credit: visitlondon.com

Underground Networks

Many types of data are graphs

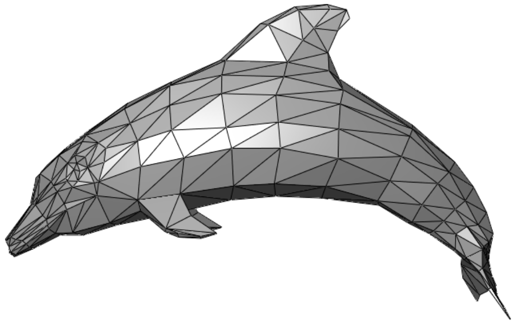
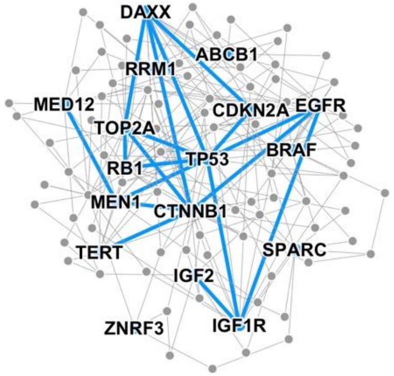


Image credit: Wikipedia

3D Shapes

Image credit: SalientNetworks

Computer Networks



Disease Pathways

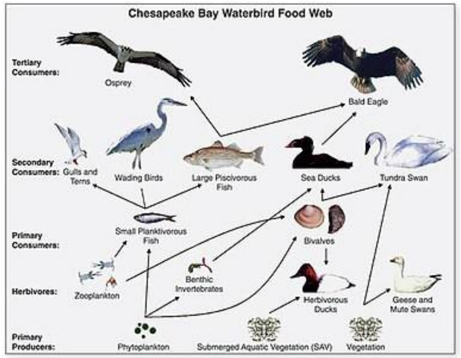


Image credit: Wikipedia

Food Webs

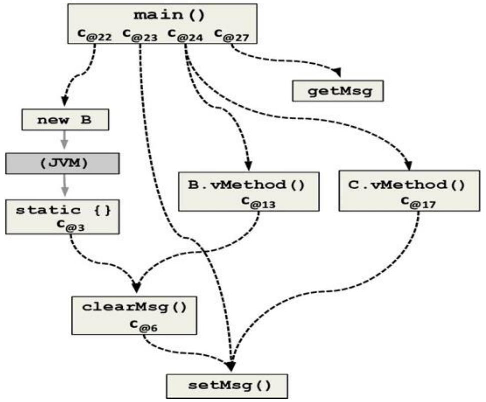


Image credit: ResearchGate

Code Graphs

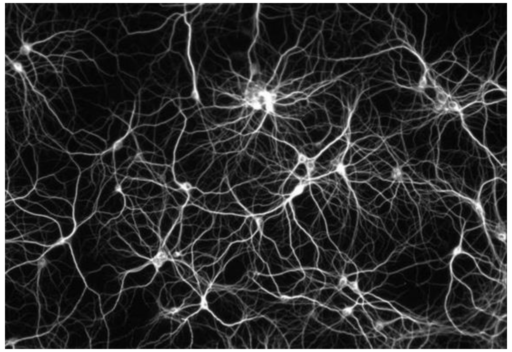


Image credit: The Conversation

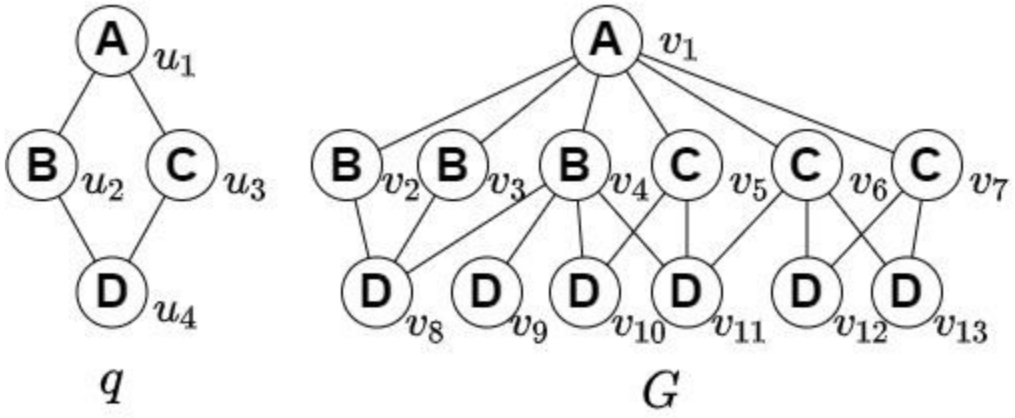
Networks of Neurons

Outline

- Graph Query Processing
  - Subgraph Isomorphism
  - Graph Similarity
  - Community Search
- Graph Data Management
  - Graph Data Quality Management
  - Graph Generation

Subgraph Isomorphism

- Query graph $q = (V, E, f_l)$
- Data graph $G = (V', E', f_l)$
- Subgraph Isomorphism: injective function f_{iso}: $V \rightarrow V'$:
  - $\forall u \in V, f_l(u) = f_l(f_{iso}(u))$
  - $\forall e(u, u') \in E, e(f_{iso}(u), f_{iso}(u')) \in E'$
- Determining the existence of subgraph isomorphism is NP-complete.

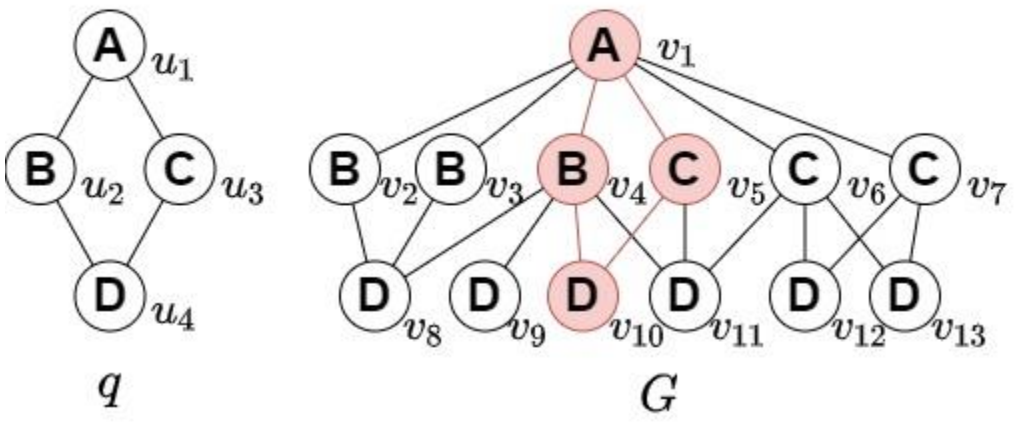


Wang, H., Hu, R., Zhang, Y., Qin, L., Wang, W., & Zhang, W. (2022, June). Neural subgraph counting with wasserstein estimator. In *Proceedings of the 2022 SIGMOD* (pp. 160-175).

Subgraph Isomorphism

Subgraph Counting

Subgraph Counting: Given a query graph q and a data graph G, the problem is to count the number of subgraphs in the data graph that match the query graph by subgraph isomorphism.



Subgraph isomorphisms

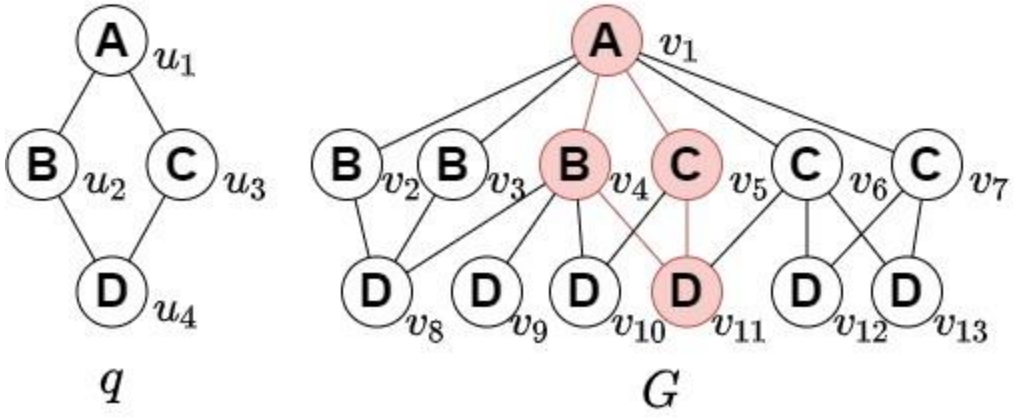
1. $(u_1, u_2, u_3, u_4) \rightarrow (v_1, v_4, v_5, v_{10})$

Wang, H., Hu, R., Zhang, Y., Qin, L., Wang, W., & Zhang, W. (2022, June). Neural subgraph counting with wasserstein estimator. In *Proceedings of the 2022 SIGMOD* (pp. 160-175).

Subgraph Isomorphism

Subgraph Counting

Subgraph Counting: Given a query graph q and a data graph G, the problem is to count the number of subgraphs in the data graph that match the query graph by subgraph isomorphism.



Subgraph isomorphisms

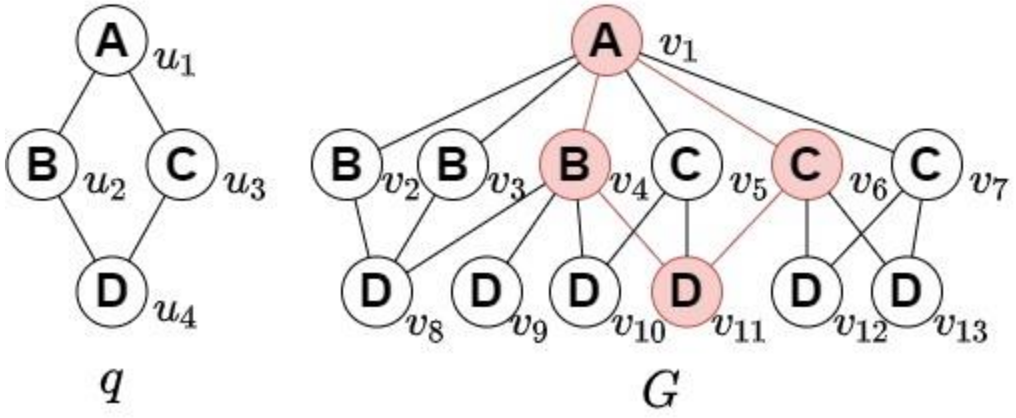
1. $(u_1, u_2, u_3, u_4) \rightarrow (v_1, v_4, v_5, v_{10})$
2. $(u_1, u_2, u_3, u_4) \rightarrow (v_1, v_4, v_5, v_{11})$

Wang, H., Hu, R., Zhang, Y., Qin, L., Wang, W., & Zhang, W. (2022, June). Neural subgraph counting with wasserstein estimator. In *Proceedings of the 2022 SIGMOD* (pp. 160-175).

Subgraph Isomorphism

Subgraph Counting

Subgraph Counting: Given a query graph q and a data graph G, the problem is to count the number of subgraphs in the data graph that match the query graph by subgraph isomorphism.



Subgraph isomorphisms

1. $(u_1, u_2, u_3, u_4) \rightarrow (v_1, v_4, v_5, v_{10})$
2. $(u_1, u_2, u_3, u_4) \rightarrow (v_1, v_4, v_5, v_{11})$
3. $(u_1, u_2, u_3, u_4) \rightarrow (v_1, v_4, v_6, v_{11})$

Wang, H., Hu, R., Zhang, Y., Qin, L., Wang, W., & Zhang, W. (2022, June). Neural subgraph counting with wasserstein estimator. In *Proceedings of the 2022 SIGMOD* (pp. 160-175).

Graph Similarity

Graph Edit Distance

Let’s start with a fundamental graph similarity metric: Graph Edit Distance.

Graph Edit Distance aims to determine the minimum number of edit operations required to transform one graph into another, and the sequence of edit operations is called a graph edit path.

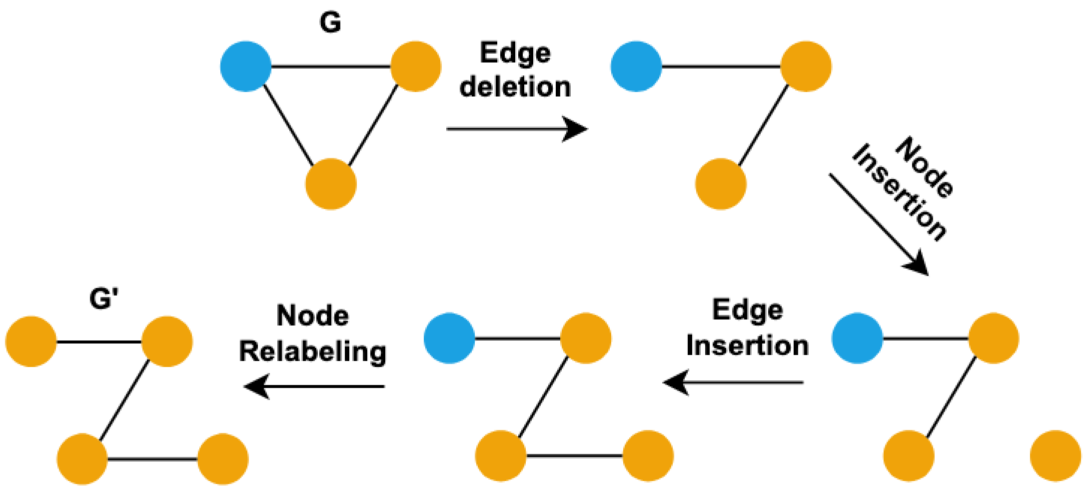
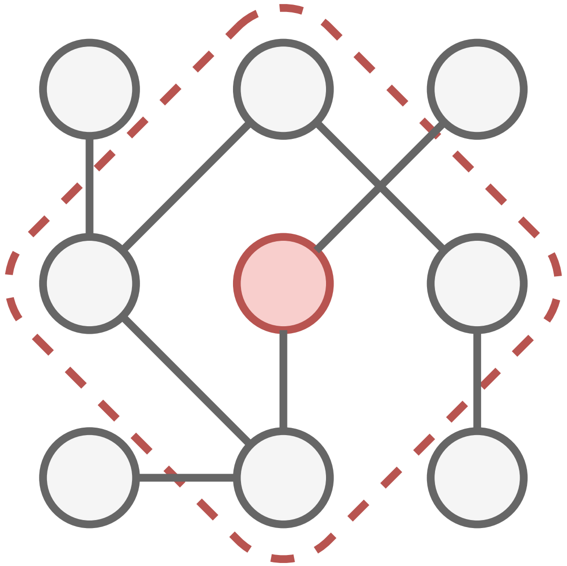


Figure 1: An optimal edit path for transforming G to G'. $\mathrm{GED}(G, G') = 4$.

Huang, Wei, et al. "DiffGED: Computing Graph Edit Distance via Diffusion-based Graph Matching." *arXiv preprint arXiv:2503.18245* (2025).

Community Search

- Definition: **Community search** (CS) is defined as the task of finding a cohesive subgraph that contains a given set of query nodes, emphasizing query-driven discovery of structurally and attributably close and well-connected communities within a larger graph.
- A query set contains one or more nodes that belong to the same community.
- We have disjoint community search and overlapping community search, depending on whether a node can only belong to one community.

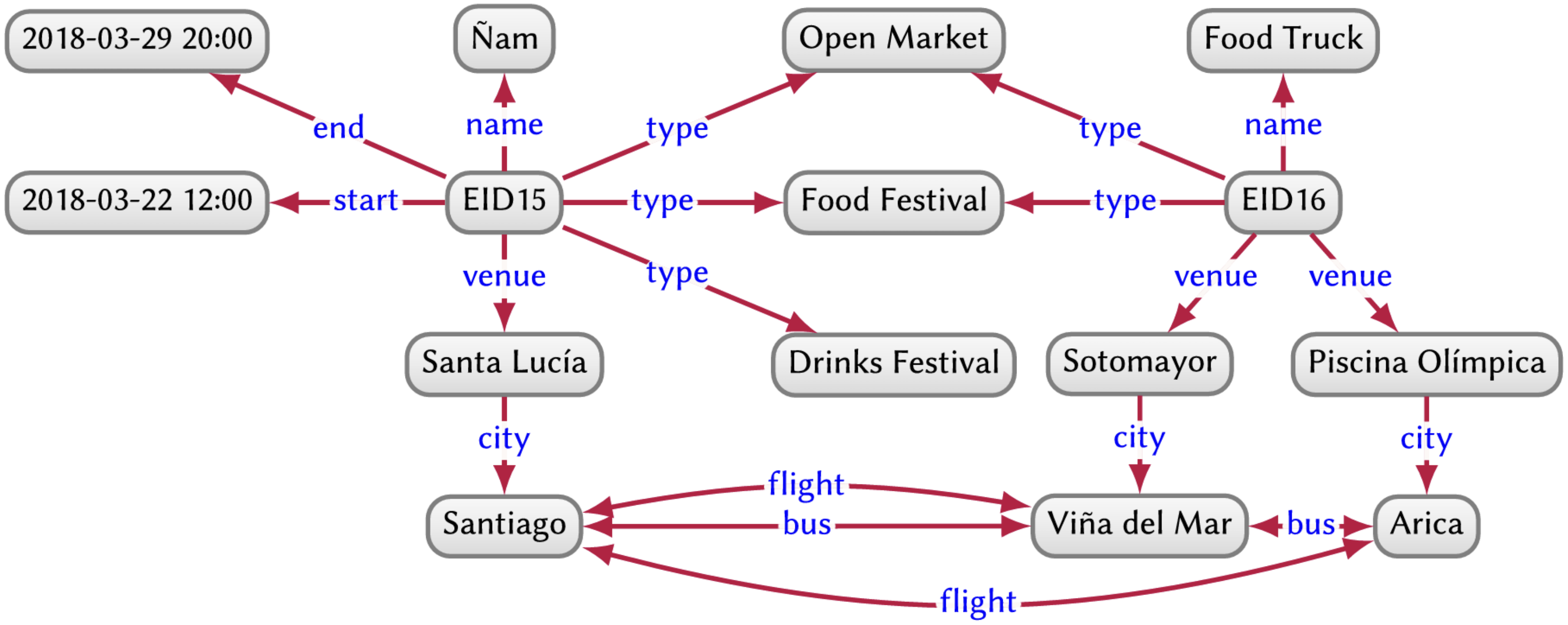


Community Search

Knowledge Graph

Definition of Knowledge Graph

Knowledge Graph is defined as a graph of data intended to accumulate and convey knowledge of the real world, whose nodes represent entities or concepts and whose edges represent relations between them, typically accompanied by ontologies and schemas.



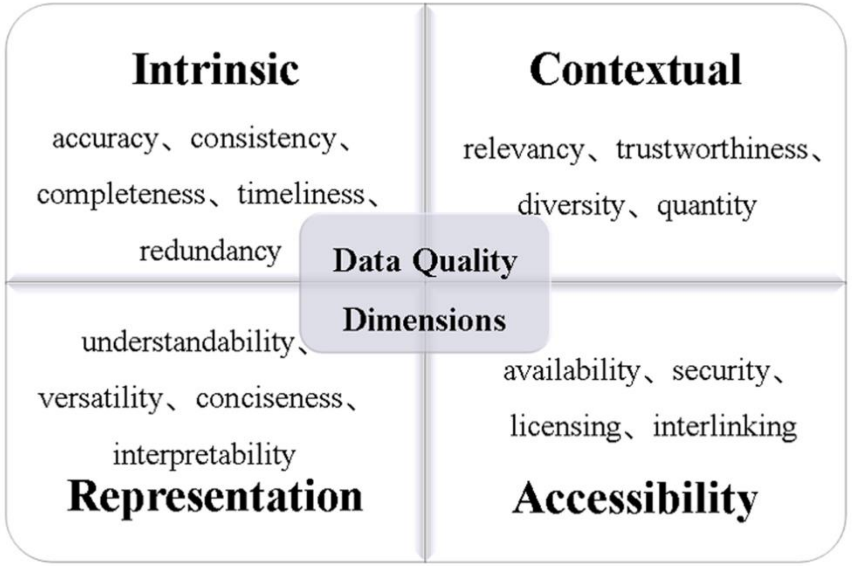
Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., Melo, G. D., Gutierrez, C., ... & Zimmermann, A. (2021). Knowledge graphs. *ACM Computing Surveys (Csur)*, *54*(4), 1-37.

Graph Quality Management

As a specfic data type, researches on knowledge graph are in the same line with general data type.

Definition

The extent to which data are **fit for a specified use** and **free of defects** with respect to explicit, context-specific criteria.



Dimension

The extent to which data are **fit for a specified use** and **free of defects** with respect to explicit, context-specific criteria.

Lifecycle

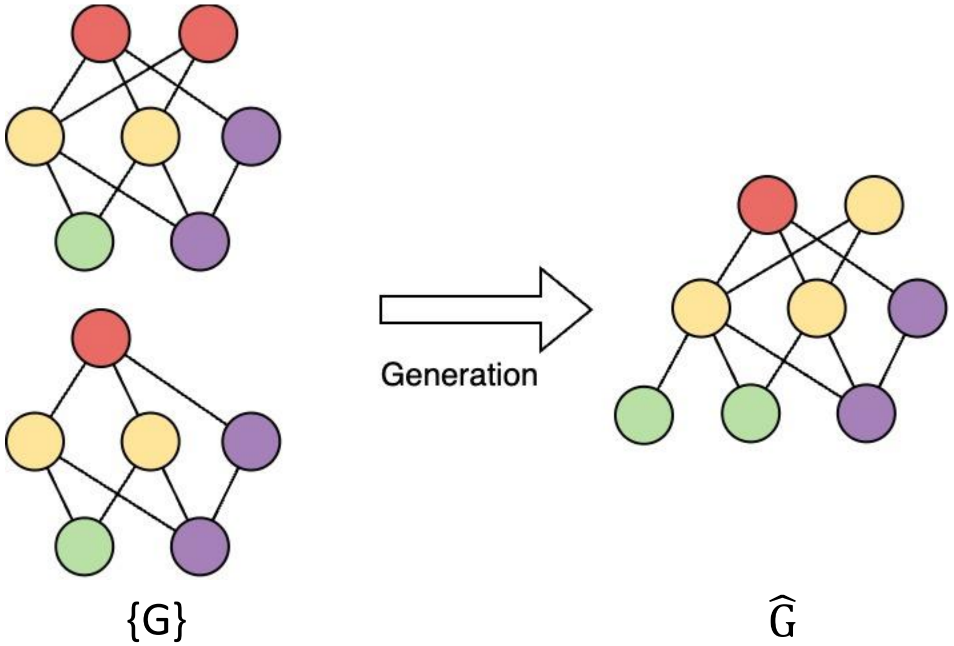
a data lifecycle pipeline contains five steps, namely, data generation, information extraction, data integration, analysis, and application.

Xue, B., & Zou, L. (2022). Knowledge graph quality management: A comprehensive survey. *IEEE Transactions on Knowledge and Data Engineering*, *35*(5), 4969-4988.

Graph Data Generation

Definition of Graph Generation

Given a set of observed graphs {G}, **graph generation** aims to construct a generative model $p_\theta(G)$ to capture the distribution of these graphs, from which new graphs can be sampled $\widehat{G} \sim p_\theta(G)$. The generation process can be conditioned on additional information s, i.e., conditional graph generation $\widehat{G} \sim p_\theta(G|s)$ to apply specific constraints on the graph generation results.



Xiang, S., Wen, D., Cheng, D., Zhang, Y., Qin, L., Qian, Z. and Lin, X., 2022. General graph generators: experiments, analyses, and improvements. The VLDB Journal, pp.1-29.

Machine Learning for Graph Data Management and Query Processing

Graph Query Processing

Speaker:
Hanchen Wang

Lecturer & ARC DECRA Fellow
Australian Artificial Intelligence Institute,
University of Technology Sydney

Contributors: Hanchen Wang, Ying Zhang and Wenjie Zhang

Graph Query Processing

- Subgraph Isomorphism
  - **Subgraph Matching**
  - Subgraph Counting
- Graph Similarity
  - Graph Edit Distance
- Community Search
  - Disjoint Community Search
  - Overlapping Community Search

Subgraph Matching

Definition

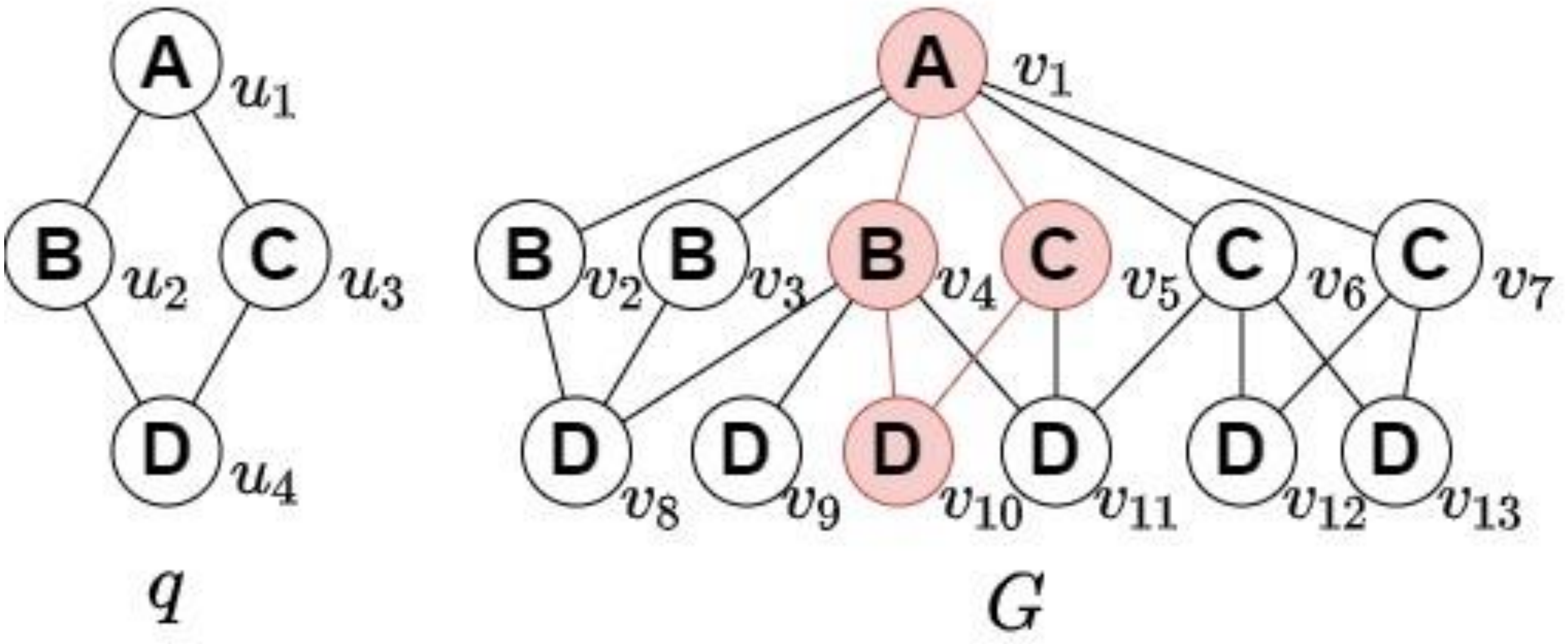
- The objective of the *subgraph matching* is searching for all *subgraph isomorphisms* from query graph q to data graph G

Definition II.1 (Subgraph Isomorphism). Given a query graph $q = (V, E)$ and a data graph $G = (V', E')$, a subgraph isomorphism is an injective function f_{iso} from V to V' such that (1) $\forall v \in V, f_l(v) = f_l(f_{iso}(v))$; and (2) $\forall e_{(u,v)} \in E, e_{(f_{iso}(u), f_{iso}(v))} \in E'$.

Wang, H., Zhang, Y., Qin, L., Wang, W., Zhang, W., & Lin, X. (2022, May). Reinforcement learning based query vertex ordering model for subgraph matching. In 2022 IEEE 38th International Conference on Data Engineering (ICDE) (pp. 245-258). IEEE.

Subgraph Matching

Definition



Wang, H., Zhang, Y., Qin, L., Wang, W., Zhang, W., & Lin, X. (2022, May). Reinforcement learning based query vertex ordering model for subgraph matching. In 2022 IEEE 38th International Conference on Data Engineering (ICDE) (pp. 245-258). IEEE.

Subgraph Matching: RLQVO

Existing Subgraph Matching Methods

The backtracking-based methods can be partitioned in three main phases:

1. The complete candidate vertex set generation.

2. Matching order generation.

3. Matching enumeration.

Wang, H., Zhang, Y., Qin, L., Wang, W., Zhang, W., & Lin, X. (2022, May). Reinforcement learning based query vertex ordering model for subgraph matching. In 2022 IEEE 38th International Conference on Data Engineering (ICDE) (pp. 245-258). IEEE.

Subgraph Matching: RLQVO

Limitations of Existing Order Generation Methods

The existing subgraph matching methods usually generate the matching order based on the heuristic values, here are some examples:

- Degree-based ordering
- Infrequent label first ordering
- Path-based ordering.

Wang, H., Zhang, Y., Qin, L., Wang, W., Zhang, W., & Lin, X. (2022, May). Reinforcement learning based query vertex ordering model for subgraph matching. In 2022 IEEE 38th International Conference on Data Engineering (ICDE) (pp. 245-258). IEEE.

Subgraph Matching: RLQVO

Limitations of Existing Order Generation Methods

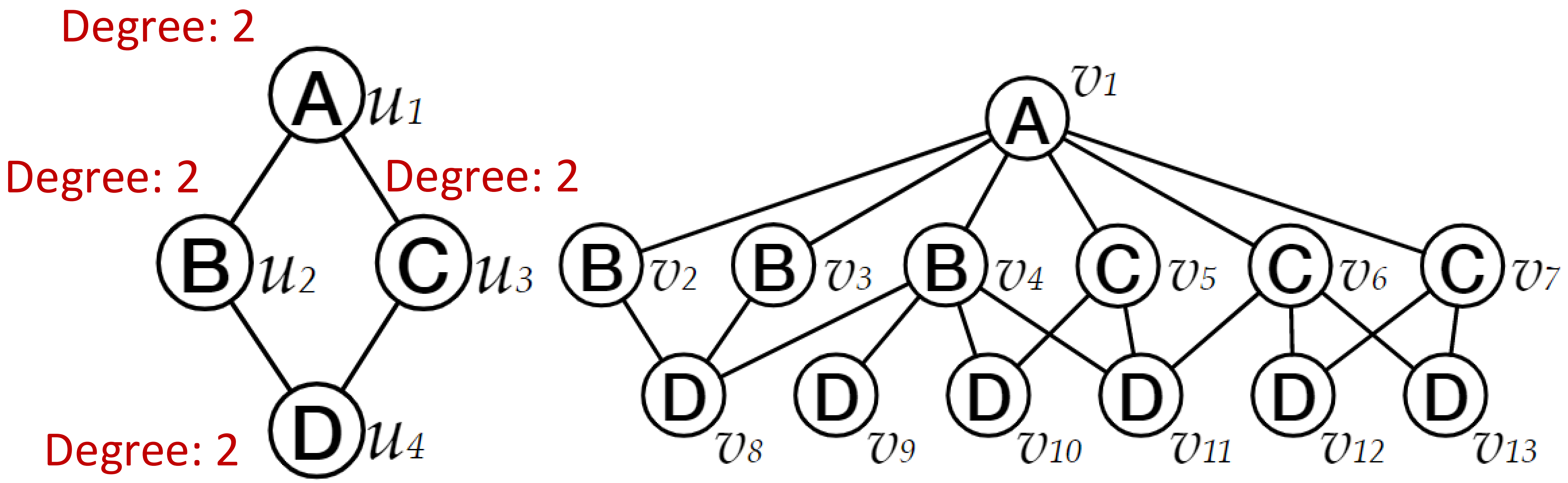
Two major limitations:

- Cannot fully use the graph information.
- Greedy heuristics can lead to local optimum.

Wang, H., Zhang, Y., Qin, L., Wang, W., Zhang, W., & Lin, X. (2022, May). Reinforcement learning based query vertex ordering model for subgraph matching. In 2022 IEEE 38th International Conference on Data Engineering (ICDE) (pp. 245-258). IEEE.

Subgraph Matching

If ordering based on degree (RI)



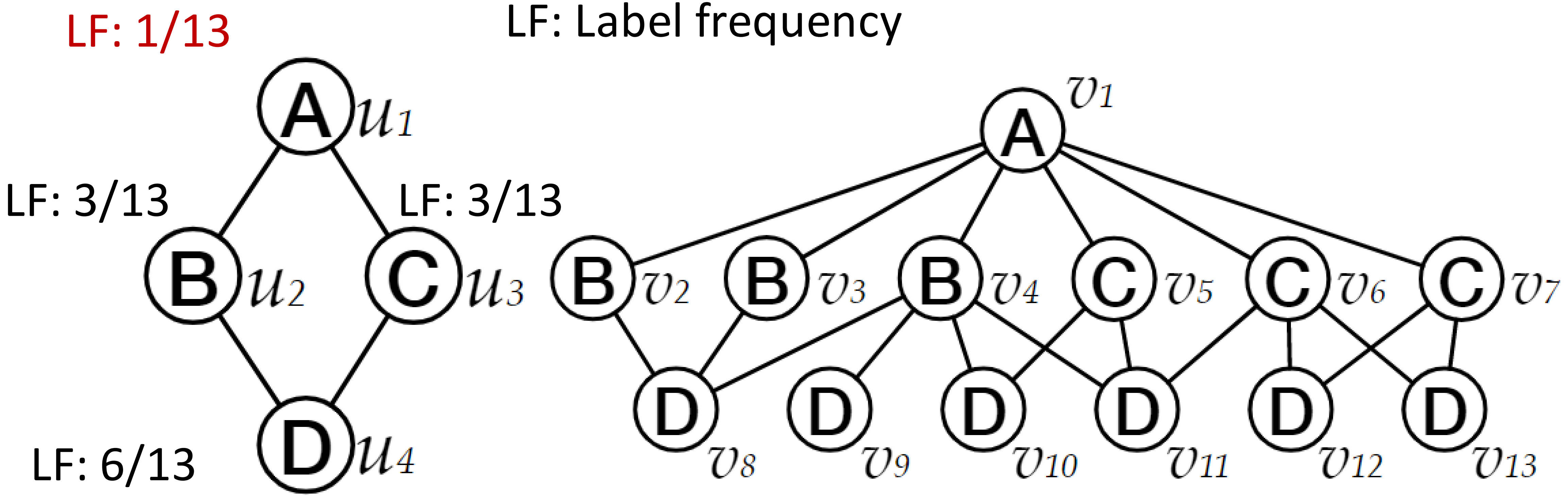
(a) Query Graph q (b) Data Graph G

Wang, H., Zhang, Y., Qin, L., Wang, W., Zhang, W., & Lin, X. (2022, May). Reinforcement learning based query vertex ordering model for subgraph matching. In 2022 IEEE 38th International Conference on Data Engineering (ICDE) (pp. 245-258). IEEE.

Subgraph Matching

If ordering based on label frequency

LF: Label frequency

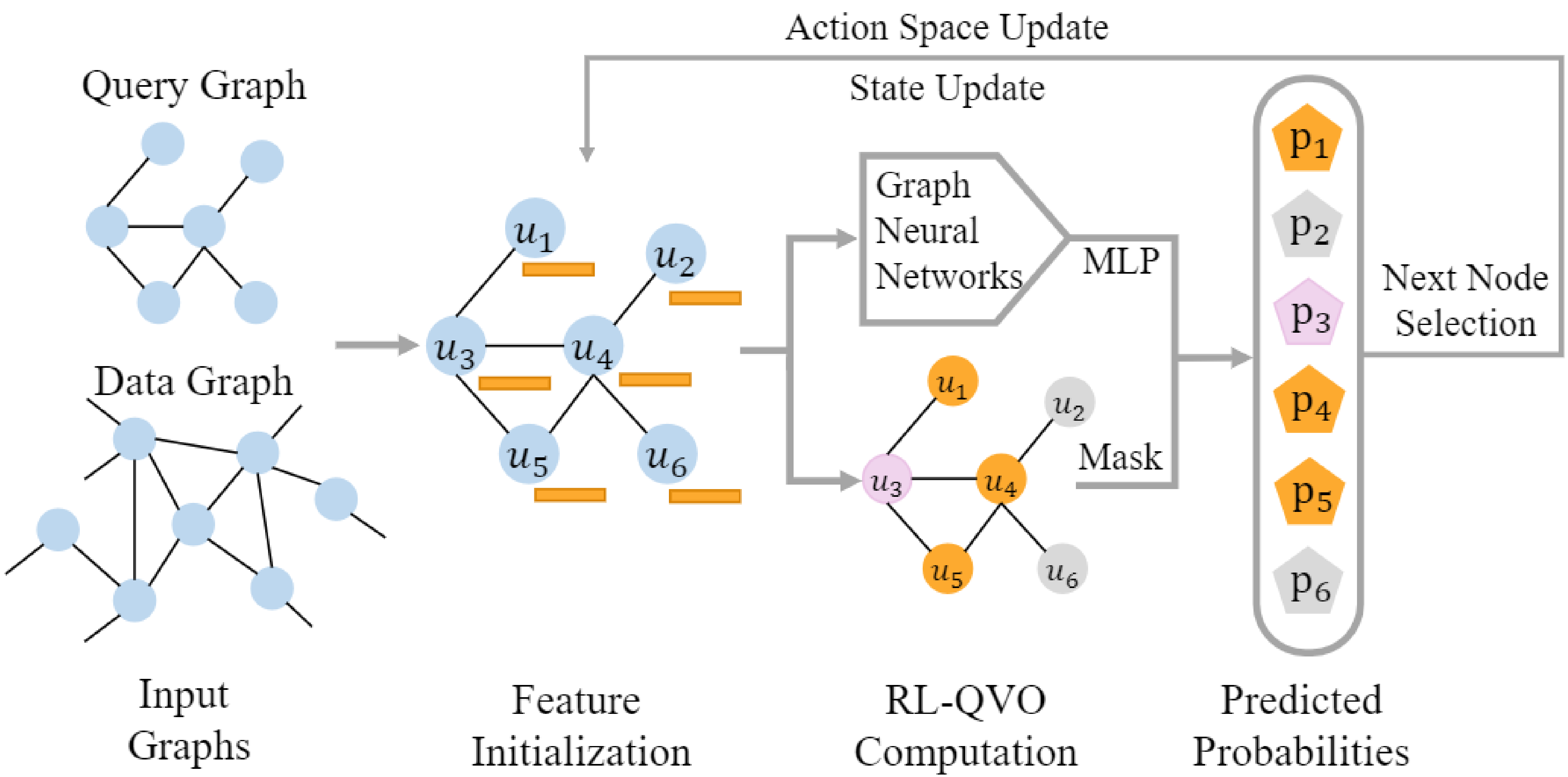


(a) Query Graph q (b) Data Graph G

Wang, H., Zhang, Y., Qin, L., Wang, W., Zhang, W., & Lin, X. (2022, May). Reinforcement learning based query vertex ordering model for subgraph matching. In 2022 IEEE 38th International Conference on Data Engineering (ICDE) (pp. 245-258). IEEE.

Subgraph Matching: RLQVO

Framework



Wang, H., Zhang, Y., Qin, L., Wang, W., Zhang, W., & Lin, X. (2022, May). Reinforcement learning based query vertex ordering model for subgraph matching. In 2022 IEEE 38th International Conference on Data Engineering (ICDE) (pp. 245-258). IEEE.

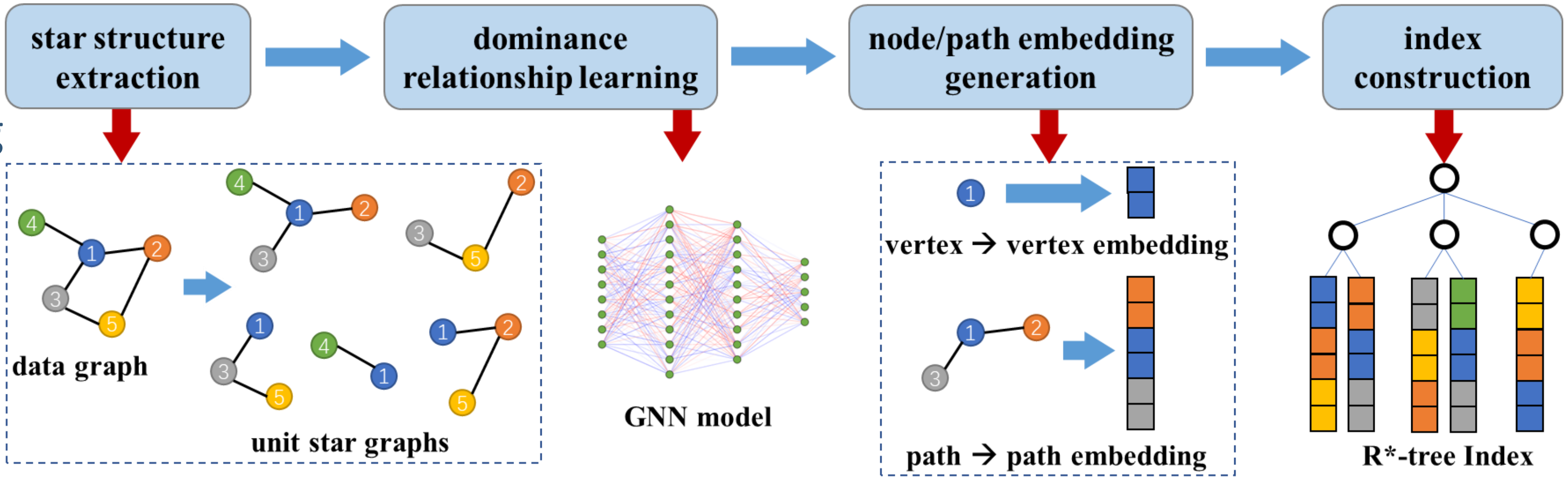
Subgraph Matching: GNN-PE

- Design **Graph Neural Network (GNN)-based embeddings for graph vertices** which enable the **subgraph matching with 100% accuracy**
  - Prior works usually trained and used GNN on distinct training and testing graph datasets
  - To enable the trained GNN to be over the same training/testing graph data set, we explore basic units of the data graph (i.e., unit star subgraphs) with a finer resolution
- Transform the subgraph matching over graphs to the **dominance search problem in the vector space**
  - Train the GNN model to learn the dominance relationship between unit star subgraphs
  - Generate node and path dominance embeddings by the trained GNN
- GNN-based path embedding (GNN-PE) framework for **efficient subgraph matching algorithm**
  - Cost-model-based query plan generation
  - Graph partitioning, pruning strategies, index construction over path embeddings, and multi-way hash join-based refinement

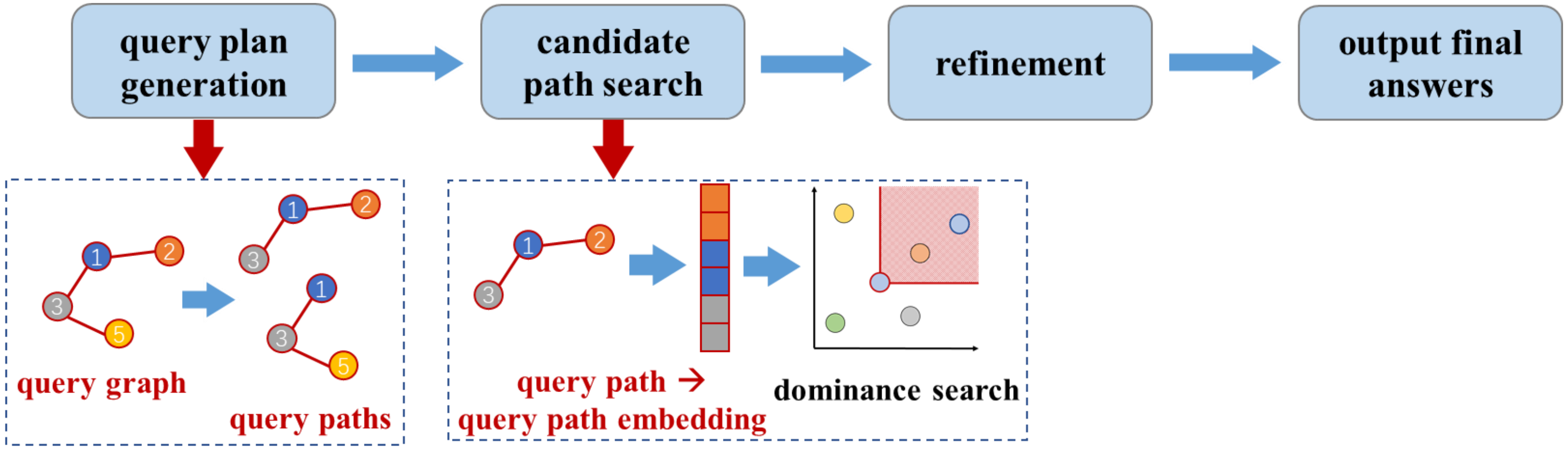
Ye, Y., Lian, X., & Chen, M. (2024). Efficient exact subgraph matching via gnn-based path dominance embedding. Proceedings of the VLDB Endowment, 17(7), 1628-1641.

