Structural Temporal Graph Neural Networks for Anomaly Detection in Dynamic Graphs

Lei Cai[1,2], Zhengzhang Chen[2], Chen Luo[3], Jiaping Gui[4], Jingchao Ni[2],
Ding Li[5], and Haifeng Chen[2]

[1]Washington State University, [2]NEC Laboratories America,
[3]Amazon, [4]Stellar Cyber, [5]Peking University

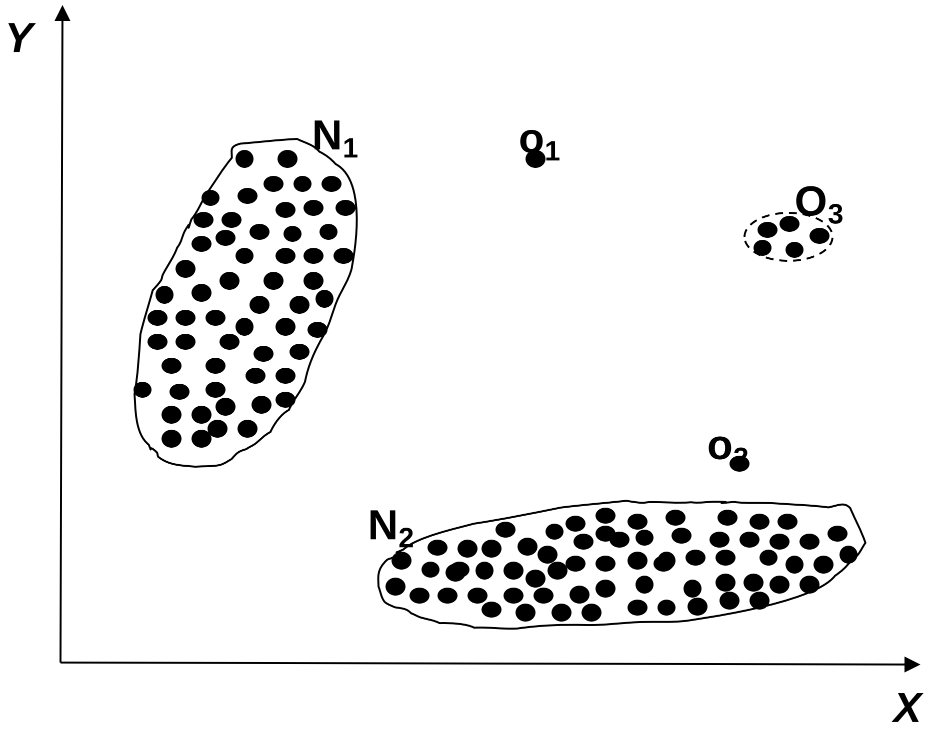
Agenda

- Background: Anomaly Detection and Anomaly Detection in Dynamic Graphs
- Existing Approaches and Challenges
- Motivation of Structural Temporal GNN (StrGNN)
- StrGNN Framework
- Experimental Results and Real-world Application
- Summary

Anomaly Detection

- Anomalies or outliers are data points within the datasets that appear to **deviate markedly from expected outputs**[1].

- Anomaly detection refers to the problem of finding patterns in data that **don't confirm to expected behavior**[2].

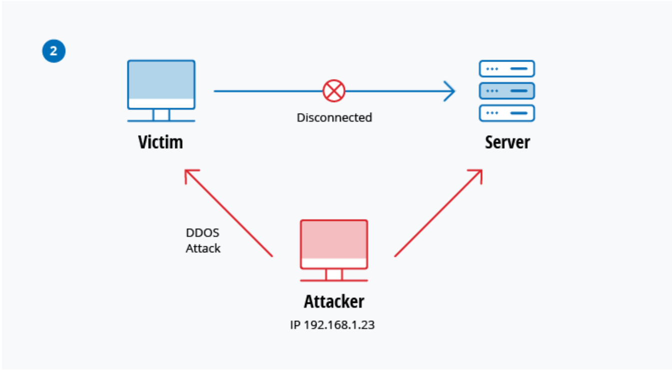


1.Anomaly Detection Techniques and Best Practices, Sri Krishnamurthy

2.Anomaly Detection: A Tutorial, Arindam Banerjee

Applications of Anomaly Detection

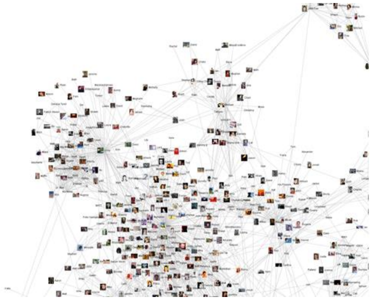
- Credit card fraud detection
- Mobile phone fraud/anomaly detection
- Insurance claim fraud detection
- Insider trading detection
- Network attack detection
- Pricing issues
- Network issues

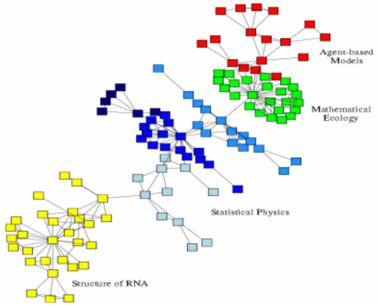


Anomaly Detection in Dynamic Graphs

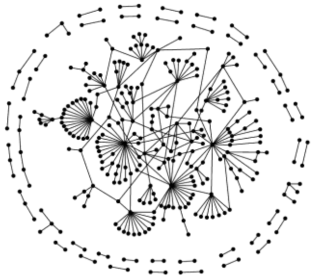
- Given a temporal network $\{G(t)=\{V(t), E(t)\}\}, t = 1, \ldots, n$, where $G(t)$ is the graph snapshot at timestamp t consisting of vertices $V(t)$ and edges $E(t)$.

