Recent advances in Graph Anomaly Detection

그래프 이상 탐지

Bonyou Koo(구본유)

Supervisor: Byungkook Oh

Graph & Language Intelligence Laboratory
Department of Computer Science and Engineering
Konkuk University

2025.07.31

CONTENTS

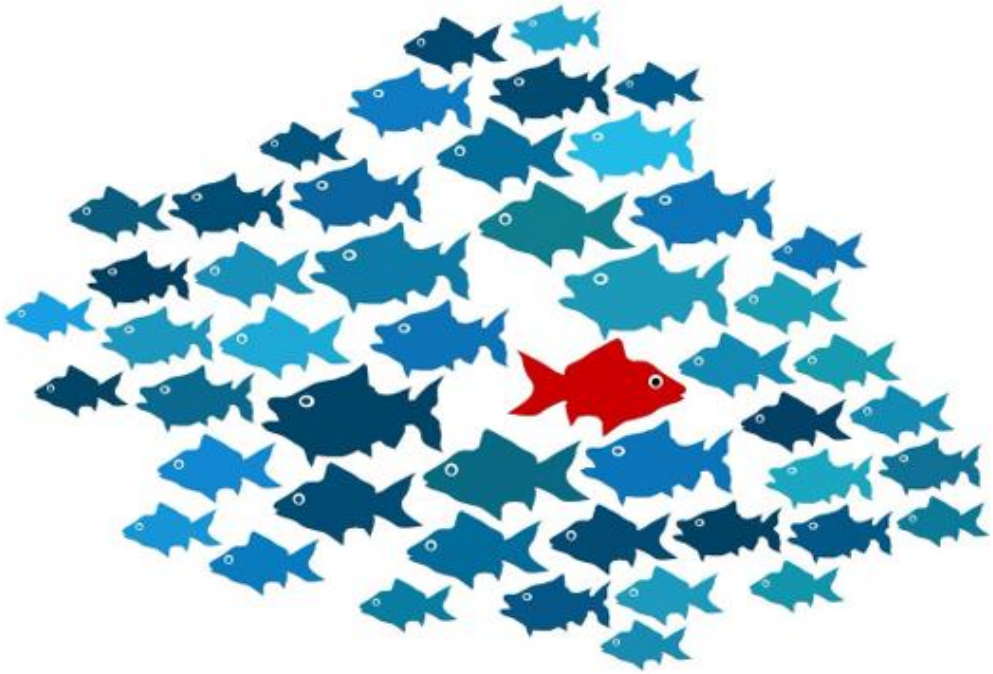
1. Introduction
   - Anomaly Detection
   - Graph anomaly detection
2. Related work
   - Generative based
     - Dominant (SDM, 2019)
     - MUSE(NIPS, 2024)
   - LLM
     - AnomalyLLM(IJCAI,2024)
   - LVLM(vision, language)
     - AnomalyGPT(AAAI, 2024)

CONTENTS

1. Introduction
   - Anomaly Detection
   - Graph anomaly detection
2. Related work
   - Generative based
     - Dominant (SDM, 2019)
     - MUSE(NIPS, 2024)
   - LLM
     - AnomalyLLM(IJCAI,2024)
   - LVLM(vision, language)
     - AnomalyGPT(AAAI, 2024)

