Beyond Neighbors: Distance-Generalized Graphlets for Enhanced Graph Characterization

Yeongho Kim, Yuyeong Kim, Geon Lee, Kijung Shin

{yeongho, youyoung, geonlee0325, kijungs} @ kaist.ac.kr

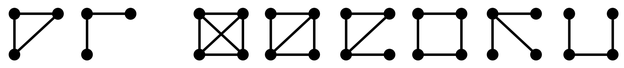
https://github.com/thisis05/EDGE

1. What is Graphlet?

[Graphlet]

- Recurring small **subgraph patterns** in graphs with *N* Nodes.
- A Key to analyze local structural patterns in complex networks.

[Examples]



3-size Graphlets 4-size Graphlets

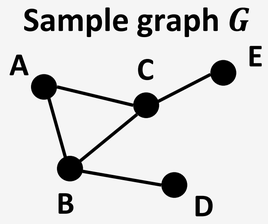
[How to use Graphlets]

- **Count each graphlet instances in graphs and normalize the counts.**
- Use the normalized values in downstream tasks.

(e.g. Graph Characterization, Graph Clustering)

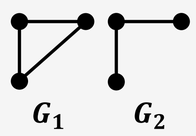
2. How to Count Graphlets ?

Sample graph G



V = {A, B, C, D, E}

3-size Graphlet



G_1 G_2

Base Approach: Enumerate all Connected Subgraphs

{A, B, C} ⟶ G_1
{A, B, D} ⟶ G_2
{A, C, E} ⟶ G_2
{B, C, D} ⟶ G_2
{B, C, E} ⟶ G_2

G_1	G_2
1	4

Efficient Approach: Combinatorial Techniques

- **Deduce the count of some graphlets based on the counts of others.**

(1) Get the count of G_1 with enumeration.

(2) Get the count of G_2 with a pre-defined combinatorial equation.

Count() = $\sum_{u \in V} \binom{|N_u|}{2}$ - 3×Count() = 7 – 3 x 1 = 4

∗ N_u : The set of neighbor nodes connected to node u

3. Limitation and Motivation to Distance-Generalized Graphlets

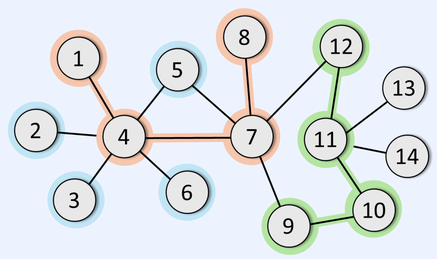
[Limitations of Original Graphlets]

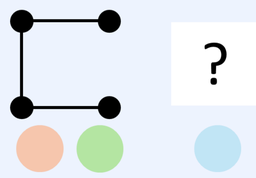
- Graphlets capture only direct connections between nodes.
- Original graphlets miss some local structures.

[Need for More Expressive Graphlets]

- We propose ***(d, s)-graphlets***, which incorporate **indirect connections up to distance d** for deeper analysis.

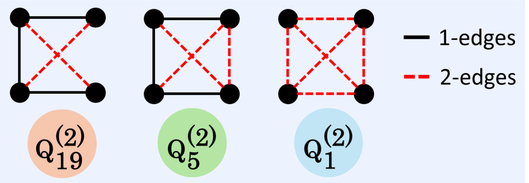
Sample graph





Size-4 Graphlets

Graphlets cannot distinguish & identify these local structures…



(2, 4)-Graphlets

Considering distance up to 2 makes them distinguishable and identifiable!