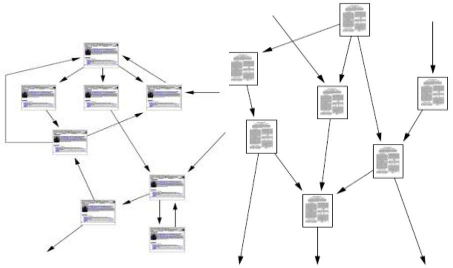
- Our goal is to detect the **anomalous edges** at any timestamp t during the testing stage.

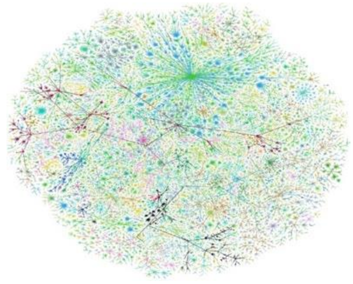

Social networks

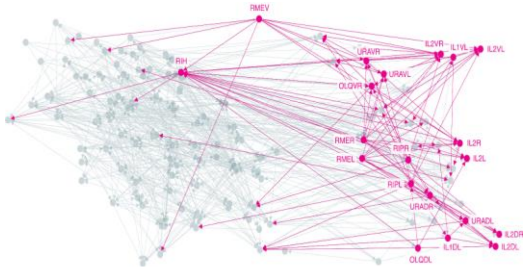


Economic networks


Biomedical networks


Information networks:
Web & citations


Internet


Networks of neurons

http://snap.stanford.edu/proj/embeddings-www/

Existing Methods and Challenges

- Two-stage approaches for anomaly detection:
  - Stage 1: data-specific features or low-dimensional representations learned from dynamic graphs.
  - Stage 2: a traditional anomaly detector, such as the support vector machines and the local outlier factor algorithm, is applied to identify anomalies.

⇨ **End-to-end framework is highly desired to improve the models**

- Most existing graph embedding approaches are designed for static graphs, and thus may not be suitable for a dynamic environment, in which the network representation has to be constantly updated.

⇨ **Effective graph neural network to learn informative features from dynamic graphs**

Motivations

(A)

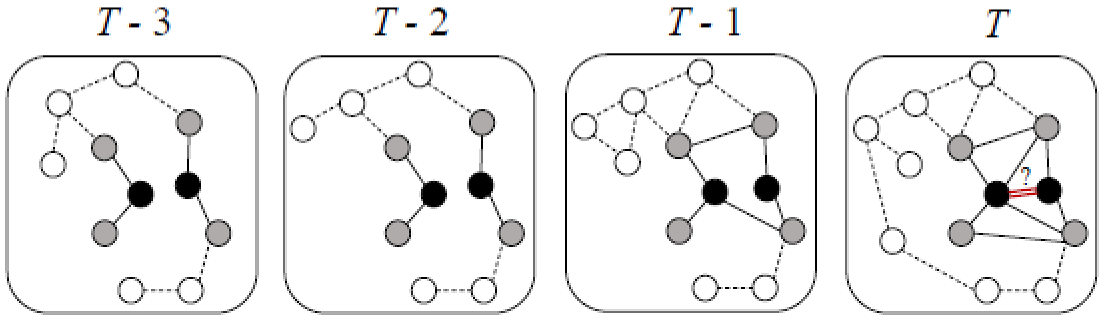


Figure (A): the interactions between nodes of the subgraph (*i.e.*, gray nodes) become more frequent. Therefore, the target edge in Figure (A) is reasonable to be a normal edge.

(B)