Anomaly Detection

Intro

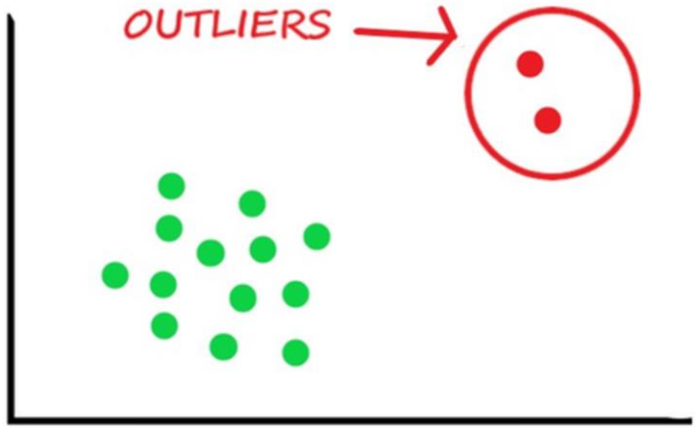


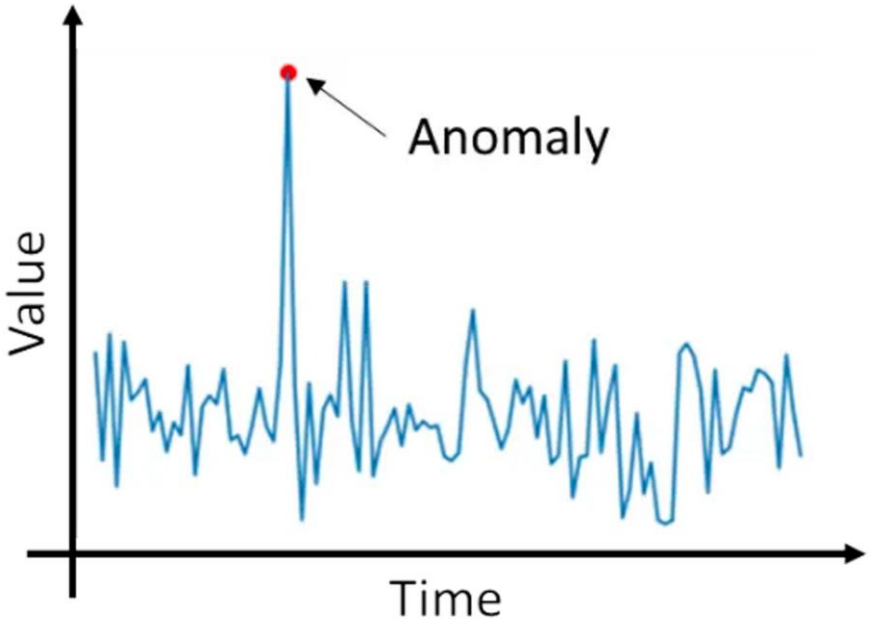
- Anomalies
  - ✓ Not conform to a notion of normal behavior.

- Anomaly detection
  - ✓ we try to distinguish anomalous samples
    - Deviate from normal samples.

Anomaly Detection

Intro





- Anomalies
  - ✓ Not conform to a notion of normal behavior.

- Anomaly detection
  - ✓ we try to distinguish anomalous samples
    - Deviate from normal samples.