Subgraph Matching: GNN-PE

- Offline pre-computation
  - Dominance relationship learning
  - Index construction over path dominance embeddings



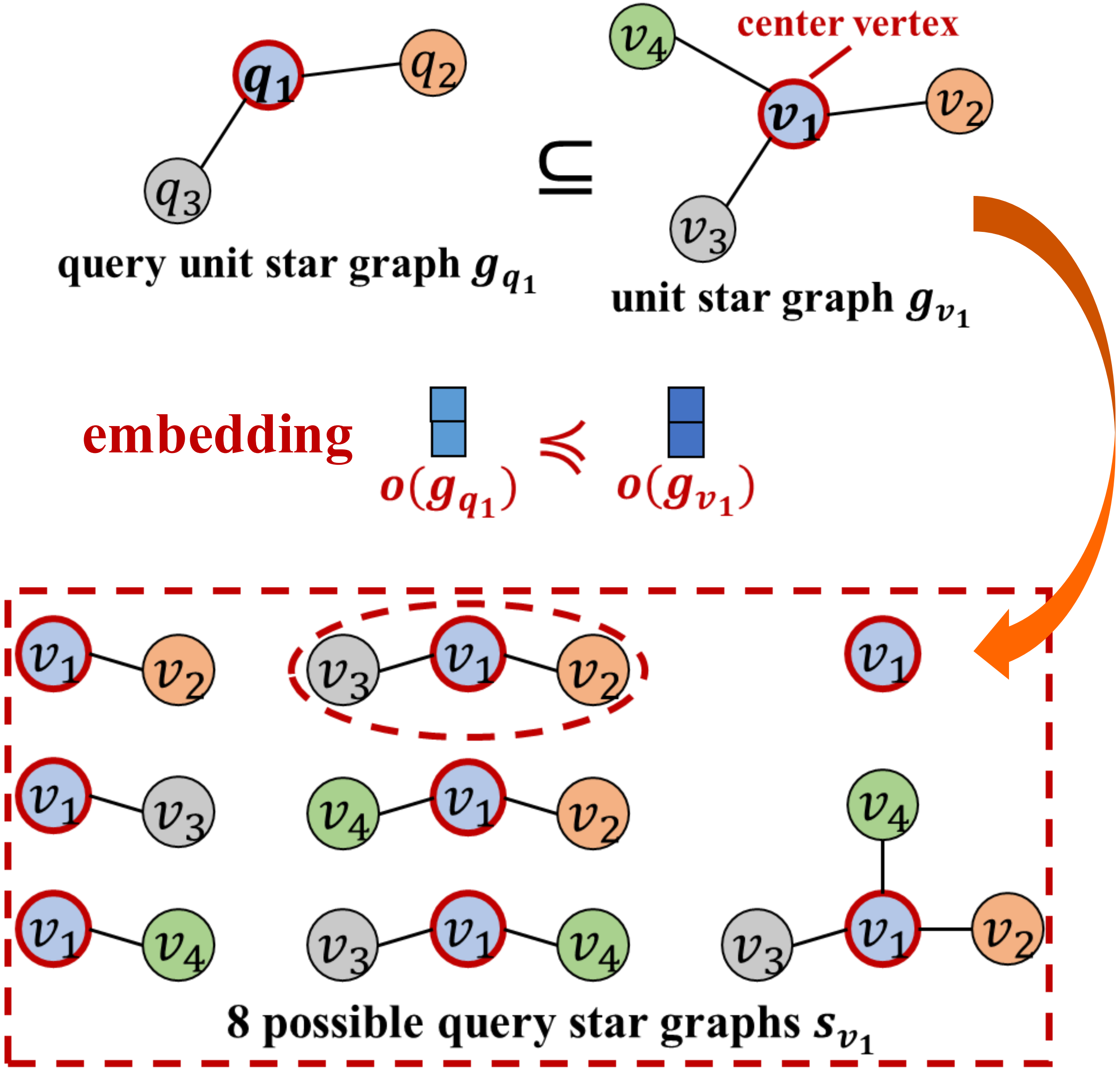
- Online subgraph matching
  - Cost-model-based query plan
  - Candidate path search in the embedding space by the index traversal



Ye, Y., Lian, X., & Chen, M. (2024). Efficient exact subgraph matching via gnn-based path dominance embedding. Proceedings of the VLDB Endowment, 17(7), 1628-1641.

Subgraph Matching: GNN-PE

- Unit structures
  - Unit star graph g_{v_i} (g_{q_i}): A star subgraph containing a center vertex $v_i \in V(G)$ ($q_i \in V(q)$) and its 1-hop neighbors
  - Unit star substructure s_{v_i}: A (star) subgraph of the unit star subgraph g_{v_i}, i.e., $s_{v_i} \subseteq g_{v_i}$
- Dominance relationship
  - If a query vertex q_i in the query graph q matches with a data vertex v_i, then it must hold that $o(g_{q_i}) \preccurlyeq o(g_{v_i})$
- Intuition
  - If $q \subseteq G$, g_{q_i} must be one of v_i's substructures s_{v_i}



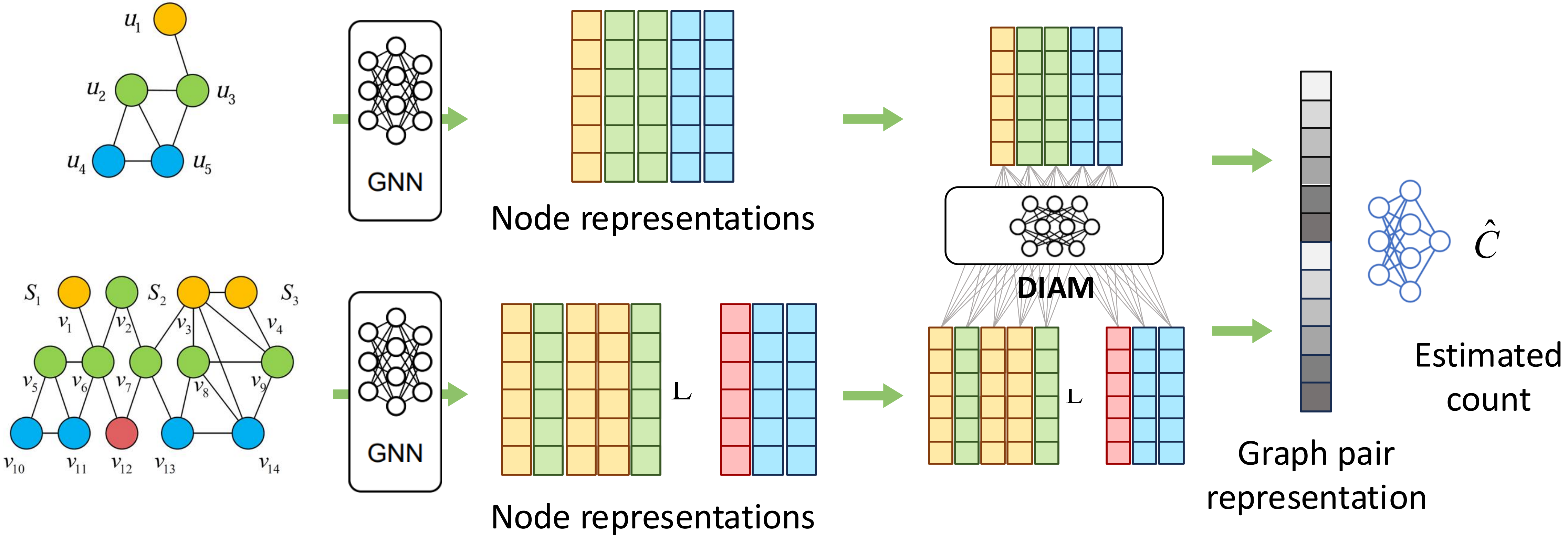
Ye, Y., Lian, X., & Chen, M. (2024). Efficient exact subgraph matching via gnn-based path dominance embedding. Proceedings of the VLDB Endowment, 17(7), 1628-1641.

Graph Query Processing

- Subgraph Isomorphism
  - Subgraph Matching
  - **Subgraph Counting**
- Graph Similarity
  - Graph Edit Distance
- Community Search
  - Disjoint Community Search
  - Overlapping Community Search

Subgraph Counting: Existing Works

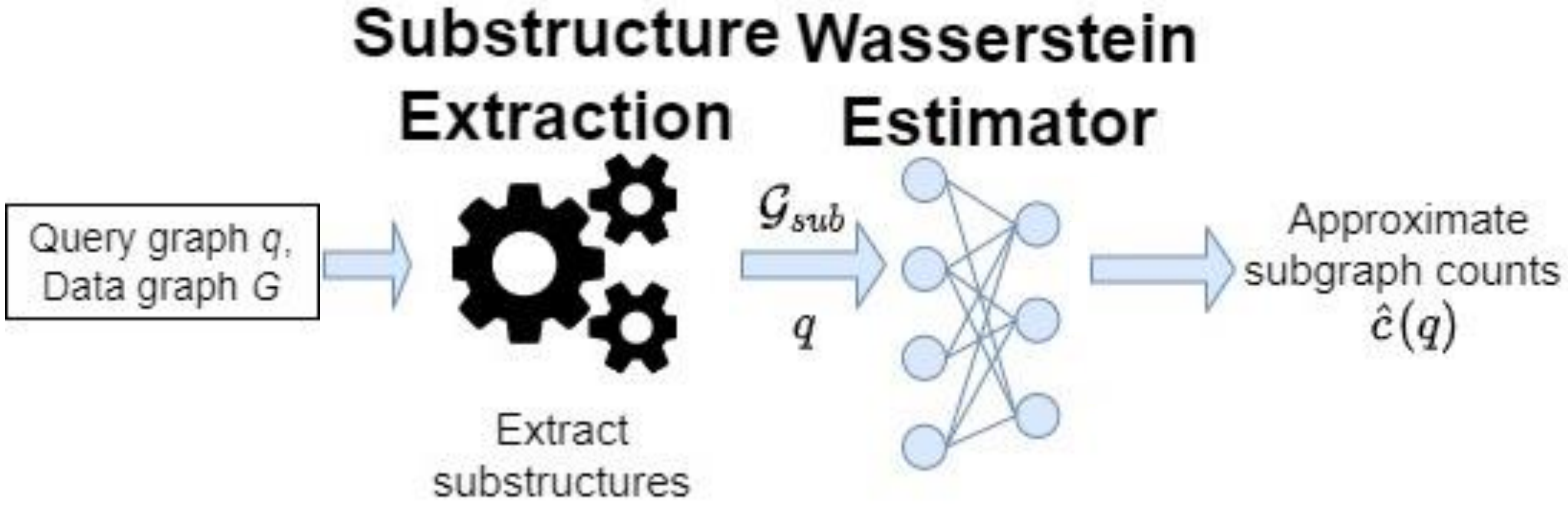
NSIC [KDD'20]



Hou, W., Zhao, X., & Tang, B. (2024, May). Learnsc: An efficient and unified learning-based framework for subgraph counting problem. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)* (pp. 2625-2638). IEEE.

Subgraph Counting: NeurSC

- ***Neur**al **S**ubgraph **C**ounting* method: *NeurSC*

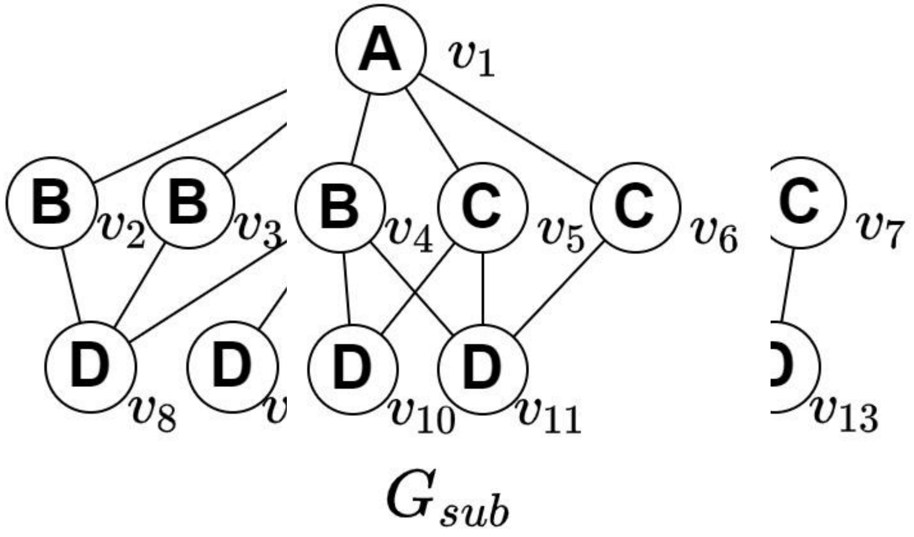


Wang, H., Hu, R., Zhang, Y., Qin, L., Wang, W., & Zhang, W. (2022, June). Neural subgraph counting with wasserstein estimator. In *Proceedings of the 2022 International Conference on Management of Data* (pp. 160-175).

Subgraph Counting: NeurSC

- ***Substructure Extraction***

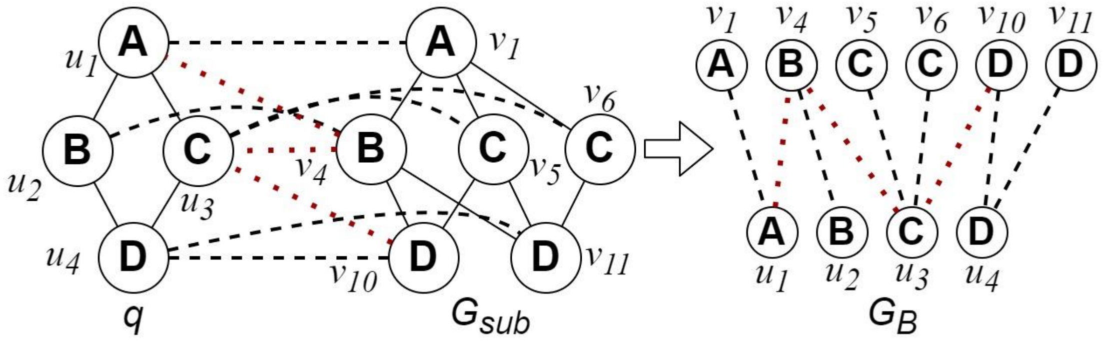
- **Complete Candidate Vertex Set** (CS):
  - $CS(u)$ for query vertex $u \in V$ is a set of data vertices $v \in V'$
  - If (u, v) exists in a match from q to G, then $v \in CS(u)$
- Candidate set of query q: $CS(\text{q}) = \cup_{u \in V} CS(u)$
- First, we determine the complete candidate vertex set for all query vertices using *local pruning* and *global refinement*.
- Based on neighboring and label information
- Induced subgraphs of G with vertices $CS(\text{q})$ are used as the candidate substructures, denoted as G_{sub}

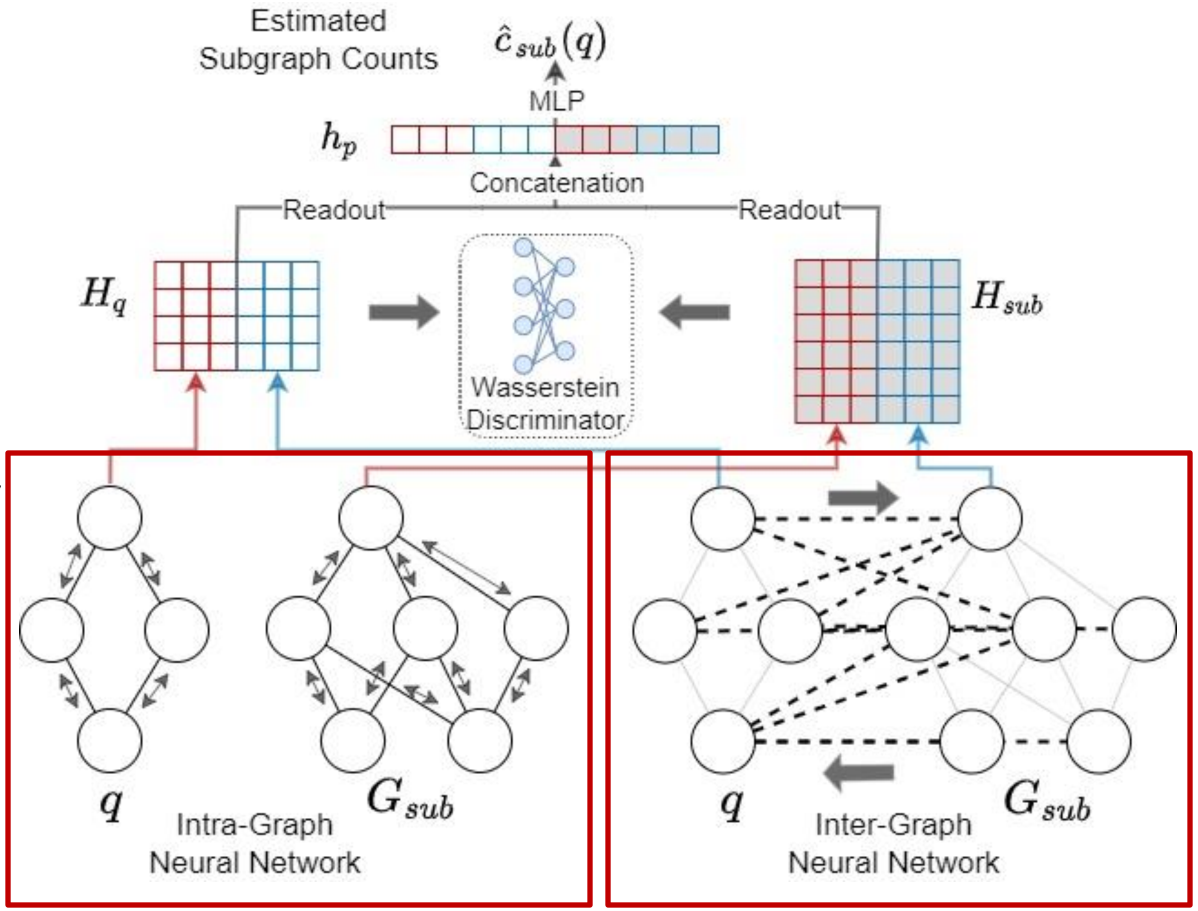


Wang, H., Hu, R., Zhang, Y., Qin, L., Wang, W., & Zhang, W. (2022, June). Neural subgraph counting with wasserstein estimator. In *Proceedings of the 2022 International Conference on Management of Data* (pp. 160-175).

Subgraph Counting: NeurSC

- ***Wasserstein Estimator***
- **Intra-Graph Neural Network**
  - For both query graph and substructure.
  - Capture structural and attribute information.
  - $\boldsymbol{h_u^{(k)}} = \boldsymbol{MLP^{(k)}}((\boldsymbol{1} + \boldsymbol{\epsilon^{(k)}})\boldsymbol{h_u^{(k-1)}}, \sum_{u' \in N_q(u)} \boldsymbol{h_{u'}^{(k)}})$
- **Inter-Graph Neural Network**
  - Construct a bipartite graph for inter-relationship.
  - Capture the mapping relationship between query vertices and corresponding candidate vertices
  - $\boldsymbol{h_u^{(k)}} = \boldsymbol{\sigma}(\boldsymbol{a_{uu}^{(k)}\theta^{(k)}h_u^{(k-1)}}, \sum_{v \in N_{G_B}(u)} \boldsymbol{a_{uv}^{(k)}\theta^{(k)}h_v^{(k)}})$

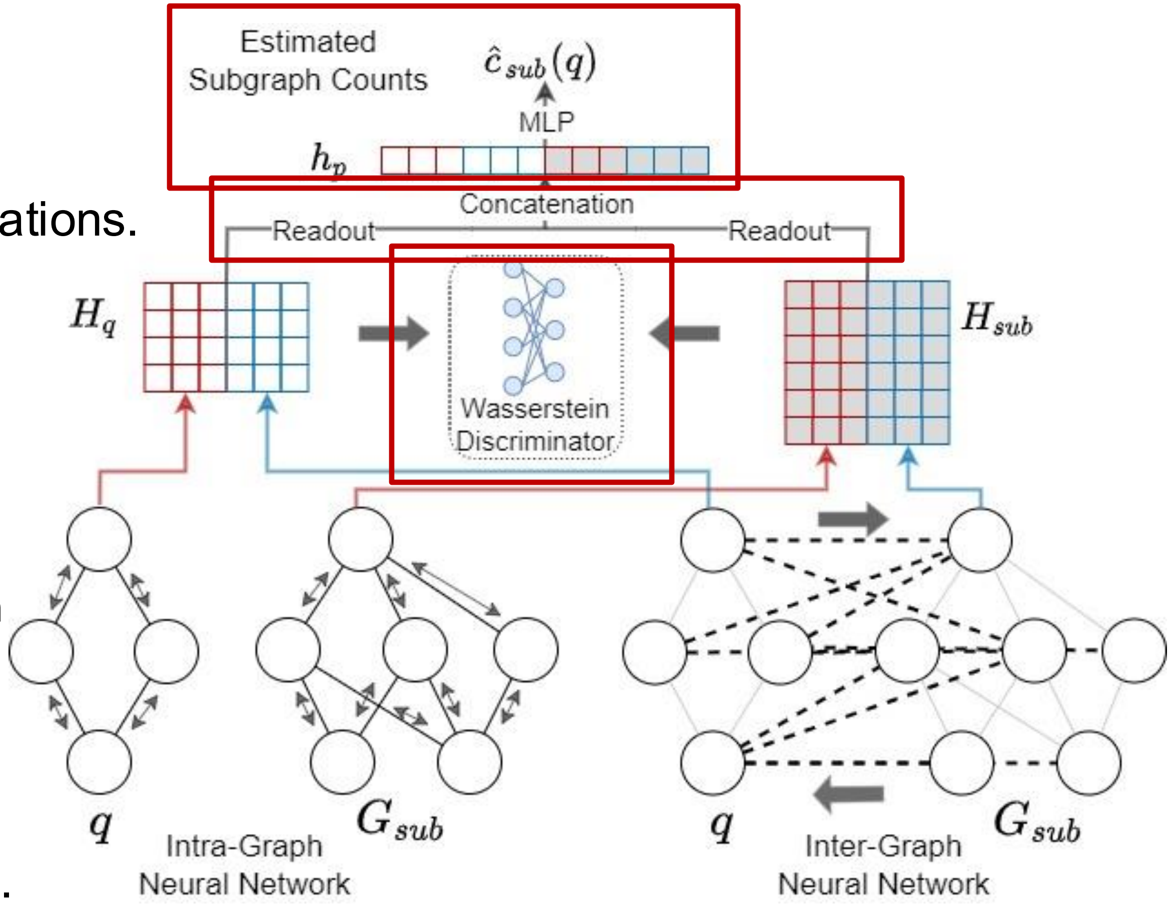




Wang, H., Hu, R., Zhang, Y., Qin, L., Wang, W., & Zhang, W. (2022, June). Neural subgraph counting with wasserstein estimator. In *Proceedings of the 2022 International Conference on Management of Data* (pp. 160-175).

Subgraph Counting: NeurSC

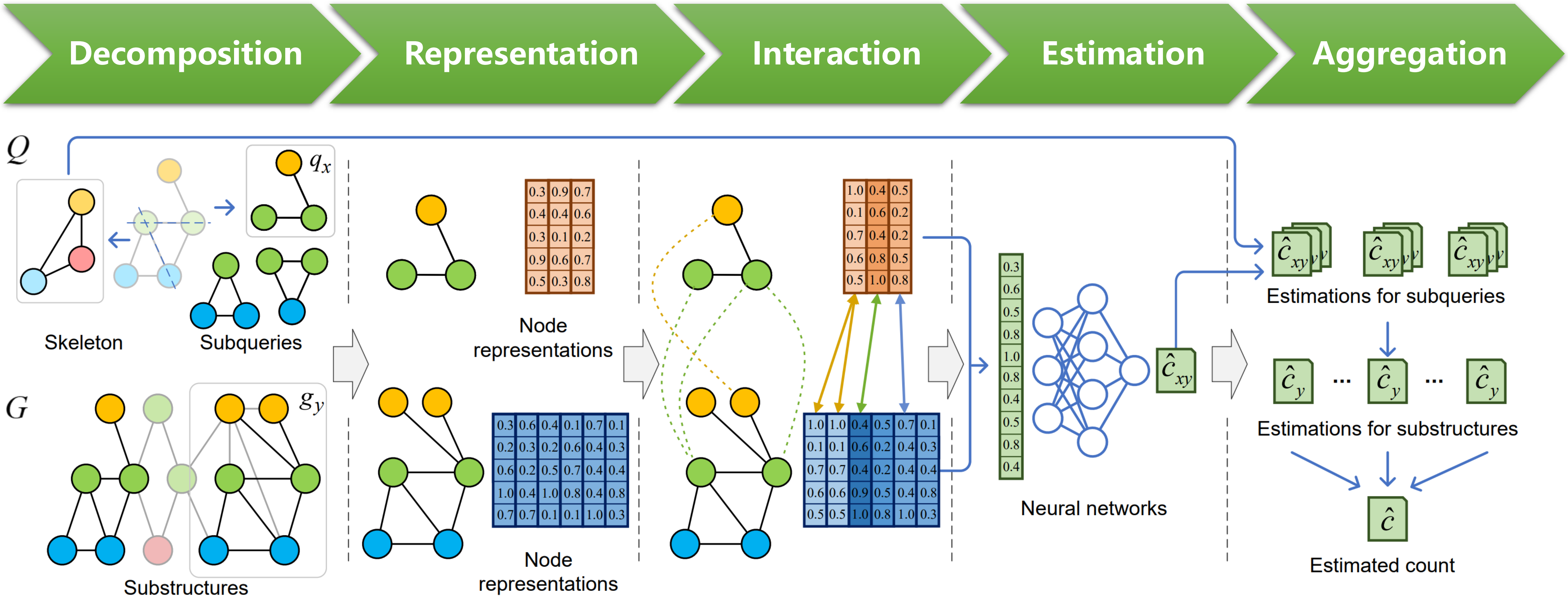
- ***Wasserstein Estimator***
- **Readout**
  - Sum Pooling
  - Concatenation of intra- and inter-graph representations.
- **Prediction**
  - Multi-layer perceptron.
- **Wasserstein Discriminator**
  - Minimize Wasserstein distance between q and G_{sub}
  - Further utilize the vertex correspondence information between q and G
  - $L_W(q, G_{sub}) = \sum_{u \in V'(q)} f_{\omega(h_u)} - \sum_{v \in V'(G_{sub})} f_{\omega(h_v)}$
- **Expressive Power**
  - *WEst* is as powerful as 1-Weisfeiler-Lehman test.



Wang, H., Hu, R., Zhang, Y., Qin, L., Wang, W., & Zhang, W. (2022, June). Neural subgraph counting with wasserstein estimator. In *Proceedings of the 2022 International Conference on Management of Data* (pp. 160-175).

Subgraph Counting: LearnSC

An efficient and unified framework, LearnSC [ICDE'24]

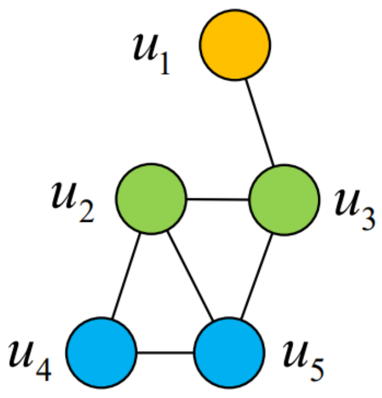


Hou, W., Zhao, X., & Tang, B. (2024, May). Learnsc: An efficient and unified learning-based framework for subgraph counting problem. In 2024 IEEE 40th International Conference on Data Engineering (ICDE) (pp. 2625-2638). IEEE.

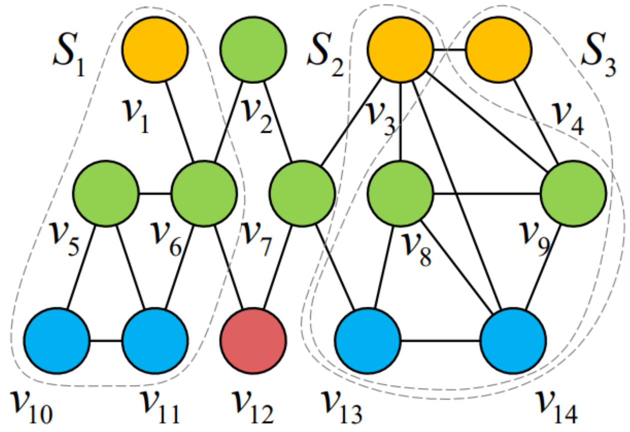
Subgraph Counting: LearnSC

LearnSC：data graph decomposition

- Data graphs are **large**, lead to heavy cost on deep learning models
- Data graph contains multiple **unqualified nodes**, which are negligible for matching results



Query graph



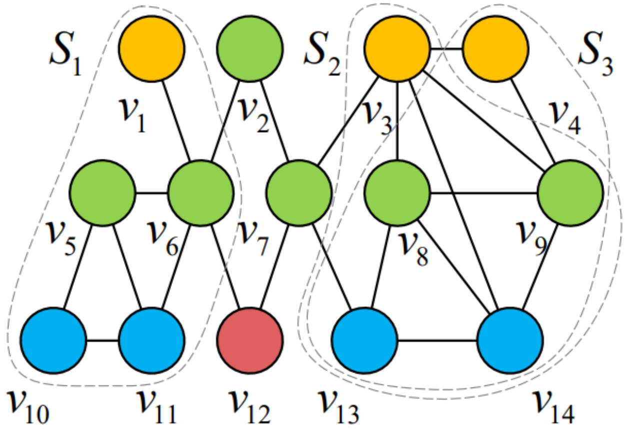
Data graph

Decompose data graph, **remove** unqualified nodes/edges, **extract** key parts

Hou, W., Zhao, X., & Tang, B. (2024, May). Learnsc: An efficient and unified learning-based framework for subgraph counting problem. In 2024 IEEE 40th International Conference on Data Engineering (ICDE) (pp. 2625-2638). IEEE.

Subgraph Counting: LearnSC

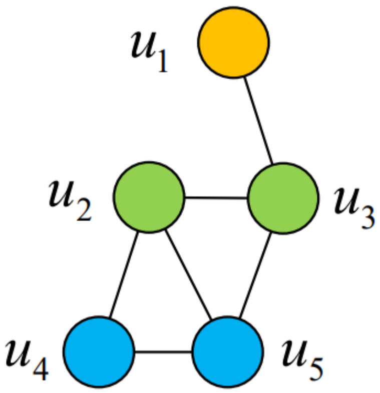
Data graph decomposition



- **Filter candidate nodes**

  To remove unqualified nodes

  - **Neighborhood** information
  - **Iterative** removal



Query graph

u1: [1, 3, 4]
u2: [2, 5, 6, 7, 8, 9]
u3: [2, 5, 6, 7, 8, 9]
u4: [10, 11, 13, 14]
u5: [10, 11, 13, 14]

Initial candidates

u1: [1, 3, 4]
u2: [5, 8, 9]
u3: [6, 8, 9]
u4: [10, 13, 14]
u5: [11, 13, 14]

Hou, W., Zhao, X., & Tang, B. (2024, May). Learnsc: An efficient and unified learning-based framework for subgraph counting problem. In 2024 IEEE 40th International Conference on Data Engineering (ICDE) (pp. 2625-2638). IEEE.

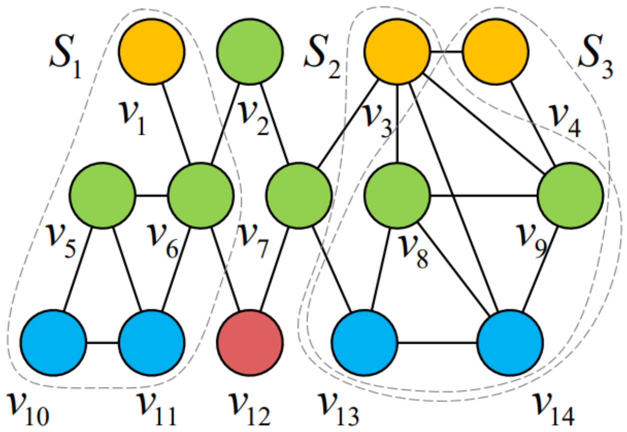
Subgraph Counting: LearnSC

Data graph decomposition

- **Extract substructures**

Extract substructures according to candidates

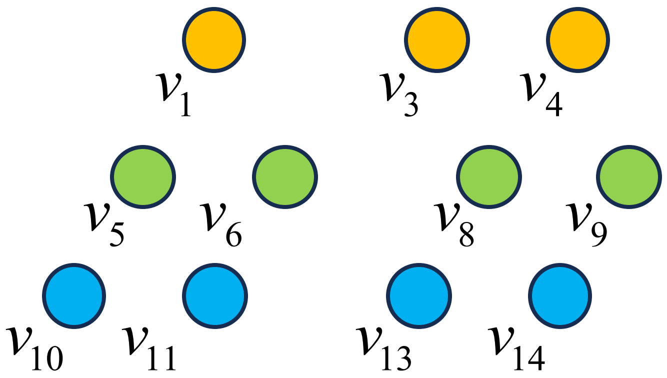
- Vertex-induced subgraphs



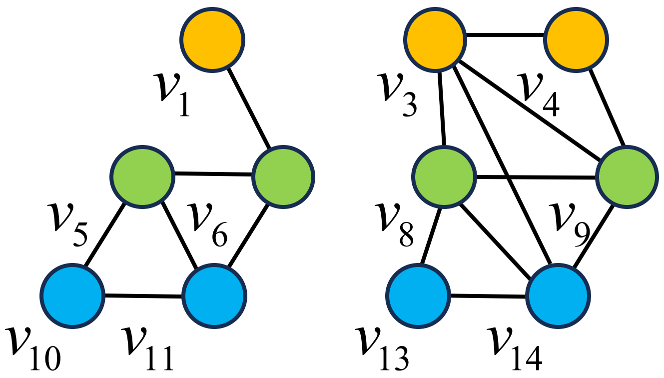
Data graph

u1: [1, 3, 4]
u2: [5, 8, 9]
u3: [6, 8, 9]
u4: [10, 13, 14]
u5: [11, 13, 14]

Filtered candidates



Valuable data graph nodes



substructures

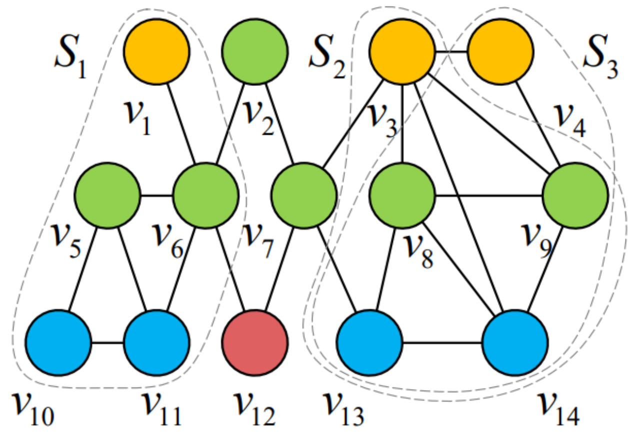
Subgraph Counting: LearnSC

Data graph decomposition

➢ **Extract substructures**

Extract substructures according to candidates

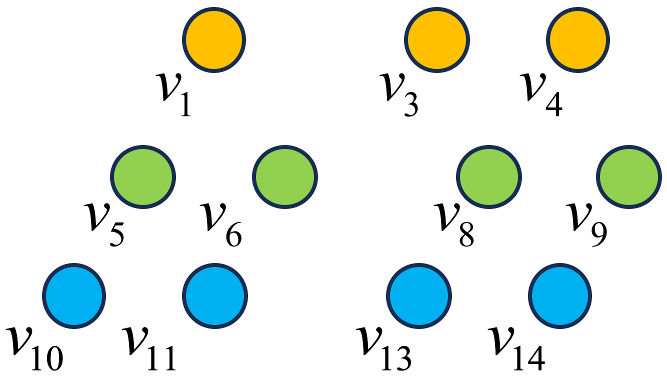
- Vertex-induced subgraphs
- But avoid redundant edges



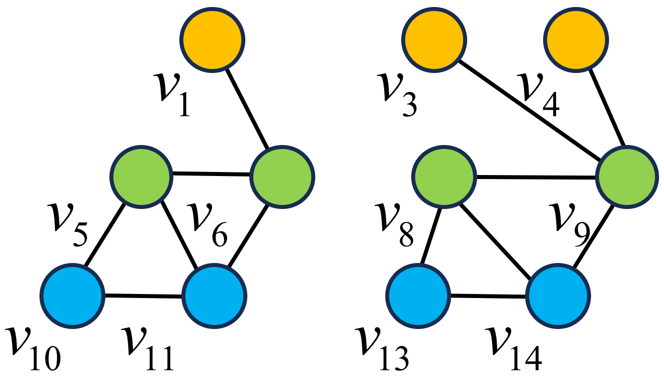
Data graph

u1: [1, 3, 4]
u2: [5, 8, 9]
u3: [6, 8, 9]
u4: [10, 13, 14]
u5: [11, 13, 14]

Filtered candidates



Valuable data graph nodes

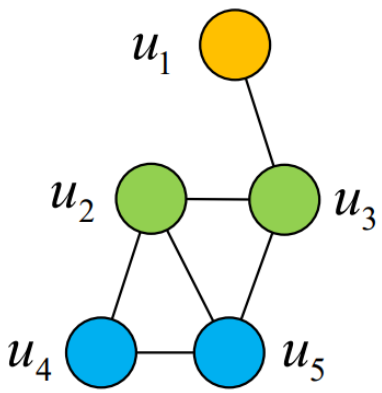


substructures

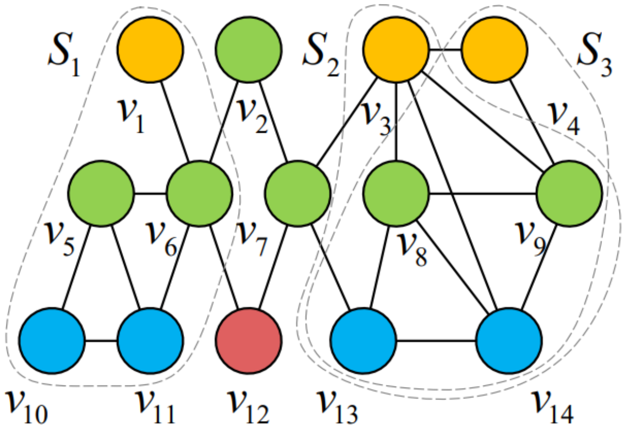
Subgraph Counting: LearnSC

LearnSC： Query graph decomposition

- Query graphs are **various**, Explicitly learn subqueries to improve the representation qualities
- **The dependency** among subqueries are supposed to be reserved



Query graph



Data graph

Decompose query graph, **reserve** dependency, **improve** representation quality

Hou, W., Zhao, X., & Tang, B. (2024, May). Learnsc: An efficient and unified learning-based framework for subgraph counting problem. In 2024 IEEE 40th International Conference on Data Engineering (ICDE) (pp. 2625-2638). IEEE.

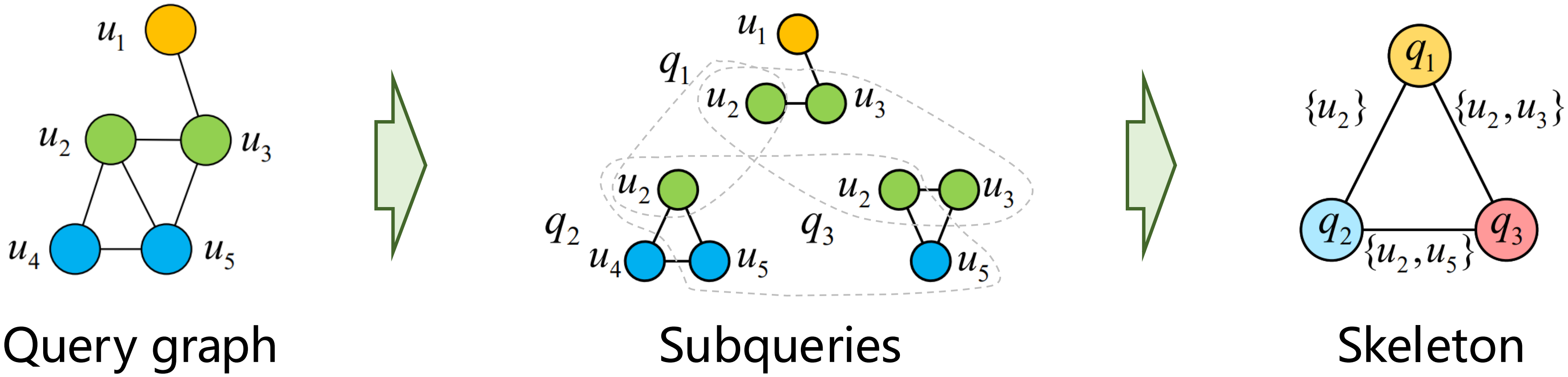
Subgraph Counting: LearnSC

LearnSC： query graph decomposition

➢ **Skeleton-based query graph decomposition**

Split query into subqueries, with a skeleton reserving dependency

- Post process after splitting
- Built a skeleton recording **connecting relations** and **shared nodes**



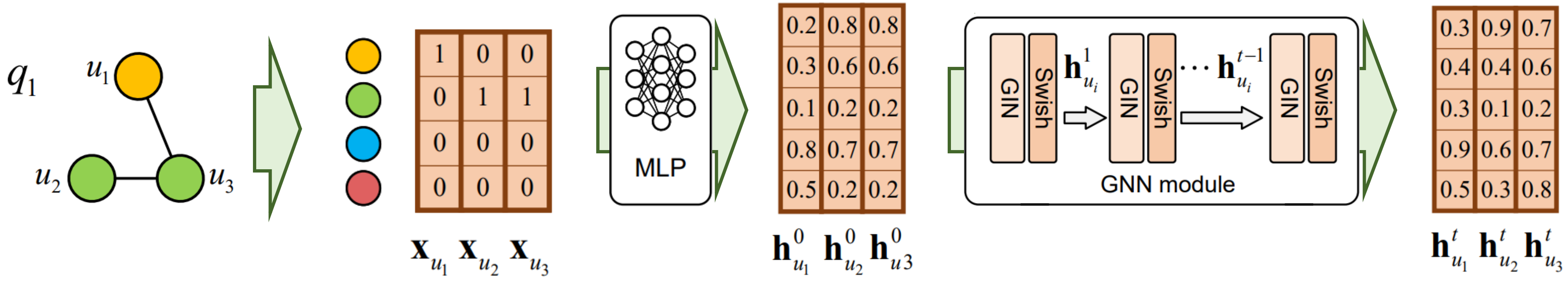
Hou, W., Zhao, X., & Tang, B. (2024, May). Learnsc: An efficient and unified learning-based framework for subgraph counting problem. In 2024 IEEE 40th International Conference on Data Engineering (ICDE) (pp. 2625-2638). IEEE.

Subgraph Counting: LearnSC

LearnSC： Representations

➢ To embed nodes in substructures and subqueries into vectors, which captures **implicit feature**

- MLP → Node attribute features
- GNN → Topology features



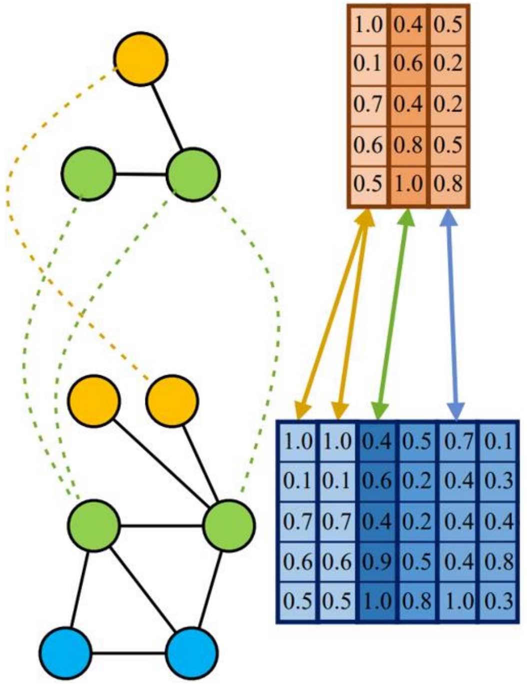
Input graph | Label encode | Label embeddings | Node embeddings

Hou, W., Zhao, X., & Tang, B. (2024, May). Learnsc: An efficient and unified learning-based framework for subgraph counting problem. In 2024 IEEE 40th International Conference on Data Engineering (ICDE) (pp. 2625-2638). IEEE.

Subgraph Counting: LearnSC

LearnSC： Interaction

- ➢ Subgraph counting is based on subgraph matching
- ➢ The **potential matching information** among query node and data graph node is essential



Interact **cross graphs**, capture **potential matching information**

Hou, W., Zhao, X., & Tang, B. (2024, May). Learnsc: An efficient and unified learning-based framework for subgraph counting problem. In 2024 IEEE 40th International Conference on Data Engineering (ICDE) (pp. 2625-2638). IEEE.

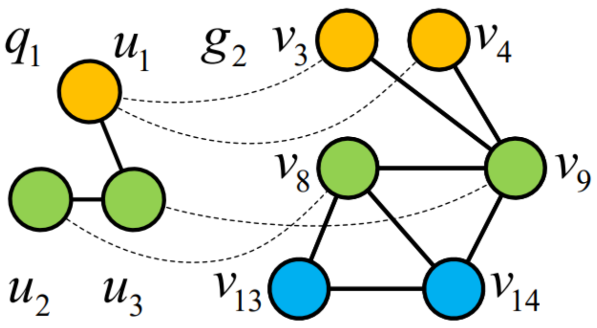
Subgraph Counting: LearnSC

LearnSC：interaction

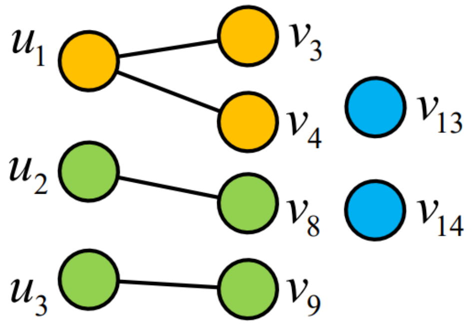
➢ **Construct intergraph**

Only potential matching nodes interacts

- Candidates are potential matching nodes
- Query nodes connect to their candidate to construct an intergraph



A subquery and a substructure



Intergraph

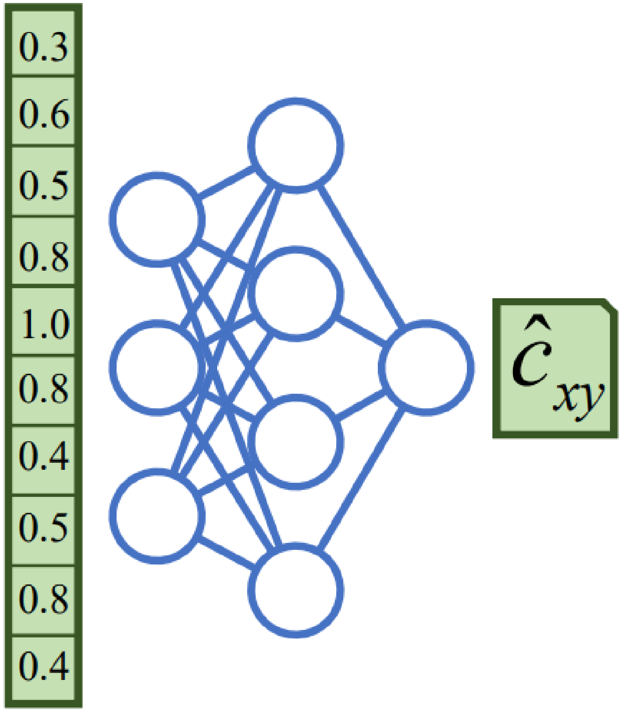
Hou, W., Zhao, X., & Tang, B. (2024, May). Learnsc: An efficient and unified learning-based framework for subgraph counting problem. In 2024 IEEE 40th International Conference on Data Engineering (ICDE) (pp. 2625-2638). IEEE.