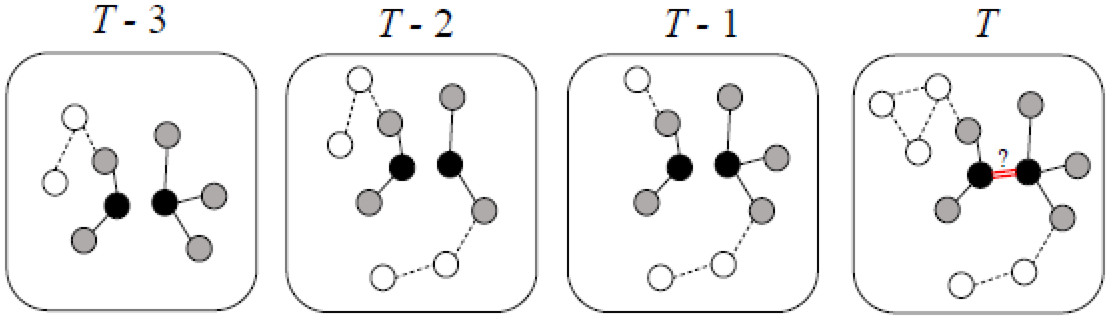
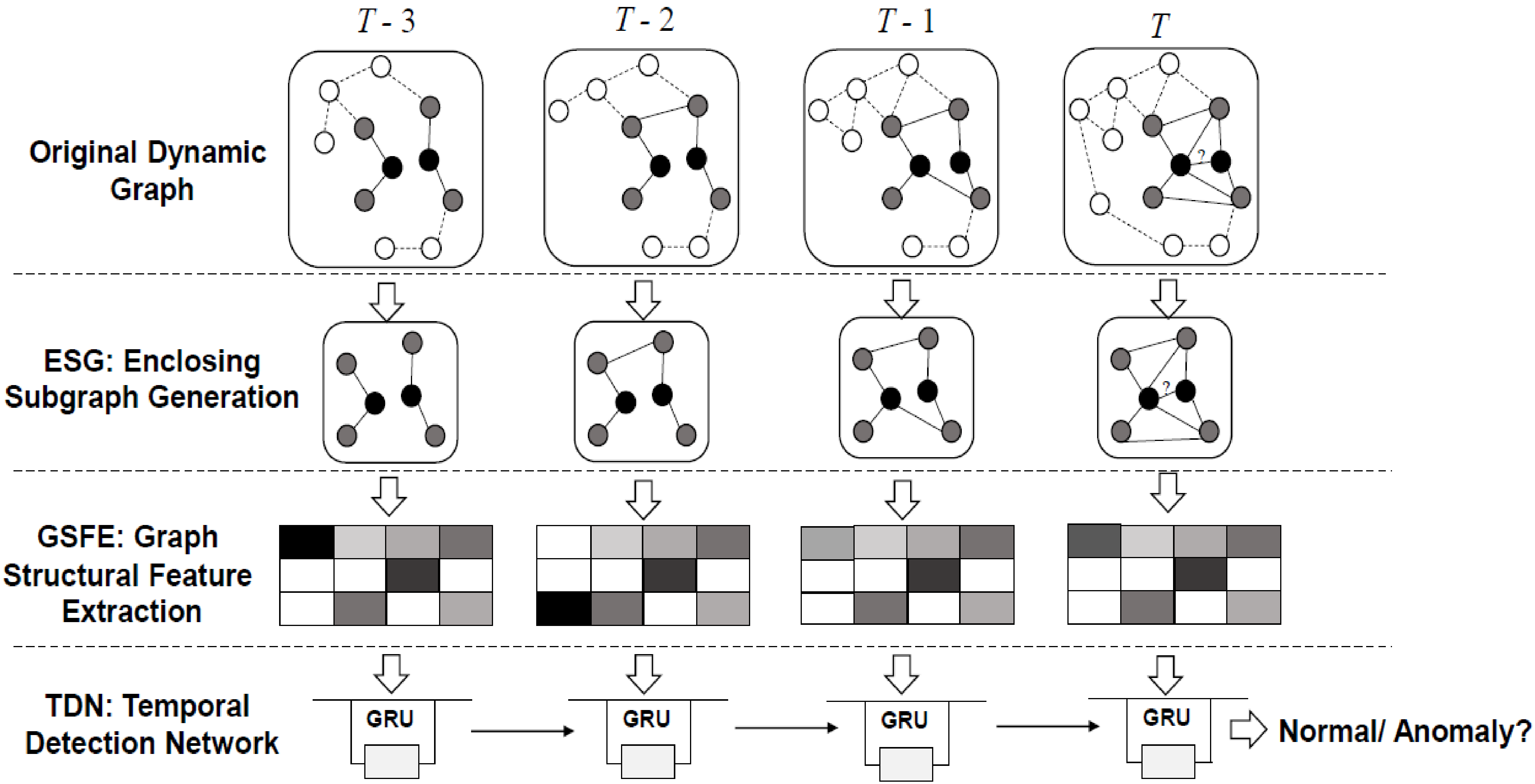


Figure (B): there are no interactions between the neighbors of the subgraph from timestamp t-3 to t-1. Therefore, the target edge at timestamp t is more likely to be an anomalous edge.

Structural temporal dynamics are key to understanding system behavior

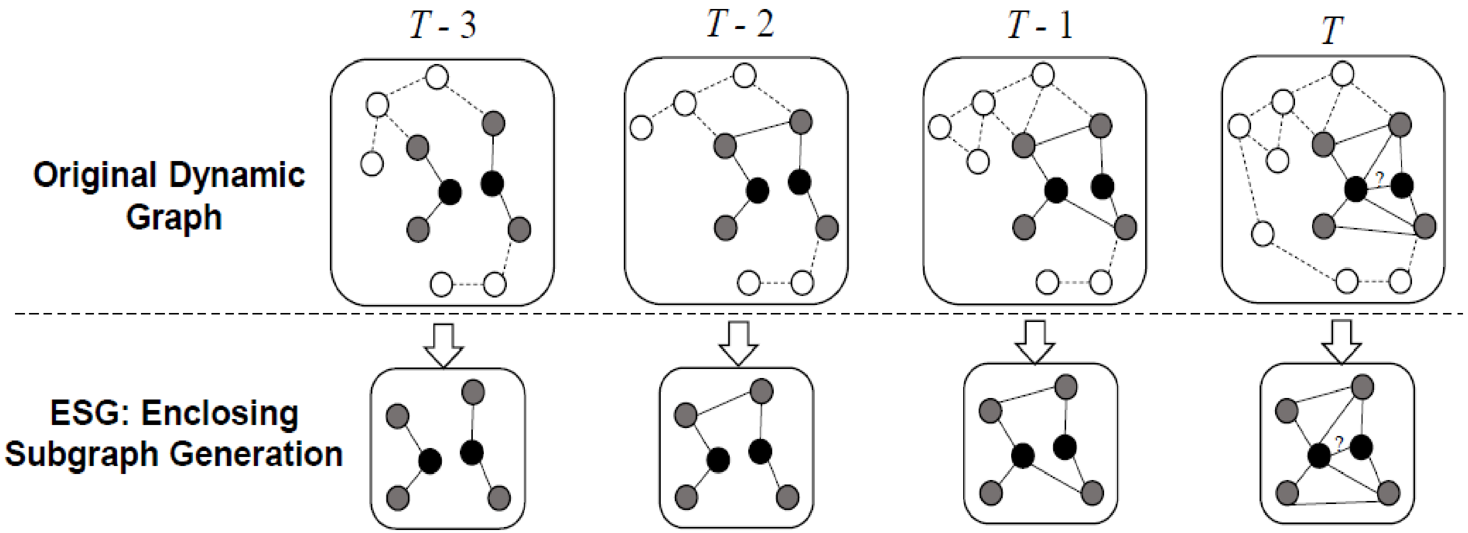
StrGNN Framework



Enclosing Subgraph Generation

- Enclosing subgraph in static graphs: For a static network G = (V, E) , given a target edge e with source node x and destination node y, the h–hop enclosing subgraph $G^{h}{}_{x,y}$ centered on edge e can be obtained by $\{i \mid d(i, x) \leq h \lor d(i,y) \leq h\}$, where $d(i, x)$ is the **shortest path distance** between node i and node x.