Anomaly Detection

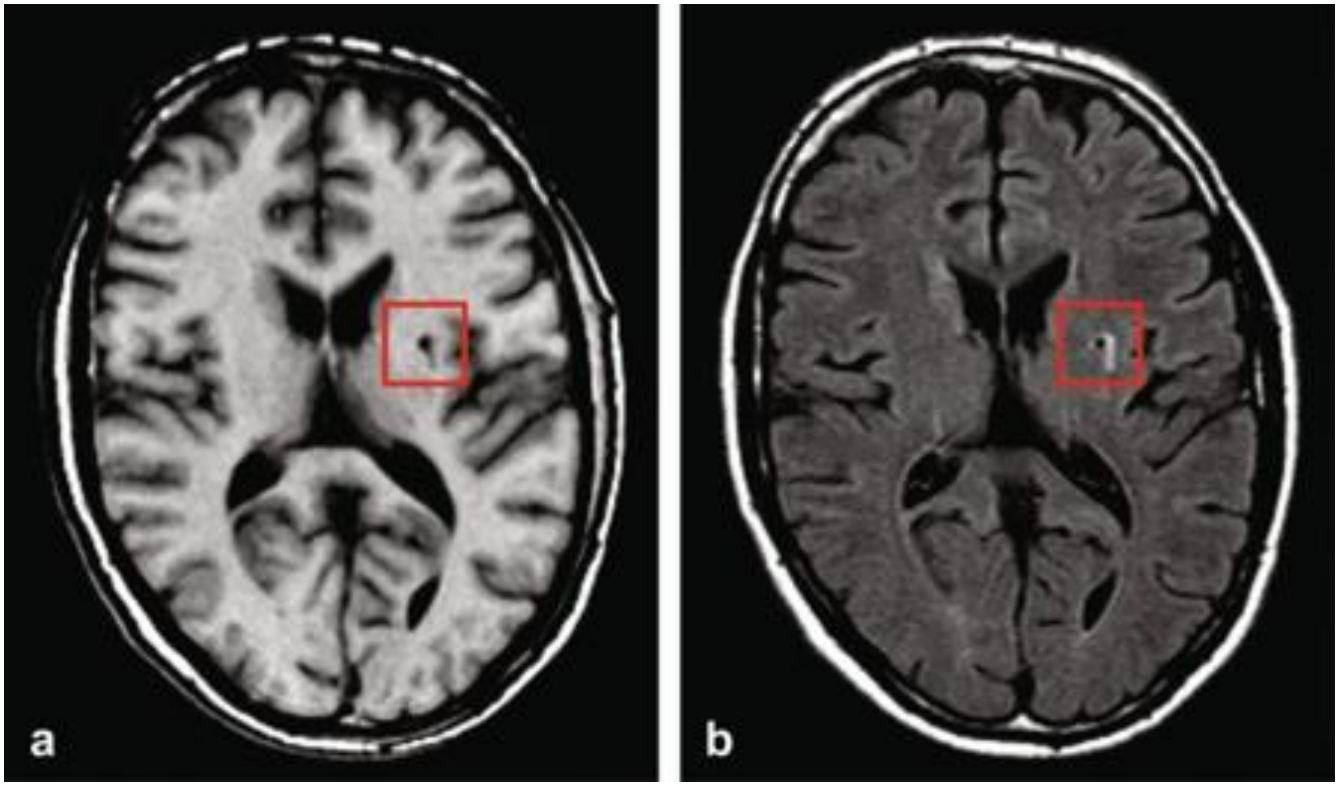
Application of Anomaly Detection

- Security and Surveillance

Anomaly Detection

Application of Anomaly Detection

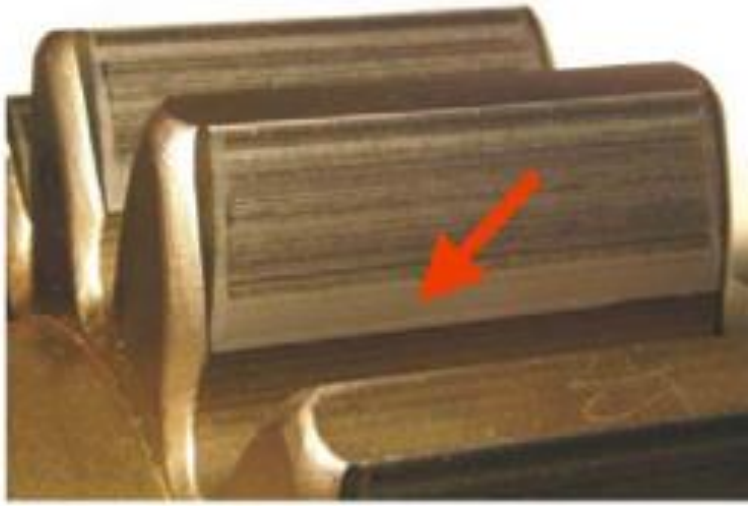
- Biomedical Application



Anomaly Detection

Application of Anomaly Detection

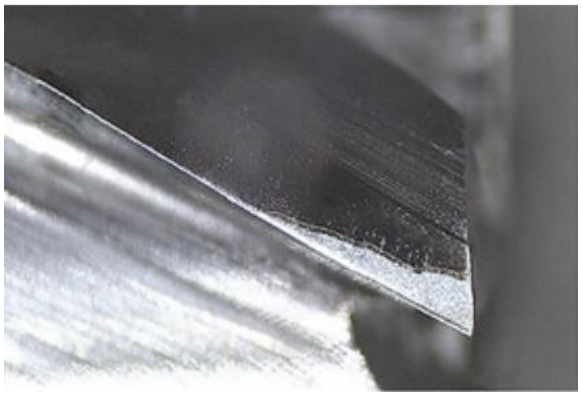
- Machinery Defects Diagnostics
  - Early alarm of malfunctioning