Subgraph Counting: LearnSC

LearnSC：Estimation

➢ Representations captures **label** features , **topology** features, **and potential matching** information

➢ Using representations to estimate the count of **a subquery in a substructure**



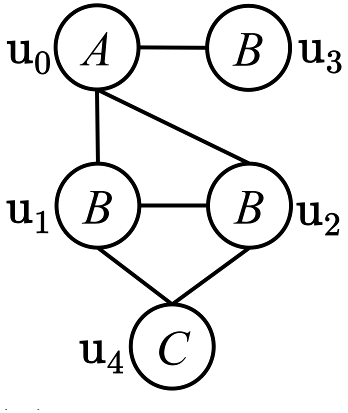
Readout representations, **estimate** counts

Hou, W., Zhao, X., & Tang, B. (2024, May). Learnsc: An efficient and unified learning-based framework for subgraph counting problem. In 2024 IEEE 40th International Conference on Data Engineering (ICDE) (pp. 2625-2638). IEEE.

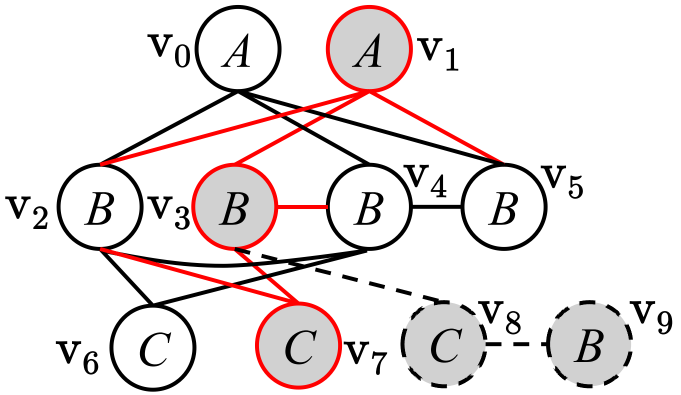
Subgraph Counting: FlowSC

Motivation

Unsatisfactory Candidate Filtering



Q



G

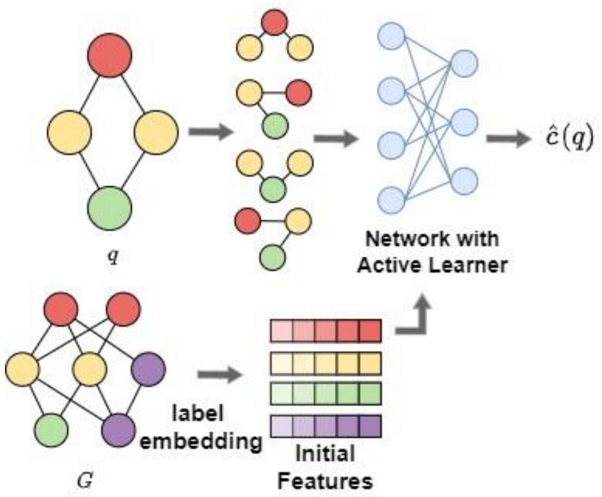
- GQL[1] and EdgeBipartite[2] do not take triangle edge (u_1, u_2) into consideration, so v_1 is not removed from $C(u_0)$.
- TriangleSafety[2] can remove v_1 from $C(u_0)$, but is limited by efficiency issue.

1 Huahai He and Ambuj K. Singh. 2008. Graphs-at-a-Time: Query Language and Access Methods for Graph Databases. In Proceedings of SIGMOD. 405–418.
2 Wonseok Shin, Siwoo Song, Kunsoo Park, and Wook-Shin Han. 2024. Cardinality Estimation of Subgraph Matching: A Filtering-Sampling Approach. In Proceedings of VLDB. 1697–1709.

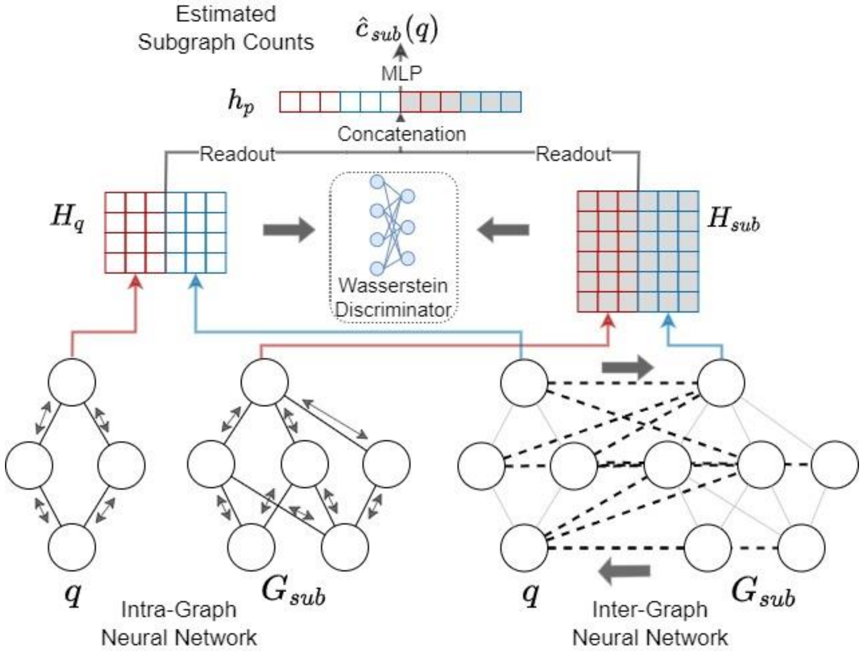
Subgraph Counting: FlowSC

Motivation

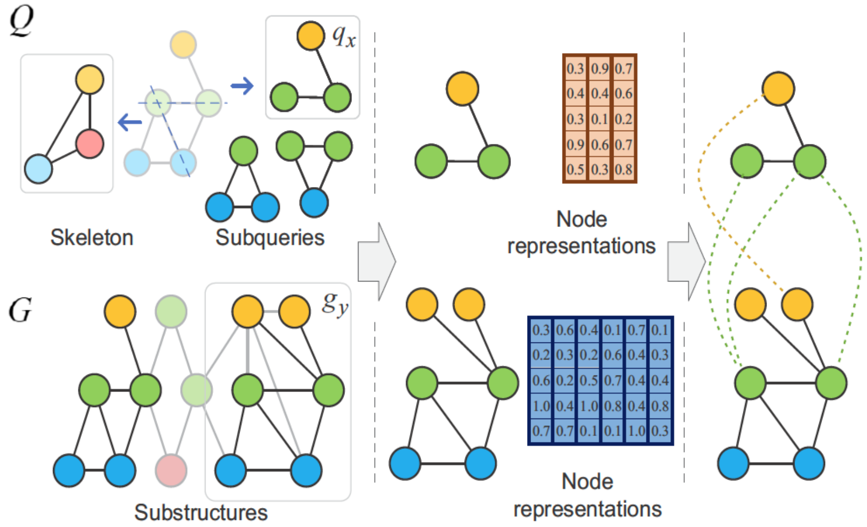
Lack of explicit modelling between structural features and subgraph counts.



LSS



NeurSC



LearnSC

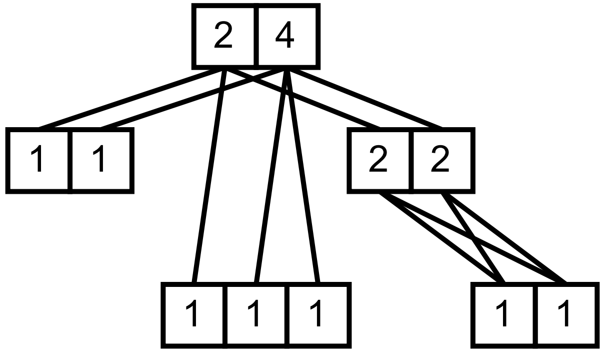
- They do not capture the explicit relationship between structure and specific counts, but regress blindly.
- unsatisfactory performance in both efficiency and accuracy.

Guo, Qiuyu, et al. "Efficient and Accurate Subgraph Counting: A Bottom-up Flow-learning Based Approach." VLDB 2025

Subgraph Counting: FlowSC

Motivation

Inspired by the candidate-tree based counting:



$$W(u,v) = \prod_{u_c \in N_c(u)} \sum_{v_c \in C(u_c|u,v)} W(u_c, v_c)$$

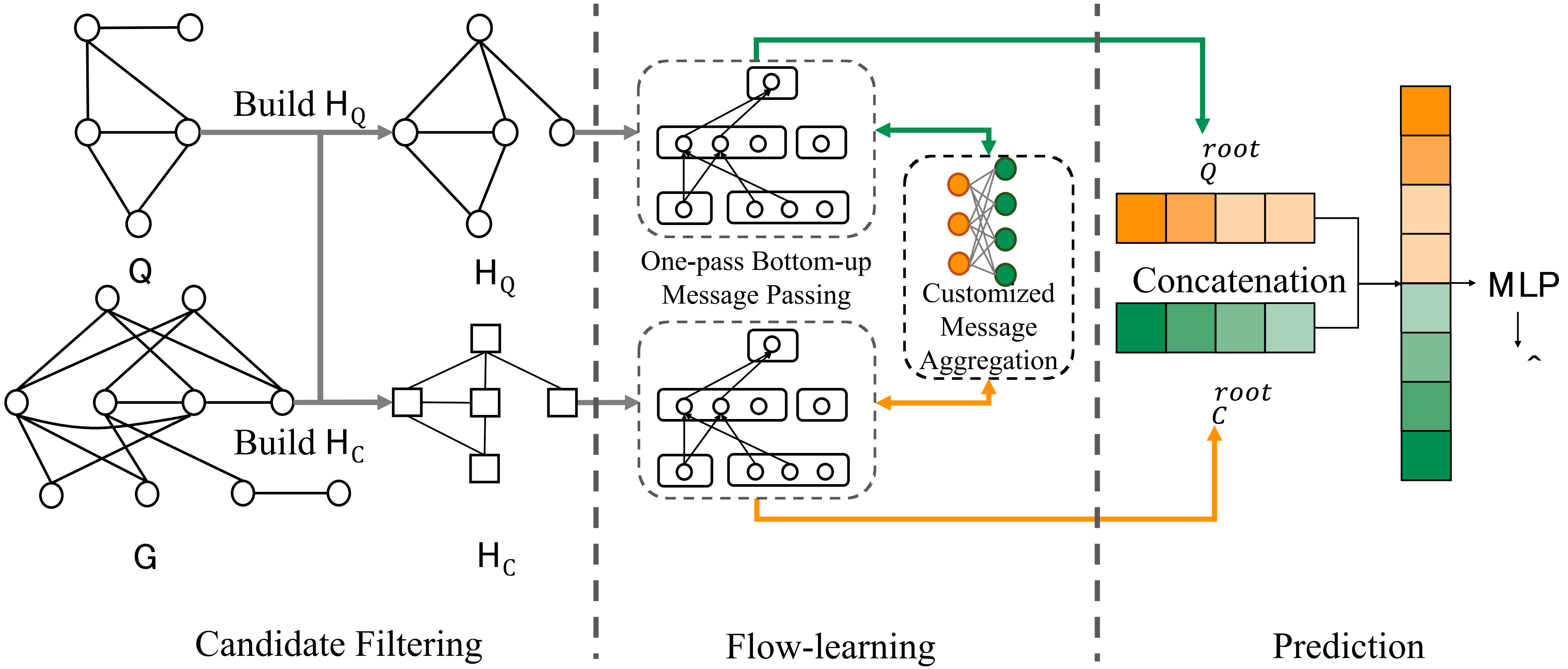
Limitations:

- Based on a spanning tree, the constraints of non-tree edges are ignored.
- Isomorphism constraints are not considered in the tree counting.

Subgraph Counting: FlowSC

Overview

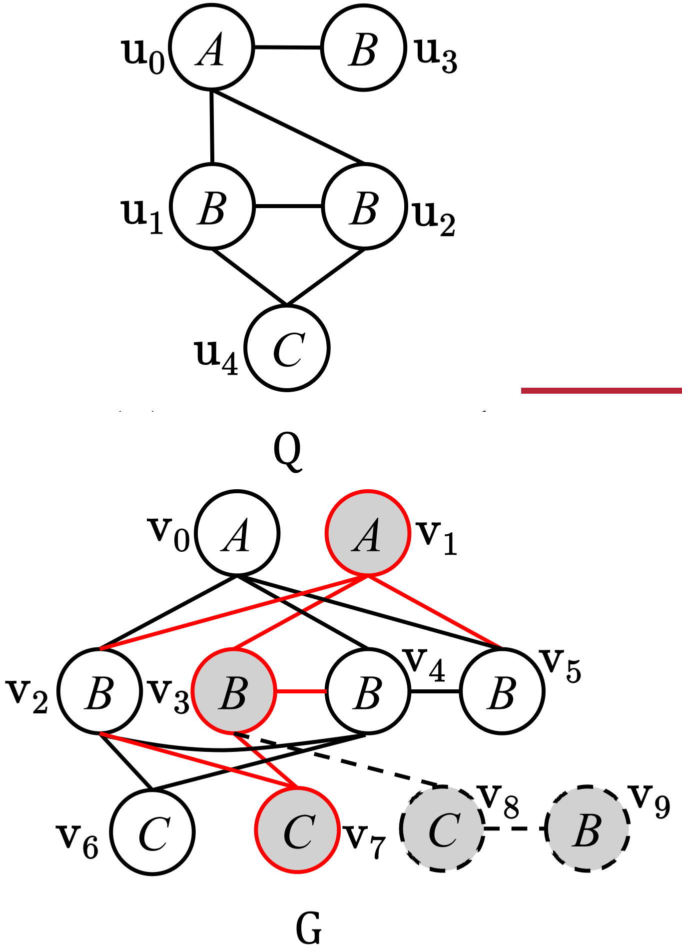
3-step learning-based method: FlowSC

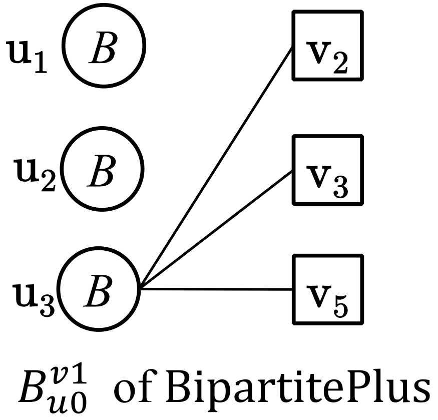


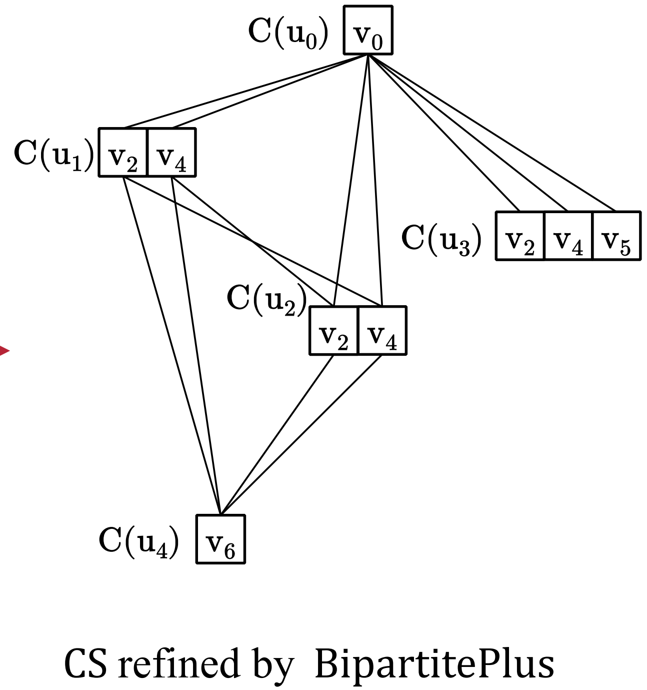
Guo, Qiuyu, et al. "Efficient and Accurate Subgraph Counting: A Bottom-up Flow-learning Based Approach." VLDB 2025

Subgraph Counting: FlowSC

Our solution - **BipartitePlus:** Bipartite graph-based filtering can be enhanced by the connectivity check for the neighbors of the matching vertex pairs.





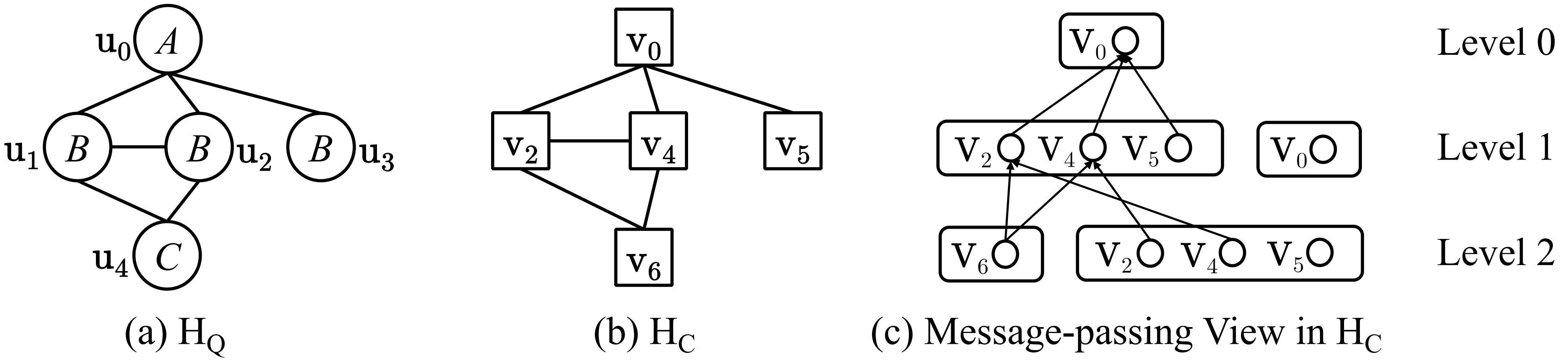


v_1 is removed from $C(u_0)$ by $B_{u_0}^{v_1}$.

Subgraph Counting: FlowSC

Flow Learning

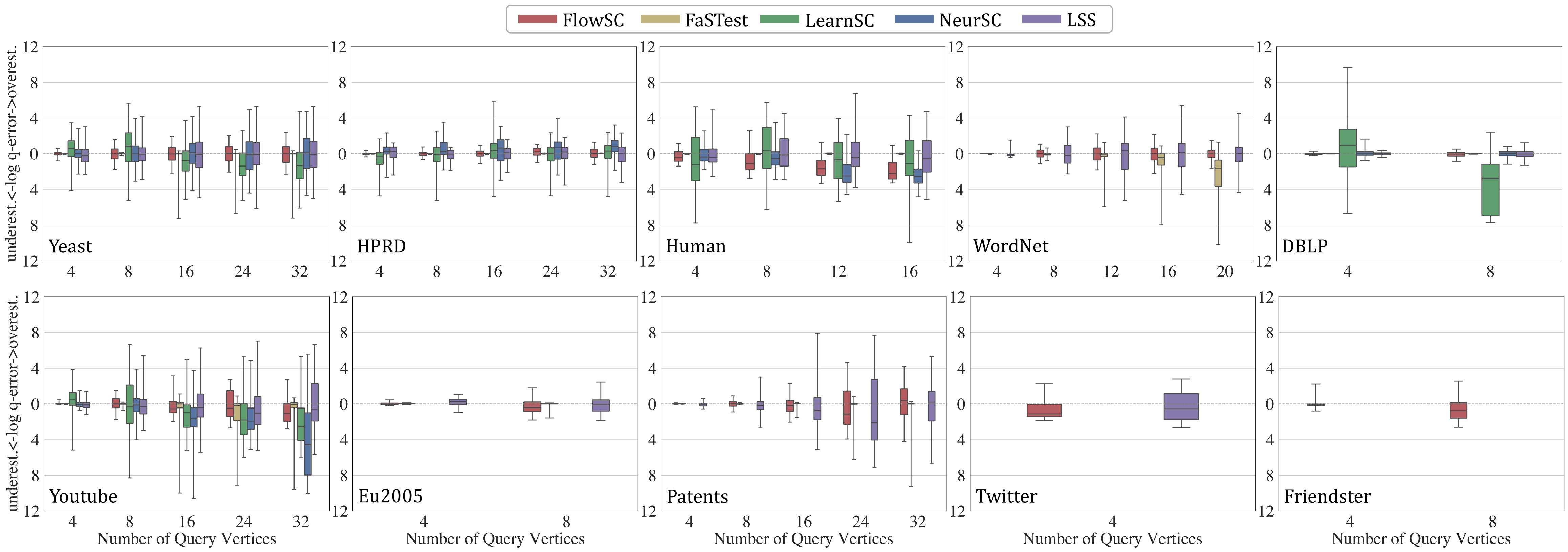
- **FlowSC:** Simulating the candidate tree-based counting by flow-learning
  - **One-pass Bottom-up Message-passing -** simulating the bottom-up dynamic programming



  - **Customized Message Aggregation –** take matching condition checks into learning
  - **Prediction** - regression

Subgraph Counting: FlowSC

Accuracy Evaluation



Graph Query Processing

- Subgraph Isomorphism
  - Subgraph Matching
  - Subgraph Counting
- Graph Similarity
  - **Graph Edit Distance**
- Community Search
  - Disjoint Community Search
  - Overlapping Community Search

Machine Learning Models for Graph Edit Distance

How about learning-based techniques for graph similarity

Let’s focus on a fundamental graph similarity metric: Graph Edit Distance.

Graph Edit Distance aims to determine the minimum number of edit operations required to transform one graph into another, and the sequence of edit operations is called a graph edit path.

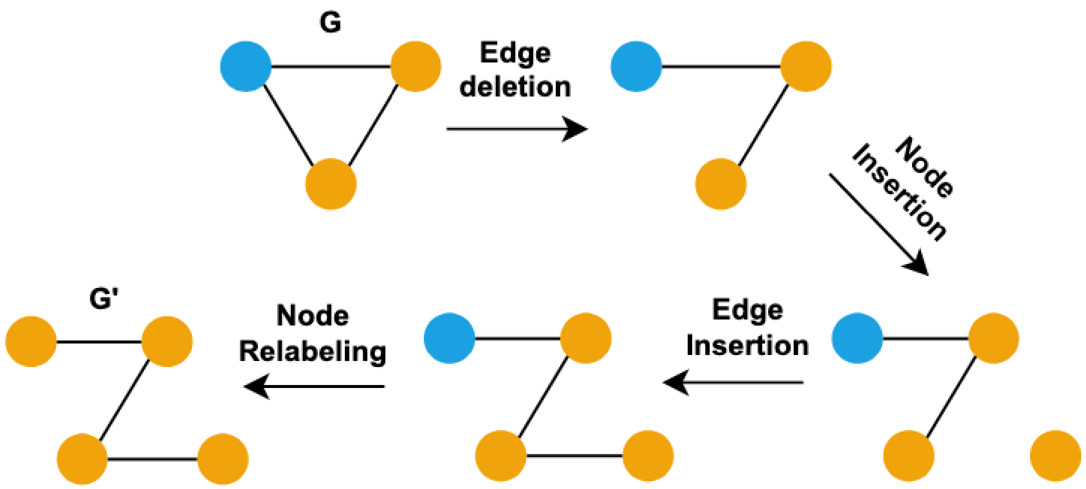


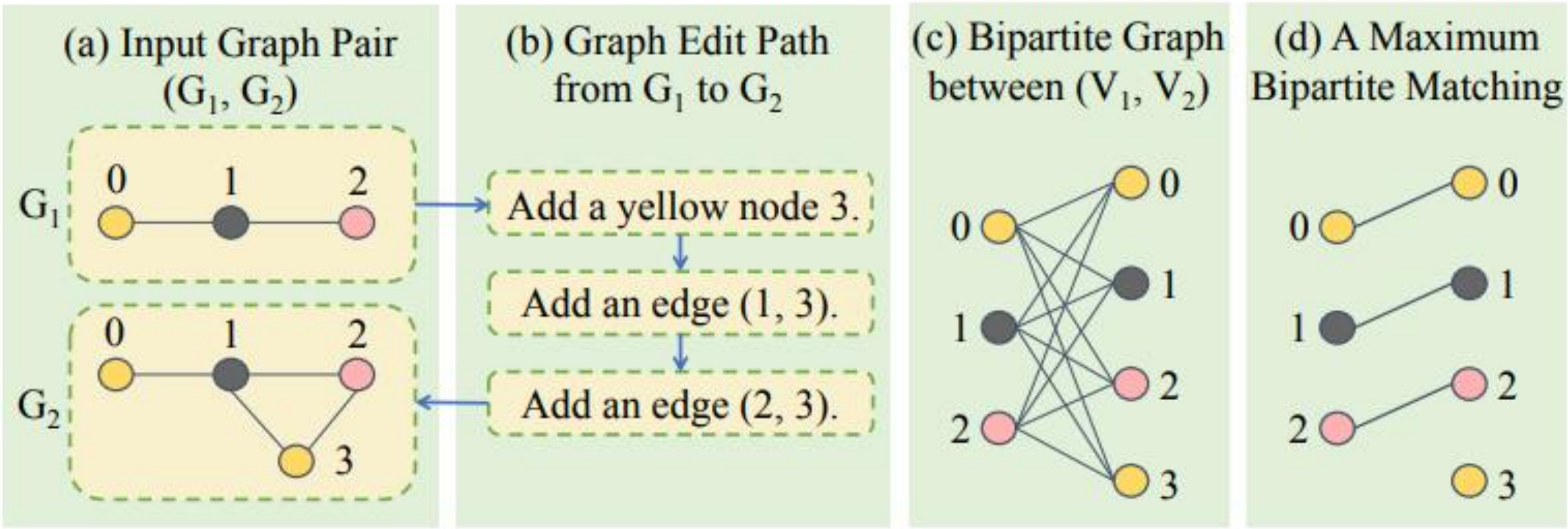
Figure 1: An optimal edit path for transforming G to G'. GED$(G, G') = 4$.

Huang, Wei, et al. "DiffGED: Computing Graph Edit Distance via Diffusion-based Graph Matching." *arXiv preprint arXiv:2503.18245* (2025).

Machine Learning Models for Graph Edit Distance

GEDGNN: Computing Graph Edit Distance via Neural Graph Matching

Graph edit distance can be modelled as maximum bipartite matching.



a, b: An instance of graph edit path.

c, d: Solving GED via bipartite matching.

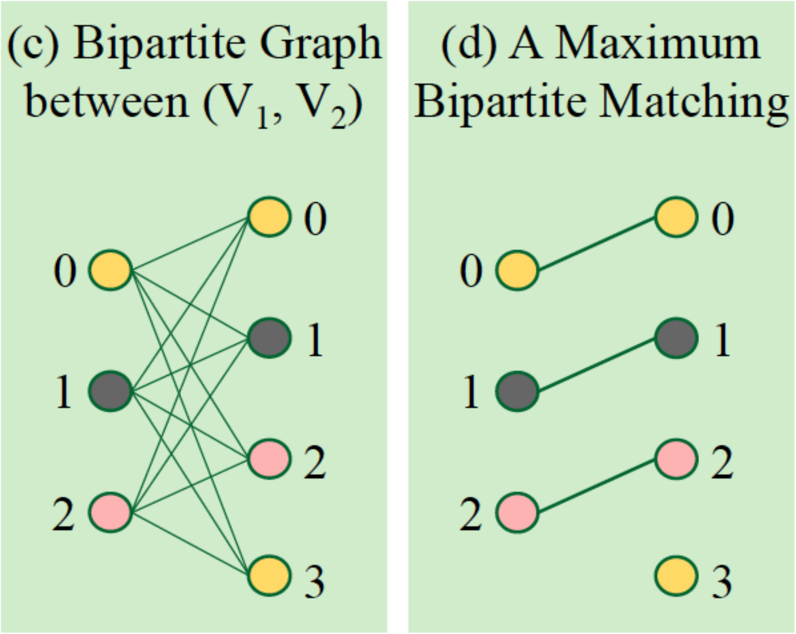
Piao, Chengzhi, et al. "Computing graph edit distance via neural graph matching." Proceedings of the VLDB Endowment 16.8 (2023): 1817-1829.

Machine Learning Models for Graph Edit Distance

GEDGNN: Computing Graph Edit Distance via Neural Graph Matching

A Two-step Framework:

- Using GNN to predict a GED and generate a node matching matrix.
- Post-processing the node matching matrix to find a short edit path.



Bipartite graph matching

Bipartite graph generation

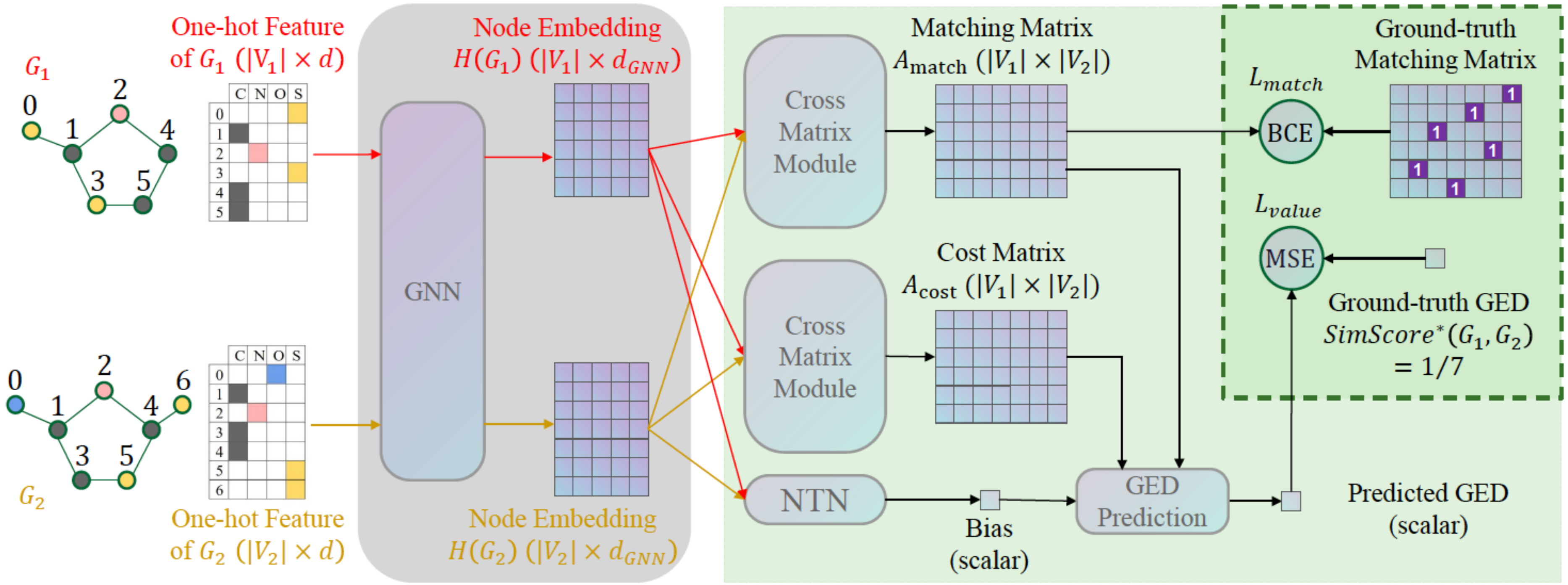
Piao, Chengzhi, et al. "Computing graph edit distance via neural graph matching." Proceedings of the VLDB Endowment 16.8 (2023): 1817-1829.

Machine Learning Models for Graph Edit Distance

GEDGNN: Computing Graph Edit Distance via Neural Graph Matching

A Two-step Framework:

- Using GNN to predict a GED and generate a node matching matrix.
- Post-processing the node matching matrix to find a short edit path.



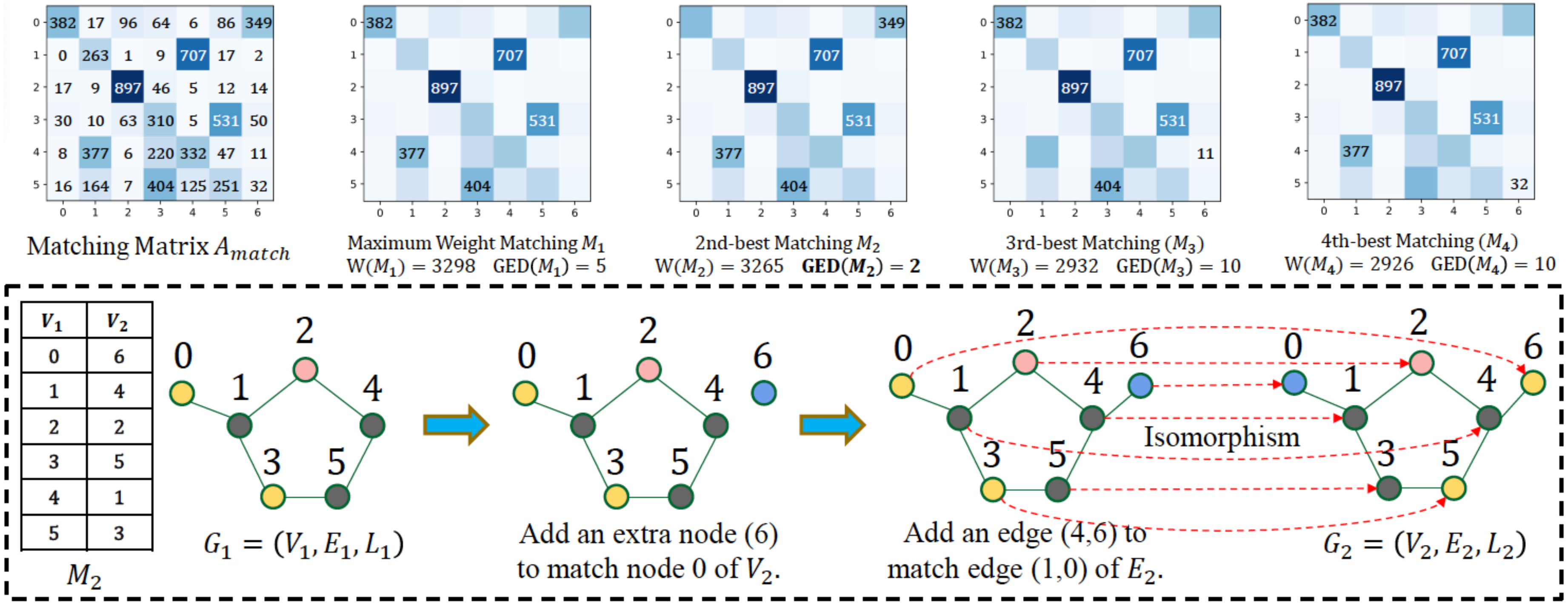
Piao, Chengzhi, et al. "Computing graph edit distance via neural graph matching." Proceedings of the VLDB Endowment 16.8 (2023): 1817-1829.

Machine Learning Models for Graph Edit Distance

GEDGNN: Computing Graph Edit Distance via Neural Graph Matching

A Two-step Framework:

- Using GNN to predict a GED and generate a node matching matrix.
- Post-processing the node matching matrix to find a short edit path.



V_1	V_2
0	6
1	4
2	2
3	5
4	1
5	3

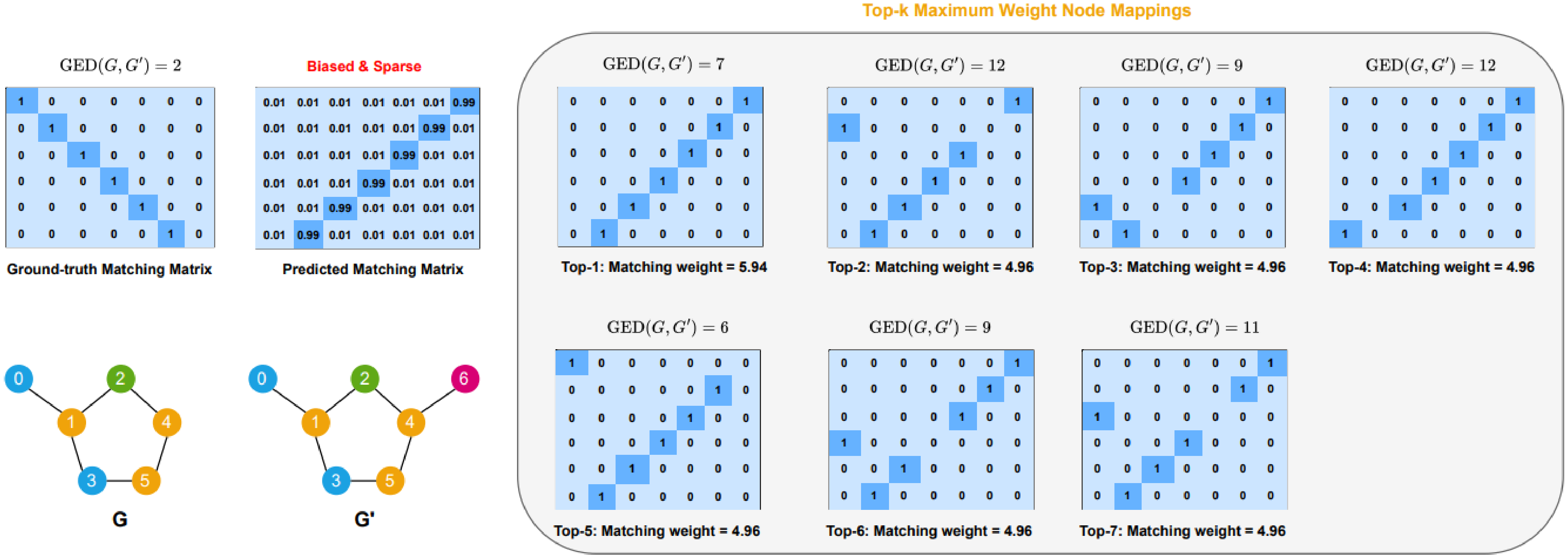
Piao, Chengzhi, et al. "Computing graph edit distance via neural graph matching." Proceedings of the VLDB Endowment 16.8 (2023): 1817-1829.

Machine Learning Models for Graph Edit Distance

DiffGED: Computing Graph Edit Distance via Diffusion-based Graph Matching

Can diffusion models be applied on Graph Edit Distance Computation?

Diffusion models for generation of (bipartite) graph matching.

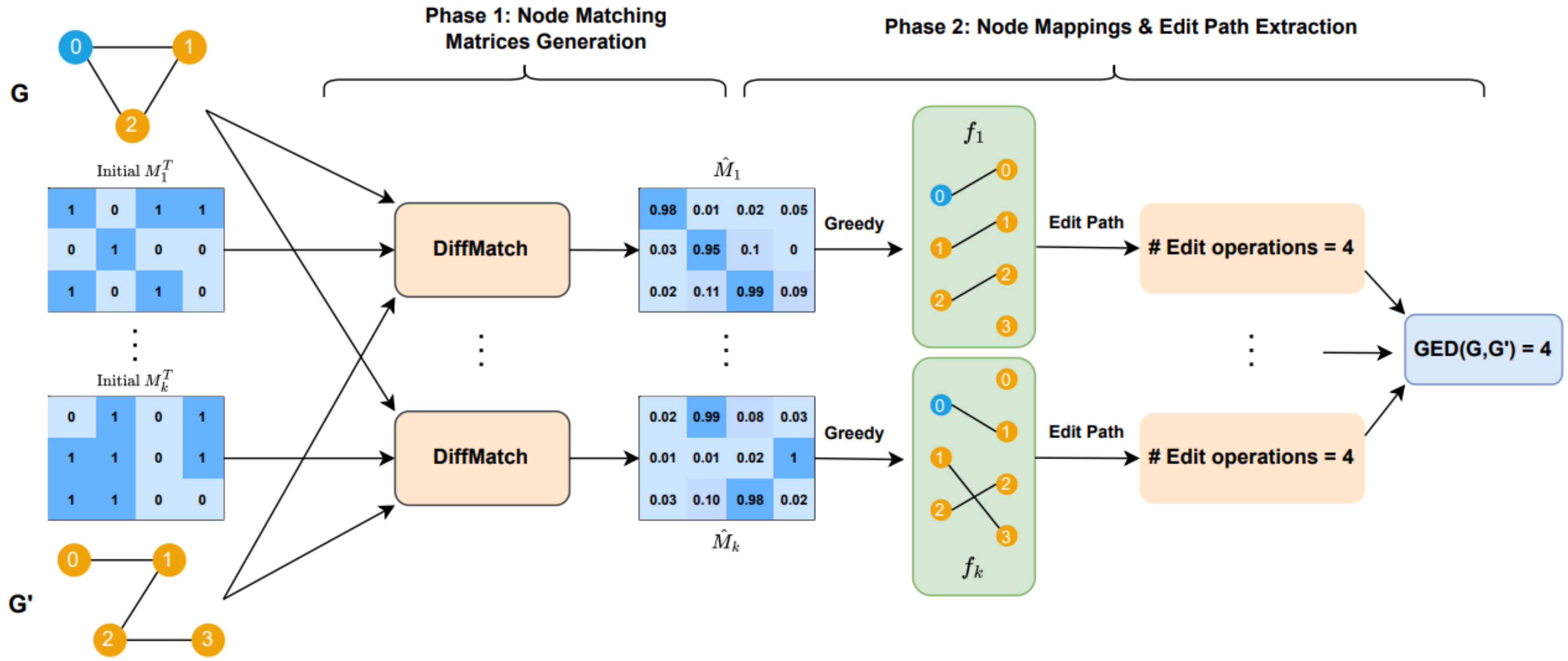


Huang, Wei, et al. "DiffGED: Computing Graph Edit Distance via Diffusion-based Graph Matching." *arXiv preprint arXiv:2503.18245* (2025).

Machine Learning Models for Graph Edit Distance

DiffGED: Computing Graph Edit Distance via Diffusion-based Graph Matching

In the first phase, DiffGED first samples k random initial node matching matrices, then DiffMatch will denoise the sampled node matching matrices.
In the second phase, one node mapping will be extracted from each node matching matrix in parallel, and edit paths will be derived from the node mappings.



Huang, Wei, et al. "DiffGED: Computing Graph Edit Distance via Diffusion-based Graph Matching." *arXiv preprint arXiv:2503.18245* (2025).

Machine Learning Models for Graph Edit Distance

DiffGED: Computing Graph Edit Distance via Diffusion-based Graph Matching

Experimental results: achieve state-of-the-art performance with nearly 100% accuracy.

Datasets	Models	MAE	Accuracy	ρ	τ	p@10	p@20	Time(s)
AIDS700	Hungarian	8.247	1.1%	0.547	0.431	52.8%	59.9%	0.00011
	VJ	14.085	0.6%	0.372	0.284	41.9%	52%	0.00017
	Noah	3.057	6.6%	0.751	0.629	74.1%	76.9%	0.6158
	GENN-A*	0.632	61.5%	0.903	0.815	85.6%	88%	2.98919
	GEDGNN	1.098	52.5%	0.845	0.752	89.1%	88.3%	0.39448
	MATA*	0.838	58.7%	0.8	0.718	73.6%	77.6%	**0.00487**
	DiffGED (ours)	**0.022**	**98%**	**0.996**	**0.992**	**99.8%**	**99.7%**	0.0763
Linux	Hungarian	5.35	7.4%	0.696	0.605	74.8%	79.6%	0.00009
	VJ	11.123	0.4%	0.594	0.5	72.8%	76%	0.00013
	Noah	1.596	9%	0.9	0.834	92.6%	96%	0.24457
	GENN-A*	0.213	89.4%	0.954	0.905	99.1%	98.1%	0.68176
	GEDGNN	0.094	96.6%	0.979	0.969	98.9%	99.3%	0.12863
	MATA*	0.18	92.3%	0.937	0.893	88.5%	91.8%	**0.00464**
	DiffGED (ours)	**0.0**	**100%**	**1.0**	**1.0**	**100%**	**100%**	0.06982
IMDB	Hungarian	21.673	45.1%	0.778	0.716	83.8%	81.9%	0.0001
	VJ	44.078	26.5%	0.4	0.359	60.1%	62%	0.00038
	Noah	-	-	-	-	-	-	-
	GENN-A*	-	-	-	-	-	-	-
	GEDGNN	2.469	85.5%	0.898	0.879	92.4%	92.1%	0.42428
	MATA*	-	-	-	-	-	-	-
	DiffGED (ours)	**0.937**	**94.6%**	**0.982**	**0.973**	**97.5%**	**98.3%**	**0.15105**

Huang, Wei, et al. "DiffGED: Computing Graph Edit Distance via Diffusion-based Graph Matching." *arXiv preprint arXiv:2503.18245* (2025).