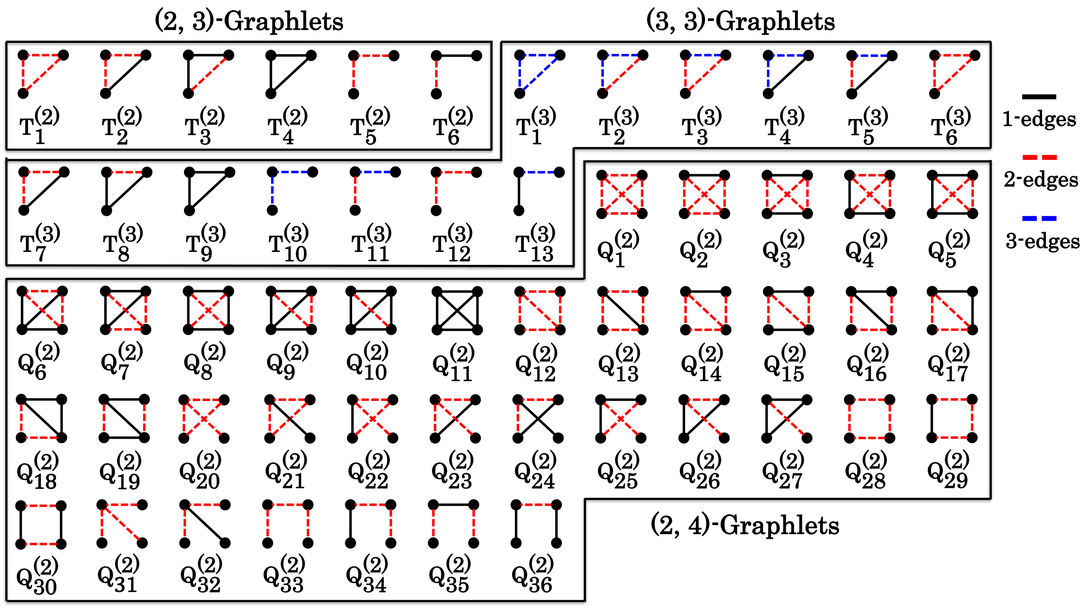
4. Proposed Concept: (d, s)-Graphlets

[Definition]

(1) d-edges: Pairs of nodes whose distance is exactly d.

(2) d-induced subgraphs: Subgraphs induced by a set of nodes that consist of 1-edge, 2-edge, … , d-edges **(up to d-edges)**.

(3) (d, s)-graphlets: d-induced subgraphs patterns with s nodes.



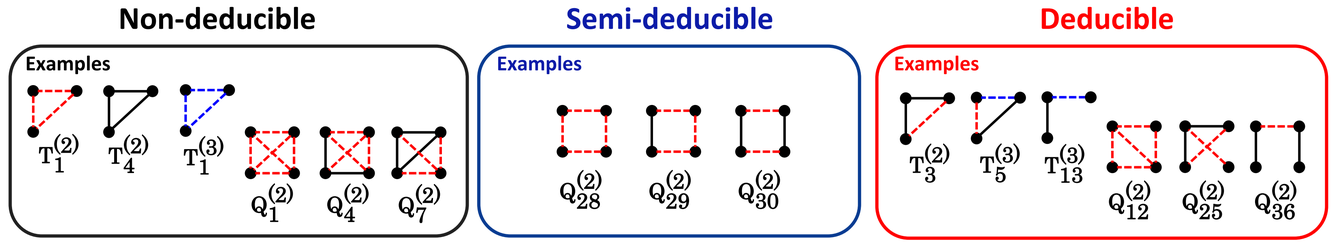
- **Size-3 graphlets T(d): 6** types for (2, 3)-graphlets / **13** types for (3, 3)-graphlets.
- **Size-4 graphlets Q(d): 36** types for (2, 4)-graphlets.

5. Proposed Counting Algorithm: EDGE

EDGE: A New Counting Algorithm for (d, s)-Graphlets Using Combinatorial Techniques

[1. Preprocessing]

- Construct **d-edges** to capture both direct and indirect (up to distance d) connections.



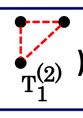
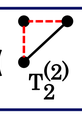
[2. Enumeration : Non-deducible / Semi-deducible (d, s)-Graphlets]

- Count **non-deducible graphlets** with enumeration and save the counts.
- Enumerate to obtain the **candidate counts of semi-deducible graphlets** and save the counts.

[3. Combinatorial Equations : Compute the counts of Deducible (d, s)-Graphlets]

- Apply **pre-defined combinatorial equations** to obtain the counts of deducible graphlets.

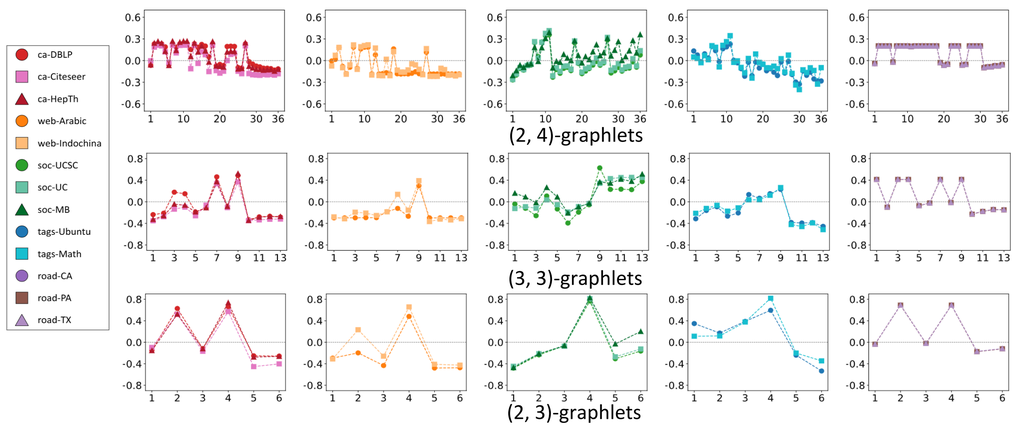
Example

Count() = $\sum_{u \in V} \binom{|N_u^{(2)}|}{2}$ - 3×Count() – Count()