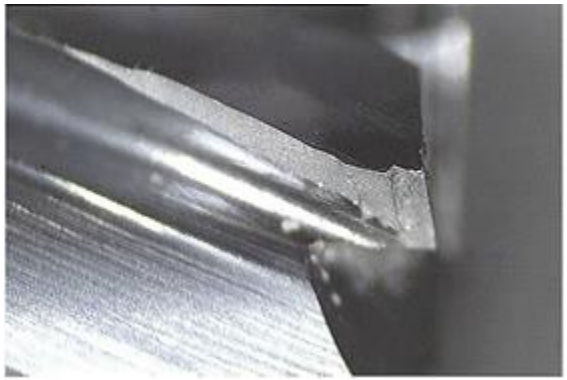
Anomaly Detection

Application of Anomaly Detection

- Industrial Damage Detection



y : No Wear



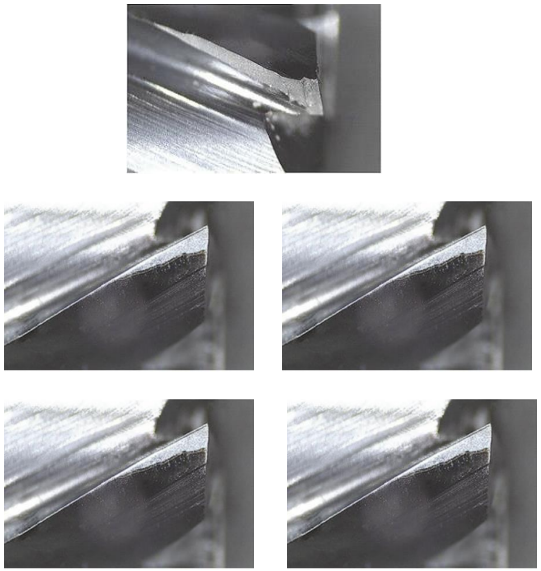
y : Wear

Anomaly Detection

Anomaly Detection y = f(x)

- Y = f(x), MLE

X = data



F

Y= label

1

0 0

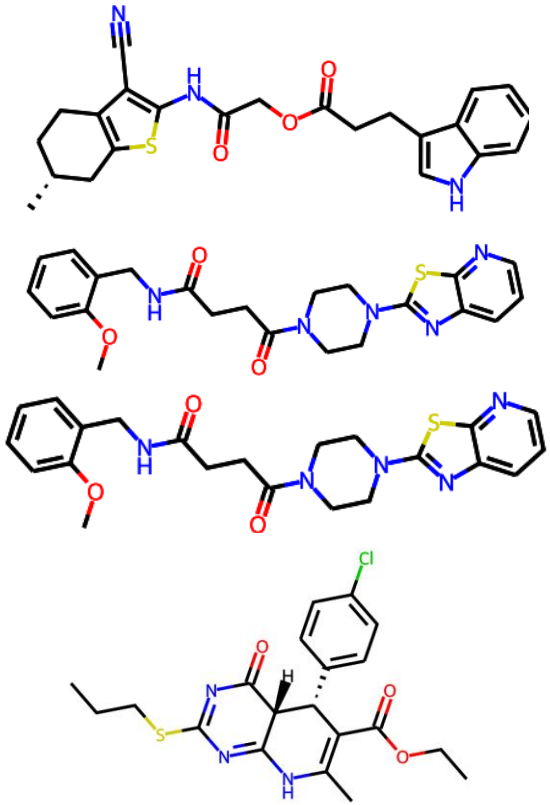
0 0

Graph Anomaly Detection

Graph Anomaly Detection y = f(x)

- y = f(x), MLE

X = graph data



F

Y= label

0

0

0

1

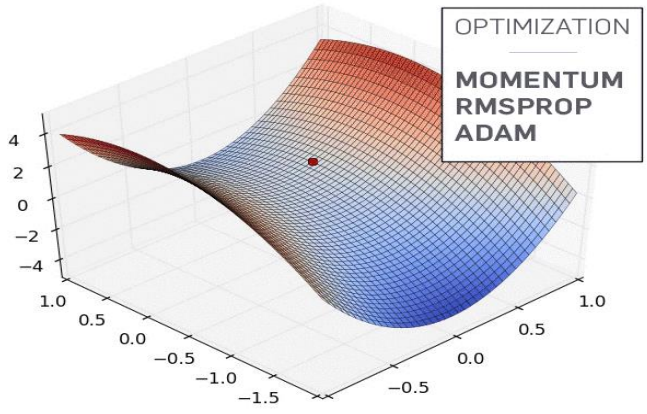
Himol (nature comm,2023)

Graph Anomaly Detection

Difficulties of anomalies

- Scarcity of anomalies
  - Not easy to get anomaly data
    - Because anomaly rarely happens