Machine Learning Models for Graph Edit Distance

Towards Unsupervised Training of Matching-based Graph Edit Distance Solver via Preference-aware GAN

Optimization objective of Matching-based GED solver:

- Given a graph pair, find an optimal node matching matrix π^* that **minimizes the edit cost $c(G_1, G_2, \pi^*)$**

Supervised training objective of Matching-based GED solver g_ϕ:

$$\mathcal{L}_{rec(\pi^*)} = \frac{1}{|V_1||V_2|}\sum_{v=1}^{|V_1|}\sum_{u=1}^{|V_2|}(\pi^*[v][u]\log(\hat{\pi}_{g_\phi}[v][u])) + (1-\pi^*[v][u])\log(1-\sigma(\hat{\pi}_{g_\phi}[v][u])))$$

What if ground-truth optimal node matching matrix π^* is not available during training?

- **A naive approach:** Starting from a random node matching matrix $\bar{\pi}$, train g_ϕ to recover $\bar{\pi}$ by $\mathcal{L}_{rec(\bar{\pi})}$, and progressively update $\bar{\pi}$ with the latest best solution predicted by g_ϕ $\longrightarrow$ **Lack of exploration**
- **A better approach:** Not only trained to exploit, but also trained to explore better $\bar{\pi}$ efficiently

$$\mathcal{L}_{g_\phi} = \mathcal{L}_{rec(\bar{\pi})} + \lambda\mathcal{L}_{explore}$$

Huang, Wei, et al. "Towards Unsupervised Training of Matching-based Graph Edit Distance Solver via Preference-aware GAN." arXiv preprint arXiv:2506.01977 (2025).

Machine Learning Models for Graph Edit Distance

Towards Unsupervised Training of Matching-based Graph Edit Distance Solver via Preference-aware GAN

How to explore better solutions? (How to design $\mathcal{L}_{explore}$?)

- **GAN-based approach:** Given a node matching matrix $\hat{\pi}_{g_\theta}$ predicted by g_ϕ, a discriminator D_θ is trained to assign a score $D_\theta(G_1, G_2, \hat{\pi}_{g_\theta})$, and g_ϕ is trained to maximize $D_\theta(G_1, G_2, \hat{\pi}_{g_\theta})$ ⟶ $\mathcal{L}_{explore} = -D_\theta(G_1, G_2, \hat{\pi}_{g_\phi})$

How to train D_θ? What score should D_θ assign?

- **A naive approach:** D_θ is trained to estimate the normalized edit cost $\mathcal{L}_{\mathcal{D}_\theta} = (D_\theta(G_1, G_2, \pi) - \exp(-\frac{c(G_1, G_2, \pi) \times 2}{|V_1| + |V_2|}))^2$
- **Ideally:** g_ϕ is trained to minimize the edit cost ⟶ **aligns with the optimization objective**

But what if the following cases occur?

- π_1 with normalized edit cost = 0.4 & π_2 with normalized edit cost = 0.6 ⟶ **π_2 is better than π_1**
- **Case 1:** $D_\theta(G_1, G_2, \pi_1) = 0.1$ & $D_\theta(G_1, G_2, \pi_2) = 0.9$ ⟶ $\mathcal{L}_{D_\theta} = (0.1 - 0.4)^2 + (0.9 - 0.6)^2 = 0.18$
- **Case 2:** $D_\theta(G_1, G_2, \pi_1) = 0.6$ & $D_\theta(G_1, G_2, \pi_2) = 0.4$ ⟶ $\mathcal{L}_{D_\theta} = (0.6 - 0.4)^2 + (0.4 - 0.6)^2 = 0.08$
- Case 2 results in lower $\mathcal{L}_{D_\theta}$ ⟶ D_θ prefers Case 2 ⟶ g_ϕ prefers π_1 ✖ **π_2 should be preferred!**

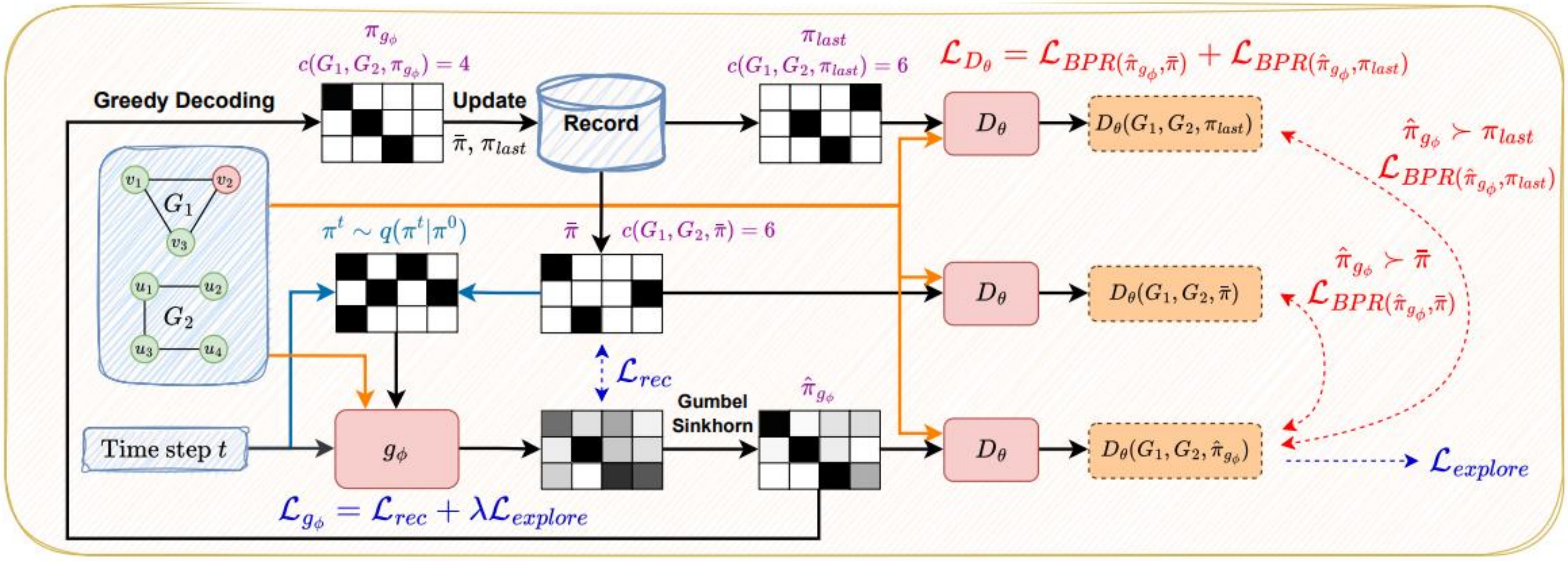
Machine Learning Models for Graph Edit Distance

Towards Unsupervised Training of Matching-based Graph Edit Distance Solver via Preference-aware GAN

How to train D_θ? What score should D_θ assign?

- **Preference optimization:** if π_2 is preferred to ($\succ$) π_1, then π_2 should be assigned a higher score than π_1
- D_θ is trained to **maximize the score margin** by minimizing the Bayes Personalized Ranking loss:

$$\mathcal{L}_{BPR(\pi_1,\pi_2)} = -\log(\sigma(D_\theta(G_1, G_2, \pi_2) - D_\theta(G_1, G_2, \pi_1)))$$



Machine Learning Models for Graph Edit Distance

Towards Unsupervised Training of Matching-based Graph Edit Distance Solver via Preference-aware GAN

Experimental results: The matching-based GED solver trained with **unsupervised** preference-aware GAN achieved performance comparable to that under supervised learning.

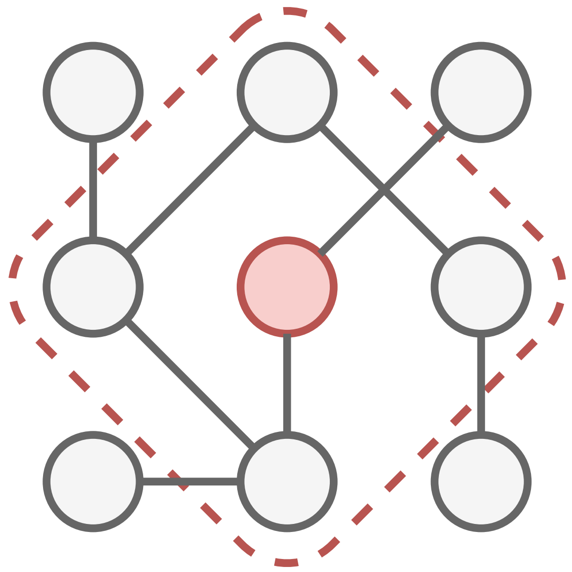
Datasets	Models	Type	MAE ↓	Accuracy ↑	ρ ↑	τ ↑	p@10 ↑	p@20 ↑	Time(s) ↓
AIDS700	Hungarian	Trad	8.247	1.1%	0.547	0.431	52.8%	59.9%	**0.00011**
	VJ	Trad	14.085	0.6%	0.372	0.284	41.9%	52%	0.00017
	GEDGW	Trad	**0.811**	**53.9%**	**0.866**	**0.78**	**84.9%**	**85.7%**	0.39255
	Noah	SL	3.057	6.6%	0.751	0.629	74.1%	76.9%	0.6158
	GENN-A*	SL	0.632	61.5%	0.903	0.815	85.6%	88%	2.98919
	MATA*	SL	0.838	58.7%	0.8	0.718	73.6%	77.6%	**0.00487**
	GEDGNN	SL	1.098	52.5%	0.845	0.752	89.1%	88.3%	0.39448
	GEDIOT	SL	1.188	53.5%	0.825	0.73	88.6%	86.7%	0.39357
	DiffGED	SL	**0.022**	**98%**	**0.996**	**0.992**	**99.8%**	**99.7%**	0.0763
	GEDRanker (Ours)	UL	**0.088**	**92.6%**	**0.984**	**0.969**	**99.1%**	**99.1%**	**0.0759**
Linux	Hungarian	Trad	5.35	7.4%	0.696	0.605	74.8%	79.6%	**0.00009**
	VJ	Trad	11.123	0.4%	0.594	0.5	72.8%	76%	0.00013
	GEDGW	Trad	**0.532**	**75.4%**	**0.919**	**0.864**	**90.5%**	**92.2%**	0.1826
	Noah	SL	1.596	9%	0.9	0.834	92.6%	96%	0.24457
	GENN-A*	SL	0.213	89.4%	0.954	0.905	99.1%	98.1%	0.68176
	MATA*	SL	0.18	92.3%	0.937	0.893	88.5%	91.8%	**0.00464**
	GEDGNN	SL	0.094	96.6%	0.979	0.969	98.9%	99.3%	0.12863
	GEDIOT	SL	0.117	95.3%	0.978	0.966	98.8%	99%	0.13535
	DiffGED	SL	**0.0**	**100%**	**1.0**	**1.0**	**100%**	**100%**	0.06982
	GEDRanker (Ours)	UL	**0.01**	**99.5%**	**0.997**	**0.995**	**100%**	**99.8%**	**0.06973**
IMDB	Hungarian	Trad	21.673	45.1%	0.778	0.716	83.8%	81.9%	**0.0001**
	VJ	Trad	44.078	26.5%	0.4	0.359	60.1%	62%	0.00038
	GEDGW	Trad	**0.349**	**93.9%**	**0.966**	**0.953**	**99.1%**	**98.3%**	0.37496
	Noah	SL	-	-	-	-	-	-	-
	GENN-A*	SL	-	-	-	-	-	-	-
	MATA*	SL	-	-	-	-	-	-	-
	GEDGNN	SL	2.469	85.5%	0.898	0.879	92.4%	92.1%	0.42428
	GEDIOT	SL	2.822	84.5%	0.9	0.878	92.3%	92.7%	0.41959
	DiffGED	SL	**0.937**	**94.6%**	**0.982**	**0.973**	**97.5%**	**98.3%**	**0.15105**
	GEDRanker (Ours)	UL	**1.019**	**94%**	**0.999**	**0.97**	**96.1%**	**97%**	**0.15111**

Graph Query Processing

- Subgraph Isomorphism
  - Subgraph Matching
  - Subgraph Counting
- Graph Similarity
  - Graph Edit Distance
- Community Search
  - **Disjoint Community Search**
  - Overlapping Community Search

Community Search

- Definition: **Community search** (CS) is defined as the task of finding a cohesive subgraph that contains a given set of query nodes, emphasizing query-driven discovery of structurally close and well-connected communities within a larger graph.
- A query set contains one or more nodes that belong to the same community.
- We have disjoint community search and overlapping community search, depending on whether a node can only belong to one community.



Community Search

Disjoint Community Search: ALICE

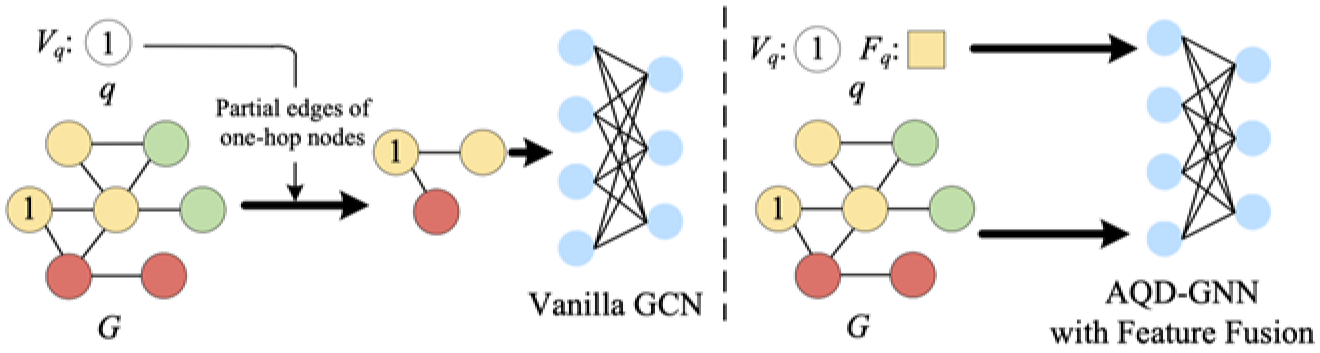
Motivation

- **Existing non-learning methods:**
  - ➢ k-core based ACS model
  - ➢ k-truss based ACS model

Structure Inflexibility

Attribute Irrelevance

- **Existing learning-based methods:**



(a) ICS-GNN (one iteration)
(b) AQD-GNN

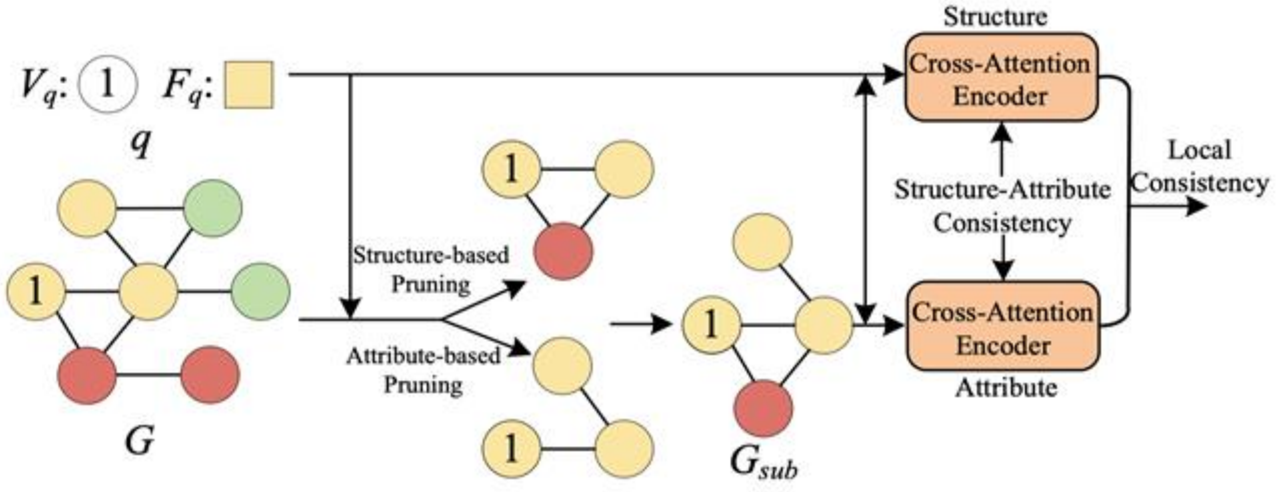
Efficiency and scalability issue for AQD-GNN

Interdependence among entities

Wang, Jianwei, et al. "Neural attributed community search at billion scale." Proceedings of the ACM on Management of Data 1.4 (2024): 1-25.

Disjoint Community Search: ALICE

Our Method



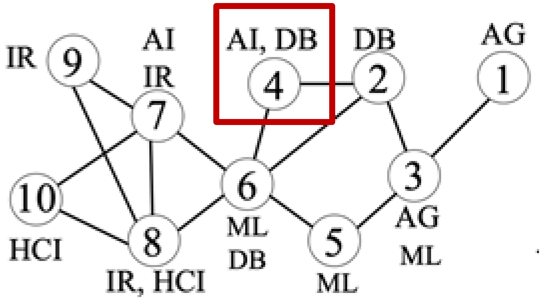
- **Candidate Subgraph Extraction**
  - ✔ Structure-based pruning with density sketch modularity
  - ✔ Attribute-based pruning
- **Consistency-aware Net (ConNet):**
  - ✔ Cross-Attention Encoder
  - ✔ Structure-Attribute Consistency & Local Consistency

Disjoint Community Search: ALICE

Candidate Subgraph Extraction

- **Structure-based pruning**

✓ k-hop neighborhood with largest density sketch modularity



➢1-hop DSM: 0.504

➢2-hop DSM: −0.094

➢3-hop DSM: 0.0

- **Attribute-based pruning:**

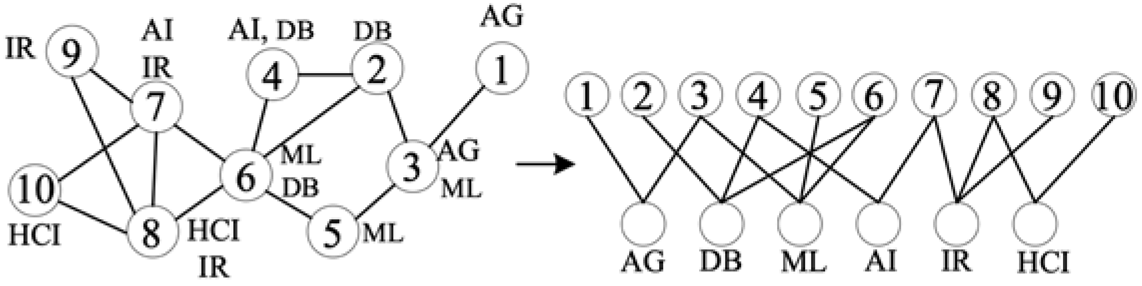


Figure 4: node-attribute bipartite graph

✓ k-hop neighborhood with largest bipartite modularity in the node-attribute bipartite graph

Wang, Jianwei, et al. "Neural attributed community search at billion scale." Proceedings of the ACM on Management of Data 1.4 (2024): 1-25.

Disjoint Community Search: ALICE

ConNet Architecture

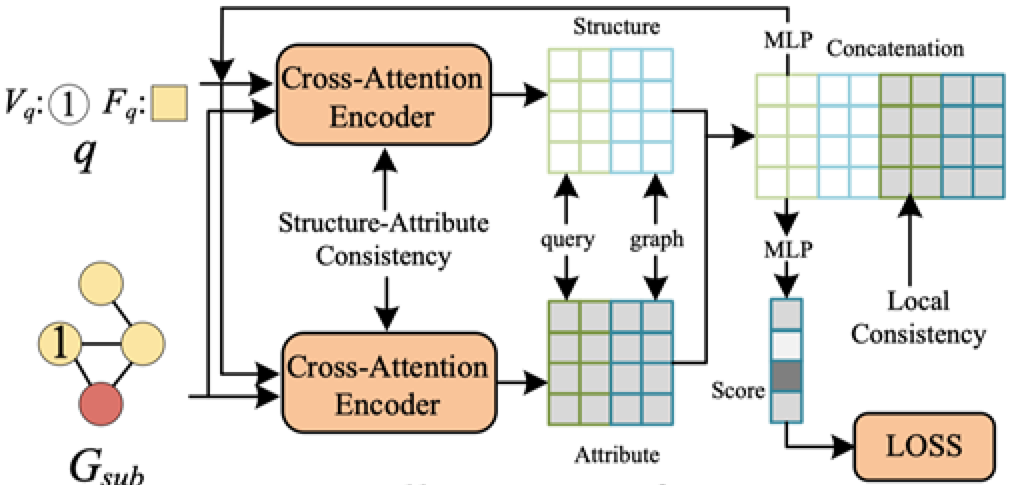


Figure 5: Illustration of *ConNet*

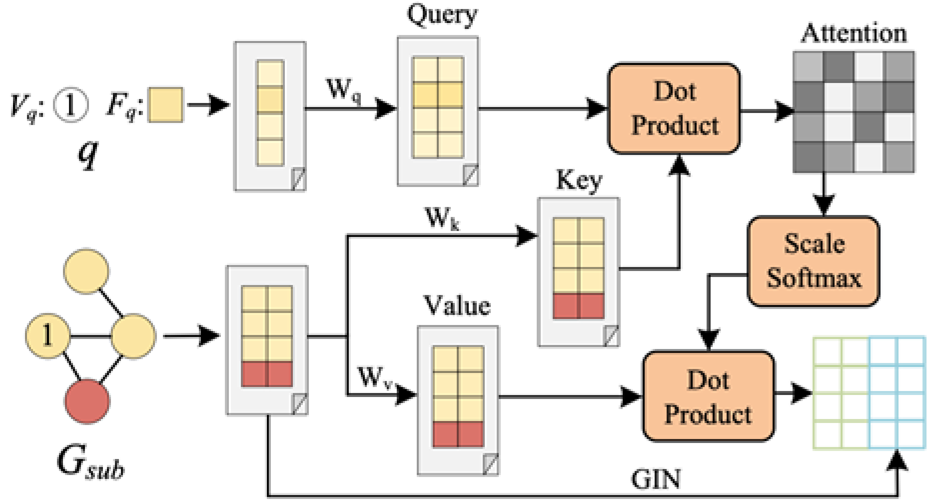


Figure 6: Illustration of Cross Attention Encoder

- Query Encoding $$X_q = H_{v_q}^{(k)} W_q^{(s,k)},\ X_k = H^{(s,k)} W_k^{(s,k)},\ X_v = H^{(s,k)} W_v^{(s,k)}$$ $$X = \text{softmax}\left(\frac{X_q X_k^T}{\sqrt{d_{k+1}}}\right),\ H_{v_q}^{(k+1)} = X X_v$$
- Graph Encoding $$h_v^{(s,k+1)} = \text{MLP}^{(s,k)}\left(1 + \epsilon^{(k)}\right) \cdot h_v^{(s,k)} + \sum_{v' \in N(v)} h_{v'}^{(s,k)}$$
- Lemma: ConNet is as powerful as the 1-WL algorithm.

Disjoint Community Search: TransZero

Motivation

- **Existing non-learning methods:**
  - ➢ *k*-core based CS model
  - ➢ *k*-truss based CS model
  - ➢ *k*-ECC based CS model

✔ Label Free
✖ Structure Flexibility

✔ Label Free
✔ Structure Flexibility

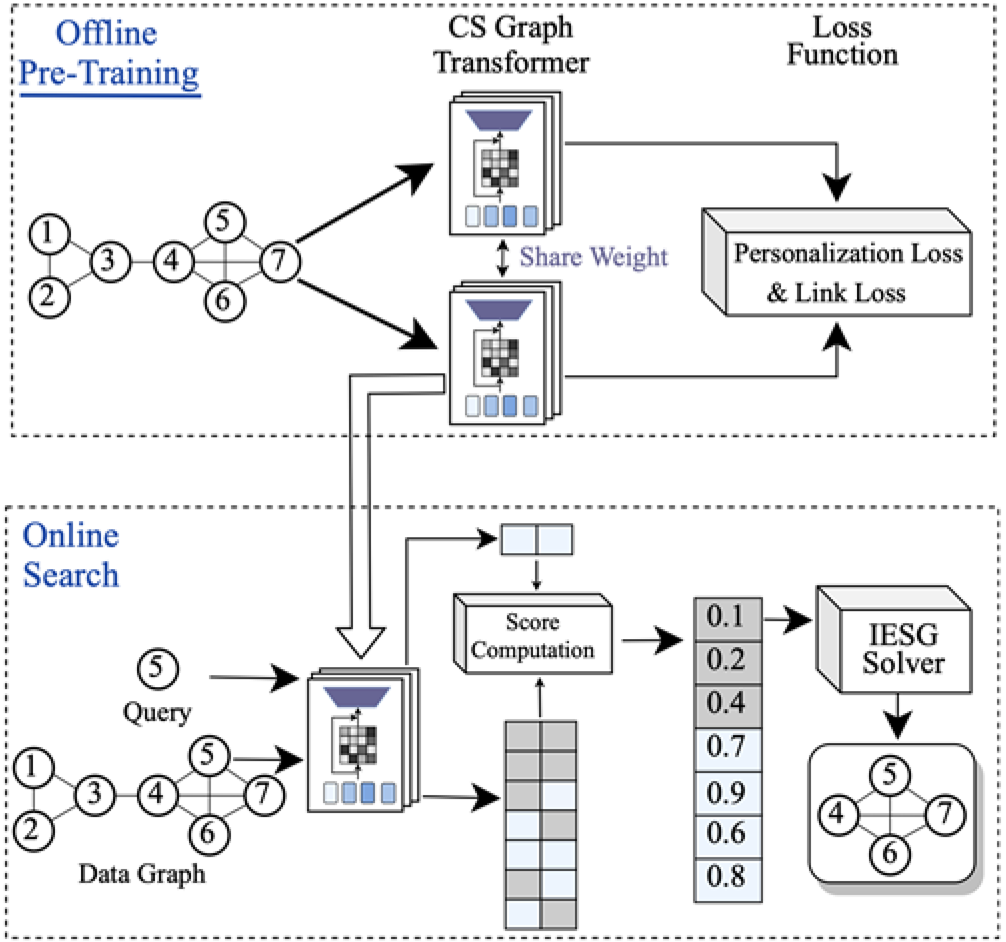
- **Existing learning-based methods:**
  - ➢ QD-GNN
  - ➢ COCELP

✖ Label Free
✔ Structure Flexibility

Wang, Jianwei, et al. "Efficient Unsupervised Community Search with Pre-Trained Graph Transformer." Proceedings of the VLDB Endowment 17.9 (2024): 2227-2240.

Disjoint Community Search: TransZero

Our Method



- Two-stage framework:
  - Offline pre-training and Online search
- Unsupervised community score learning:
  - Offline pre-training with CSGphormer
  - && Online score computation via similarity
- Unsupervised community identification:
  - Identification with Expected Score Gain
  - && Local Search && Global Search

Wang, Jianwei, et al. "Efficient Unsupervised Community Search with Pre-Trained Graph Transformer." Proceedings of the VLDB Endowment 17.9 (2024): 2227-2240.

Disjoint Community Search: TransZero

Offline Pre-training: Augmented Subgraph Sampler

DEFINITION 2. *(Conductance [6, 46]). Given a graph $G(V, E)$ and a community C, the conductance of C is defined as:*

$$\Phi(G, C) = \frac{|e(C, \overline{C})|}{min(d_C, d_{\overline{C}})} \tag{1}$$

where $\overline{C} = V \backslash C$ is complement of C. $e(C, \overline{C})$ is the edges between nodes in C and nodes in $\overline{C}$. d_C is the sum of degrees of the nodes in C.

- Conductance-based augmented subgraph sampler
- K-hop subgraph with lowest conductance value

Wang, Jianwei, et al. "Efficient Unsupervised Community Search with Pre-Trained Graph Transformer." Proceedings of the VLDB Endowment 17.9 (2024): 2227-2240.

Disjoint Community Search: TransZero

Offline Pre-training: CSGphormer

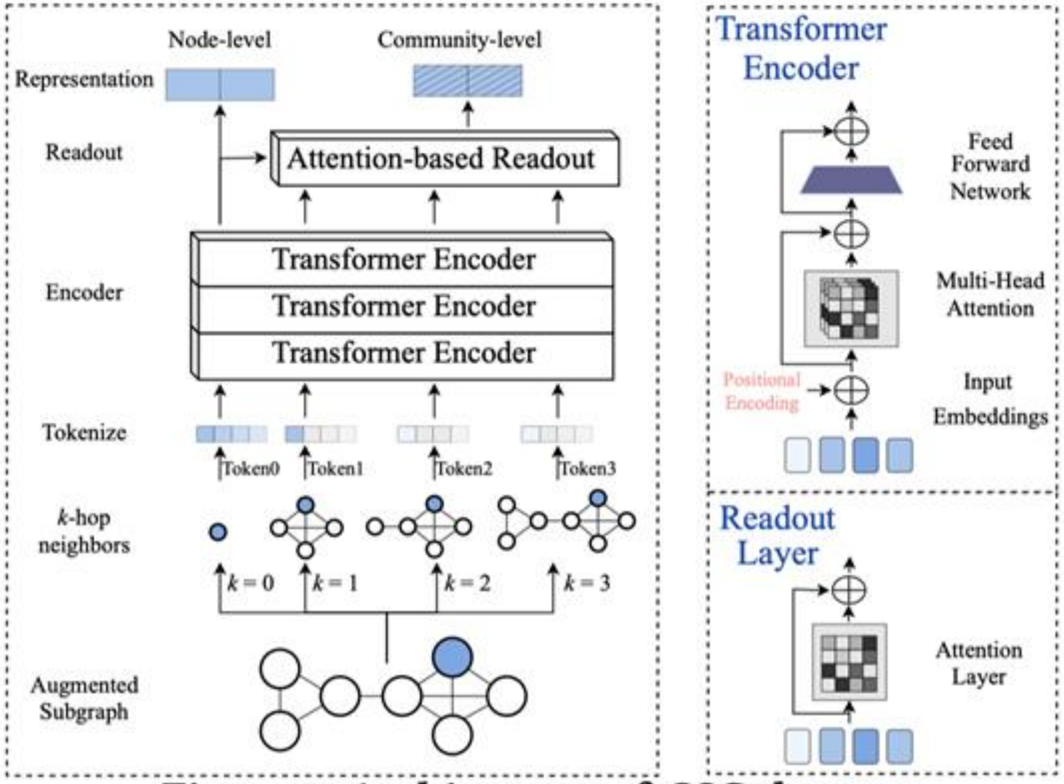


Figure 3: Architecture of *CSGphormer*

Algorithm 1: Forward Propagation of *CSGphormer*.

Input: center node v, feature matrix X, adjacent matrix A, transformer layers L.

Output: The node representation Z_v^{node} and community-level representation Z_v^{com} .

$\mathcal{X}_v \leftarrow \{{}^0x_v, {}^1x_v, \cdots, {}^Kx_v\}$

$H_v^{(0)} \leftarrow \mathcal{X}_v W$

`// L-layers transformer encoder.`

for $l = 0, \cdots, L-1$ **do**

- $P \leftarrow$ Position Encoding Construction
- $H_v^{(l)} \leftarrow H_v^{(l)} + P$
- $H_v^{(l+1)} = \mathrm{MHA}(\mathrm{LN}(H_v^{(l)})) + H_v^{(l)}$
- $H_v^{(l+1)} = \mathrm{FFN}(\mathrm{LN}(H_v^{(l+1)})) + H_v^{(l+1)}$

`// Readout layer.`

$Z_v^{node} \leftarrow {}^0H_v^{(L)}$; $Z_v^{com} \leftarrow$ Zero Tensor

for $k = 1, \cdots, K$ **do**

- $\alpha_k = \dfrac{\exp(({}^0H_v^{(L)} || {}^kH_v^{(L)})W_a^T)}{\sum_{i=1}^{K} \exp(({}^0H_v^{(L)} || {}^iH_v^{(L)})W_a^T)}$
- $Z_v^{com} \leftarrow Z_v^{com} + \alpha_k \, {}^kH_v^{(L)}$

return Z_v^{node}, Z_v^{com}

Disjoint Community Search: TransZero

Online Search: IESG

- Expected Score Gain: $$ESG(S, C, G) = \frac{1}{|V_C|^{\tau}}\left(\sum_{v \in V_C} s_v - \frac{\sum_{u \in V} s_u}{|V|}|V_C|\right)$$

\# of internal nodes

τ is a hyperparameter to control granularity

sum of internal scores

expected score for nodes in the community

- Identification with expected score gain

Definition 4. *(Identification with Expected Score Gain). Given a graph $G(V, E)$, the query V_q, the community score S and a profit function $ESG(\cdot)$, IESG aims to select a community C of G, such that:*

(1) V_C contains nodes in V_q, and C is connected;

(2) $ESG(S, C, G)$ is maximized among all feasible choices for C.

query-driven && cohesive constraint

nodes with high community score

- The problem of IESG is NP-hard

Disjoint Community Search: TransZero

Experiment results

Table 4: F1-score results under different settings

Settings	Models	Texas	Cornell	Wisconsin	Cora	Citeseer	Photo	DBLP	CoCS	Physics	Reddit	Average +/-
Inductive	CST	0.1986	0.1975	0.2251	0.2111	0.1423	0.2019	0.2854	0.1252	0.2276	0.1463	-27.12%
	EquiTruss	0.3120	0.3168	0.3079	0.2384	0.2240	0.2166	0.3252	0.1225	0.2471	0.2163	-21.46%
	M*k*ECS	0.3581	0.3177	0.3404	0.2364	0.2015	0.1975	0.2768	0.1152	0.2193	0.2068	-22.03%
	CTC	0.3211	0.3482	0.3327	0.2558	0.2418	0.2626	0.3417	0.1059	0.2511	0.2431	-19.69%
	QD-GNN	0.0821	0.0669	0.0683	0.0322	0.0536	0.0018	0.0372	0.0145	OOM	OOM	-41.50%
	COCLEP	0.4044	0.2960	0.1804	0.3094	0.3058	0.4413	0.3066	0.4253	0.3389	0.2696	-13.95%
	TransZero-LS	0.1801	0.1583	0.2074	0.5467	0.3906	0.5725	**0.4407**	0.4292	0.5075	**0.4879**	-7.52%
	TransZero-GS	**0.4283**	**0.3716**	**0.3755**	**0.5764**	**0.4535**	**0.6018**	0.4326	**0.4374**	**0.5113**	0.4848	-
Transductive	QD-GNN	0.6703	0.8408	0.6247	0.5062	0.4726	0.2205	0.4918	0.6356	OOM	OOM	+9.81%
	COCLEP	0.4020	0.3167	0.3206	0.3685	0.3331	0.5060	0.3763	0.3549	0.4388	0.3270	-9.29%
Hybrid	QD-GNN	0.3852	0.3644	0.5956	0.4789	0.4097	0.0833	0.3902	0.4969	OOM	OOM	-5.91%
	COCLEP	0.3883	0.3313	0.2938	0.3615	0.3067	0.4388	0.3733	0.4027	0.4693	0.3071	-10.01%

* CST, EquiTruss, M*k*ECS, CTC and *TransZero* have consistent results under three settings as they are label-free. *TransZero* with *Local Search* is denoted as *TransZero*-LS, and *TransZero* with *Global Search* is denoted as *TransZero*-GS. OOM indicates out-of-memory. The last column presents the average margin compared to *TransZero*-GS.

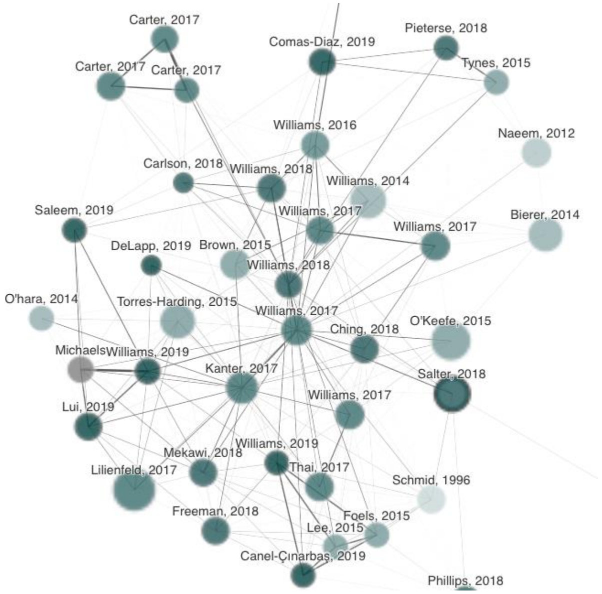
- ✔ TransZero has an outstanding performance, especially under the inductive setting.

Graph Query Processing

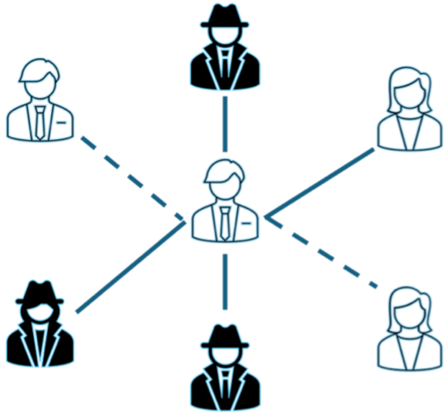
- Subgraph Isomorphism
  - Subgraph Matching
  - Subgraph Counting
- Graph Similarity
  - Graph Edit Distance
- Community Search
  - Disjoint Community Search
  - **Overlapping Community Search**

Overlapping Community Search

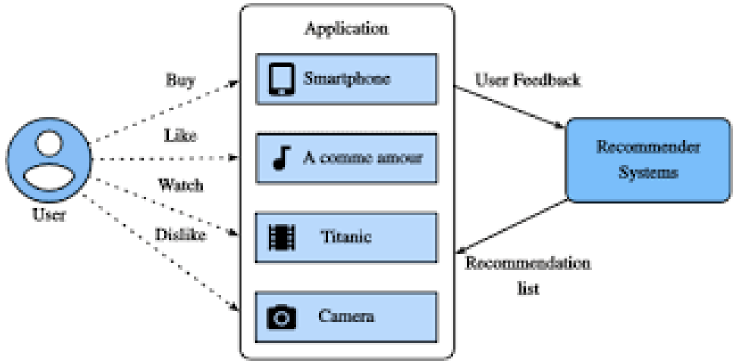
Motivation and Application



(1) Literature Discovery



(2) Fraud Detection



(3) Recommender System

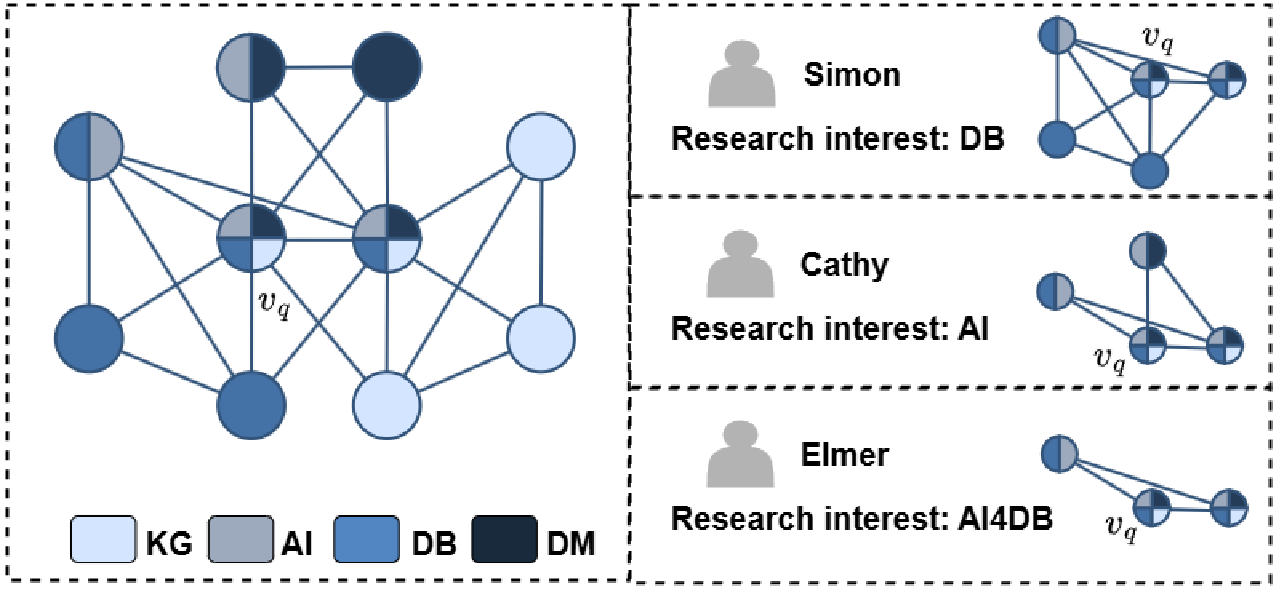
1. Efficiently isolate the most relevant publications within huge citation networks.
2. Accurately detect fraudulent entities hidden in highly imbalanced transactional datasets.
3. Deliver a list of products closely aligned with each user’s interests from extensive catalogues.

Sima, Q., Yu, J., Wang, X., Zhang, W., Zhang, Y., & Lin, X. (2025). Deep overlapping community search via subspace embedding. SIGMOD.

Overlapping Community Search

Background

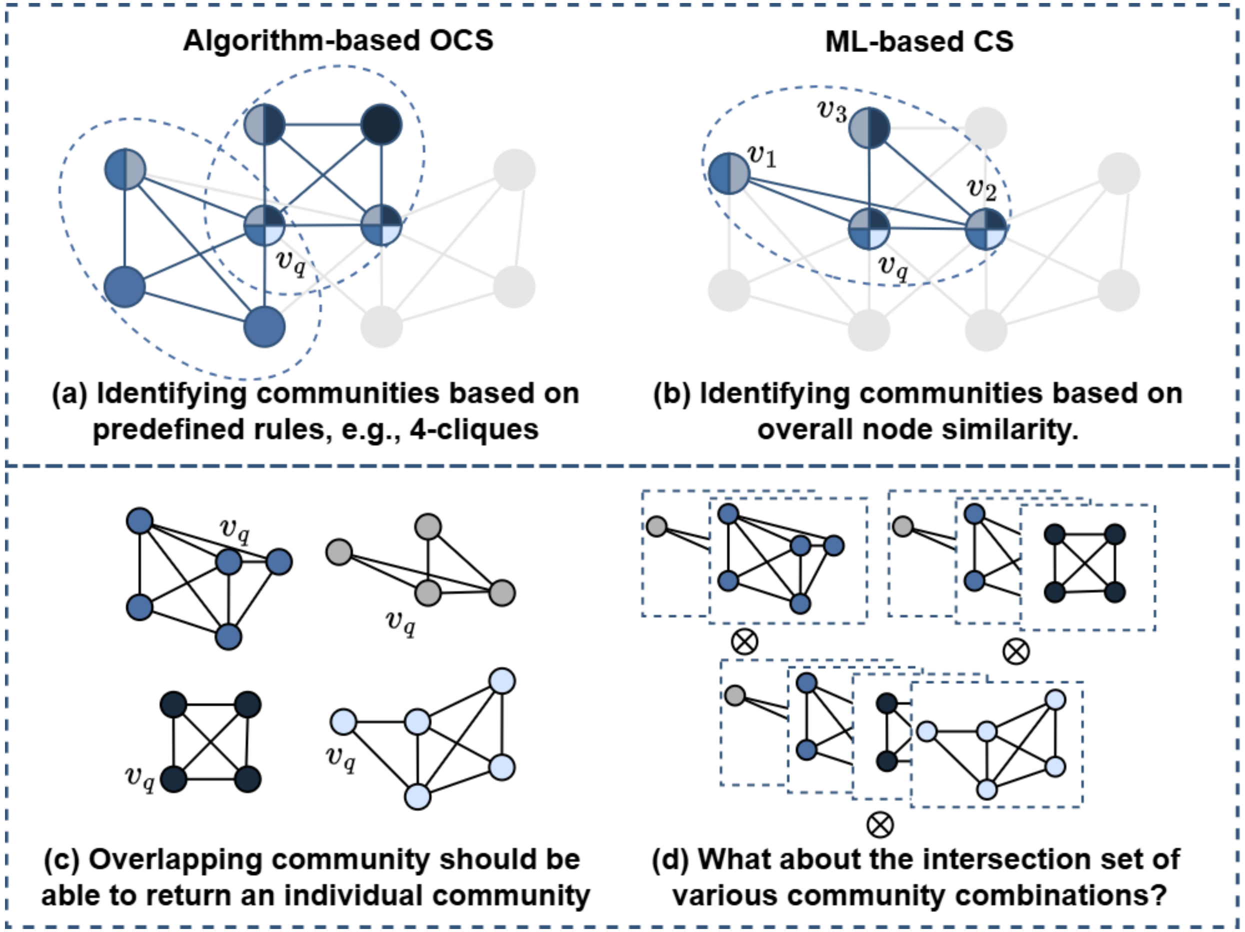
- Nodes is allowed to have multiple community affiliations.
- Colors on nodes represent community label, where nodes have multiple colors means they belongs to different communities.
- Each community exhibiting distinct characteristics such as sizes, levels of cohesiveness, and attribute patterns.
- Given the same query node, different users may seek different communities.



Sima, Q., Yu, J., Wang, X., Zhang, W., Zhang, Y., & Lin, X. (2025). Deep overlapping community search via subspace embedding. SIGMOD.

Overlapping Community Search

Existing Solutions and Research Gaps



Overlapping Community Search (OCS)

- **Algorithm-based:** Popular algorithm-based approaches use different structural constraints, such as k-core, k-truss, and k-clique (Example(a)).
- **Machine learning-based:** ML-based community search models are task orientated and identify communities by prior knowledge learned from ground truth labels (Example(b)).