- Enclosing subgraph in dynamic graphs: For a temporal network $\{G(i) = \{V(i), E(i)\}\}$, where $i = t\text{-}w+1$ to t and w is window size, given a target edge e^t with source node x^t and destination node y^t, the h-hop enclosing subgraph centered on edge e^t is **a collection of all subgraphs** centered on e^t in the temporal network.



Node Labeling

- Node labeling function should convey the following information:
  - Which edge is **the target edge in the current subgraph**
  - **The contribution of each node** in identifying the category of each edge

Node Label Function:

$$f(i, x, y) = 1 + \min(d(i, x), d(i, y)) + (d_{sum}/2)[(d_{sum}/2)+(d_{sum}\%2)-1]$$

$d(i, y)$

$d(i, x)$

	1	2	3	4	5
1	2	3	4	6	8
2		5	7	9	12
3			10	13	16
4				17	21
5					26

Graph Structural Feature Extraction

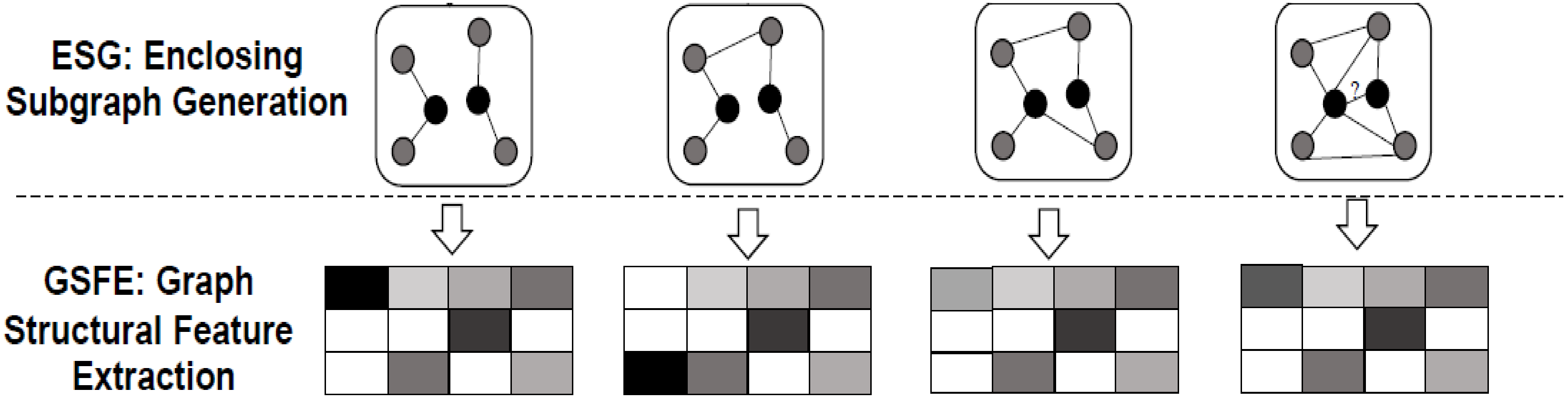
- Graph Convolution layers

$$G(X, A) = \sigma(\hat{D}^{-1/2}\hat{A}\hat{D}^{-1/2}XW)$$

- Graph Pooling layers

$$S(H_i, A) = \sigma(\hat{D}^{-1/2}\hat{A}\hat{D}^{-1/2}H_iW^1)$$

X	Node input
A	Adjacency matrix
W	Weight matrix
W^1	Weight matrix with 1 output channel
$\sigma(\cdot)$	Activation function