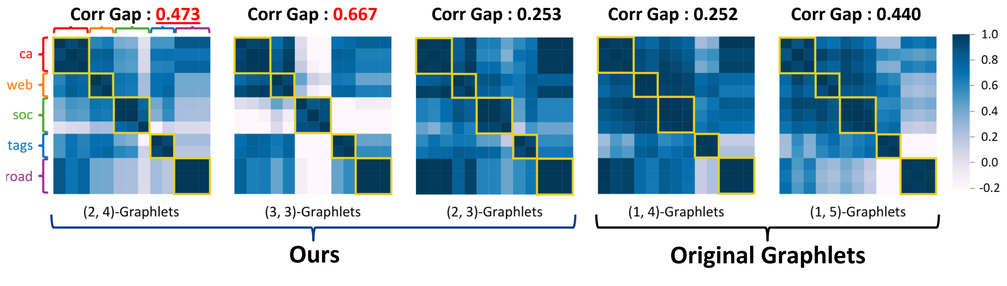
∗ $N_u^{(2)}$: The set of nodes connected to node u **by a 2-edge**

6. Experimental Results

[R1. Graph Characterization]



- Graphs within the **same domain** exhibit **similar normalized counts patterns.**

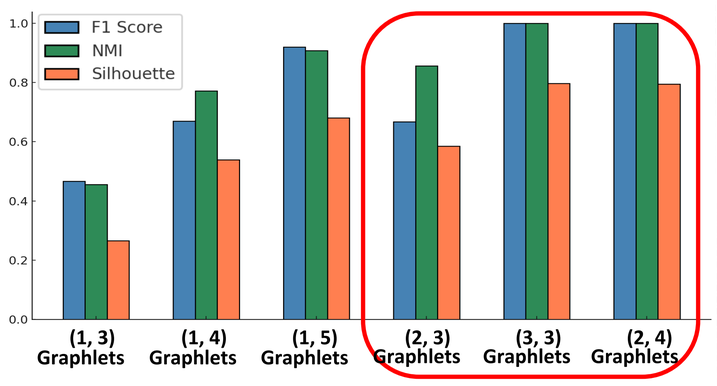


- **Domain-based similarity is much higher** for (d, s)-graphlets than for original graphlets.

➡ **(d, s)-graphlets** accurately **characterize** real-world graphs.

[R2. Graph Clustering & Link Prediction]

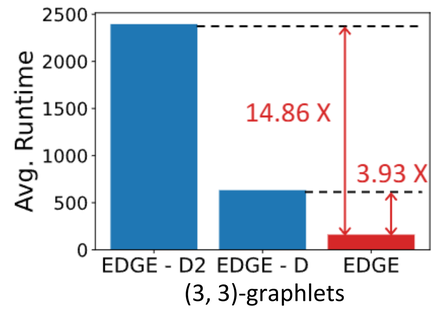
Graph Clustering



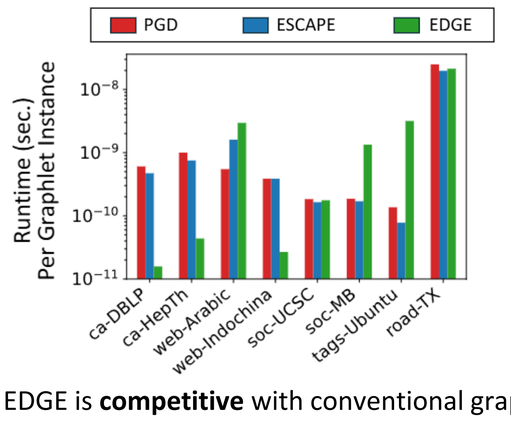
Link Prediction (w/o explicit node features)

Dataset	Metrics	(1, 3)	(1, 4)	(2, 3)	(2, 4)	(3, 3)
web-Indochina	ACC	0.720	0.772	0.774	**0.803**	0.799
	AP	0.837	0.918	0.934	**0.982**	0.965
	AUC	0.838	0.921	0.934	**0.984**	0.970
soc-UC	ACC	0.645	0.684	0.696	**0.735**	0.728
	AP	0.694	0.799	0.817	**0.866**	0.851
	AUC	0.727	0.800	0.824	**0.869**	0.859
ca-HepTh	ACC	0.696	0.733	0.736	**0.765**	0.756
	AP	0.781	0.886	0.864	**0.944**	0.916
	AUC	0.802	0.888	0.878	**0.947**	0.922

[R3. Speed and Scalability]



- EDGE achieves **up to 14.86×** speedup over **naive enumeration methods.**



- EDGE is **competitive** with conventional graphlet counting methods in **runtime per graphlet instance**.