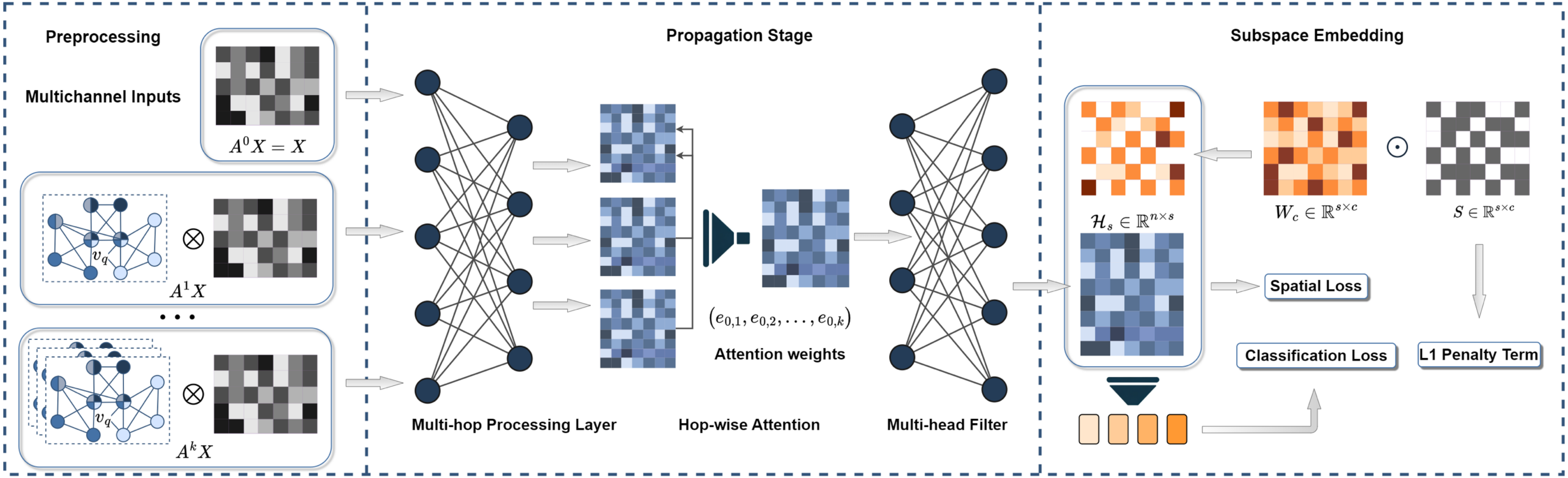
Research Gaps

- ❑ Both approach failed to isolate a 'pure' community according to user specified requirement.
- ❑ User should allow to select multiple targeted communities and only search for the intersect set.

Sima, Q., Yu, J., Wang, X., Zhang, W., Zhang, Y., & Lin, X. (2025). Deep overlapping community search via subspace embedding. SIGMOD.

Overlapping Community Search

Efficient and effective model structure - SMN



Framework of Simplified Multi-hop Attention Networks (SMN)

Sima, Q., Yu, J., Wang, X., Zhang, W., Zhang, Y., & Lin, X. (2025). Deep overlapping community search via subspace embedding. SIGMOD.

Overlapping Community Search

Simplified Multi-hop Attention Network - SMN

- **Aggregation:** Inspired by SGC [2], we removes the non-linear activation functions during aggregation to improve the model training speed. This simplified model structure reduce the model training complexity to a multilayer perceptron levels, which significantly increase the model training efficiency. Then, we generate multi-hop features channel during preprocessing stage such as $\boldsymbol{X}, \widehat{\boldsymbol{A}}^{\mathbf{1}}\boldsymbol{X}, \ldots \widehat{\boldsymbol{A}}^{\boldsymbol{k}}\boldsymbol{X}$.
-
- **Normalization:** Removing the self-loop reduces redundancy in message passing and differentiates messages from each hop.

- **Propagation**: Hop-wise attention is adopted to propagate and fuse the embeddings learned from different hops as:

$$\begin{aligned} \boldsymbol{e_i} &= a(\mathbf{W}^l \boldsymbol{H}_0, \mathbf{W}^l \boldsymbol{H}_i), \quad \forall i \in [0..k] \\ &= (\overrightarrow{\boldsymbol{a}}^T \mathbf{W}^l \boldsymbol{H}_0 + \overrightarrow{\boldsymbol{a}}^T \mathbf{W}^l \boldsymbol{H}_i), \end{aligned}$$

$$\boldsymbol{\alpha}_i = \frac{\exp\left(\mathrm{LeakyReLU}\left(\boldsymbol{e_i}\right)\right)}{\sum_{j\in[0..k]} \exp\left(\mathrm{LeakyReLU}\left(\boldsymbol{e_j}\right)\right)}.$$

$$\mathcal{H}_s = \sigma\left(\mathbf{W}^r \left(\Vert_{i=1}^{I} \sum_{k=0}^{K} \boldsymbol{\alpha}_k^i \mathbf{W}^l \boldsymbol{H}_k\right)\right),$$

Sima, Q., Yu, J., Wang, X., Zhang, W., Zhang, Y., & Lin, X. (2025). Deep overlapping community search via subspace embedding. SIGMOD.

Overlapping Community Search

Effectiveness OCS & OCIS

Table 2: SMN performance in overlapping community search

	Task	Overlapping Community Search, OCS								Overlapping Communities Intersection Search, OCIS								
Metric	Model	k-clique	CTC	k-core	ICS GNN	QD GNN	COC LEP	SMN Topk	SMN CS	k-clique	CTC	k-core	ICS GNN	QD GNN	COC LEP	SMN Topk	SMN CS	Ave+
F1	FB0	0.2478	0.2588	0.2423	0.7058	0.6710	0.2424	0.7427	**0.7630**	0.0572	0.0622	0.0551	0.6122	0.5982	0.6667	0.6547	**0.7147**	35%
	FB107	0.2781	0.3024	0.2537	0.6835	0.6361	-	0.9035	**0.9103**	0.0712	0.0829	0.0609	0.5127	0.5760	-	**0.7520**	0.6520	46%
	FB348	0.1543	0.1366	0.1443	0.8041	0.7338	0.6907	**0.8517**	0.7913	0.0916	0.0949	0.0840	0.7508	0.7316	0.6822	**0.8114**	0.8031	39%
	FB414	0.2882	0.3119	0.2718	0.7941	0.6923	0.7286	0.8745	**0.9006**	0.0798	0.0907	0.0681	0.4107	0.4813	0.2080	0.7493	**0.7533**	45%
	FB686	0.0947	0.0881	0.1013	0.6366	0.6006	0.6512	0.6776	**0.7075**	0.0691	0.0825	0.0615	0.4077	0.4351	0.4201	0.4958	**0.5966**	32%
	ENG	0.0471	0.0529	0.0553	0.6680	0.7422	0.1530	**0.8172**	0.7618	0.0471	0.0529	0.0553	0.6406	0.6792	0.1659	**0.8096**	0.7973	52%
	CS	0.0395	0.0433	0.0408	0.6187	0.5878	0.1400	**0.8301**	0.8242	0.0395	0.0433	0.0408	0.6426	0.6472	0.1507	0.7383	**0.7504**	53%
	CHEM	0.0594	0.0615	0.0623	0.5732	0.6151	0.1812	**0.8585**	0.8487	0.0594	0.0615	0.0623	0.6047	0.6940	0.2199	0.8734	**0.8758**	59%
	MED	-	0.0503	0.0622	0.6630	0.5704	0.1628	0.8416	**0.8540**	-	0.0503	0.0622	0.6760	0.6927	0.1651	0.8405	**0.8514**	53%
JAC	FB0	0.1972	0.2115	0.1903	0.5446	0.5049	0.1379	0.5907	**0.6168**	0.0559	0.0609	0.0538	0.6022	0.5811	0.5172	0.6500	**0.7080**	34%
	FB107	0.2386	0.2768	0.2048	0.5192	0.4664	-	0.8783	**0.8913**	0.0709	0.0827	0.0606	0.5113	0.5760	-	**0.7520**	0.6520	49%
	FB348	0.1116	0.0940	0.1128	0.6724	0.5796	0.5275	**0.7417**	0.6547	0.0874	0.0924	0.0771	0.6649	0.6447	0.5460	**0.7233**	0.7157	36%
	FB414	0.2538	0.2931	0.2294	0.6585	0.5294	0.5731	0.7769	**0.8191**	0.0795	0.0903	0.0673	0.3987	0.4680	0.2080	0.7380	**0.7420**	45%
	FB686	0.0661	0.0599	0.0728	0.4669	0.4292	0.4828	0.5124	**0.5474**	0.0641	0.0796	0.0554	0.3623	0.4002	0.2793	0.4645	**0.5597**	29%
	ENG	0.0260	0.0296	0.0311	0.5015	0.5901	0.0828	**0.6908**	0.6152	0.0260	0.0296	0.0311	0.6259	0.6634	0.0917	**0.7799**	0.7659	49%
	CS	0.0224	0.0249	0.0233	0.4479	0.4162	0.0753	**0.7096**	0.7009	0.0224	0.0249	0.0233	0.6124	0.6244	0.0839	0.7101	**0.7206**	51%
	CHEM	0.0349	0.0363	0.0369	0.4017	0.4442	0.0996	**0.7522**	0.7372	0.0349	0.0363	0.0369	0.5744	0.6728	0.1298	**0.8403**	0.8392	58%
	MED	-	0.0288	0.0368	0.4959	0.3990	0.0886	0.7266	**0.7453**	-	0.0288	0.0368	0.6404	0.6472	0.0933	0.7946	**0.8054**	52%
NMI	FB0	0.1788	0.1245	0.2069	0.1535	0.2007	0.1029	**0.3182**	0.2905	0.1788	0.1245	0.2069	0.2117	0.2021	0.1673	0.5212	**0.5418**	25%
	FB107	0.3790	0.5479	0.2054	0.1590	0.2794	-	**0.6176**	0.5937	0.3790	0.5479	0.2054	0.1554	0.2043	-	**0.6395**	0.6197	31%
	FB348	0.3338	0.4700	0.3321	0.4626	0.4155	0.2345	0.5301	**0.6550**	0.3338	0.3380	0.3321	0.2023	0.1760	0.0771	**0.3829**	0.3582	17%
	FB414	0.3695	0.4281	0.4250	0.4449	0.3914	0.3189	0.5669	**0.6186**	0.3695	0.4281	0.4250	0.3529	0.4286	0.1375	0.6318	**0.6325**	24%
	FB686	0.2862	0.2790	0.2225	0.2047	0.2864	0.1773	**0.4040**	0.3777	0.2862	0.2790	0.2225	0.2474	0.2662	0.2608	**0.4723**	0.4495	17%
	ENG	0.0424	0.0545	0.0687	0.3201	0.4550	0.0325	**0.5803**	0.4810	0.0424	0.0545	0.0687	0.3094	0.4986	0.0333	**0.7696**	0.7590	48%
	CS	-	0.0377	-	0.2936	0.2983	0.0097	0.5954	**0.6033**	-	0.0377	-	0.2985	0.4734	0.0047	0.6891	**0.7043**	47%
	CHEM	0.0393	0.0396	0.0411	0.2745	0.2961	0.0297	**0.6546**	0.6405	0.0393	0.0396	0.0411	0.2636	0.4930	0.0107	**0.7028**	0.6896	54%
	MED	-	0.0556	0.0430	0.3916	0.2744	0.0746	0.6419	**0.6726**	-	0.0556	0.0430	0.3806	0.4606	0.0303	0.6728	**0.6876**	49%

Machine Learning for Graph Data Management and Query Processing

Graph Data Management

Speaker:
Hanchen Wang

Lecturer & ARC DECRA Fellow
Australian Artificial Intelligence Institute,
University of Technology Sydney

Contributors: Hanchen Wang, Ying Zhang and Wenjie Zhang

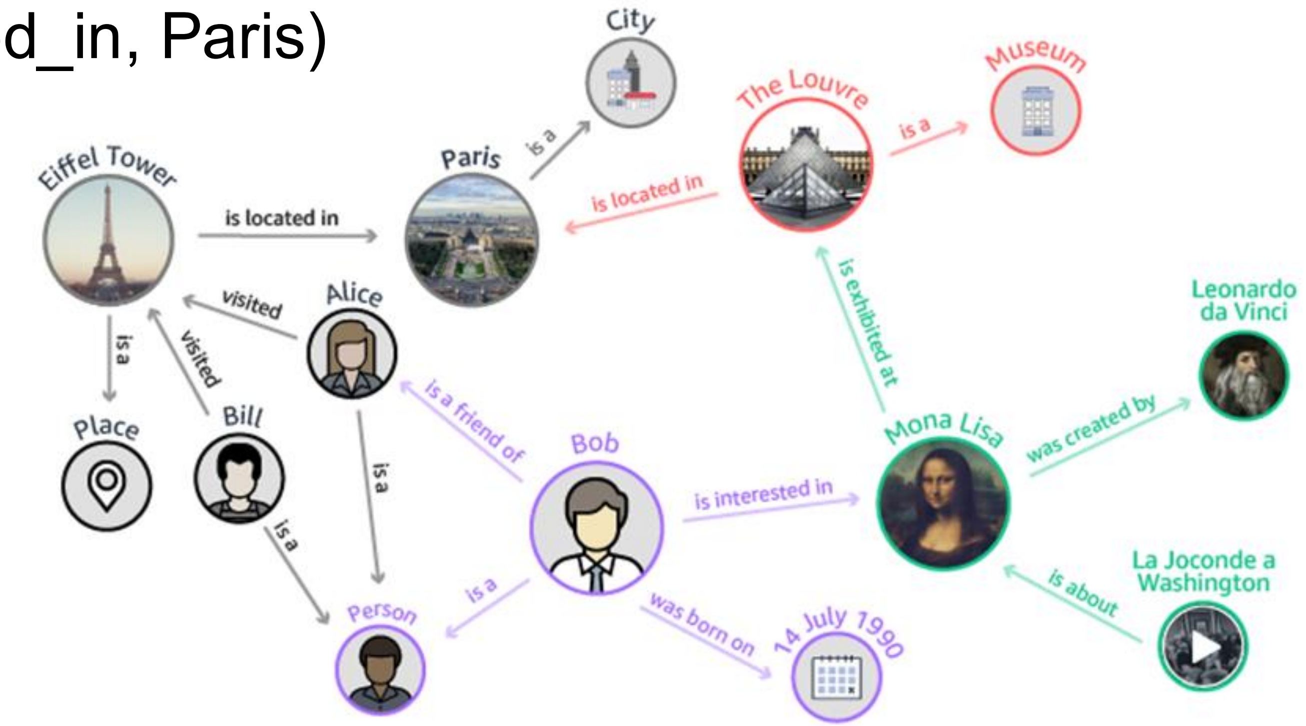
Graph Data Management

- Graph Data Quality Management
  - **Data Quality Assessment**
  - Data Quality Enhancement
- Graph Generation
  - Learning-based Graph Generation
  - Function-driven Graph Generation

Introduction: Knowledge Graphs (KGs)

- Structured, Multi-relational
- $\mathcal{G} = \{\mathcal{E}, \mathcal{R}, \mathcal{F}\}$ A Triple → (Head Entity, Relation, Tail Entity)

  $(h, r, t) \in \mathcal{F}$ $h, t \in \mathcal{E};\ r \in \mathcal{R}$
- e.g. (Eiffel_Tower, is_located_in, Paris)



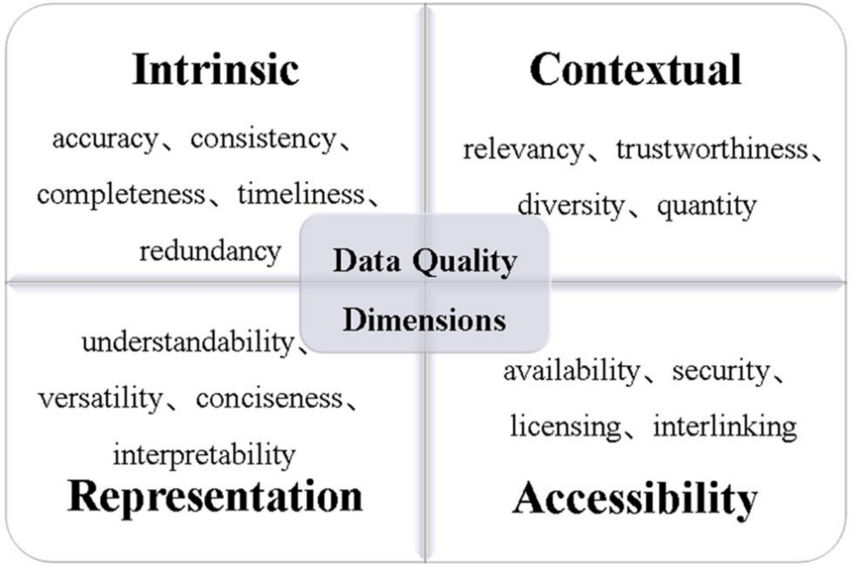
Zhang, Q., Dong, J., Duan, K., Huang, X., Liu, Y., & Xu, L. (2022, October). Contrastive knowledge graph error detection. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (pp. 2590-2599).

Knowledge Graph Quality Management

As a specfic data type, researches on knowledge graph are in the same line with general data type.

Definition

The extent to which data are **fit for a specified use** and **free of defects** with respect to explicit, context-specific criteria.



Dimension

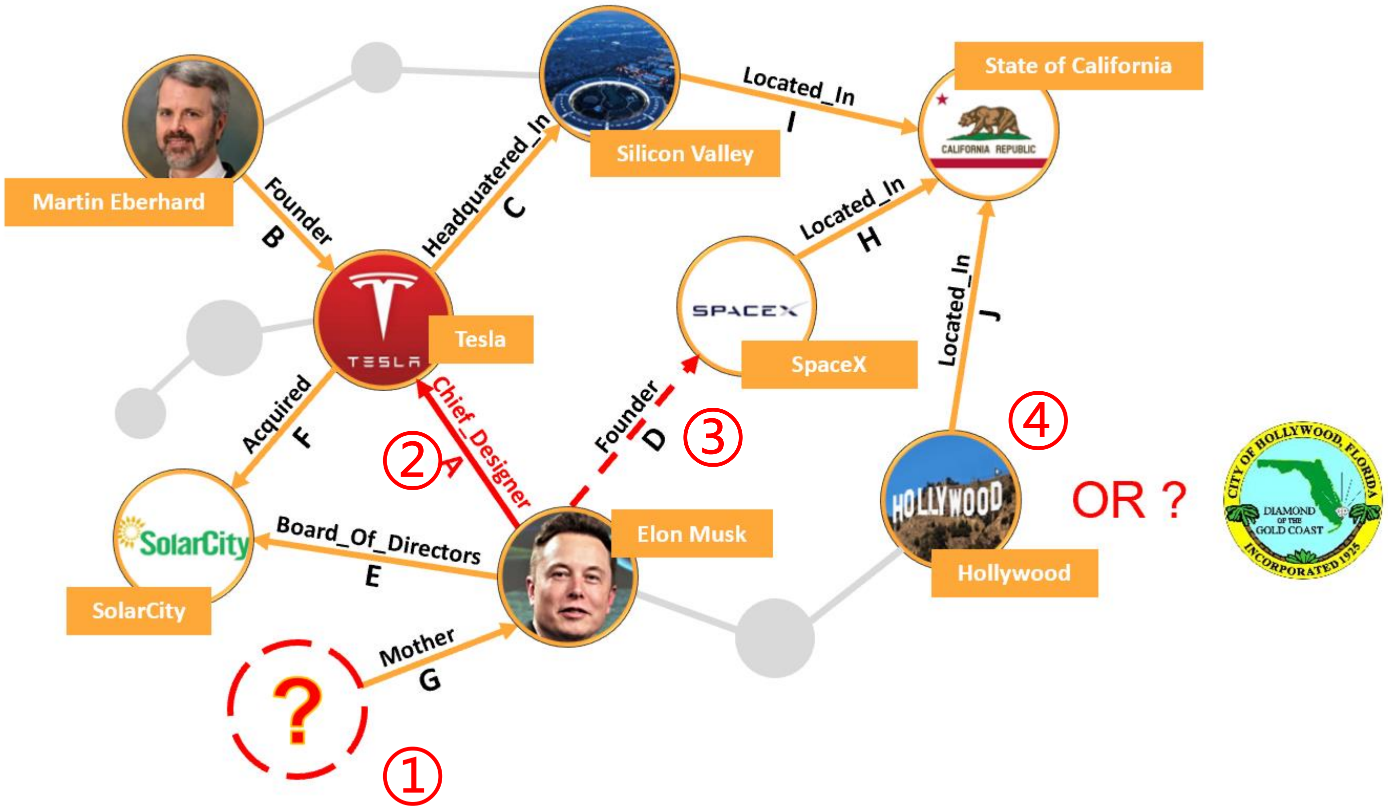
The extent to which data are **fit for a specified use** and **free of defects** with respect to explicit, context-specific criteria.

Lifecycle

a data lifecycle pipeline contains five steps, namely, data generation, information extraction, data integration, analysis, and application.

Xue, B., & Zou, L. (2022). Knowledge graph quality management: A comprehensive survey. *IEEE Transactions on Knowledge and Data Engineering*, *35*(5), 4969-4988.

Challenges for KG Error: Diverse Error Types



① Missing Entity

② Wrong Relation

③ Missing Relation

④ Entity Confusion

……

Unknown Types → Unavailable Labeled errors

Zhang, Q., Dong, J., Duan, K., Huang, X., Liu, Y., & Xu, L. (2022, October). Contrastive knowledge graph error detection. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (pp. 2590-2599).

KG Error Detection

Problem Statement

Given a knowledge graph $\mathcal{G}$ with potential errors

The proposed framework could learn a confidence score for each triple

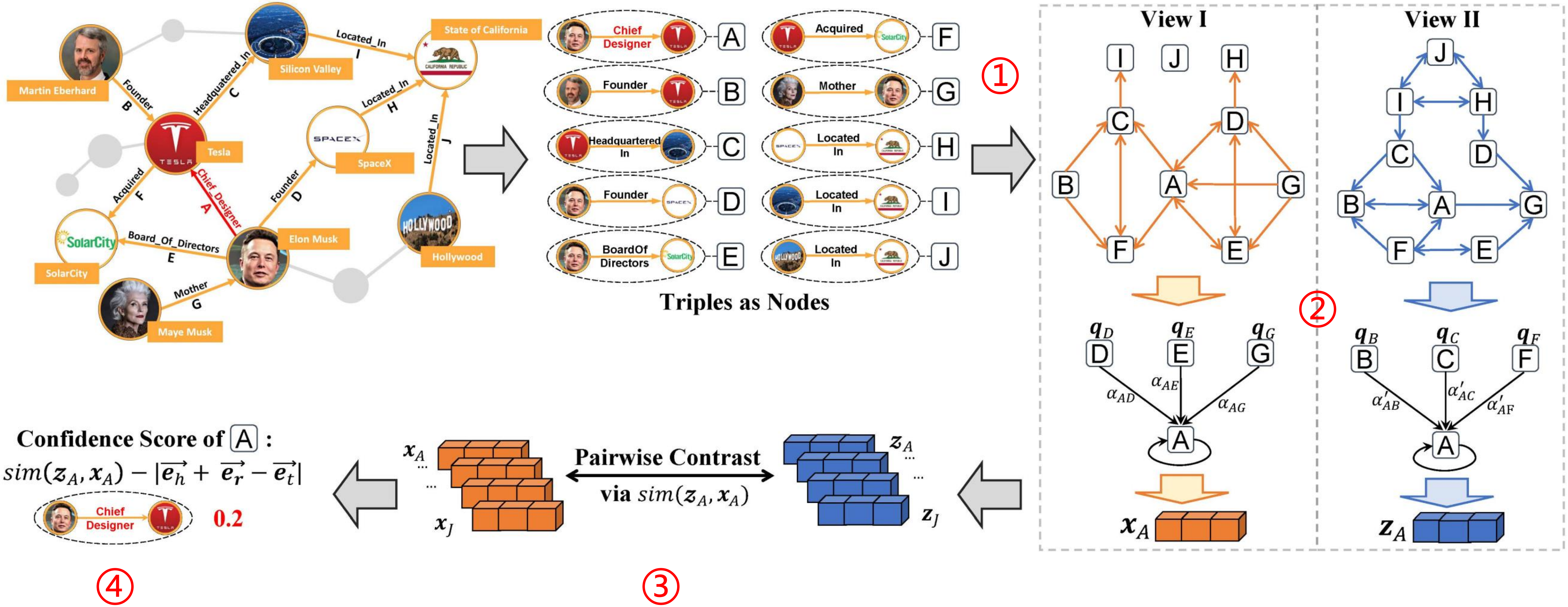
Detecting errors by ranking all the scores

- **How to design an augmentation mechanism for KGs?**
- **How to design a tailored encoder for KGs ?**

Zhang, Q., Dong, J., Duan, K., Huang, X., Liu, Y., & Xu, L. (2022, October). Contrastive knowledge graph error detection. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (pp. 2590-2599).

ContrAstive Knowledge Graph Error Detection (CAGED)

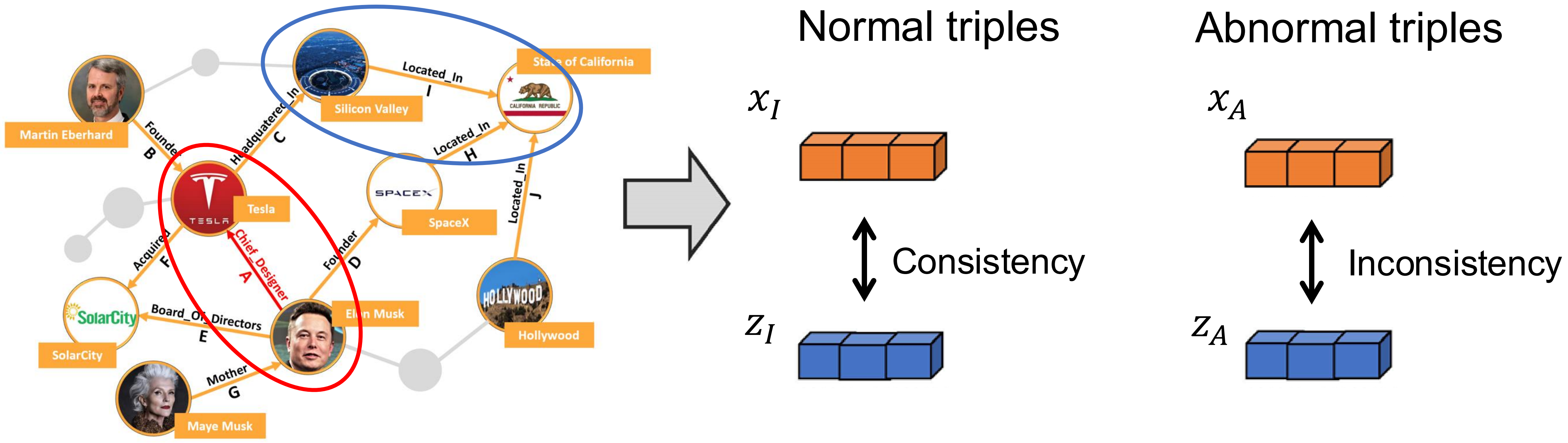
CAGED Model



Zhang, Q., Dong, J., Duan, K., Huang, X., Liu, Y., & Xu, L. (2022, October). Contrastive knowledge graph error detection. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (pp. 2590-2599).

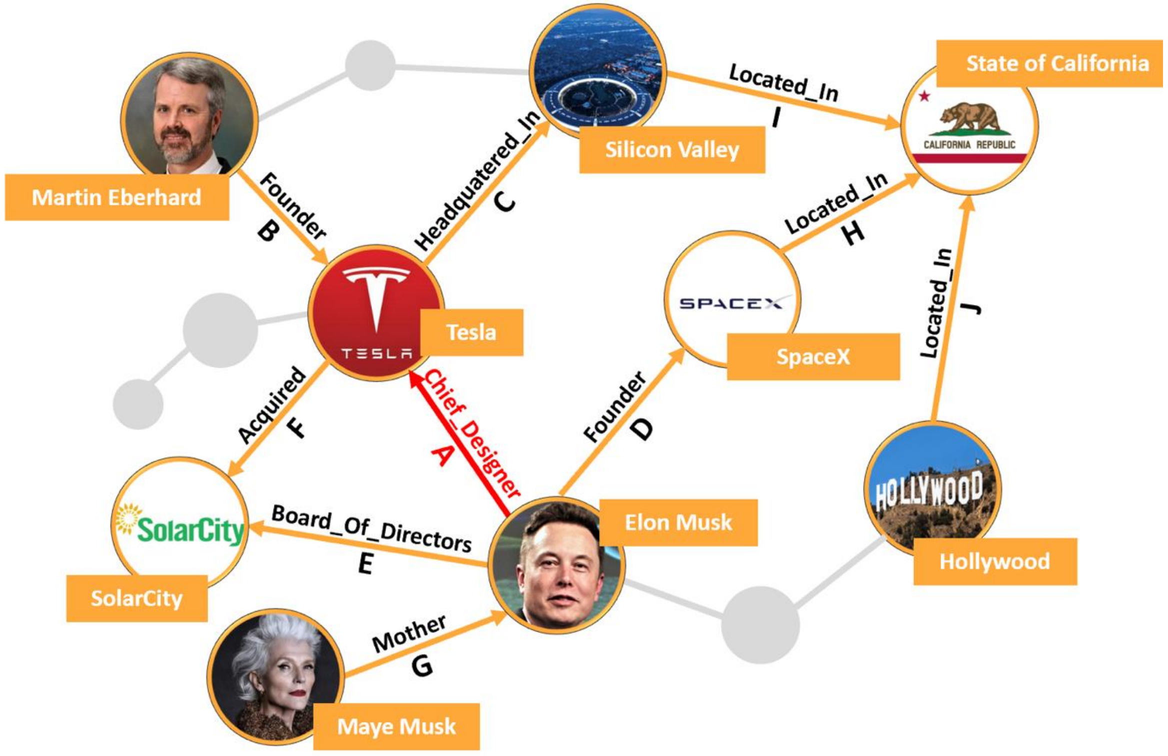
Augmentation Rules

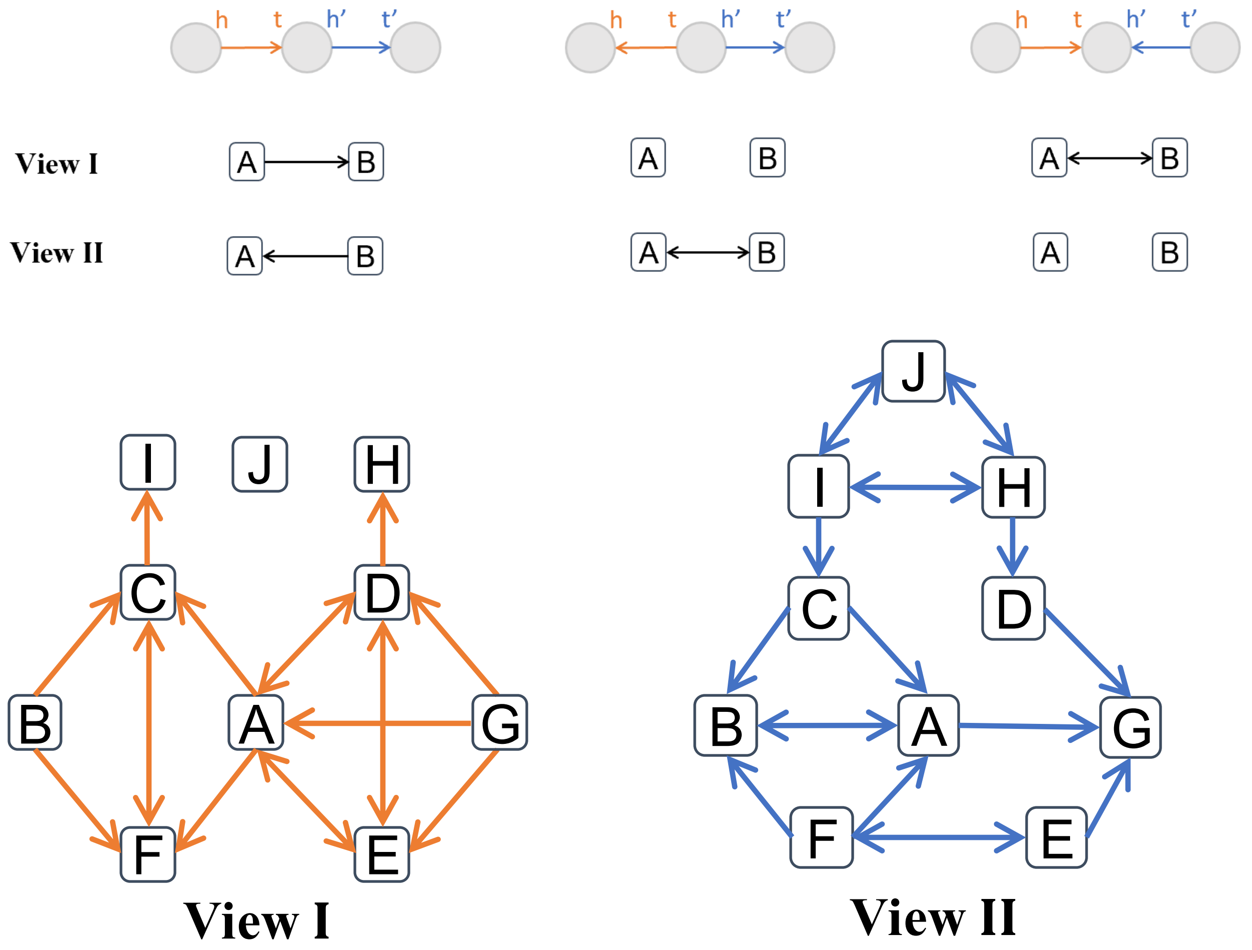
- Augmentation rules are used to generate two views of KG in triple-level.



Zhang, Q., Dong, J., Duan, K., Huang, X., Liu, Y., & Xu, L. (2022, October). Contrastive knowledge graph error detection. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (pp. 2590-2599).

Augmentation Rules

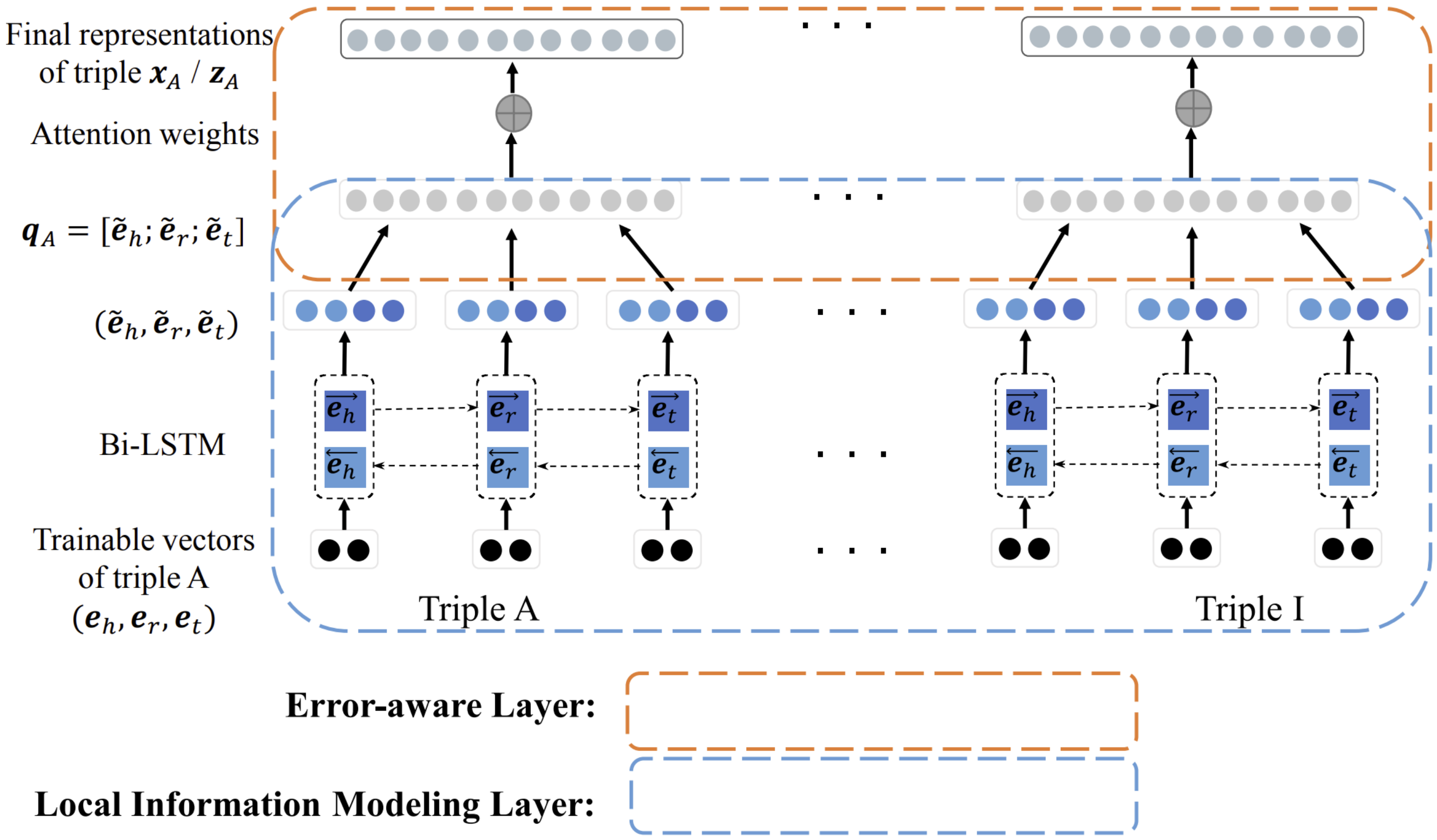




Zhang, Q., Dong, J., Duan, K., Huang, X., Liu, Y., & Xu, L. (2022, October). Contrastive knowledge graph error detection. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (pp. 2590-2599).

EaGNN Encoder

Error-aware Knowledge Graph Neural Network



Zhang, Q., Dong, J., Duan, K., Huang, X., Liu, Y., & Xu, L. (2022, October). Contrastive knowledge graph error detection. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (pp. 2590-2599).

EaGNN Attention Layer

➢ A tailored encoder is required to alleviate the impact of errors.

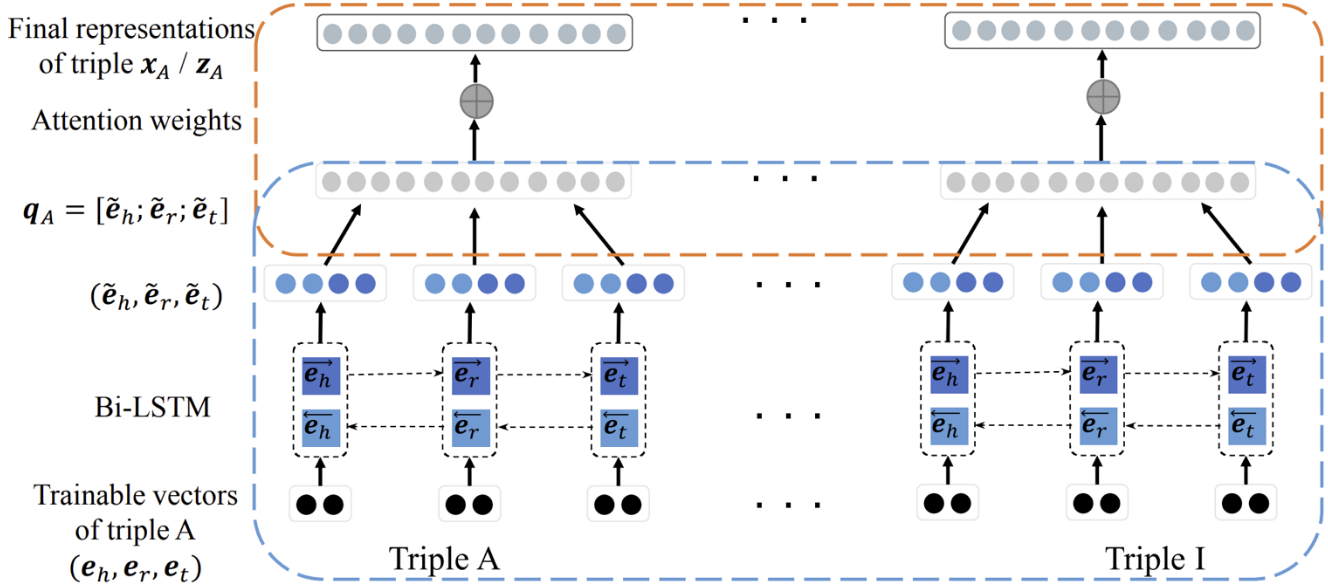
neighbors of $\boldsymbol{q_i}$ $\rightarrow \{\boldsymbol{q_1}, \boldsymbol{q_2}, \ldots, \boldsymbol{q_m}\}$

Attention Coefficient

$$\alpha_{ij} = \mathcal{A}(\mathbf{W}\mathbf{q}_\mathrm{i}, \mathbf{W}\mathbf{q}_\mathrm{j})$$

Attentional Function

$$\alpha_{ij} = \frac{\exp(\alpha_{ij})}{\sum_{k=1}^{m} \exp(\alpha_{ik})}$$ (Softmax Function)



Zhang, Q., Dong, J., Duan, K., Huang, X., Liu, Y., & Xu, L. (2022, October). Contrastive knowledge graph error detection. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (pp. 2590-2599).

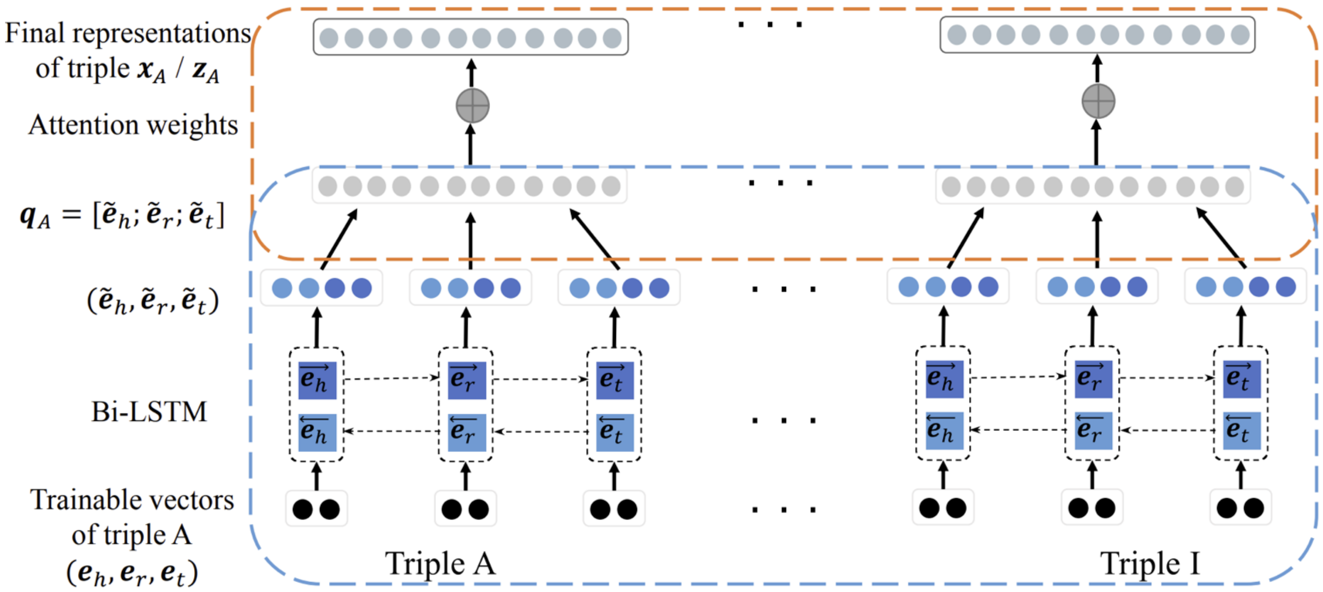
EaGNN Attention Layer

➢ A tailored encoder is required to alleviate the impact of errors.

$$\alpha_{ij} = \begin{cases} \alpha_{ij}, & \alpha_{ij} \geq \mu \\ 0, & \alpha_{ij} < \mu \end{cases}$$

Attention Threshold

$$\begin{cases} \mathbf{x}_i = \sigma\left(\sum_{j=1}^{m} \alpha_{ij}\mathbf{W}\mathbf{q}_j\right) \\ \mathbf{z}_i = \sigma\left(\sum_{j=1}^{m} \alpha'_{ij}\mathbf{W}\mathbf{q}_j\right) \end{cases}$$



Zhang, Q., Dong, J., Duan, K., Huang, X., Liu, Y., & Xu, L. (2022, October). Contrastive knowledge graph error detection. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (pp. 2590-2599).

Joint Optimization

- Translational Loss with Negative Sampling

$$E(h, r, t) = \|\tilde{\mathbf{e}}_h + \tilde{\mathbf{e}}_r - \tilde{\mathbf{e}}_t\|_2$$

$$\mathcal{L}_{\text{trans}} = \sum_{(h,r,t)\in\mathcal{G}} \sum_{(\hat{h},\hat{r},\hat{t})\in\hat{\mathcal{G}}} \max(0, \gamma + E(h,r,t) - E\left(\hat{h},\hat{r},\hat{t}\right)$$

- Contrastive Loss

$$sim(\boldsymbol{x}_i, \boldsymbol{z}_i) = \frac{\boldsymbol{x}_i \ \boldsymbol{z}_i}{|\boldsymbol{x}_i| \ |\boldsymbol{z}_i|}$$

$$\mathcal{L}_{\text{con}}(\mathbf{x}_i, \mathbf{z}_i) = -\log \frac{\exp\left(\operatorname{sim}\left(\mathbf{x}_i, \mathbf{z}_i\right)/\tau\right)}{\sum_{j\in\{1,2,\ldots,N\}\setminus\{i\}} \exp\left(\operatorname{sim}\left(\mathbf{x}_i, \mathbf{z}_j\right)/\tau\right)}$$

Zhang, Q., Dong, J., Duan, K., Huang, X., Liu, Y., & Xu, L. (2022, October). Contrastive knowledge graph error detection. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (pp. 2590-2599).