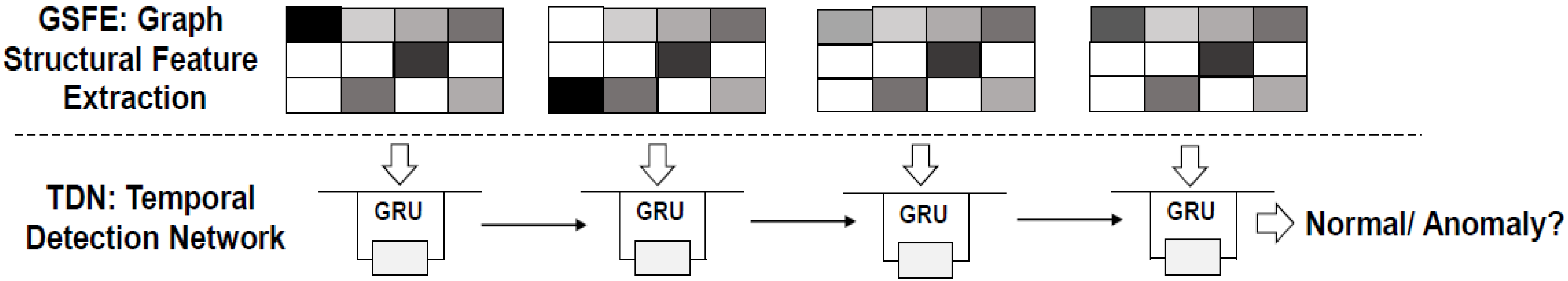


Temporal Detection Network

- Gated Recurrent Units (GRUs) to capture temporal information

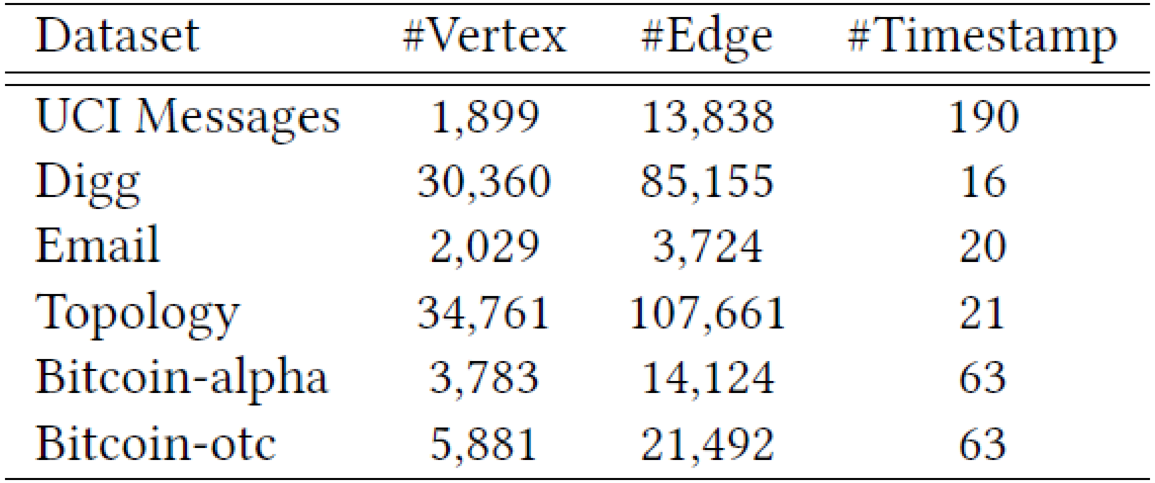
$$
\begin{aligned}
z_t &= \sigma(W_z \hat{H}_t + U_z h_{t-1} + b_z) \\
r_t &= \sigma(W_r \hat{H}_t + U_r h_{t-1} + b_r) \\
h_t^{'} &= \tanh(W_h \hat{H}_t + U_h (r_t \circ h_{t-1}) + b_h) \\
h_t &= z_t \circ h_{t-1} + (1 - z_t) \circ h_t^{'},
\end{aligned}
$$

Where $\hat{H}_t$ is the output of graph pooling layer at timestamp *t*



Experimental Setting

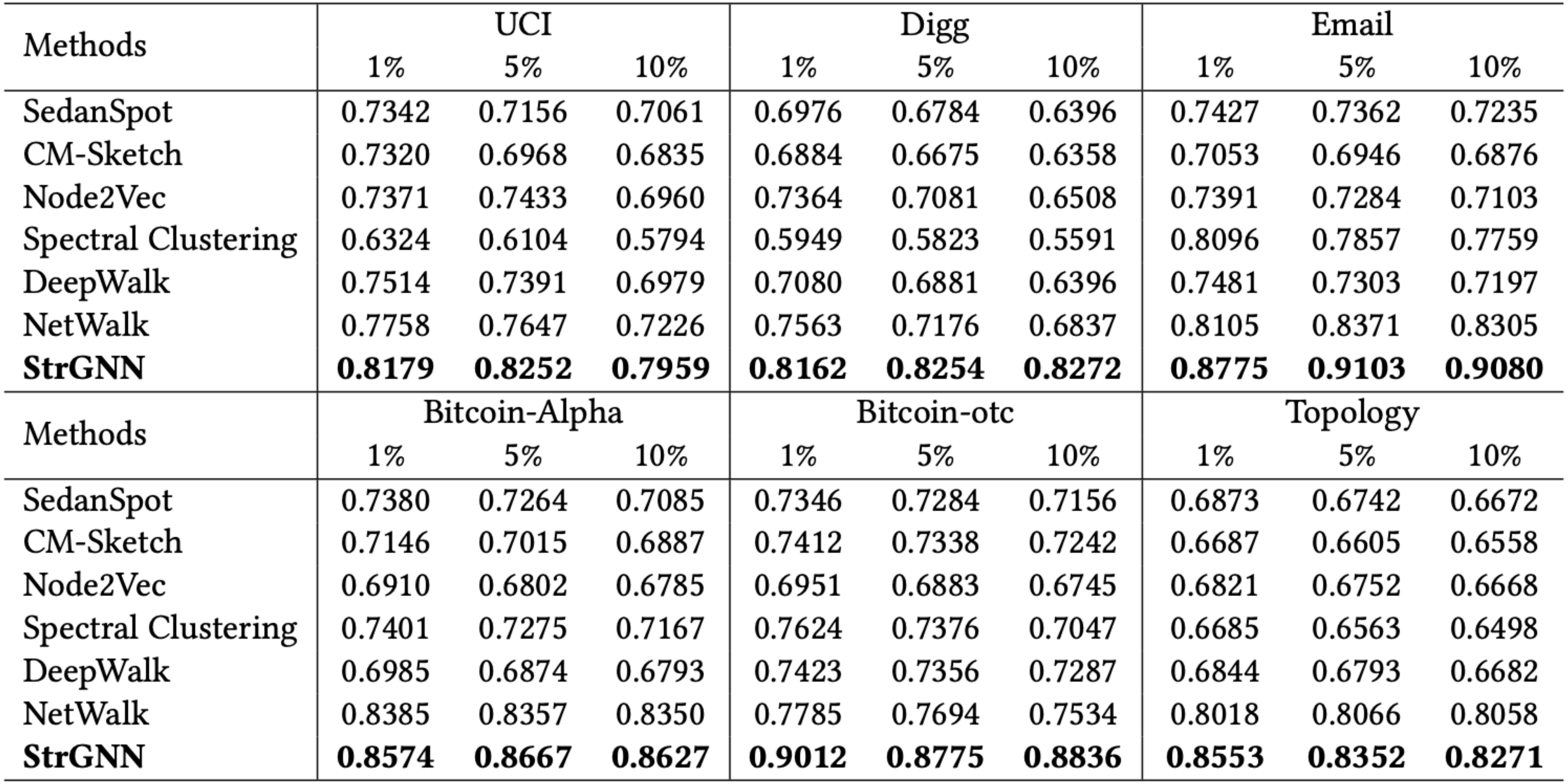
- Datasets



Dataset	#Vertex	#Edge	#Timestamp
UCI Messages	1,899	13,838	190
Digg	30,360	85,155	16
Email	2,029	3,724	20
Topology	34,761	107,661	21
Bitcoin-alpha	3,783	14,124	63
Bitcoin-otc	5,881	21,492	63