Graph Data Management

- Graph Data Quality Management
  - Data Quality Assessment
  - **Data Quality Enhancement**
- Graph Generation
  - Learning-based Graph Generation
  - Function-driven Graph Generation

Missing Data Imputation

Problem Definition

Presidential information				
Name	**Nation**	**Birth**	**Age**	**Place of birth**
Obama	USA	1961	?	Hawaii
Trump	USA	1946	77	New York
Biden	?	1942	81	Pennsylvania
Putin	Russia	1952	71	Saint Petersburg

Birth && Age

Nation && Place of birth

Presidential information				
Name	**Nation**	**Birth**	**Age**	**Place of birth**
Obama	USA	1961	62	Hawaii
Trump	USA	1946	77	New York
Biden	USA	1942	81	Pennsylvania
Putin	Russia	1952	71	Saint Petersburg

x_{11}	x_{12}	x_{13}	?	x_{15}
x_{21}	?	x_{23}	x_{24}	x_{25}
?	x_{32}	x_{33}	?	x_{35}

Data Matrix X

1	1	1	0	1
1	0	1	1	1
0	1	1	0	1

Mask Matrix M

x_{11}	x_{12}	x_{13}	$\tilde{x}_{14}$	x_{15}
x_{21}	$\tilde{x}_{22}$	x_{23}	x_{24}	x_{25}
$\tilde{x}_{31}$	x_{32}	x_{33}	$\tilde{x}_{34}$	x_{35}

Imputed Matrix $\tilde{X}$

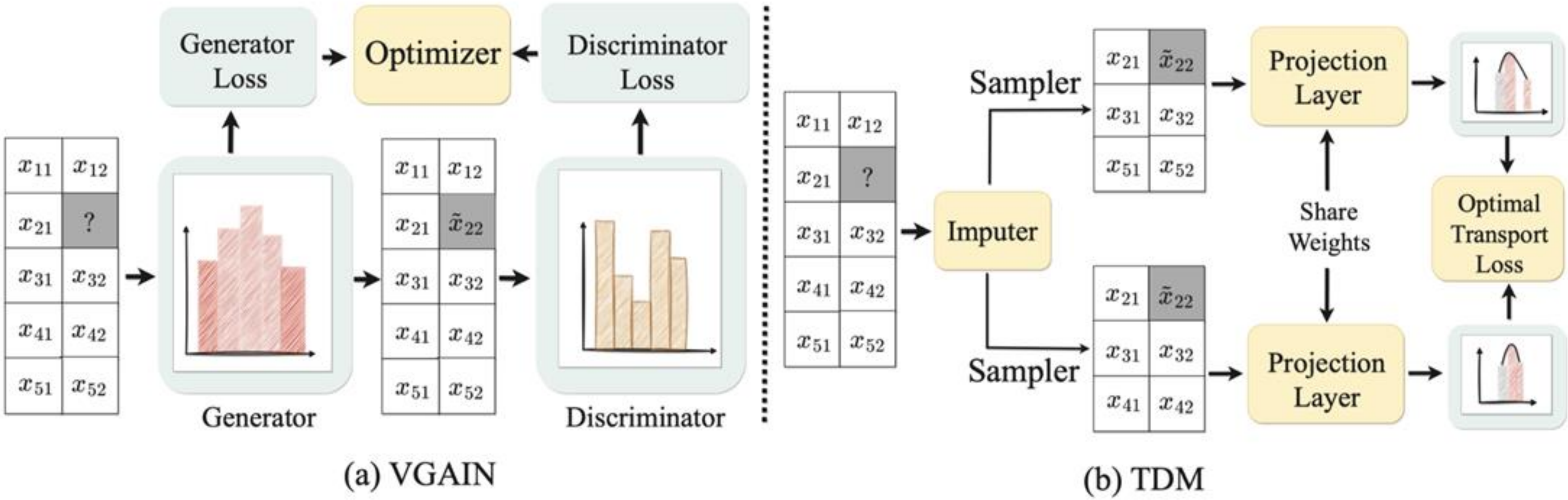
$\tilde{x}_{11}$	$\tilde{x}_{12}$	$\tilde{x}_{13}$	$\tilde{x}_{14}$	$\tilde{x}_{15}$
$\tilde{x}_{21}$	$\tilde{x}_{22}$	$\tilde{x}_{23}$	$\tilde{x}_{24}$	$\tilde{x}_{25}$
$\tilde{x}_{31}$	$\tilde{x}_{32}$	$\tilde{x}_{33}$	$\tilde{x}_{34}$	$\tilde{x}_{35}$

Intermediate Matrix $\tilde{X}^{in}$

Wang, J., Zhang, Y., Wang, K., Lin, X., & Zhang, W. (2024). Missing data imputation with uncertainty-driven network. *Proceedings of the ACM on Management of Data*, *2*(3), 1-25.

Missing Data Imputation

Motivation



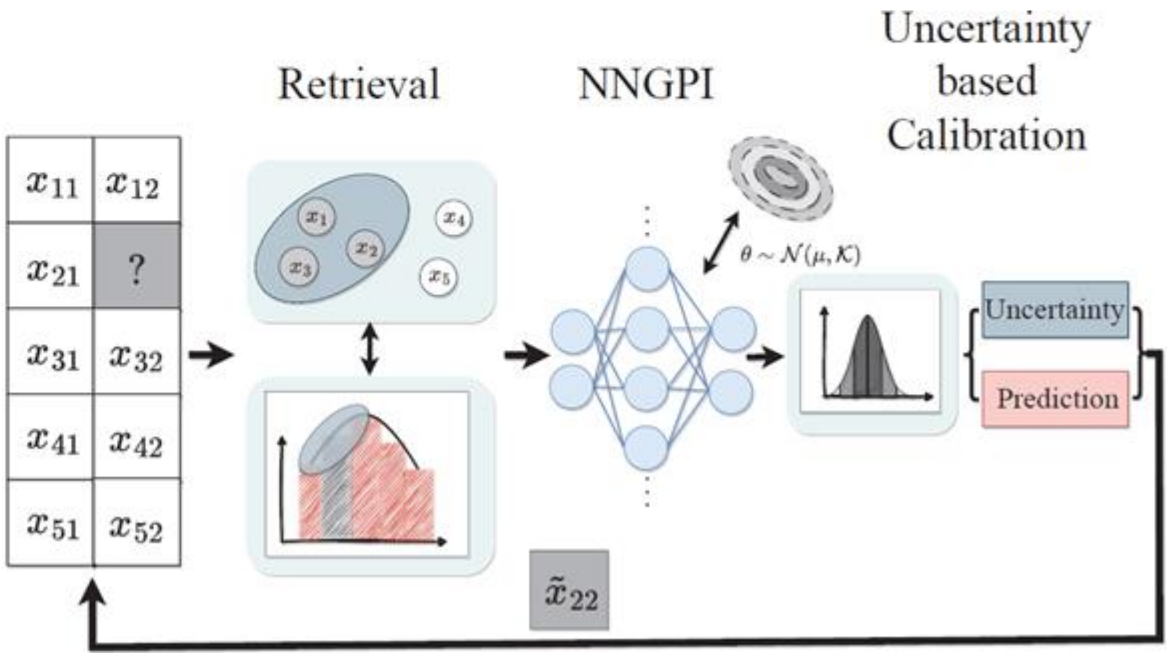
Heavily rely on the global distribution

Deploy a sophisticated deep learning (DL) model

Wang, J., Zhang, Y., Wang, K., Lin, X., & Zhang, W. (2024). Missing data imputation with uncertainty-driven network. *Proceedings of the ACM on Management of Data*, *2*(3), 1-25.

Missing Data Imputation

Our Method



- **Framework design-NOMI**
  - ✓ Neural Network Gaussian Process Imputator (NNGPI)
  - ✓ Retrieval Module and Uncertainty-based Calibration
- **Theoretical foundation**
  - ✓ NOMI can be reformulated as an instance of the EM algorithm

Wang, J., Zhang, Y., Wang, K., Lin, X., & Zhang, W. (2024). Missing data imputation with uncertainty-driven network. *Proceedings of the ACM on Management of Data*, *2*(3), 1-25.

Missing Data Imputation

Retrieval Module

- Similarity computation

$$S(x_i, x_j) = \frac{1}{L_2(x_i, x_j)} = \frac{1}{\sqrt{\sum_{p=1}^{d} (x_{ip} - x_{jp})^2}}$$

- Select top-k similar neighbors

$$\mathrm{idx} = \mathrm{top\_rank}(S(x_i, X), K)$$

- Input construction

$$x_i = x_i || \{S^N(x_i, x_j) \times x_j, \forall j \in \mathrm{idx}\}$$

Wang, J., Zhang, Y., Wang, K., Lin, X., & Zhang, W. (2024). Missing data imputation with uncertainty-driven network. *Proceedings of the ACM on Management of Data*, *2*(3), 1-25.

Missing Data Imputation

Neural Gaussian Network Imputation

- L-layer neural network

number of neurons in layer l

$$g_i^l(x) = b_i^l + \sum_{j=1}^{\rho_l} w_{ij}^l f_j^l(x)$$

$$f_j^l(x) = \phi(g_j^{l-1}(x))$$

the non-linearity function output of previous layer

- Assume that $g_j^{l-1}(x)$ represents a Gaussian Process, thus $f_j^l(x)$ is also a GP.
- $g_i^l(x)$ is a summation of i.i.d. terms. According to the Central Limit Theorem, $g_i^l(x)$ approach a Gaussian distribution when ρ_l grows towards infinity.

Wang, J., Zhang, Y., Wang, K., Lin, X., & Zhang, W. (2024). Missing data imputation with uncertainty-driven network. *Proceedings of the ACM on Management of Data*, *2*(3), 1-25.

Missing Data Imputation

Training Objectives

$$\mathcal{L}(X,T) = \frac{1}{n}\sum_{i=1}^{n}\left(\left(g\left(x_i\right) - t_i\right)^2\right)$$

Algorithm 1: The Forward Propagation of NOMI.

Input: The test sample x_*, training dataset $\{X, T\}$, neighbor set size K, threshold τ. Batch size n_B.

Output: The imputation $\tilde{x}_*$

Initialize the missing value in x_*

$\{X_B, T_B\} \leftarrow$ Randomly select from $\{X, T\}$ with size n_B

while *True* **do**

  `// Retrieval phase.`

  for $x_i \in X_B \cup x_*$ **do**

    $\text{idx} = \text{top\_rank}\left(S\left(x_i, X\right), K\right)$

    $x_i = x_i || \{S^N\left(x_i, x_j\right) \times x_j, \forall j \in \text{idx}\}$

  `// Imputation by NNGPI.`

  for $l = 1, \cdots, L$ **do**

    for $x_p, x_q \in X_B \cup x_*$ **do**

      $\mathcal{K}^l\left(x_p, x_q\right) = \sigma_b^2 + \sigma_w^2 F_\phi(\mathcal{K}^{l-1}(x_p, x_q), \mathcal{K}^{l-1}(x_p, x_p), \mathcal{K}^{l-1}(x_q, x_q))$

  $\overline{\mu}_{x_*} = \mathcal{K}^T_{X_B, x_*}(\mathcal{K}_{X_B, X_B} + \sigma_b^2 I)^{-1} g(X_B)$

  $\overline{\mathcal{K}}_{x_*} = \mathcal{K}_{x_*, x_*} - \mathcal{K}^T_{X_B, x_*}(\mathcal{K}_{X_B, X_B} + \sigma_b^2 I)^{-1} \mathcal{K}_{X_B, x_*}$

  `// Uncertainty-based calibration.`

  $t_*^{\{m\}} = (1 - \frac{\beta}{\overline{\mathcal{K}}_{x_*}}) t_*^{\{m-1\}} + \frac{\beta}{\overline{\mathcal{K}}_{x_*}} \overline{\mu}_{x_*}$

  $\tilde{x}_* = t_*^{\{m\}}$

  `// Termination Check.`

  if $\overline{\mathcal{K}}_{x_*} < \tau$ **then**

    Break

return $\tilde{x}_*$

Wang, J., Zhang, Y., Wang, K., Lin, X., & Zhang, W. (2024). Missing data imputation with uncertainty-driven network. *Proceedings of the ACM on Management of Data*, *2*(3), 1-25.

Missing Data Imputation

Experiments

Table 2. Statistics of the datasets

Dataset	# of data sample	# numerical	# categorical
Wine	178	13	1
Heart	303	7	7
Breast	699	9	1
Car	1,728	0	7
Wireless	2,000	7	1
Abalone	4,177	8	0
Turkiye	5,820	0	33
Letter	20,000	0	16
Chess	28,056	0	7
Shuttle	43,500	0	10
Retail	1,067,371	5	1
WISDM	15,630,426	3	2

- **Metric**

$$RMSE(X,\tilde{X}) = \sqrt{\frac{\sum_{j=1}^{d}\sum_{i=1}^{n}(1-m_{ij})(x_{ij}-\tilde{x}_{ij})^2}{\sum_{j=1}^{d}\sum_{i=1}^{n}(1-m_{ij})}}$$

$$MAE(X,\tilde{X}) = \frac{\sum_{j=1}^{d}\sum_{i=1}^{n}(1-m_{ij})\left|x_{ij}-\tilde{x}_{ij}\right|}{\sum_{j=1}^{d}\sum_{i=1}^{n}(1-m_{ij})}$$

- **Missing Mechanism**

✔ MCAR $P(M|X) = P(M|X^o, X^m) = P(M)$

✔ MAR $P(M|X^o, X^m) = P(M|X^o)$

✔ MNAR $P(M|X^o, X^m) = P(M|X^m)$

Wang, J., Zhang, Y., Wang, K., Lin, X., & Zhang, W. (2024). Missing data imputation with uncertainty-driven network. *Proceedings of the ACM on Management of Data*, *2*(3), 1-25.

Missing Data Imputation

Experiments

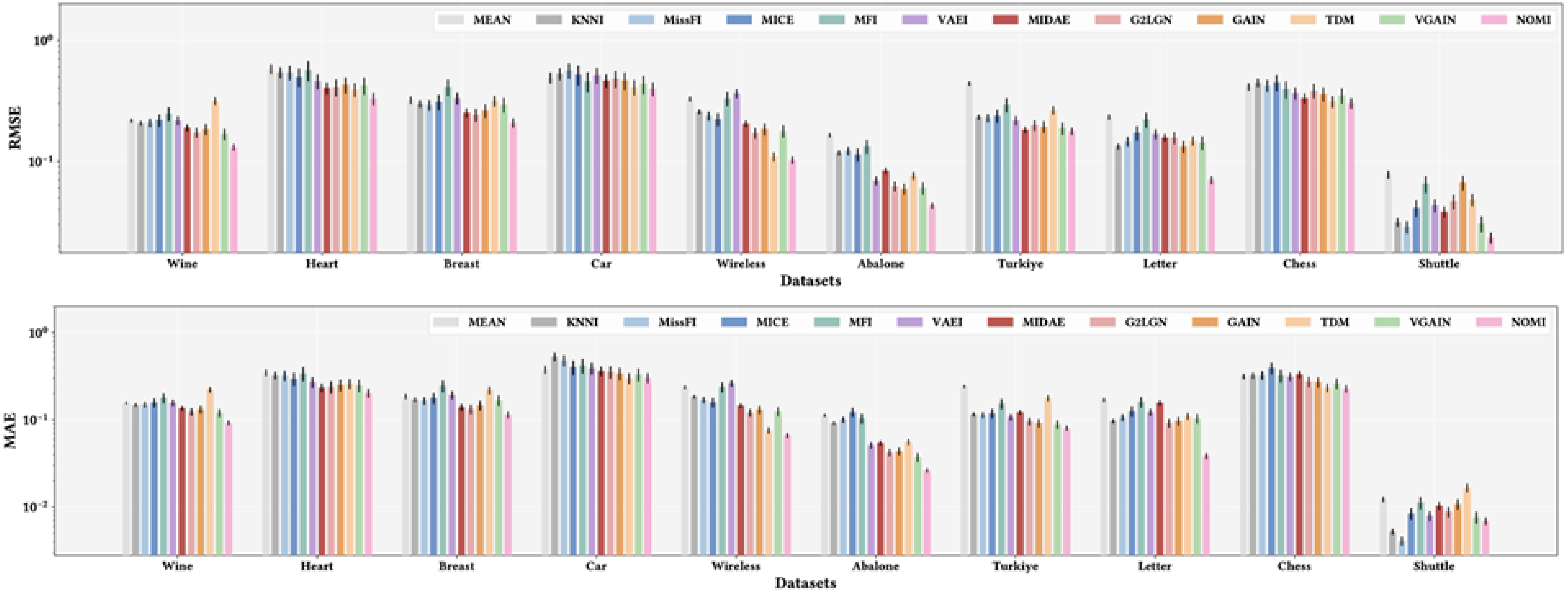


Fig. 3. Results of missing data imputation (20% MCAR)

- NOMI reduces the imputation RMSE, by 24.58% and 56.64% compared to VGAIN and TDM
- NOMI reduces the imputation MAE, by 25.14% and 37.16% compared to VGAIN and TDM

Wang, J., Zhang, Y., Wang, K., Lin, X., & Zhang, W. (2024). Missing data imputation with uncertainty-driven network. *Proceedings of the ACM on Management of Data*, *2*(3), 1-25.

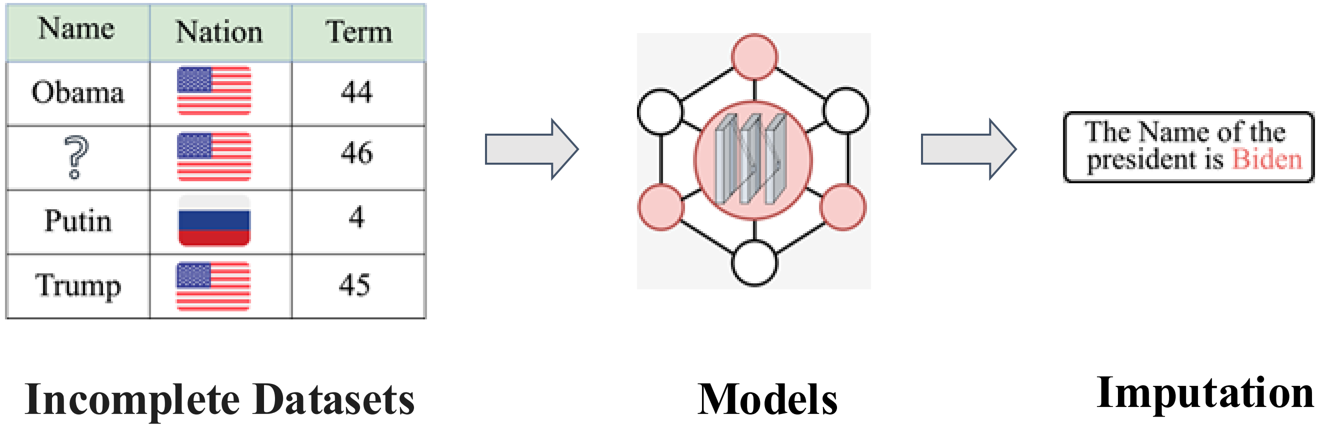
Mixed-type Missing Data Imputation

On LLM-enhanced mixed-type data imputation with high-order message passing

Problem Definition

- **Mixed-type Missing Data Imputation**

Aims to impute the unobserved elements in the raw data, i.e., X_{miss} , and make the imputed matrix $\tilde{X}$ as close to the real complete dataset $\bar{X}$ as possible. The raw data matrix X may contain numerical, categorical and text data.



Mixed-type Missing Data Imputation

Background and Motivations

- **Existing learning-based methods and rule-based methods:**
  - ➢MICE
  - ➢GAIN / VGAIN
  - ➢GRAPE / IGRM
  - ➢TDM, ReMasker and so on

Num. and Cate data

Text data

- **Existing LLM-based methods:**
  - ➢DFMs
  - ➢Table-GPT / Jellyfish

Num. and Cate data

Text data

Num. and Cate data

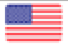
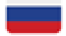
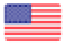
Text data

Wang, Jianwei, et al. "On llm-enhanced mixed-type data imputation with high-order message passing." VLDB (2025)

Mixed-type Missing Data Imputation

Motivation 1: Global-Local Information

The name of the president is relevant not only to their nation and term but also to the sequential relationship of terms.

Name	Nation	Term
Obama		44
?		46
Putin		4
Trump		45

Theorem 3.1: *Consider two imputation models, θ^{g+l} and θ^{l}, where θ^{g+l} captures both global and local information in the latent space, and θ^{l} captures only the local information. Assuming that interactions of global and local information are independent, then we have:*

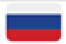
$$\Psi(\theta^{g+l}, X_{miss}) \leq \Psi(\theta^{l}, X_{miss}),$$

indicating that a model capable of capturing both global and local information achieves a lower imputation error.

Mixed-type Missing Data Imputation

Motivation 2: High-order Relationship

Neither the nation nor the term alone is sufficient to fully determine the Name.

Name	Nation	Term
Obama		44
?		46
Putin		4
Trump		45

Theorem 3.2: *Consider two imputation models, $\theta^{[0:r]}$ and $\theta^{[0:s]}$, where $\theta^{[0:r]}$ captures interactions up to order r in the latent space, and $\theta^{[0:s]}$ captures interactions up to order s, with $r > s$. We have*

$$\Psi(\theta^{[0:r]}, X_{miss}) \leq \Psi(\theta^{[0:s]}, X_{miss}),$$

indicating that the model capable of capturing higher-order interactions exhibits a lower imputation error.

Wang, Jianwei, et al. "On llm-enhanced mixed-type data imputation with high-order message passing." VLDB (2025)

Mixed-type Missing Data Imputation

Motivation 3: Inter-column Heterogeneity and Intra-Column Homogeneity

The name is text data and the nation is categorical data. Furthermore, the name format remains consistent across rows.

Name	Nation	Term
Obama		44
?		46
Putin		4
Trump		45

Theorem 3.3: *Consider two imputation models, θ^{cp} and θ, where θ^{cp} captures the column patterns including intra-column heterogeneity and intra-column homogeneity, while θ does not. Then, we have:*

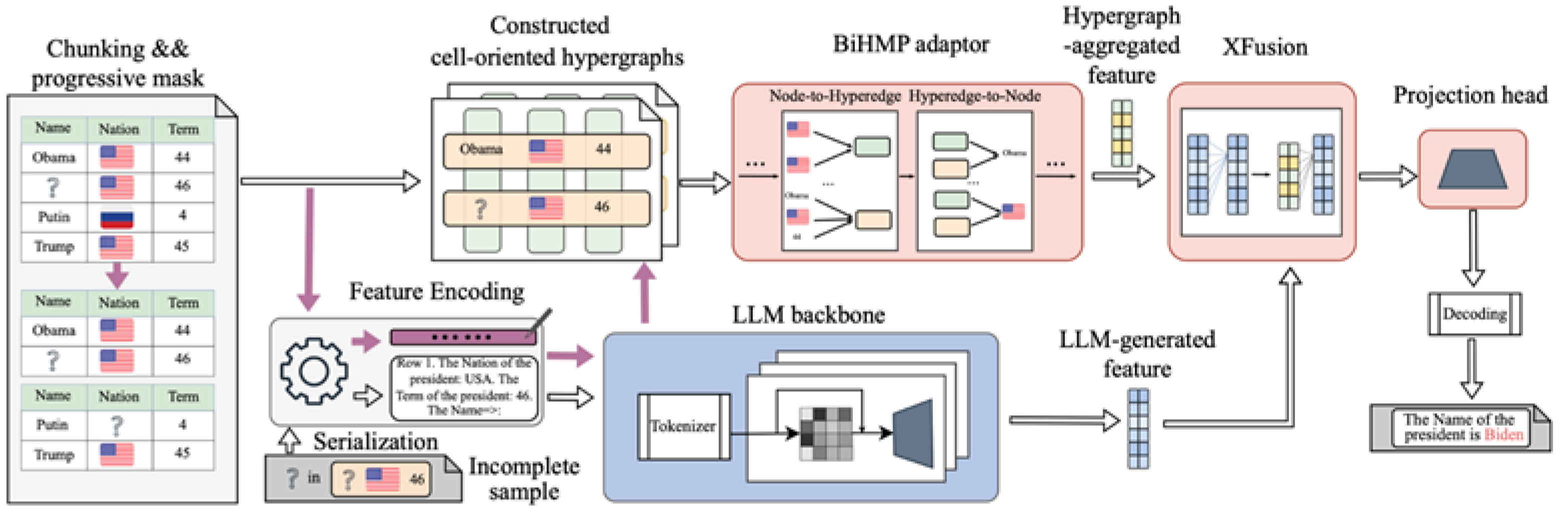
$$\Psi(\theta^{cp}, X_{miss}) \leq \Psi(\theta, X_{miss}),$$

indicating that model θ^{cp} capturing the column patterns achieves a lower imputation error.

Wang, Jianwei, et al. "On llm-enhanced mixed-type data imputation with high-order message passing." VLDB (2025)

Mixed-type Missing Data Imputation

Our Method: UnIMP



- Cell-Oriented Hypergraph Modeling: Serialization and Tokenization and Propagation of LLM backbone
- Bidirectional High-order Message Passing: Node-to-Hyperedge and Hyperedge-to-Node
- XFusion Block and Projection Head

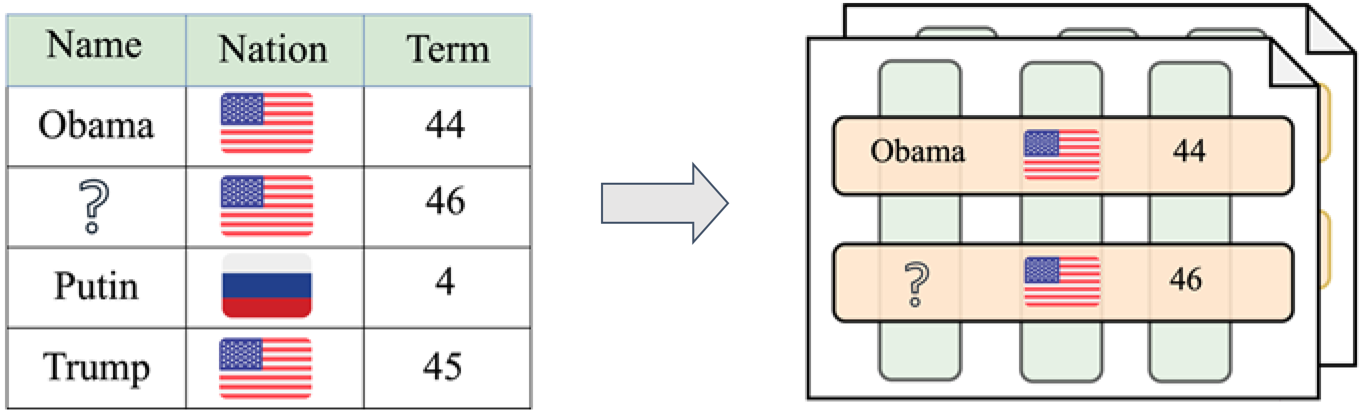
Wang, Jianwei, et al. “On llm-enhanced mixed-type data imputation with high-order message passing.” VLDB (2025)

Mixed-type Missing Data Imputation

Our Method: Cell-Oriented Hypergraph

Given a tabular dataset X with n samples, each containing d features, we construct a hypergraph $HG(\mathcal{V}, \mathcal{E})$ as follows:

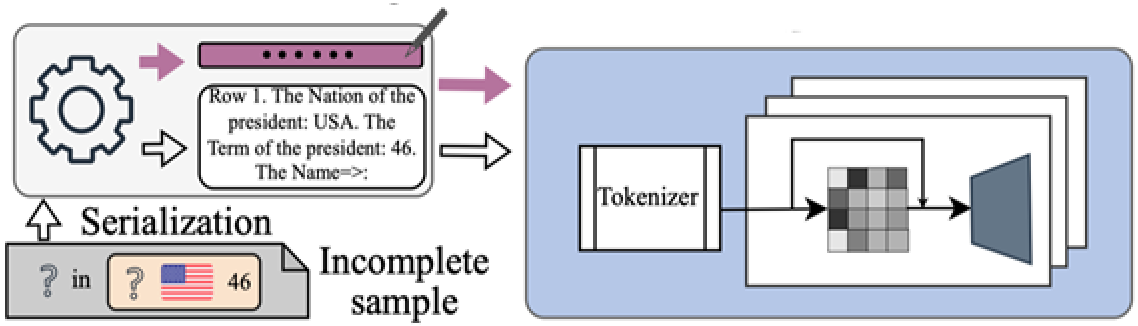
- For each cell $x_{ij} \in X$, we create a corresponding node $v_{idx} \in \mathcal{V}$, where $idx = i * d + j$.
- For nodes in the same column (i.e., nodes corresponding to $\{x_{0j}, x_{1j}, \cdots\}$), we construct a hyperedge $e_j \in \mathcal{E}$.
- Similarly, nodes in the same row (i.e., nodes corresponding to $\{x_{i0}, x_{i1}, \cdots\}$) form a hyperedge $e_{i+d} \in \mathcal{E}$.



Wang, Jianwei, et al. "On llm-enhanced mixed-type data imputation with high-order message passing." VLDB (2025)

Mixed-type Missing Data Imputation

Our Method: Feature Encoding



- Attribute-Value Serialization

  Row i, col_name=>{node data} EOS

  This is row (or col): i (or col_name) EOS

- Tokenization

  $$\{t_0, t_1, \cdots, t_s\} = \text{tokenizer}(\text{prompt-text})$$

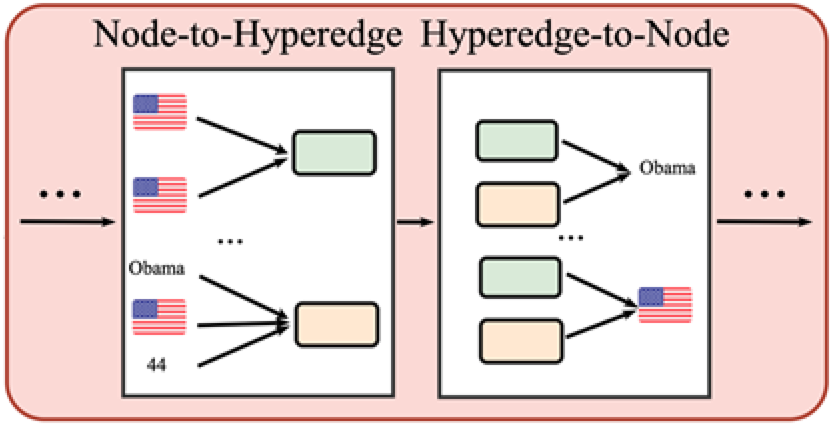
- Propagation of LLM backbone

  $$z_p : \{z_{t_0}, z_{t_1}, \cdots, z_{t_s}\} = \text{LLM-backbone}(t_0, t_1, \cdots, t_s)$$

Wang, Jianwei, et al. "On llm-enhanced mixed-type data imputation with high-order message passing." VLDB (2025)

Mixed-type Missing Data Imputation

Our Method: Bidirectional High-order Message Passing



Node-to-Hyperedge

$$z_{e_j}^{temp} = \frac{1}{|e_j|} \sum_{v_i \in e_j} \sigma\left(f_1^l(z_{v_i}^l)\right)$$

$$z_{e_j}^{l+1} = \sigma\left(f_2^l\left(\mathrm{CONCAT}\left(z_{e_j}^l, z_{e_j}^{temp}\right)\right)\right)$$

- Update the representation of hyperedge.

Hyperedge-to-Node

$$z_{v_i}^{l+1} = \sigma\left(f_3^l\left(\mathrm{CONCAT}\left[z_{v_i}^l, z_{e_{v_i}^c}^l, z_{e_{v_i}^r}^l\right]\right)\right)$$

- Updates node representations.

Wang, Jianwei, et al. "On llm-enhanced mixed-type data imputation with high-order message passing." VLDB (2025)

Mixed-type Missing Data Imputation

Evaluation: Accuracy Over Numerical and Categorical Data

Table 4: Results of missing data imputation (20% MCAR)

Model	RMSE								MAE							
	Blogger	Zoo	Parkinsons	Bike	Chess	Shuttle	Power	Improve%	Blogger	Zoo	Parkinsons	Bike	Chess	Shuttle	Power	Improve%
MEAN	0.4315	0.4260	0.2062	0.2239	0.3079	0.0914	0.0869	45.22%	0.3631	0.3663	0.1473	0.1561	0.2584	0.0467	0.0559	57.24%
KNNI	0.4417	0.2749	0.1891	0.2023	0.3246	0.0456	OOT	35.75%	0.3337	0.1473	0.1207	0.1277	0.2606	0.0192	OOT	41.33%
MICE	0.4134	0.2645	0.1263	0.1796	0.2966	0.0426	0.0650	28.27%	0.3605	0.1731	0.0667	0.1097	0.2453	0.0131	0.0370	36.49%
VGAIN	0.4316	0.4114	0.1913	0.2219	0.2797	0.0786	0.0811	42.48%	0.3643	0.1891	0.1156	0.1363	0.2537	0.0352	0.0509	48.88%
TDM	0.4384	0.2949	0.1862	0.2444	0.3027	0.0769	0.0926	41.48%	0.3229	0.1490	0.0830	0.1499	0.2345	0.0350	0.0600	42.96%
GINN	0.4657	0.2761	0.1466	0.1641	0.2911	0.0734	OOM	32.41%	0.3444	0.1445	0.0884	0.0921	0.2339	0.0434	OOM	36.97%
GRAPE	0.4304	0.3211	0.1064	0.1481	0.2749	0.0242	OOM	20.89%	0.3151	0.1605	0.0535	0.0796	0.2200	0.0073	OOM	21.39%
IGRM	0.4551	0.3063	0.1035	OOM	OOM	OOM	OOM	12.11%	0.3423	0.1621	0.0497	OOM	OOM	OOM	OOM	8.74%
DFMs	0.4413	0.4445	0.2412	0.2483	OOT	OOT	OOT	41.53%	0.3676	0.2934	0.1647	0.1529	OOT	OOT	OOT	49.81%
Table-GPT	0.4237	0.4315	0.2246	0.2547	OOT	OOT	OOT	39.71%	0.3572	0.2713	0.1761	0.1442	OOT	OOT	OOT	48.99%
Jellyfish	0.4133	0.4177	0.2127	0.1935	OOT	OOT	OOT	36.43%	0.3557	0.2719	0.1548	0.1478	OOT	OOT	OOT	47.38%
NOMI	0.4112	0.2576	0.1322	0.1582	0.3042	0.0237	0.0731	28.32%	0.3102	0.1442	0.0710	0.0740	0.2298	0.0071	0.0463	30.86%
ReMasker	0.4068	0.3309	0.1508	0.1277	0.2662	0.1111	OOT	29.61%	0.3293	0.1807	0.0995	0.0655	0.2113	0.0545	OOT	35.36%
UnIMP	0.4171	0.2979	0.1407	0.1730	0.2628	0.0398	0.0485	25.84%	0.3384	0.1822	0.0966	0.1121	0.2050	0.0238	0.0256	36.08%
UnIMP-ft	0.3972	0.2474	0.0990	0.1172	0.2142	0.0134	0.0425	–	0.3082	0.1428	0.0475	0.0602	0.1438	0.0040	0.0225	–

* Red text indicates the best result. Blue text indicates the second best result. 'OOT' indicates out of time (with a limit of 10 hours). 'OOM' indicates out of memory.

- These results highlight the excellence of UnIMP and UnIMP-ft in imputing numerical and categorical data.

Wang, Jianwei, et al. "On llm-enhanced mixed-type data imputation with high-order message passing." VLDB (2025)

Mixed-type Missing Data Imputation

Evaluation: Accuracy Over Text Data

Table 5: Results of imputation over text data

Model	ROUGE-1_{F_1}			Cos-Sim		
	Buy	Restaurant	Walmart	Buy	Restaurant	Walmart
DFMs	0.1535	0.0822	0.1420	0.8251	0.7609	0.7943
Table-GPT	0.1784	0.1398	0.1344	0.8345	0.8137	0.8254
Jellfish	0.2153	0.1675	0.2067	0.8418	0.8145	0.778
UnIMP	0.3327	0.4017	0.5594	0.8610	0.8774	0.9025
UnIMP-ft	0.4273	0.4326	0.5931	0.8892	0.8923	0.9177

- Both UnIMP and UnIMP-ft outperform previous LLM-based methods consistently.

Wang, Jianwei, et al. "On llm-enhanced mixed-type data imputation with high-order message passing." VLDB (2025)

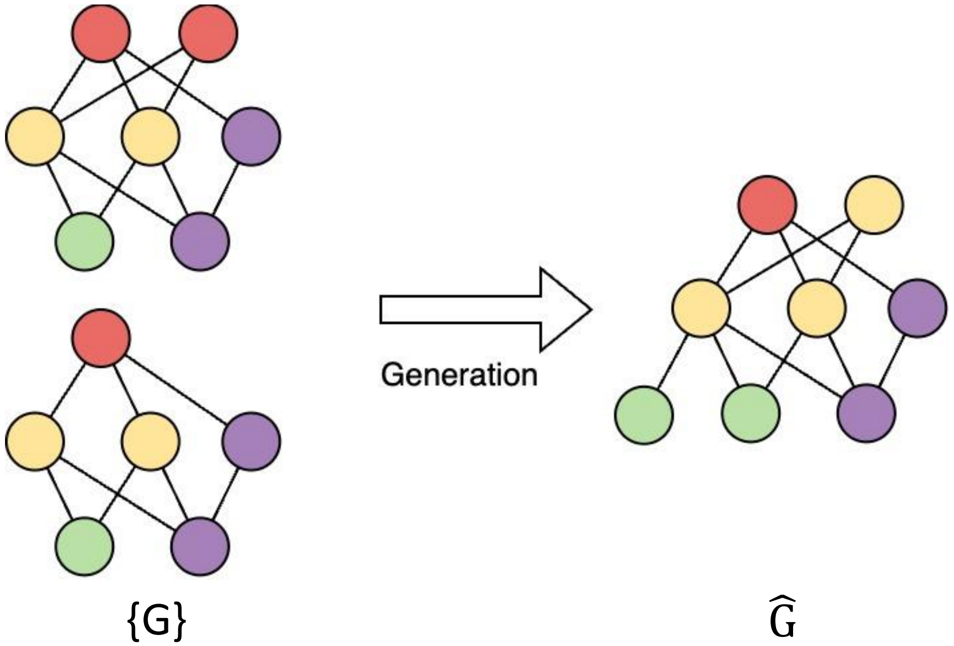
Graph Data Management

- Graph Data Quality Management
  - Data Quality Assessment
  - Data Quality Enhancement
- Graph Generation
  - **Learning-based Graph Generation**
  - Function-driven Graph Generation

Graph Data Generation

Definition of Graph Generation

Given a set of observed graphs {G}, **graph generation** aims to construct a generative model $p_\theta(G)$ to capture the distribution of these graphs, from which new graphs can be sampled $\widehat{G} \sim p_\theta(G)$. The generation process can be conditioned on additional information s, i.e., conditional graph generation $\widehat{G} \sim p_\theta(G|s)$ to apply specific constraints on the graph generation results.



Xiang, S., Wen, D., Cheng, D., Zhang, Y., Qin, L., Qian, Z. and Lin, X., 2022. General graph generators: experiments, analyses, and improvements. The VLDB Journal, pp.1-29.

Recursive: Kronecker

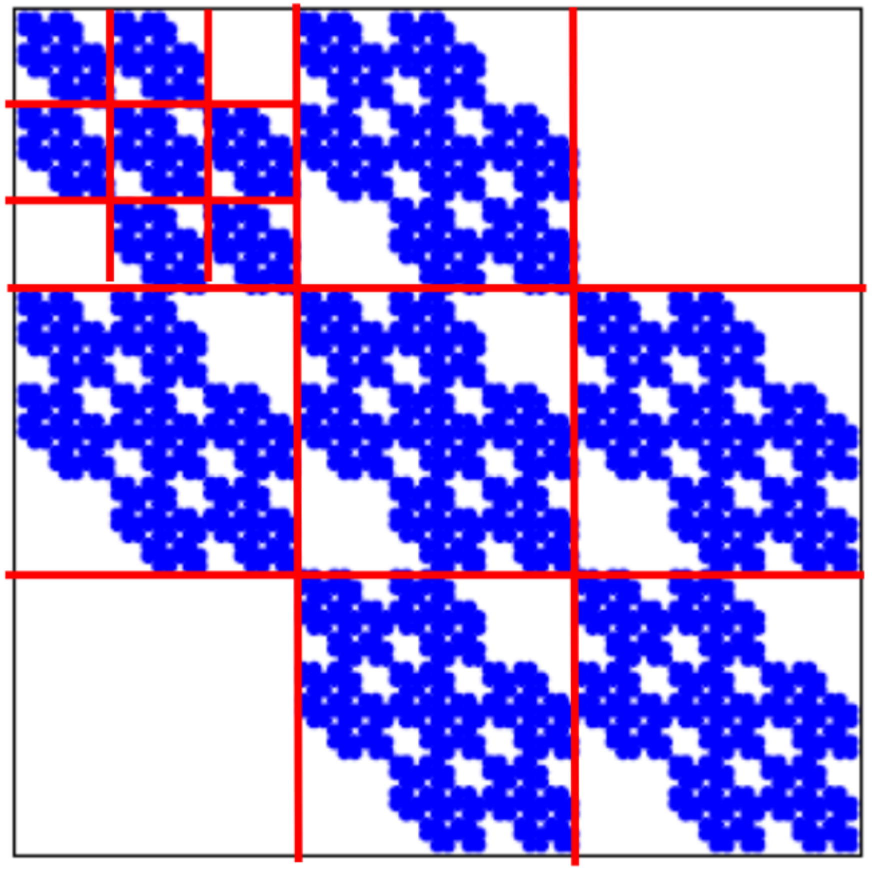
Decomposing Graph Generation into Recursive Expansion:

1	1	0
1	1	1
0	1	1

K_1

K_1	K_1	0
K_1	K_1	K_1
0	K_1	K_1

$$K_2 = K_1 \otimes K_1$$



3 x 3

9 x 9

81 x 81 adjacency matrix

Recursive: Kronecker

Decomposing Graph Generation into Recursive Expansion:

Kronecker product of matrices A and B is given by

$$\underset{N \times M}{\mathbf{C} = \mathbf{A}} \otimes \underset{K \times L}{\mathbf{B}} \doteq \underbrace{\begin{pmatrix} a_{1,1}\mathbf{B} & a_{1,2}\mathbf{B} & \dots & a_{1,m}\mathbf{B} \\ a_{2,1}\mathbf{B} & a_{2,2}\mathbf{B} & \dots & a_{2,m}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1}\mathbf{B} & a_{n,2}\mathbf{B} & \dots & a_{n,m}\mathbf{B} \end{pmatrix}}_{N*K \times M*L}$$

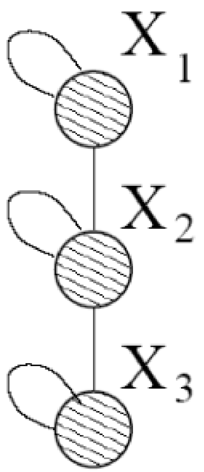
Recursive: Kronecker

Decomposing Graph Generation into Recursive Expansion:

- **Kronecker graph:** a growing sequence of graphs by iterating the **Kronecker product:**

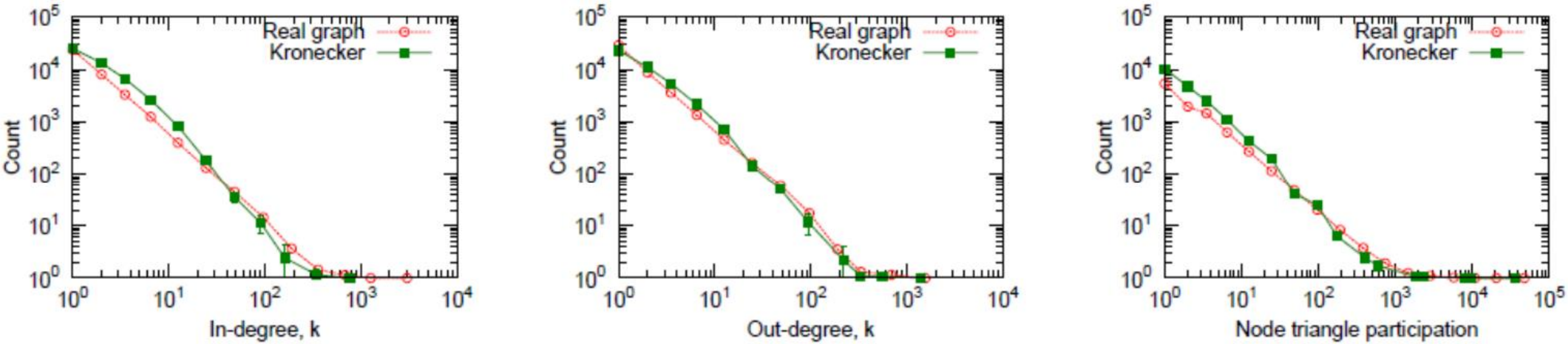
$$K_1^{[m]} = K_m = \underbrace{K_1 \otimes K_1 \otimes \dots K_1}_{m \ \textit{times}} = K_{m-1} \otimes K_1$$

1	1	0
1	1	1
0	1	1



K_1

Recursive: Kronecker



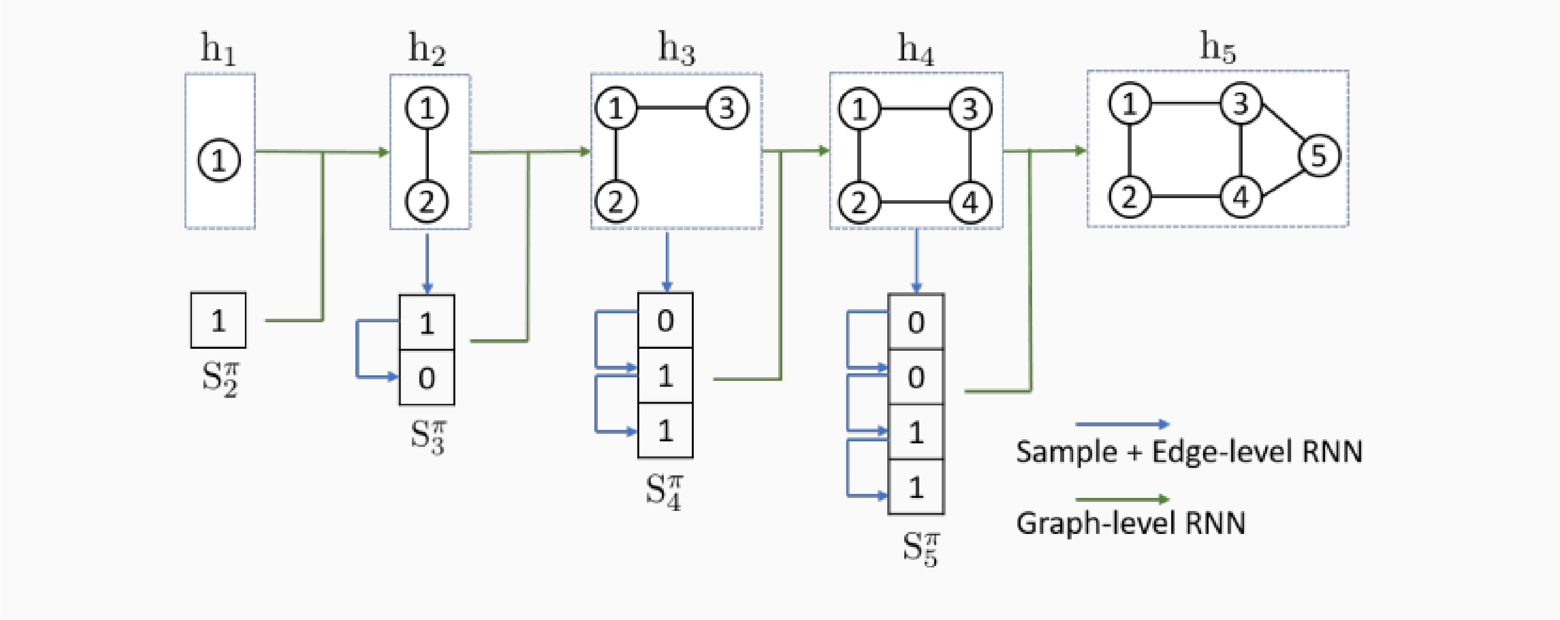
(a) In-Degree (b) Out-degree (c) Triangle participation

Autoregressive: GraphRNN

Decomposing Graph Generation into two RNNs:

- Graph-level: generates sequence of nodes

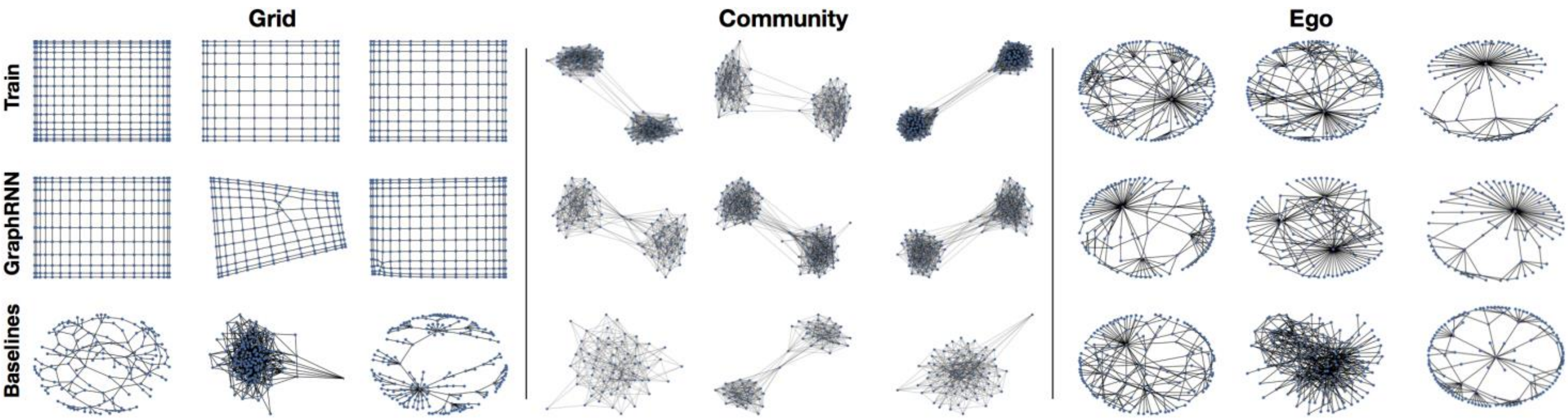
- Edge-level: generates sequence of edges for each new node



You, Jiaxuan, et al. "Graphrnn: Generating realistic graphs with deep auto-regressive models." International conference on machine learning. PMLR, 2018.