- Baselines
  - Two traditional graph anomaly detection methods: SedanSpot and CM-Sketch
  - Four network embedding methods: Node2Vec, Spectral Clustering, DeepWalk, and NetWalk
- Evaluation Metrics
  - AUC: the area under the ROC curve

Results On Six Benchmark Datasets



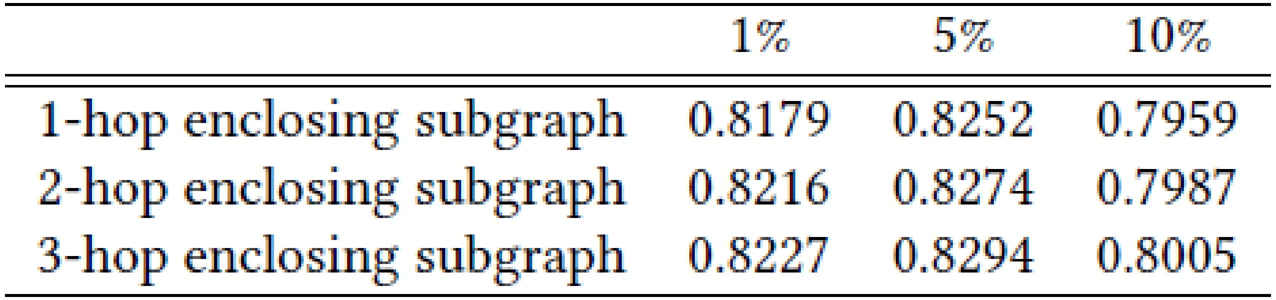
Methods	UCI			Digg			Email		
	1%	5%	10%	1%	5%	10%	1%	5%	10%
SedanSpot	0.7342	0.7156	0.7061	0.6976	0.6784	0.6396	0.7427	0.7362	0.7235
CM-Sketch	0.7320	0.6968	0.6835	0.6884	0.6675	0.6358	0.7053	0.6946	0.6876
Node2Vec	0.7371	0.7433	0.6960	0.7364	0.7081	0.6508	0.7391	0.7284	0.7103
Spectral Clustering	0.6324	0.6104	0.5794	0.5949	0.5823	0.5591	0.8096	0.7857	0.7759
DeepWalk	0.7514	0.7391	0.6979	0.7080	0.6881	0.6396	0.7481	0.7303	0.7197
NetWalk	0.7758	0.7647	0.7226	0.7563	0.7176	0.6837	0.8105	0.8371	0.8305
StrGNN	**0.8179**	**0.8252**	**0.7959**	**0.8162**	**0.8254**	**0.8272**	**0.8775**	**0.9103**	**0.9080**

Methods	Bitcoin-Alpha			Bitcoin-otc			Topology		
	1%	5%	10%	1%	5%	10%	1%	5%	10%
SedanSpot	0.7380	0.7264	0.7085	0.7346	0.7284	0.7156	0.6873	0.6742	0.6672
CM-Sketch	0.7146	0.7015	0.6887	0.7412	0.7338	0.7242	0.6687	0.6605	0.6558
Node2Vec	0.6910	0.6802	0.6785	0.6951	0.6883	0.6745	0.6821	0.6752	0.6668
Spectral Clustering	0.7401	0.7275	0.7167	0.7624	0.7376	0.7047	0.6685	0.6563	0.6498
DeepWalk	0.6985	0.6874	0.6793	0.7423	0.7356	0.7287	0.6844	0.6793	0.6682
NetWalk	0.8385	0.8357	0.8350	0.7785	0.7694	0.7534	0.8018	0.8066	0.8058
StrGNN	**0.8574**	**0.8667**	**0.8627**	**0.9012**	**0.8775**	**0.8836**	**0.8553**	**0.8352**	**0.8271**

StrGNN outperforms all baseline methods.

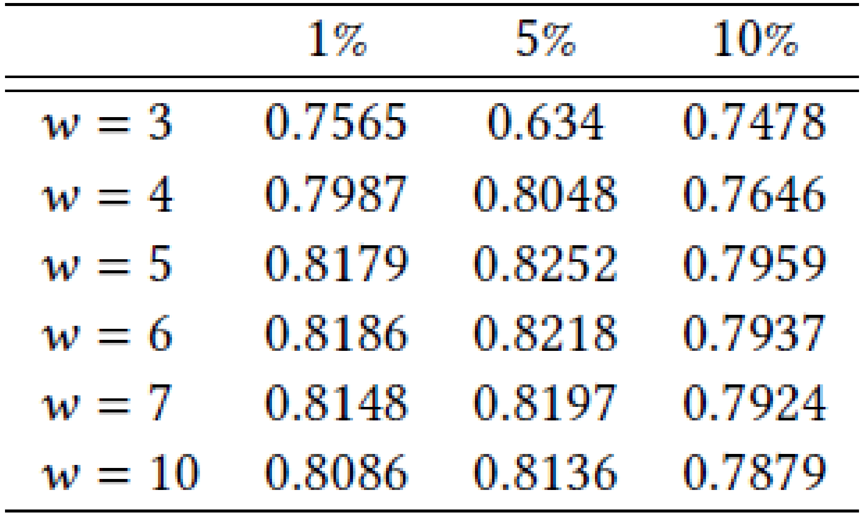
Hyperparameter Analysis

AUC results with different hops of enclosing subgraph on UCI Messages



	1%	5%	10%
1-hop enclosing subgraph	0.8179	0.8252	0.7959
2-hop enclosing subgraph	0.8216	0.8274	0.7987
3-hop enclosing subgraph	0.8227	0.8294	0.8005