Autoregressive: GraphRNN

Visualization of input graphs and generated graphs



You, Jiaxuan, et al. "Graphrnn: Generating realistic graphs with deep auto-regressive models." International conference on machine learning. PMLR, 2018.

Autoregressive: GraphRNN

Quantitative Comparison on Generative Performance

Table 1. Comparison of GraphRNN to traditional graph generative models using MMD. $(\max(|V|), \max(|E|))$ of each dataset is shown.

	Community (160,1945)			Ego (399,1071)			Grid (361,684)			Protein (500,1575)		
	Deg.	Clus.	Orbit	Deg.	Clus.	Orbit	Deg.	Clus.	Orbit	Deg.	Clus.	Orbit
E-R	0.021	1.243	0.049	0.508	1.288	0.232	1.011	0.018	0.900	0.145	1.779	1.135
B-A	0.268	0.322	0.047	0.275	0.973	0.095	1.860	0	0.720	1.401	1.706	0.920
Kronecker	0.259	1.685	0.069	0.108	0.975	0.052	1.074	0.008	0.080	0.084	0.441	0.288
MMSB	0.166	1.59	0.054	0.304	0.245	0.048	1.881	0.131	1.239	0.236	0.495	0.775
GraphRNN-S	0.055	0.016	0.041	0.090	**0.006**	0.043	0.029	10^{-5}	0.011	0.057	**0.102**	**0.037**
GraphRNN	**0.014**	**0.002**	**0.039**	**0.077**	0.316	**0.030**	$\mathbf{10^{-5}}$	**0**	$\mathbf{10^{-4}}$	**0.034**	0.935	0.217

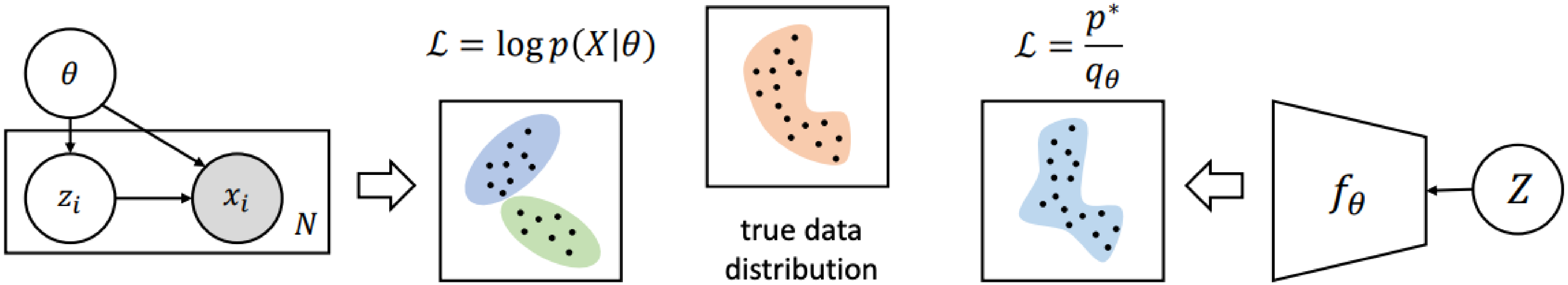
You, Jiaxuan, et al. "Graphrnn: Generating realistic graphs with deep auto-regressive models." International conference on machine learning. PMLR, 2018.

Random Walk: NetGAN

Explicit vs. implicit models

- **Explicit models** have a parametric specification of the data distribution
- Observe patterns and manually specify a model to capture them
- Learn via MLE, …

- **Implicit models** define a stochastic process that directly generates data
- Likelihood free: learn by comparison with the true data distribution (e.g. class probability estimation, GANs)



Bojchevski, Aleksandar, et al. "Netgan: Generating graphs via random walks." *International conference on machine learning*. PMLR, 2018.

Random Walk: NetGAN

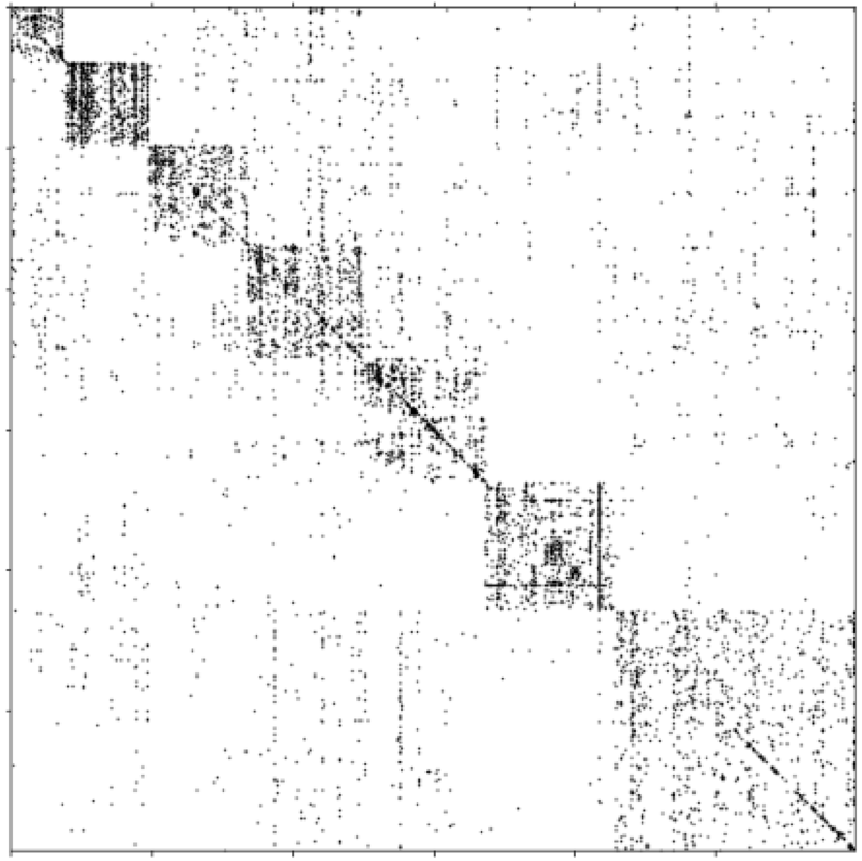
Challenges

1. Single large graph as input
- Compared to e.g. many images in computer vision

2. Quadratic scaling and sparsity
- For N nodes there are N^2 possible edges
- Real graphs have $|E| \ll N^2$ significantly fewer edges

3. Discrete output samples
- Can't easily backpropagate through sampling step

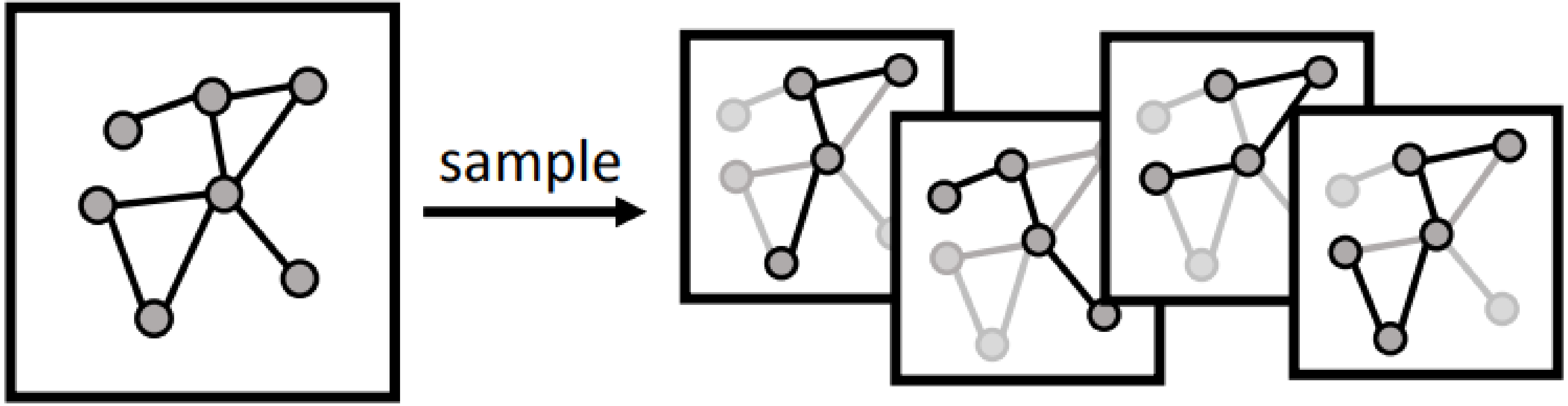
4. Permutation invariance



Bojchevski, Aleksandar, et al. "Netgan: Generating graphs via random walks." *International conference on machine learning*. PMLR, 2018.

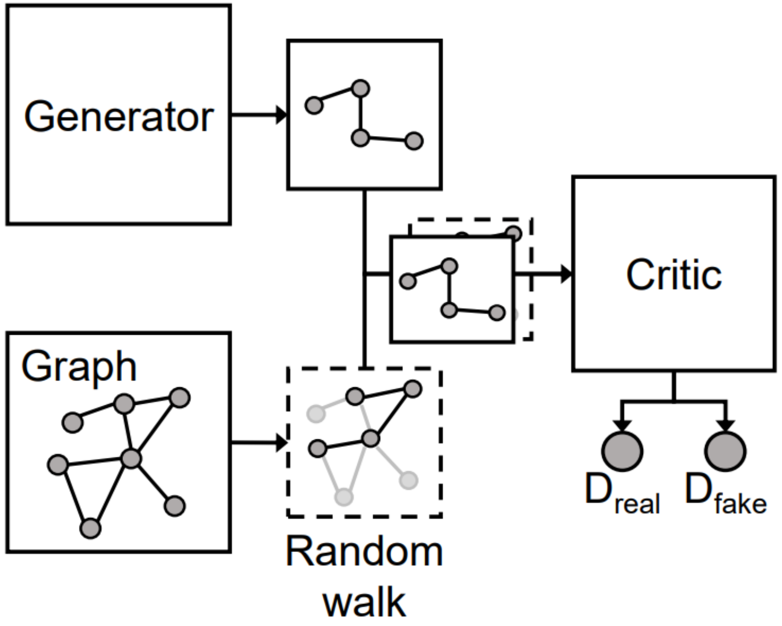
Random Walk: NetGAN

Decomposing Graph Generation into learning a distribution of random walks over the graph

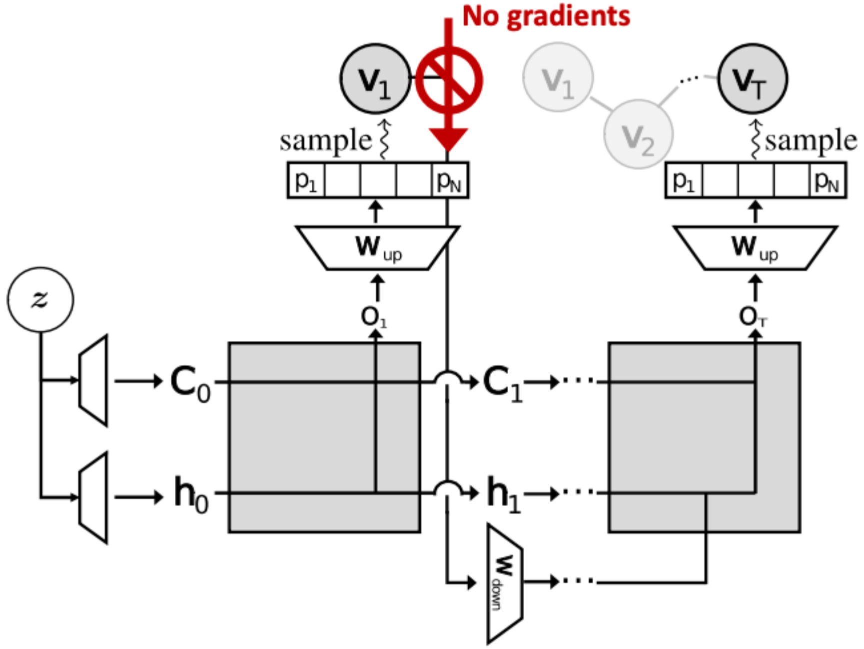


Bojchevski, Aleksandar, et al. "Netgan: Generating graphs via random walks." *International conference on machine learning*. PMLR, 2018.

Random Walk: NetGAN



Wasserstein GAN

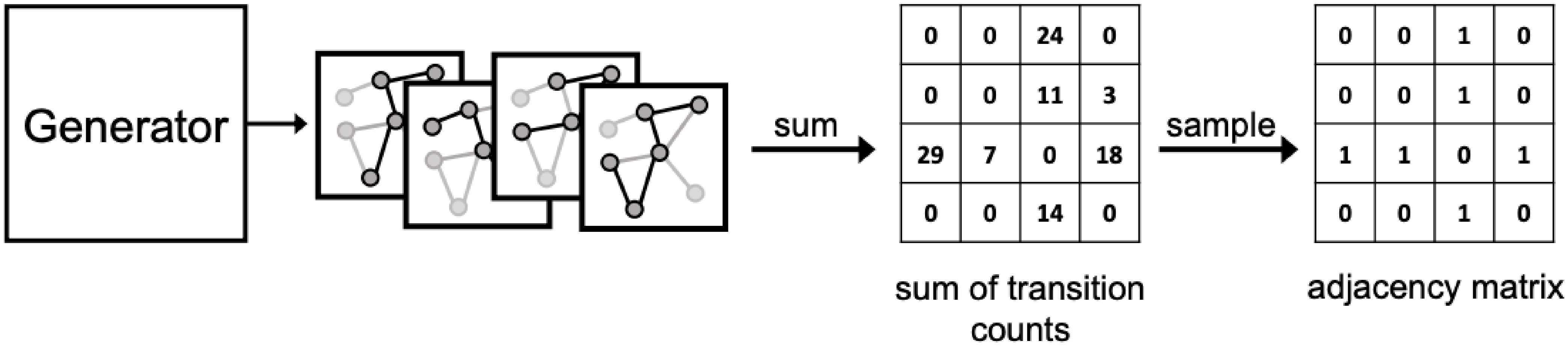


Model Framework

Generator

Bojchevski, Aleksandar, et al. "Netgan: Generating graphs via random walks." *International conference on machine learning*. PMLR, 2018.

Random Walk: NetGAN

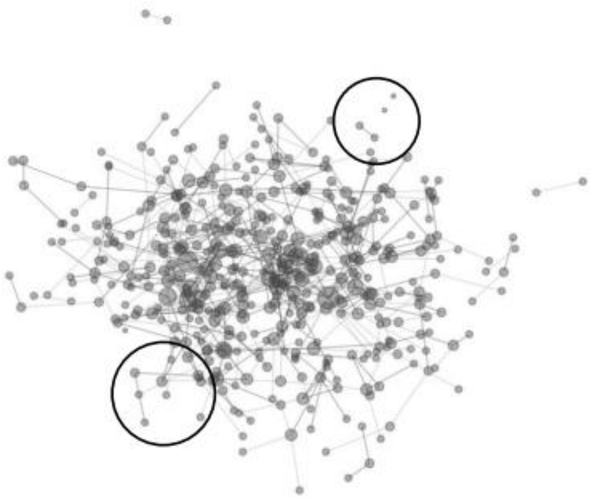


Graph assembly: sample edges with probability proportional to their transition counts

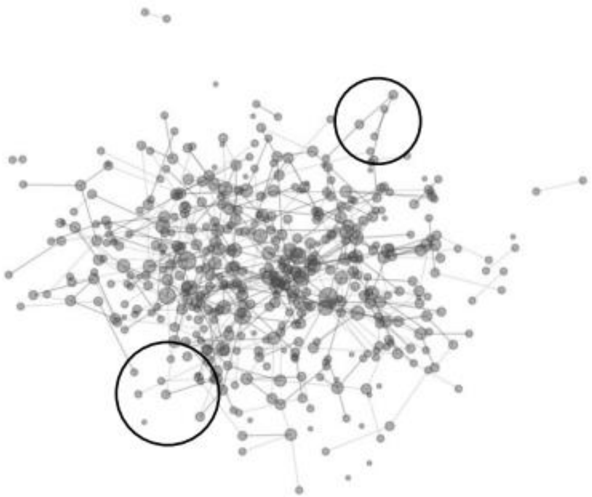
Bojchevski, Aleksandar, et al. "Netgan: Generating graphs via random walks." *International conference on machine learning*. PMLR, 2018.

Random Walk: NetGAN

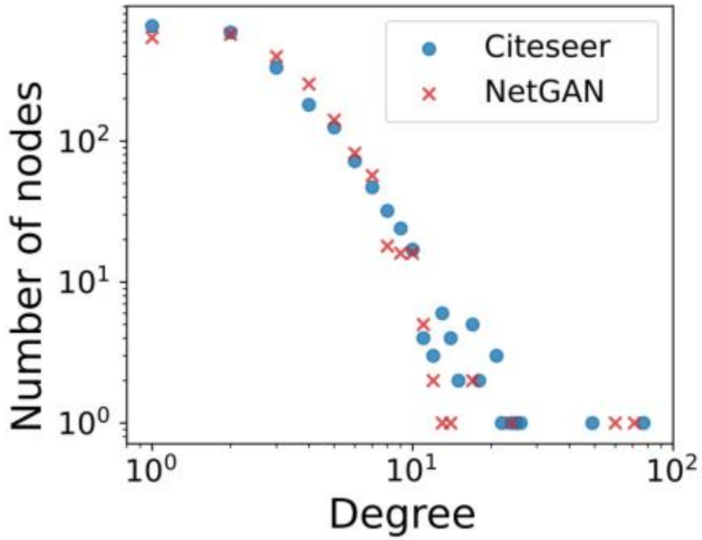
Key point: Generate graphs that have similar structure but are not replicas



Original graph



Graph generated by NetGAN

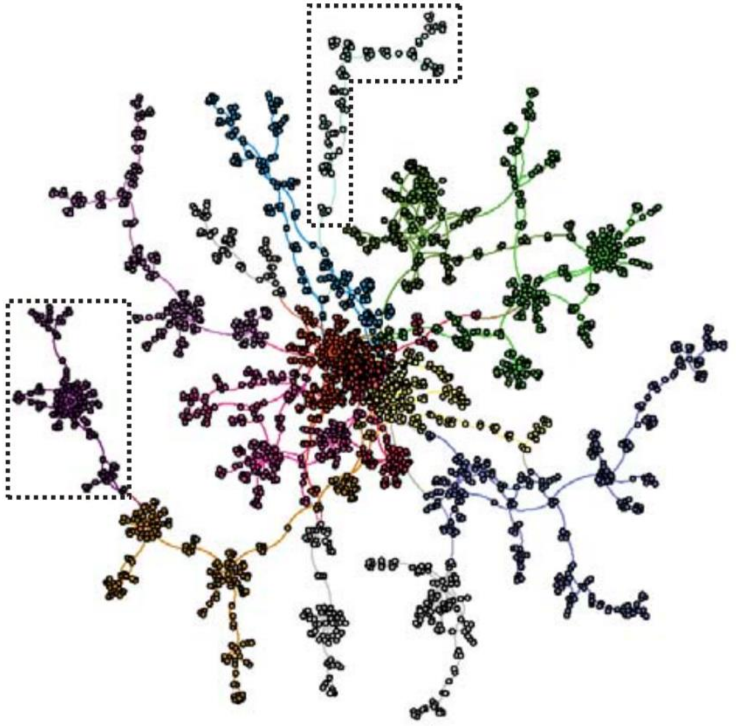


Bojchevski, Aleksandar, et al. "Netgan: Generating graphs via random walks." *International conference on machine learning*. PMLR, 2018.

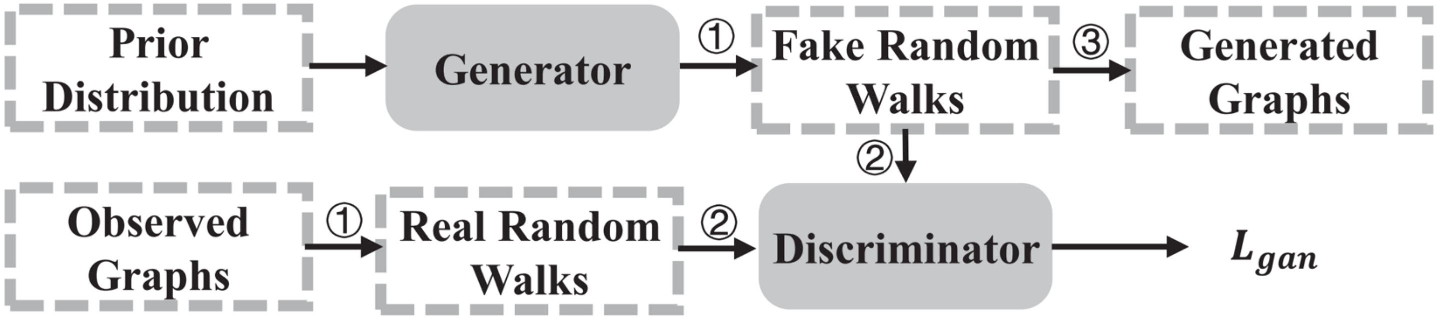
Beyond NetGAN: Community Preserving GAN

◆ Motivation

- **Community Structure,** as the main character of graph data, existing graph generation solutions cannot handle this property.
- Existing autoregressive and random walk-based solutions are not efficient.



real-world community

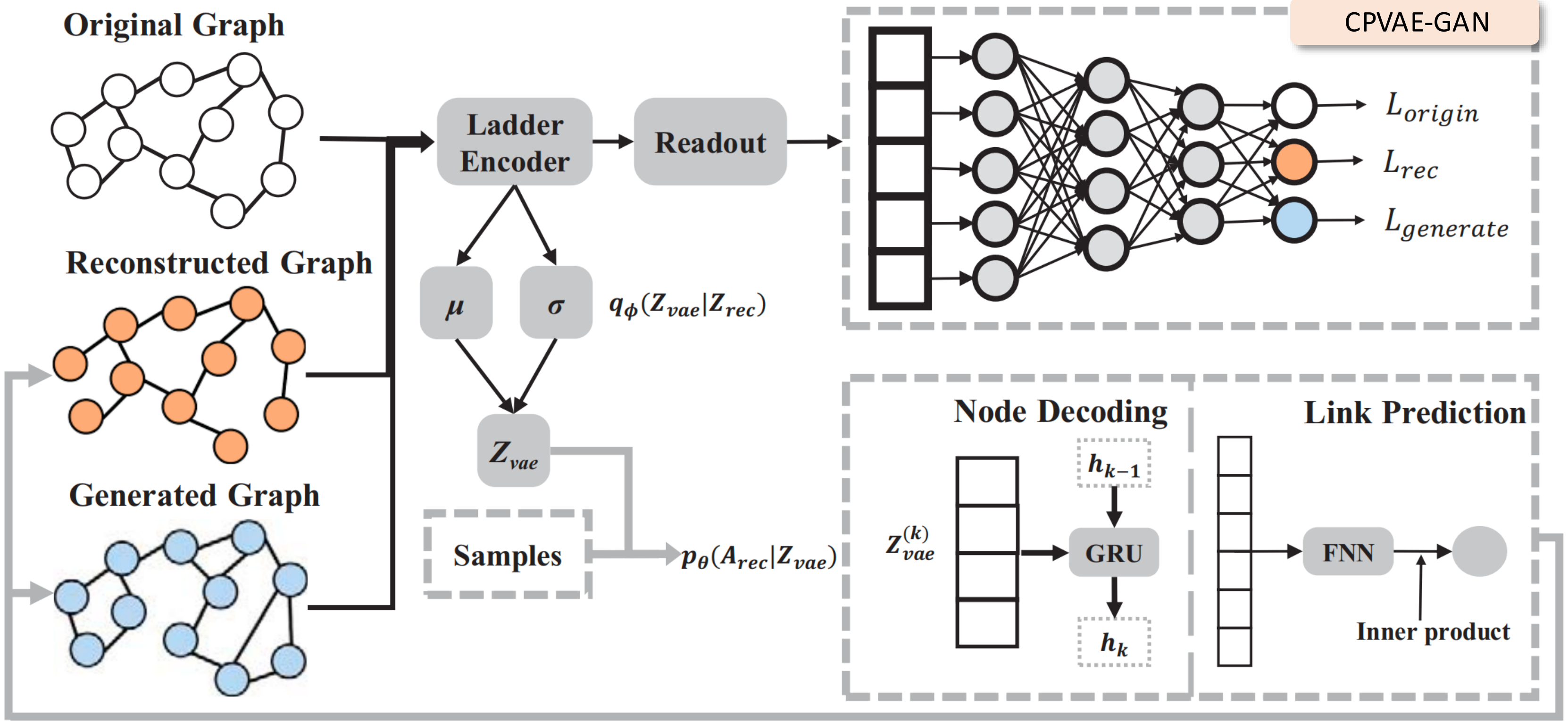


NetGAN is Random Walk-based, with low efficiency

Beyond NetGAN: Community Preserving GAN

◆ Contribution

- ☐ Community preserving graph generator: **CPVAE-GAN (CPGAN)**
- ☐ Community-preserving graph encoder: **Ladder Encoder**
- ☐ deprecate random walk sampling, leveraging autoencoder, which is efficient.



Xiang, S., Cheng, D., Zhang, J., Ma, Z., Wang, X., & Zhang, Y. (2022, May). Efficient learning-based community-preserving graph generation. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)* (pp. 1982-1994). IEEE.

Beyond NetGAN: Community-Preserving GAN

◆ Experimental Results

Graph	Citeseer		Pubmed		PPI		3D Point Cloud		Facebook		Google	
	NMI(e-2)	ARI(e-2)	NMI(e-2)	ARI(e-2)	NMI(e-2)	ARI(e-2)	NMI(e-2)	ARI(e-2)	NMI(e-2)	ARI(e-2)	NMI(e-2)	ARI(e-2)
SBM	19.7±0.9	1.9±0.1	4.4±0.2	0.3±0.1	11.3±0.7	1.2±0.1	37.0±1.3	11.4±0.7	14.5±2.0	2.1±0.3	24.4±0.9	1.3±0.4
DCSBM	27.1±0.8	1.7±0.1	18.9±0.2	0.3±0.1	18.6±0.8	1.8±0.3	37.3±1.4	11.5±0.8	17.5±1.5	1.9±0.3	29.4±0.6	5.7±0.5
BTER	27.3±0.7	1.8±0.1	19.1±0.2	0.3±0.1	19.0±0.7	1.7±0.1	38.1±1.2	12.1±0.8	17.9±1.2	2.1±0.2	30.3±0.7	5.8±0.5
MMSB	26.7±0.9	4.4±1.0	OOM	OOM	15.4±0.6	0.8±0.4	7.1±0.4	1.3±0.3	OOM	OOM	OOM	OOM
VGAE	63.0±0.4	29.0±1.5	42.0±0.3	15.0±0.4	50.4±0.6	40.0±1.2	57.0±0.8	8.2±1.1	OOM	OOM	OOM	OOM
Graphite	62.8±0.7	28.2±2.1	43.0±0.5	15.1±0.4	52.3±0.8	33.4±1.9	58.8±0.4	13.2±0.3	OOM	OOM	OOM	OOM
SBMGNN	62.6±0.5	21.5±1.0	39.3±0.5	14.1±0.5	56.9±0.4	31.0±1.6	59.2±0.9	15.9±1.1	OOM	OOM	OOM	OOM
NetGAN	57.9±0.5	20.1±0.3	OOM	OOM	55.2±0.5	30.2±0.3	67.4±0.9	37.8±2.6	OOM	OOM	OOM	OOM
CPGAN	**72.5±0.4**	**44.3±1.5**	**45.8±0.9**	**34.1±1.1**	**57.0±0.7**	**44.2±1.3**	**70.6±0.6**	**39.9±1.4**	**54.7±1.0**	**28.4±1.6**	**38.7±0.5**	**30.8±0.5**

Performance on Community-preserving graph generation

#Nodes	0.1k	1k	10k	100k
MMSB	0.11	0.91	40.3	-
Kronecker	1.39	1.55	**3.25**	**4.73**
GraphRNN-S	1.63	15.4	161	-
VGAE	**0.06**	**0.42**	9.75	-
Graphite	0.07	0.47	10.6	-
SBMGNN	0.08	0.63	12.4	-
NetGAN	0.27	2.80	31.1	-
CondGEN-R	0.18	25.3	-	-
CPGAN	0.35	0.70	6.39	32.9

Comparison on training time

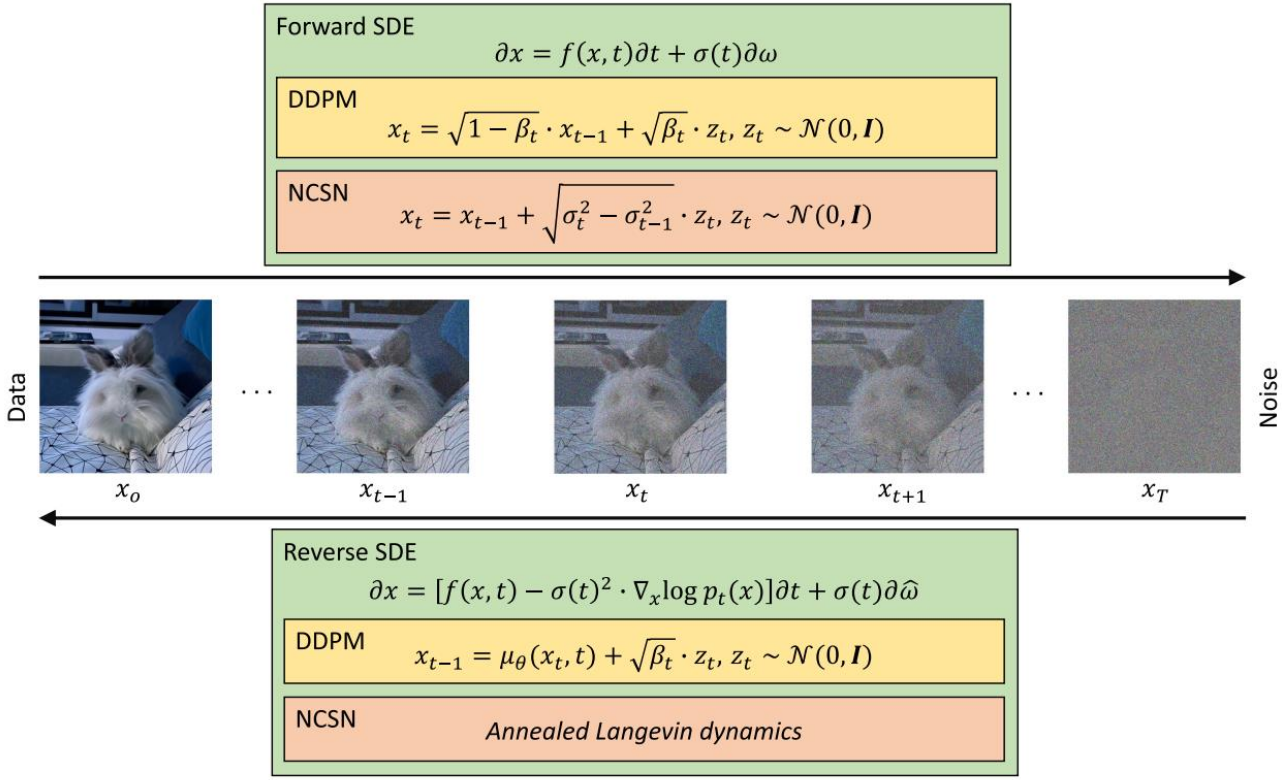
#Nodes	0.1k	1k	10k	100k
E-R	$\mathbf{4.6}e^{-4}$	$9.0e^{-3}$	0.46	10.1
B-A	$1.0e^{-3}$	$1.2e^{-2}$	**0.11**	1.17
Chung-Lu	$7.2e^{-4}$	$2.5e^{-3}$	0.18	2.38
SBM	$6.1e^{-3}$	0.09	2.58	37.1
DCSBM	$6.2e^{-3}$	0.09	2.69	39.3
BTER	$1.28e^{-3}$	$\mathbf{1.9}e^{-3}$	0.16	**0.25**
MMSB	$6.1e^{-3}$	0.09	2.56	-
Kronecker	$8.5e^{-3}$	0.08	1.00	9.69
GraphRNN-S	0.27	4.74	63.6	-
VGAE	$4.2e^{-3}$	0.04	0.38	-
Graphite	$6.1e^{-3}$	0.06	0.64	-
SBMGNN	0.01	0.11	1.18	-
NetGAN	$8.7e^{-3}$	0.09	1.12	-
CondGEN-R	$8.3e^{-3}$	0.15	-	-
CPGAN	$9.1e^{-3}$	0.08	0.95	86.1

Comparison on infernece time

Diffusion Model: DiGress

Diffusion models: Two major processes.

- **Forward process** transforms data into noise.
- **Generative process** learns to transform the noise back into data.

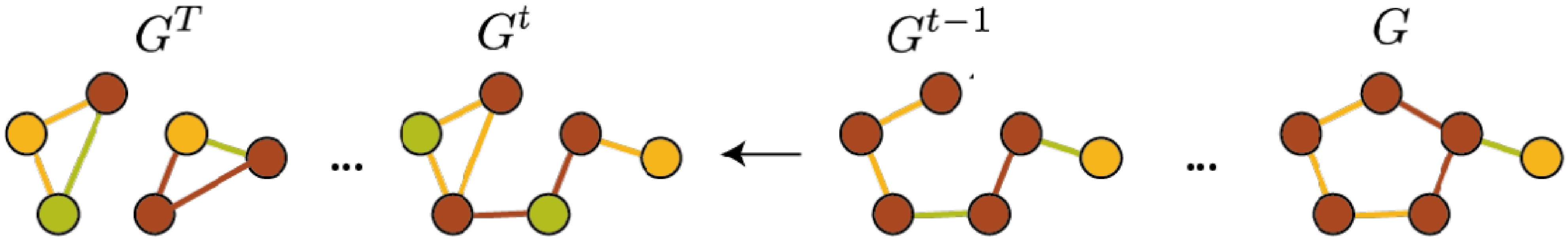


Croitoru, Florinel-Alin, et al. "Diffusion models in vision: A survey." IEEE Transactions on Pattern Analysis and Machine Intelligence 45.9 (2023): 10850-10869.

Diffusion Model: DiGress

Diffusion model for graph generation: Motivation

- Motivation for discrete diffusion: no need to predict continuous values that do not exist in the data + do not break sparsity
- Adding noise = sampling node or edge types from a categorical distribution.
- No edge = one particular edge type.
- The noise is sampled independently on each node and edge.

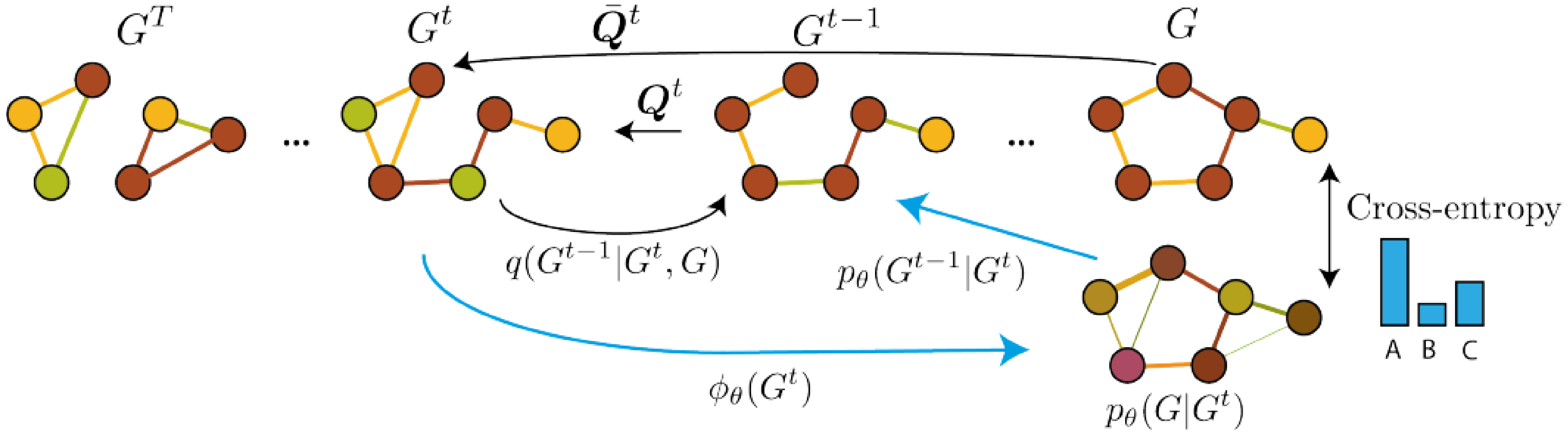


Vignac, Clément, et al. "DiGress: Discrete Denoising diffusion for graph generation." *ICLR*. 2023.

Diffusion Model: DiGress

Diffusion model for graph generation.

- **Forward process** adds noise using Markov transition matrix Q^t.
- **Generative process** learns to transform the noise back into data. A discrete G^{t-1} is sampled from the learned categorical distribution.
- Graph generation becomes a sequence of node and edge classification tasks.



Vignac, Clément, et al. "DiGress: Discrete Denoising diffusion for graph generation." *ICLR*. 2023.

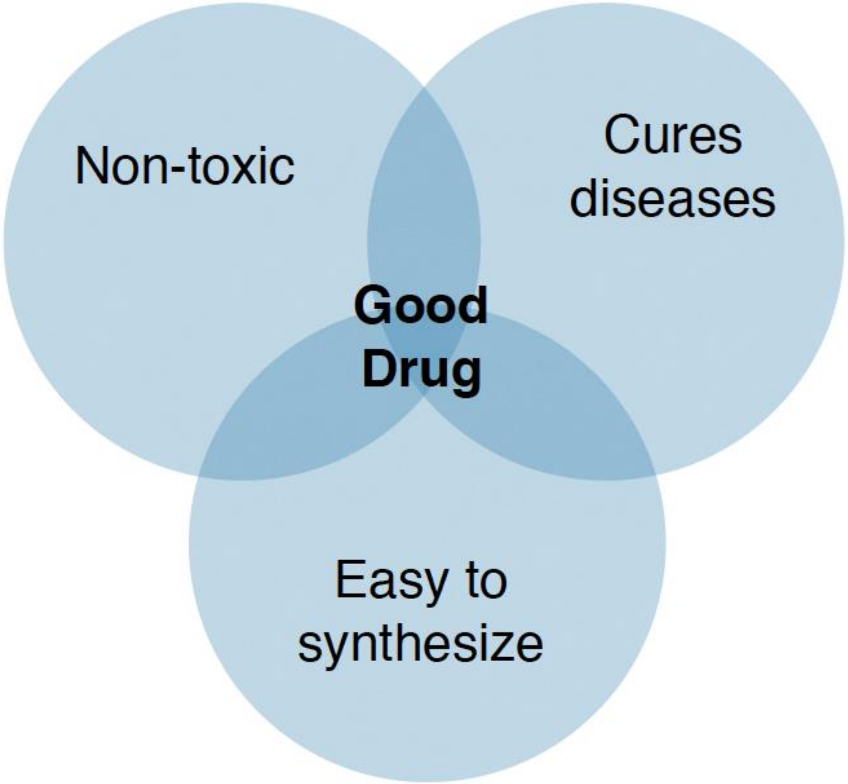
Graph Data Management

- Graph Data Quality Management
  - Data Quality Assessment
  - Data Quality Enhancement
- Graph Generation
  - Learning-based Graph Generation
  - **Function-driven Graph Generation**

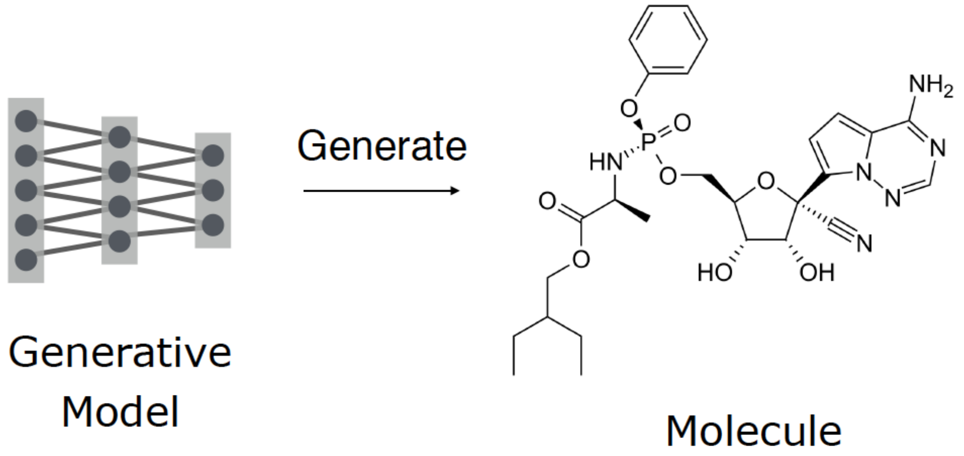
Molecular Graph Generation: applications

Drug Discovery: finding molecules with desired chemical properties.