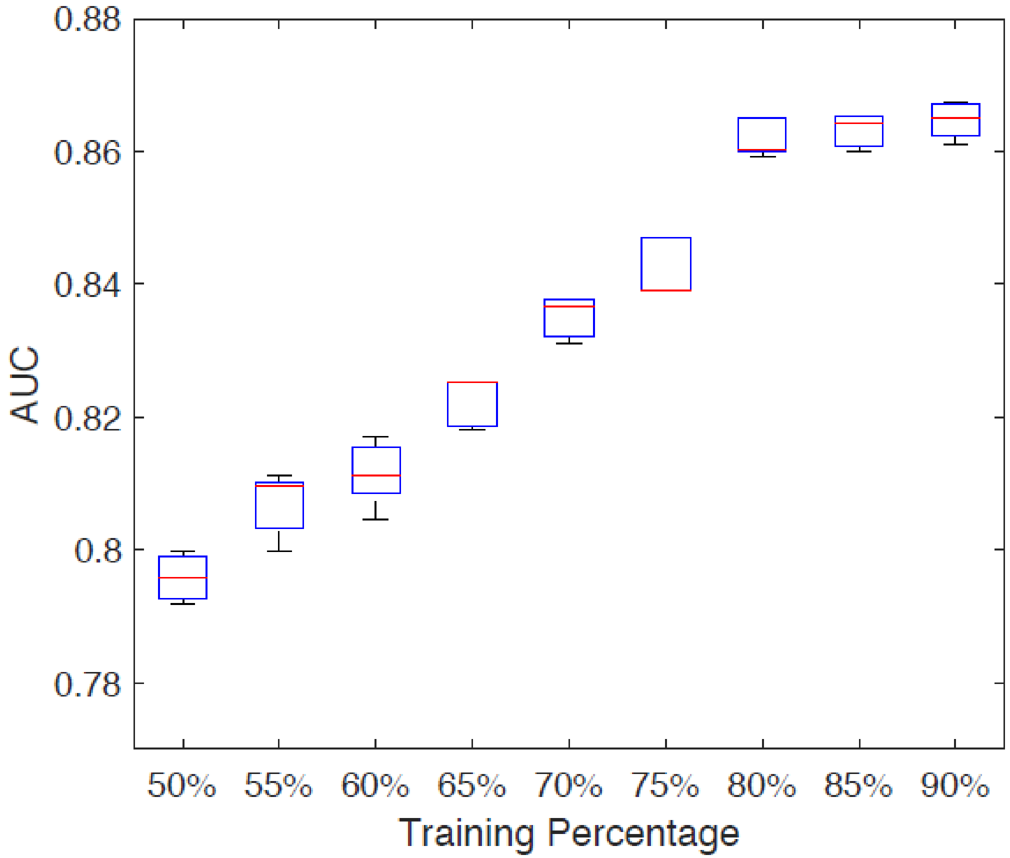
AUC results with different sizes of time Window on UCI Messages



	1%	5%	10%
$w = 3$	0.7565	0.634	0.7478
$w = 4$	0.7987	0.8048	0.7646
$w = 5$	0.8179	0.8252	0.7959
$w = 6$	0.8186	0.8218	0.7937
$w = 7$	0.8148	0.8197	0.7924
$w = 10$	0.8086	0.8136	0.7879

Overall performance haven’t been significantly affected.

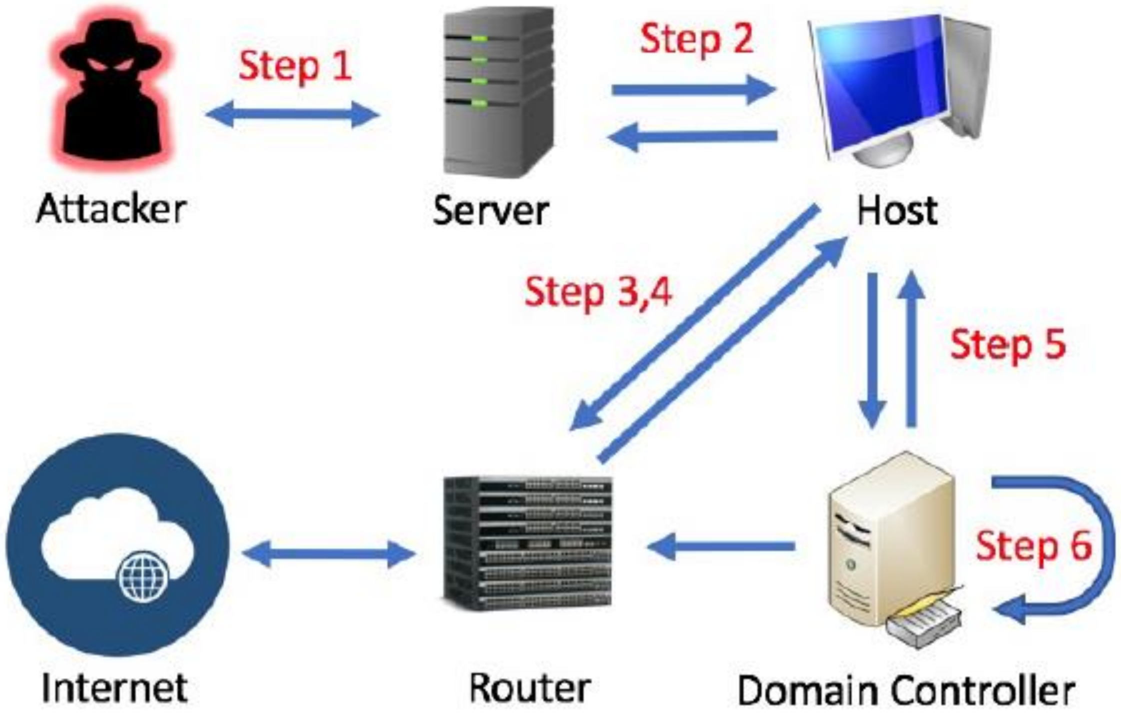
Stability Analysis



- Model is evaluated by different percentage of training samples.