A good drug needs to satisfy multiple objectives:



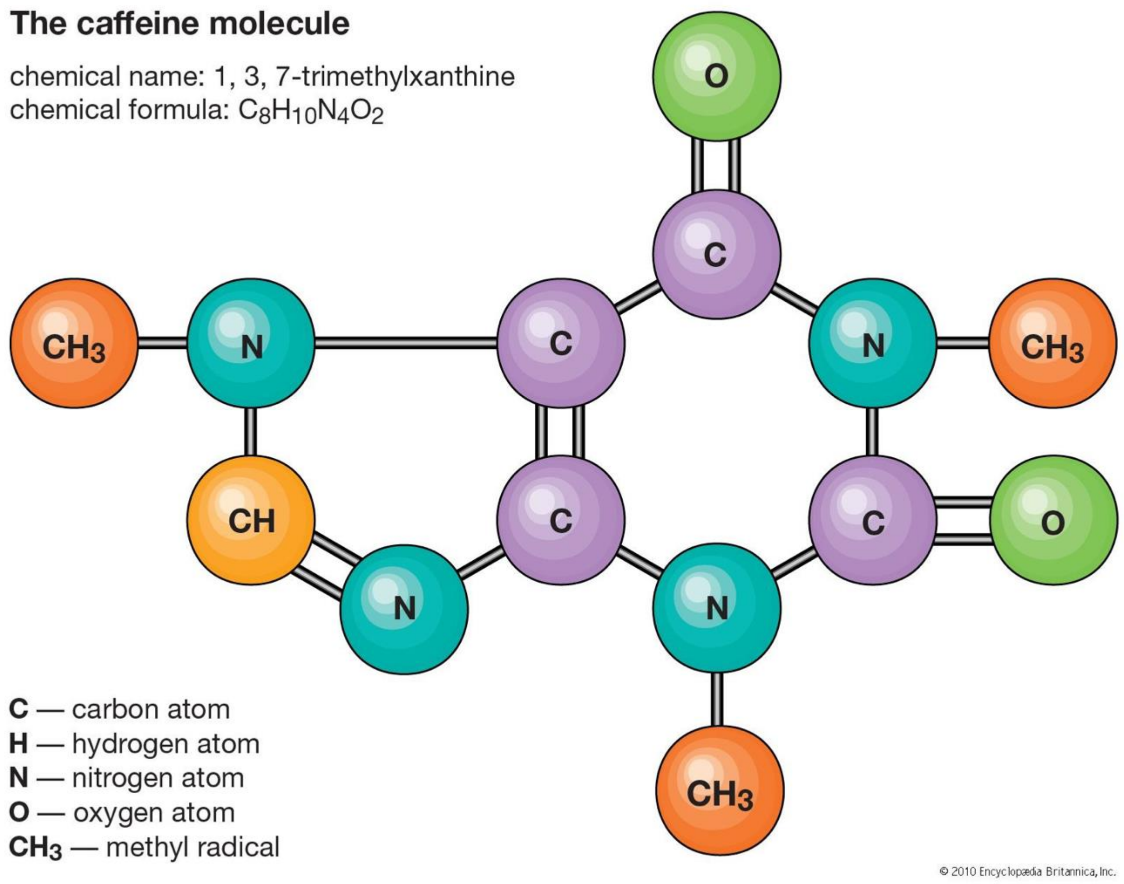
- The scale of potential drug-like molecules: $10^{33} \sim 10^{60}$
- The scale of existing chemical database: 10^{6}
- A huge gap!



Jin, Wengong, Regina Barzilay, and Tommi Jaakkola. "Multi-objective molecule generation using interpretable substructures." *ICML.* PMLR, 2020.
Zang, Chengxi, and Fei Wang. "Moflow: an invertible flow model for generating molecular graphs." *Proceedings of the 26th ACM SIGKDD.* 2020.

Molecular Graph Generation: representations

Representation of molecular graphs: graphs



Nodes: Atoms

Edges: Chemical bonds between atoms

Edwards, Carl, Qingyun Wang, and Heng Ji. "Language+ molecules." *ACL: Tutorial Abstracts*. 2024.

Molecular Graph Generation: Models

Goal of Molecule Graph Generation

Generating realistic, novel and unique molecules with desired property.

e.g. drug-likeness, octanol-water partition coefficient

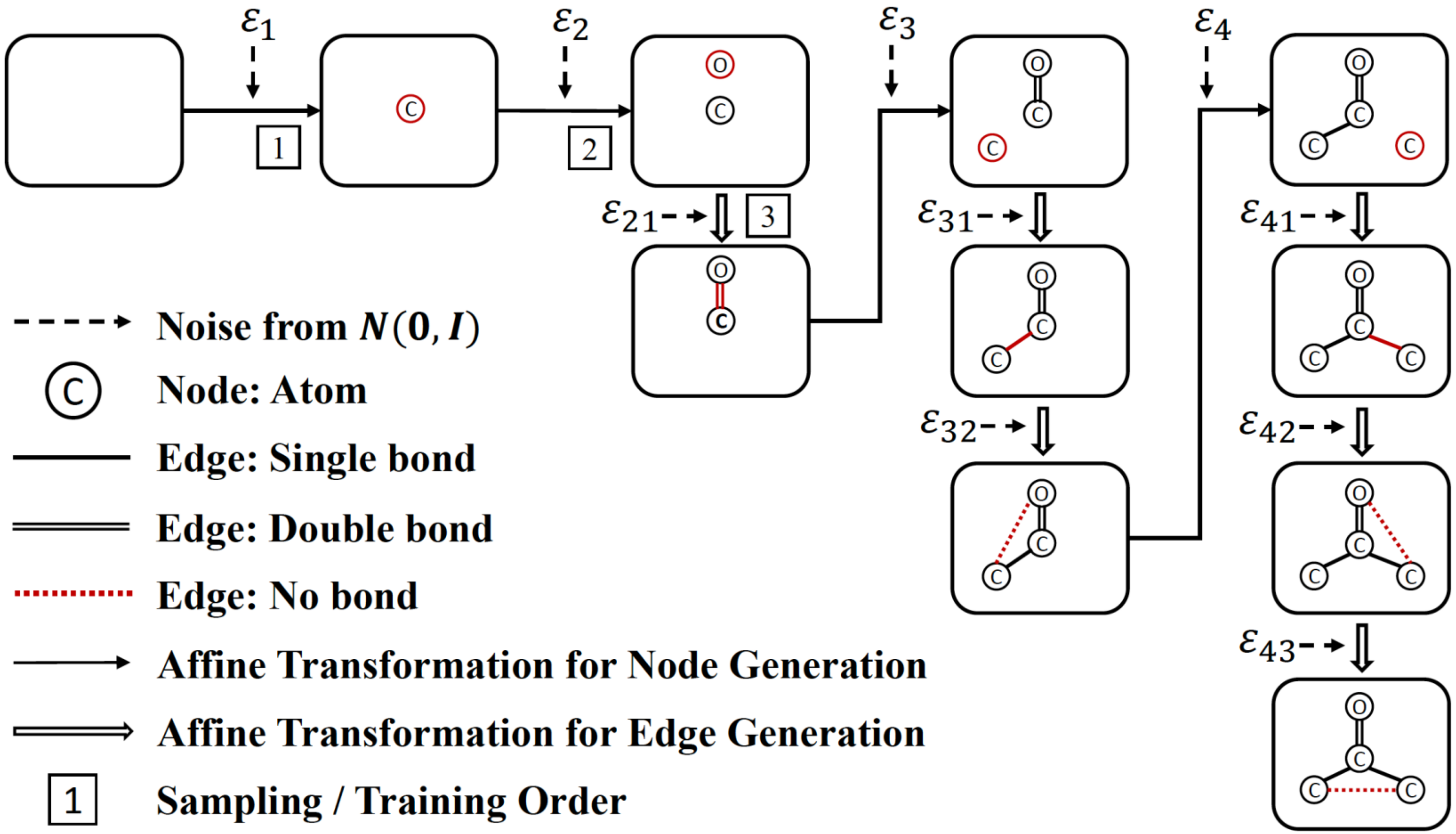
Shi, Chence, et al. "GraphAF: a Flow-based Autoregressive Model for Molecular Graph Generation." ICLR 2020.

Molecular Graph Generation: Models

GraphAF: a Flow-based Autoregressive Model for Molecular Graph Generation

Key Idea

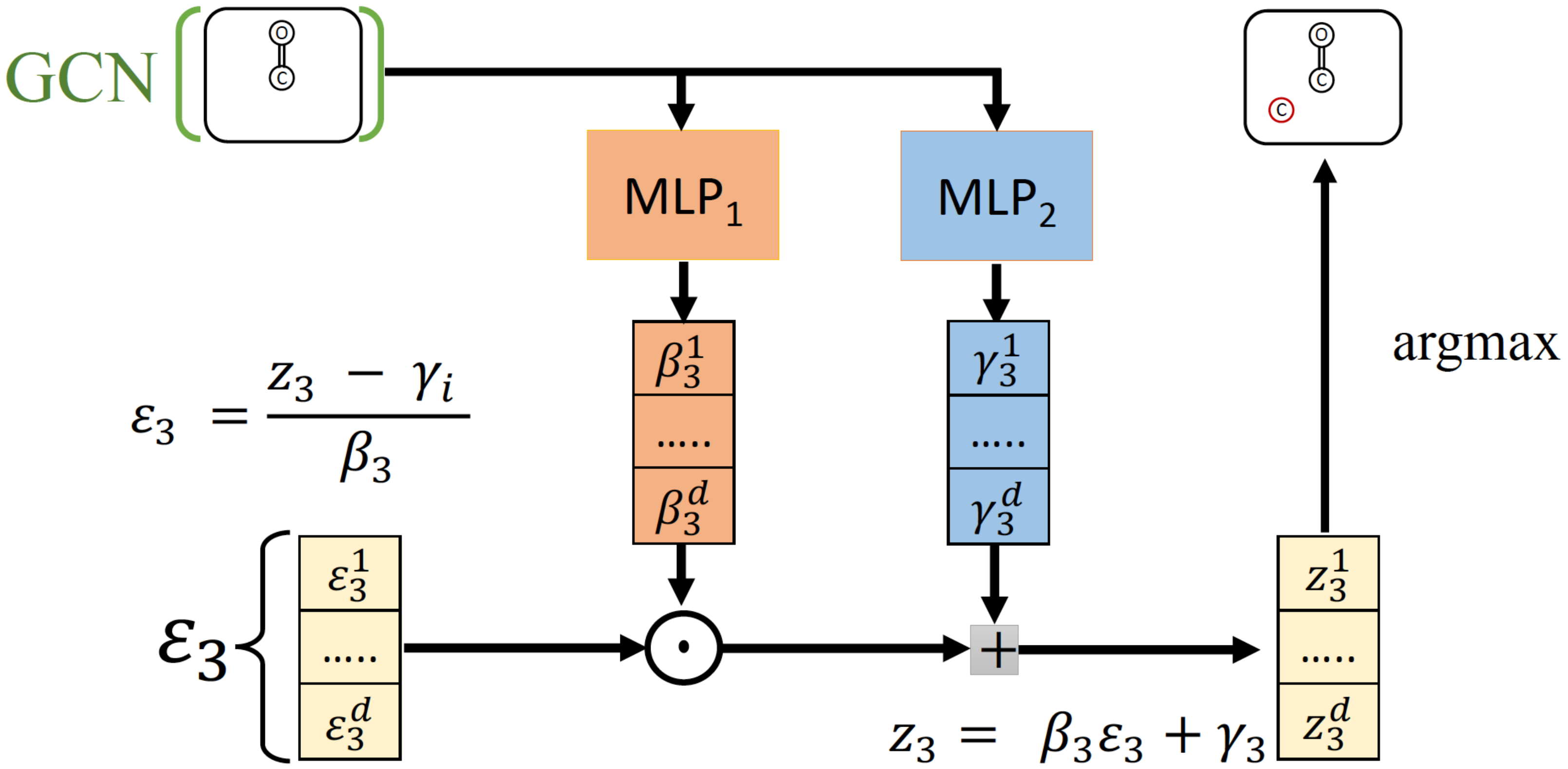
- Decompose molecular graphs into sequences
- Use autoregressive flows to model the sequences



Shi, Chence, et al. "GraphAF: a Flow-based Autoregressive Model for Molecular Graph Generation." ICLR 2020.

Molecular Graph Generation: Models

GraphAF: Model Framework



Shi, Chence, et al. "GraphAF: a Flow-based Autoregressive Model for Molecular Graph Generation." ICLR 2020.

Molecular Graph Generation: Models

GraphAF: Goal-Directed Molecule Generation with RL

For drug discovery, we also want model to be able to optimize the chemical properties of generated molecule.

- **State**: current sub-graph.
- **Policy**: autoregressive flow to generate node/edge based on current subgraph.
- **Reward**: intermediate reward and final reward

Shi, Chence, et al. "GraphAF: a Flow-based Autoregressive Model for Molecular Graph Generation." ICLR 2020.

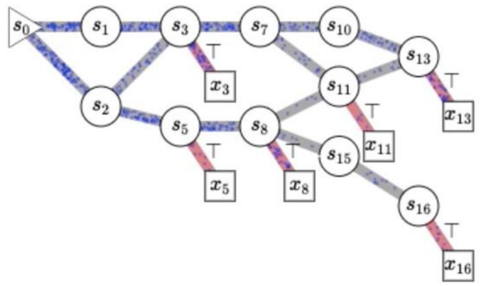
Molecular Graph Generation: Models

RGFN: Synthesizable Molecular Generation Using GFlowNets: Why FlowNet?

Generative Flow Networks (GFlowNets) are a relatively new family of generative models.

Goal: generating **high reward**, **diverse** samples in an **amortized** manner. All crucial in drug discovery!

Shortcomings of the existing methods:
MCMC - lack of amortization,
RL - mean-seeking behaviour; mode collapse.

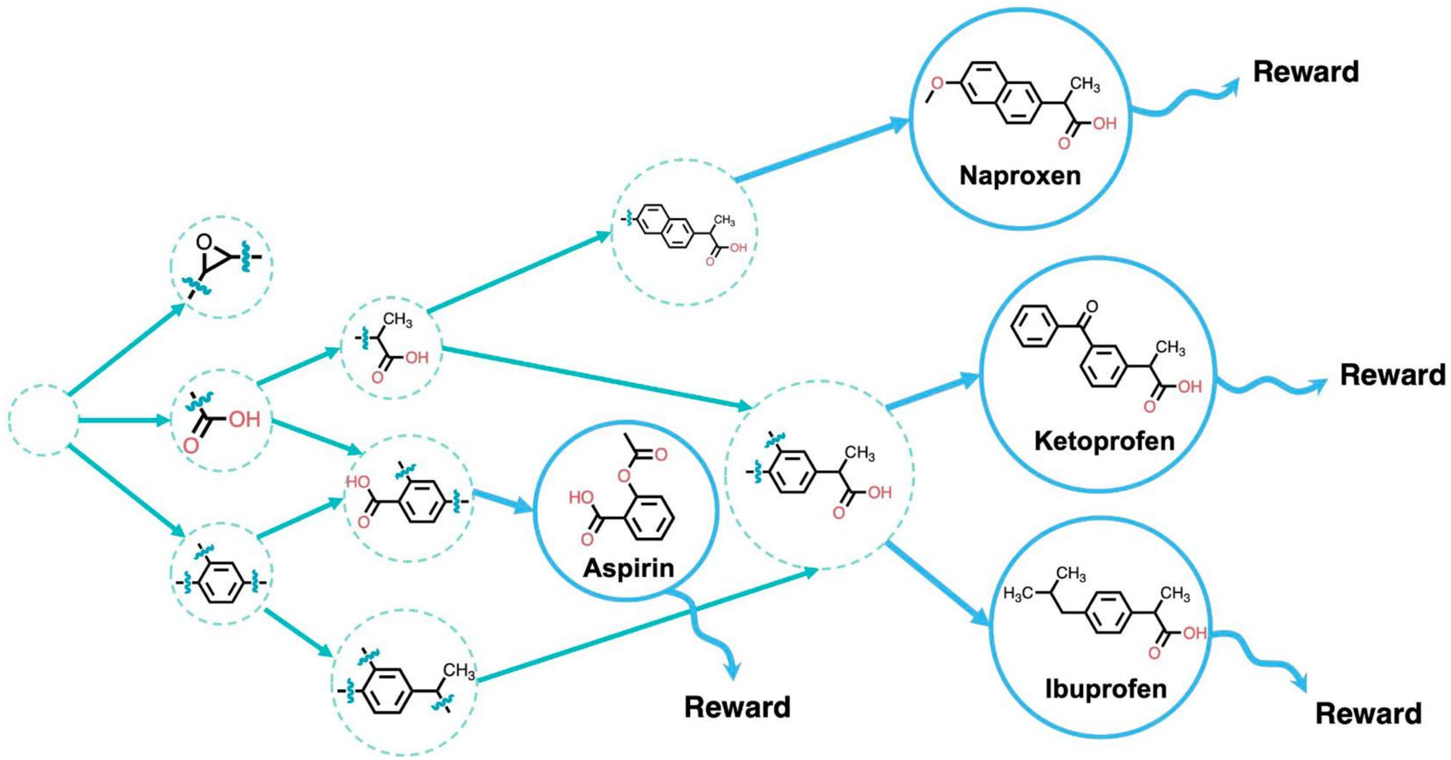


How to do it? On high level: ensure that the probability of generating a sample is proportional to its reward: ***p(x) ~ R(x)***. This can be done by training a sampling policy ***π(x)*** (a machine learning model).

Koziarski, Michał, et al. "Rgfn: Synthesizable molecular generation using gflownets." Advances in Neural Information Processing Systems 37 (2024): 46908-46955.

Molecular Graph Generation: Models

GFlowNet for Molecule Design



Key ingredients of GFlowNets:

State = current molecule
Action space = fragments to add
Reward function = property of interest

How do we ensure molecules are synthesizable?

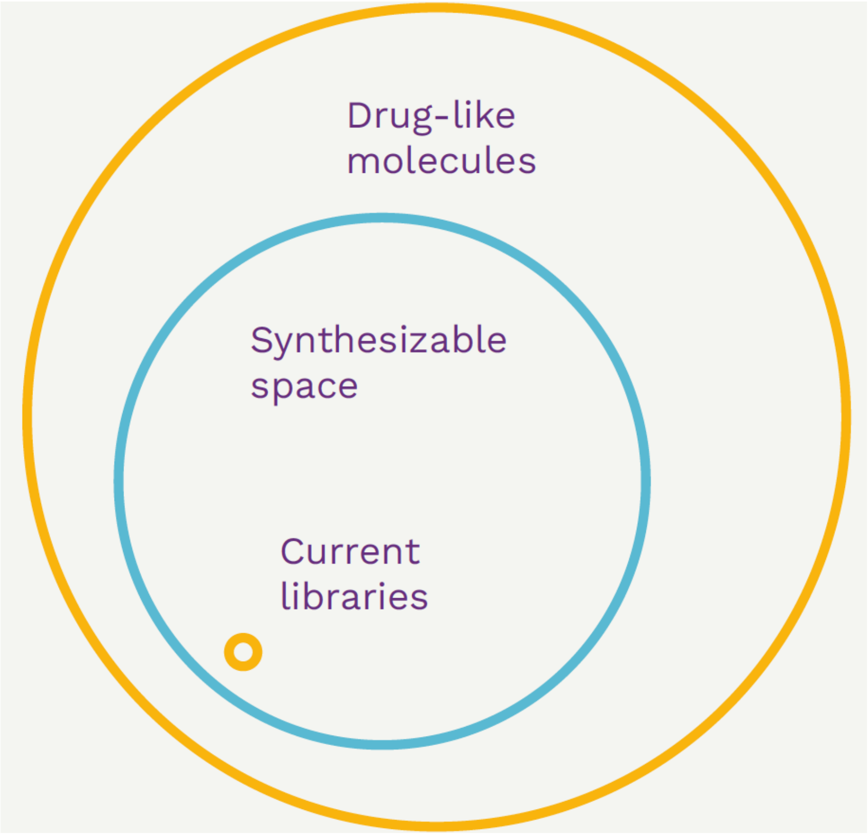
Koziarski, Michał, et al. "Rgfn: Synthesizable molecular generation using gflownets." Advances in Neural Information Processing Systems 37 (2024): 46908-46955.

Molecular Graph Generation: Models

GFlowNet for Molecule Design

The goal: constrain the searchable space to highly synthesizable compounds.

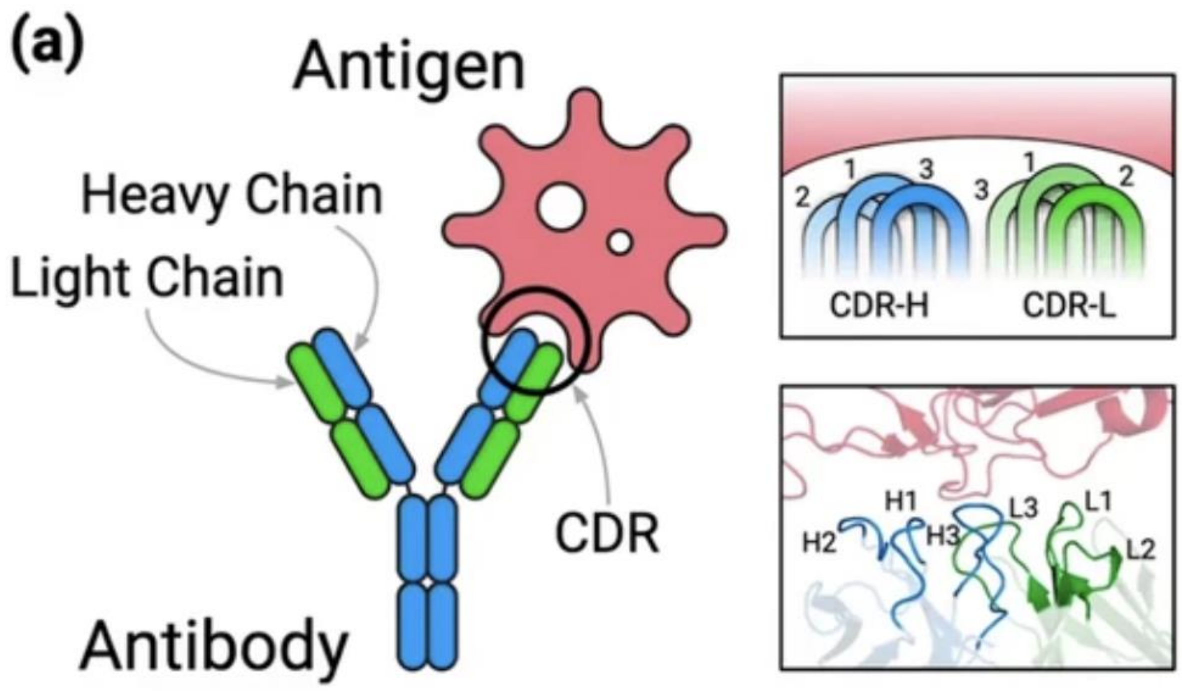
(while increasing the search space size as much as possible!)



Koziarski, Michał, et al. "Rgfn: Synthesizable molecular generation using gflownets." Advances in Neural Information Processing Systems 37 (2024): 46908-46955.

Protein Generation

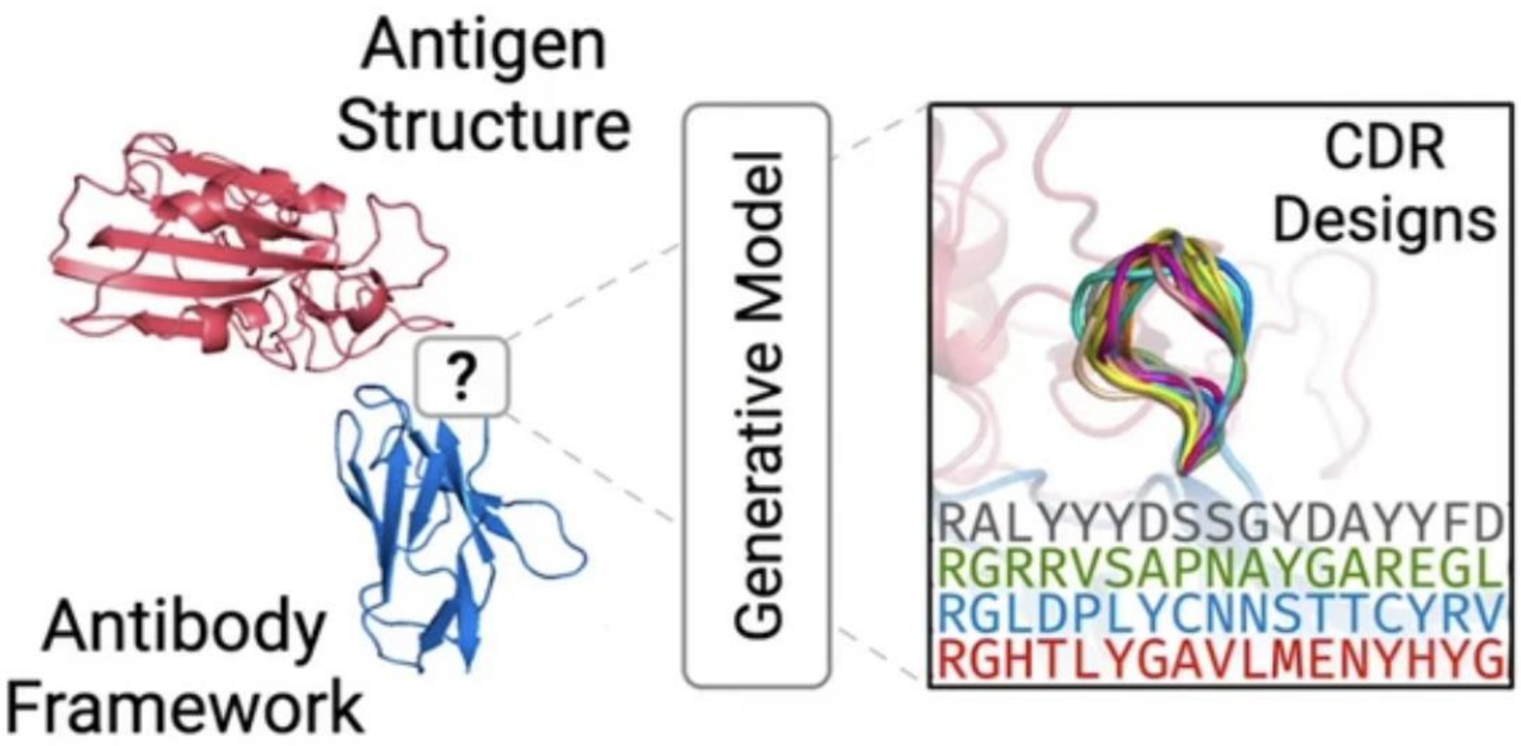
Antibody generation



Luo, Shitong, et al. "Antigen-specific antibody design and optimization with diffusion-based generative models for protein structures." NeurIPS (2022): 9754-9767.

Protein Generation

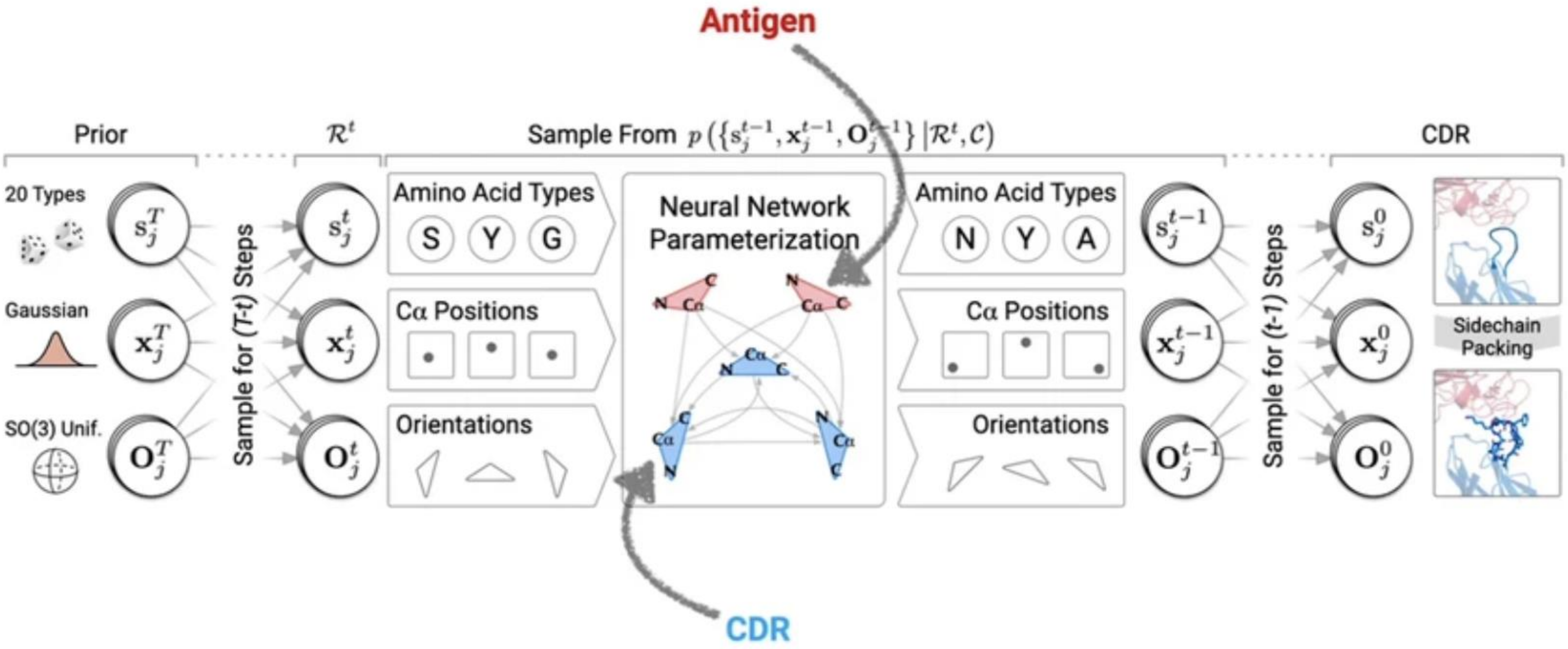
Antibody generation



Luo, Shitong, et al. "Antigen-specific antibody design and optimization with diffusion-based generative models for protein structures." NeurIPS (2022): 9754-9767.

Protein Generation

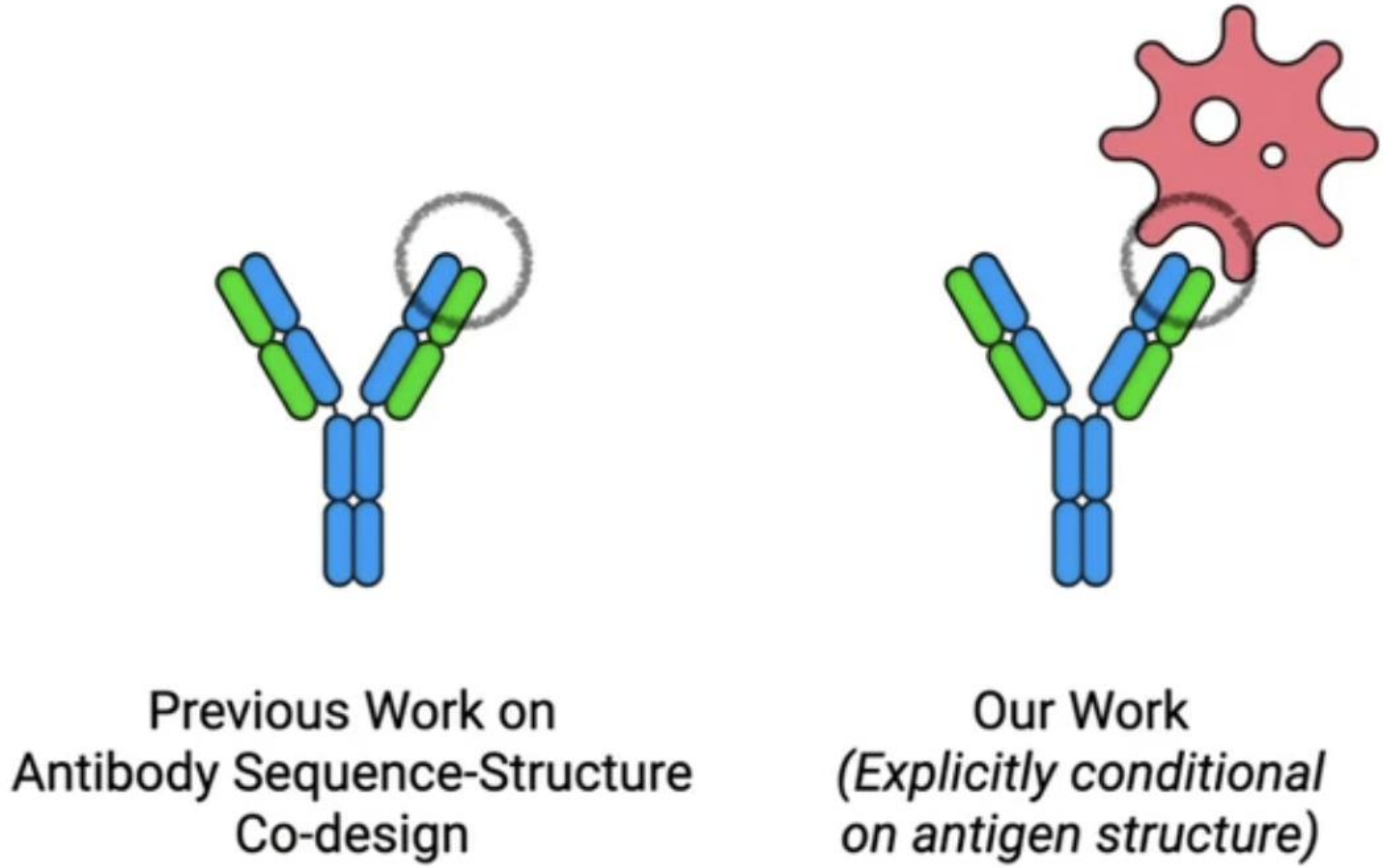
Antibody generation



Luo, Shitong, et al. "Antigen-specific antibody design and optimization with diffusion-based generative models for protein structures." NeurIPS (2022): 9754-9767.

Protein Generation

Antibody generation

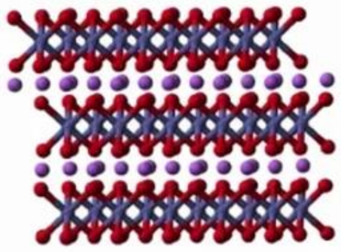


Luo, Shitong, et al. "Antigen-specific antibody design and optimization with diffusion-based generative models for protein structures." NeurIPS (2022): 9754-9767.

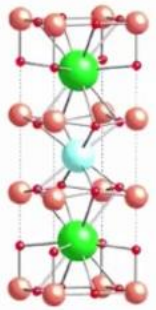
Material Graph Generation

What are Material Graphs

Materials are infinite periodic arrangements of atoms in 3D



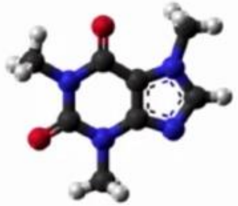
$LiCoO_2$
Cathode material for Li-ion battery
2019 Nobel Prize in Chemistry



$YBa_2Cu_3O_7$
First high-T superconductor
1987 Nobel Prize in Physics

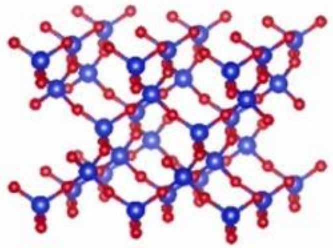
Small molecules

- Non-periodic, finite
- 5 – 10 elements
- Simple 2D graph
- Relatively simple valency rules



Materials

- Periodic, infinite
- All 94 naturally occurring elements
- Graph difficult to define
- No general valency rules



3D material structures must be directly generated, rather than relying on intermediate graphs.

Xie, Tian, et al. "Crystal Diffusion Variational Autoencoder for Periodic Material Generation." *International Conference on Learning Representations 2022.*

Material Graph Generation

Why Generate Materials?

Belsky, et al. *Acta Crystallographica Section B: Structural Science* 58.3 (2002): 364-369.

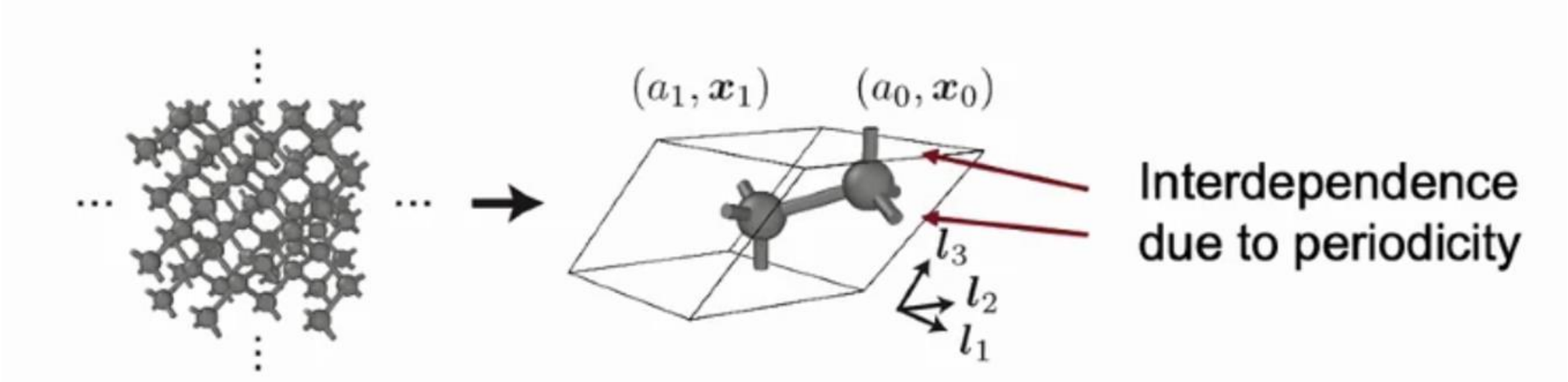
There are only **~200k** unique materials that are experimentally known (in contrast, **ZINC** includes close to a billion drug-like molecules).

Today's material discovery is centred on these **~200k known materials**. Moving beyond them could offer exciting new opportunities for multiple domains in materials science.

Xie, Tian, et al. "Crystal Diffusion Variational Autoencoder for Periodic Material Generation." *International Conference on Learning Representations 2022*.

Material Graph Generation

Representation of Periodic Materials



Key Components:
The unit cell (smallest repeating unit) of a material *M* can be fully defined by three lists:
Atom types: $A=(a_1,\ldots,a_N) \in \mathrm{A}^N$
Atom coordinates: $X = (x_1, \ldots, x_N) \in \mathbb{R}^{N\times 3}$
Periodic lattice: $L = (l_1, l_2, l_3) \in \mathbb{R}^{3\times 3}$
Infinite Periodic Structure:

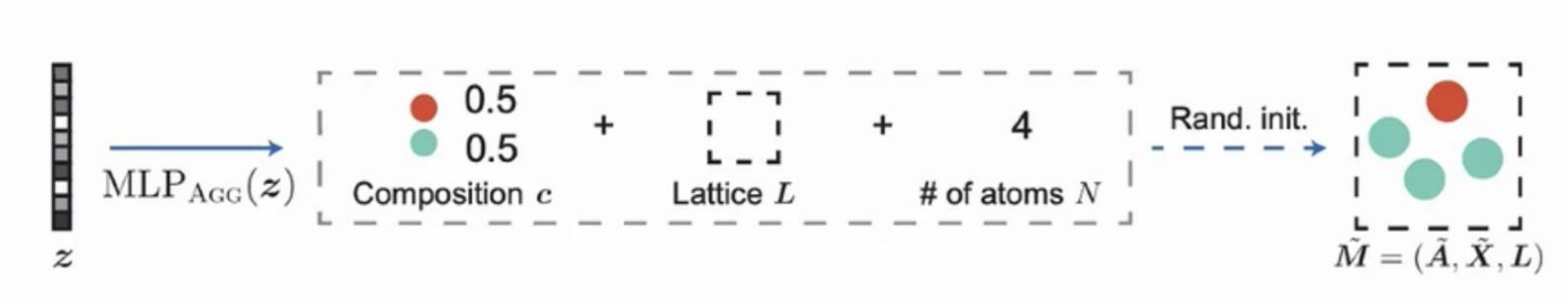
$$\{(zi', r_i') \mid zi' = zi,\ ri' = ri + k_1 l_1 + k_2 l_2 + k_3 l_3,\ k_1, k_2, k_3 \in \mathbb{Z}\}$$

This represents the repetition of the unit cell across all integer translations of the lattice vectors.
Interdependence Due to Periodicity:
The periodic lattice L and atomic coordinates X are interdependent, as the lattice defines how atoms repeat in 3D space.
Goal: Jointly generate $M = (A, X, L)$ that corresponds to a **stable material**.

Xie, Tian, et al. "Crystal Diffusion Variational Autoencoder for Periodic Material Generation." *International Conference on Learning Representations 2022*.

Material Graph Generation

Generate a close random structure



Use **$\mathbf{MLP_{AGG}(z)}$** (a neural network) to predict three aggregated properties for material generation:

Composition c: Sparse probability distribution over 100 element.

Lattice L: Rotation-invariant representation of the periodic lattice.

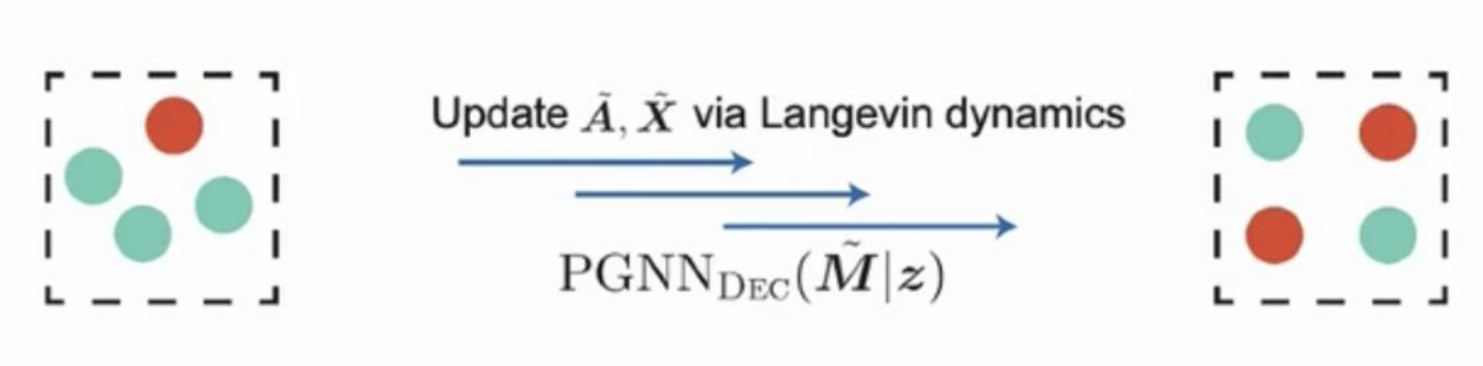
Number of atoms N: Probability distribution over possible atom counts.

Motivation: Use these easy-to-predict properties to simplify the task.

Xie, Tian, et al. "Crystal Diffusion Variational Autoencoder for Periodic Material Generation." *International Conference on Learning Representations 2022*.

Material Graph Generation

Denoise the random structure



Gradually deform $\widetilde{M}$ into a stable material structure $M = (A, X, L)$ by iteratively:

- Adjusting atom coordinates.
- Updating atom types.

Physics-Guided Design: The GNN's architecture inherently preserves physical constraints (e.g., lattice periodicity, bond lengths).

Efficiency: Focuses updates on critical regions of instability.

Xie, Tian, et al. "Crystal Diffusion Variational Autoencoder for Periodic Material Generation." *International Conference on Learning Representations 2022*.

Material Graph Generation

Generate novel realistic materials

Task: Sample from latent space to generate 10,000 materials

Evaluation metrics:

- **Validity**: Generated materials satisfy struc./comp. requirements
- **COV**: How many test materials are covered with a similar one
- **Property statistics**: Similarity of property distributions

Result: Significantly outperforming all baselines

Method	Data	Validity (%)[3] ↑		COV (%) ↑		Property Statistics ↓		
		Struc.	Comp.	R.	P.	ρ	E	# elem.
FTCP[4]	Perov-5	0.24	54.24	0.00	0.00	10.27	156.0	0.6297
	Carbon-24	0.08	–	0.00	0.00	5.206	19.05	–
	MP-20	1.55	48.37	4.72	0.09	23.71	160.9	0.7363
Cond-DFC-VAE	Perov-5	73.60	82.95	73.92	10.13	2.268	4.111	0.8373
G-SchNet	Perov-5	99.92	**98.79**	0.18	0.23	1.625	4.746	**0.03684**
	Carbon-24	99.94	–	0.00	0.00	0.9427	1.320	–
	MP-20	99.65	75.96	38.33	**99.57**	3.034	42.09	0.6411
P-G-SchNet	Perov-5	79.63	**99.13**	0.37	0.25	0.2755	1.388	0.4552
	Carbon-24	48.39	–	0.00	0.00	1.533	134.7	–
	MP-20	77.51	76.40	41.93	**99.74**	4.04	2.448	**0.6234**
CDVAE	Perov-5	**100.0**	**98.59**	**99.45**	**98.46**	**0.1258**	**0.0264**	0.0628
	Carbon-24	**100.0**	–	**99.80**	**83.08**	**0.1407**	**0.2850**	–
	MP-20	**100.0**	**86.70**	**99.15**	**99.49**	**0.6875**	**0.2778**	1.432

Xie, Tian, et al. "Crystal Diffusion Variational Autoencoder for Periodic Material Generation." *International Conference on Learning Representations 2022*.

VLDB 2025

WELCOME TO LONDON

Q & A

Thank you!

Hanchen Wang
University of Technology Sydney
hanchen.wang@uts.edu.au

Homepage: https://hanchen-wang.com/

Contributors: Hanchen Wang, Ying Zhang and Wenjie Zhang