- AUC increases with the percentage of training data ranging from 50% to 75%, and then the performance stays relatively stable.

Intrusion Detection Application



Attack Testbed Example Related to
The Diversifying Attack Vectors Attack

Attack Types:

- Diversifying Attack Vectors
- Emulating Enterprise Environment
- Domain Controller Penetration
- MLS Attack
- Snowden Attack
- Botnet Attack

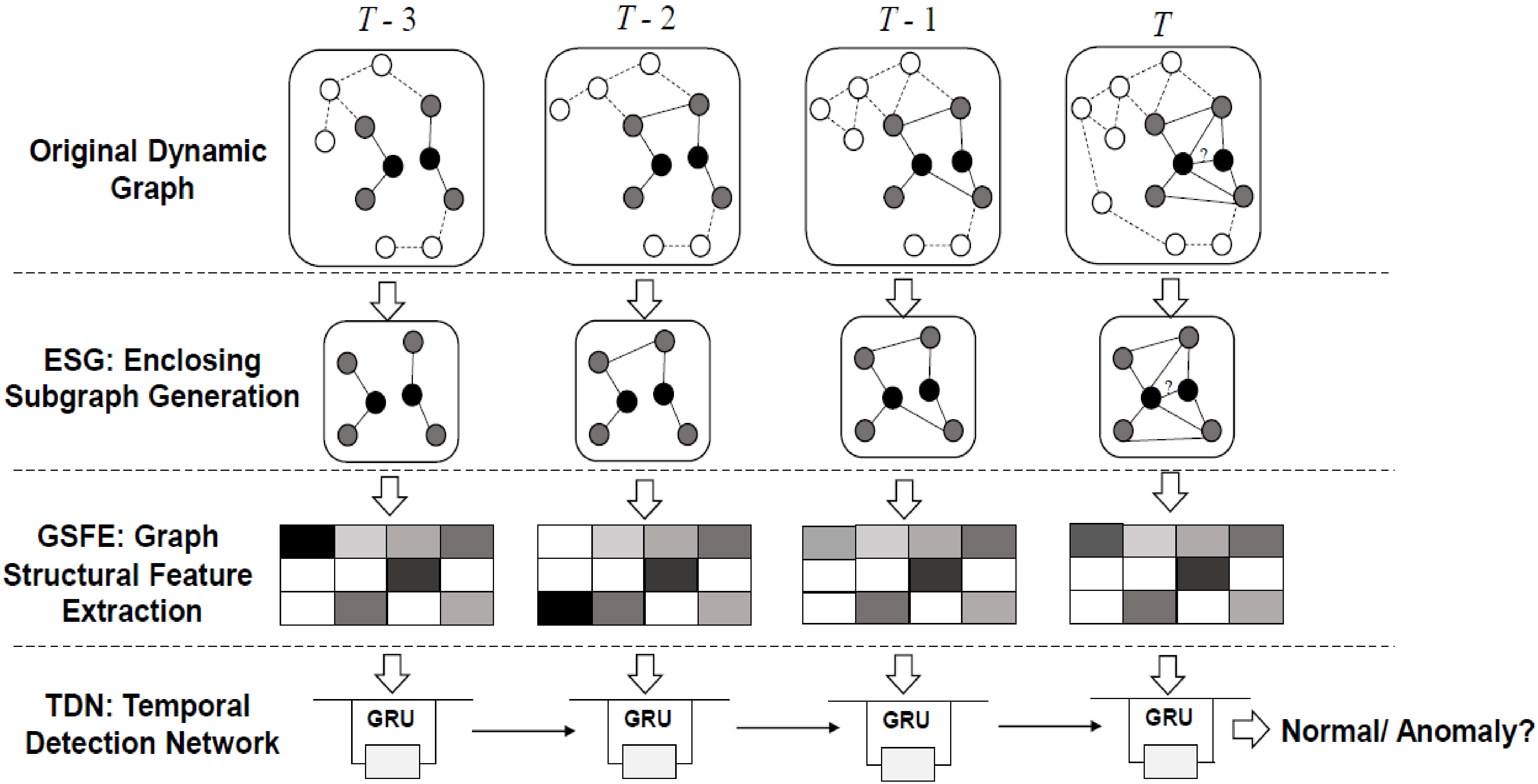
Method	AUC
SedanSpot	0.76
CM-Sketch	0.68
Node2Vec	0.71
Spectral Clustering	0.65
DeepWalk	0.76
Netwalk	0.90
StrGNN	**0.99**

Summary

- We proposed StrGNN, a structural temporal Graph Neural Network to detect anomalous edges by mining the **unusual temporal subgraph structure**s.
- StrGNN can be **trained end-to-end**, and it is **not sensitive to the percentage of anomalies**.
- We implemented and deployed our approach to a **real enterprise security system** and evaluated the proposed algorithm for **intrusion detection tasks**.
- We also evaluated the proposed framework using extensive experiments on **six benchmark datasets**. The experimental results demonstrated the effectiveness of our approach.

Thanks

- **QA**


T - 3
T - 2
T - 1
T
Original Dynamic Graph
ESG: Enclosing Subgraph Generation
GSFE: Graph Structural Feature Extraction
TDN: Temporal Detection Network
GRU
GRU
GRU
GRU
Normal/ Anomaly?