  - Diverse types of anomalies
    - So many cause of anomalies



digitalocean

Graph Anomaly Detection

Difficulties of anomalies

- Technique in anomaly detection

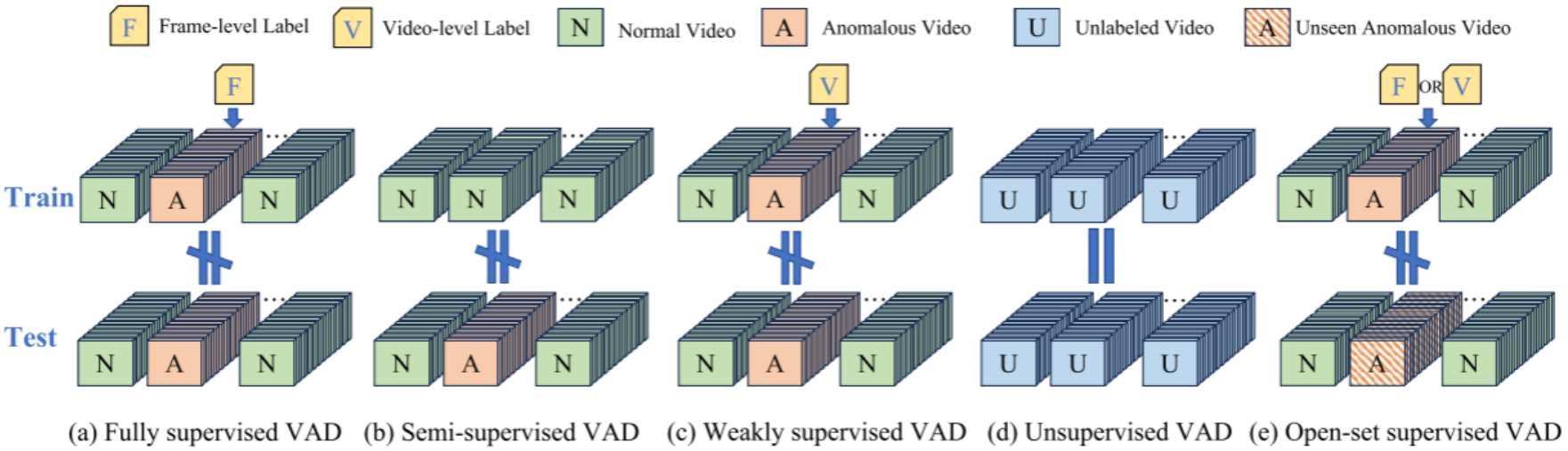
  ✓ Supervised
  - Utilize labeled data samples
    - Limits their scalability for training.

  ✓ Unsupervised
  - To capture deviations from normal pattern.
    - Hard to extend for anomaly types.

  ✓ Semi Supervised
  - Supervised + unsupervised
    - Use hundreds of labels
      - » Not for a few anomaly type

Graph Anomaly Detection

Difficulties of anomalies



https://arxiv.org/pdf/2409.05383

CONTENTS

1. Introduction
   - Anomaly Detection
   - Graph anomaly detection
2. Related work
   - Generative based
     - Dominant (SDM, 2019)
     - MUSE(NIPS, 2024)
   - LLM
     - AnomalyLLM(IJCAI,2024)
   - LVLM(vision, language)
     - AnomalyGPT(AAAI, 2024)

GAD Survey

Intro

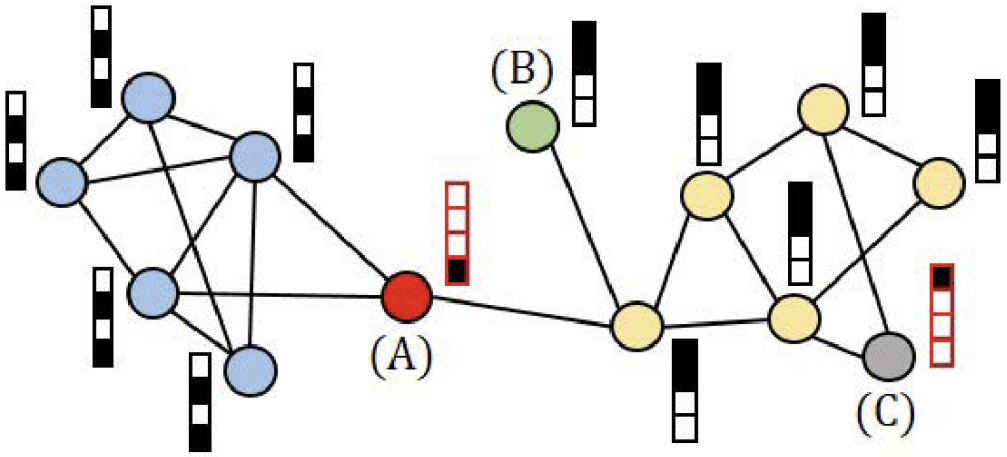


FIGURE 1. Example of graph anomaly detection. Nodes A and C are detected anomalous attribute-wise. Nodes A and B are detected anomalous structure-wise (as they do not belong to any community). Using GNN detects node A to be anomalous both attribute-wise and structure-wise.

Graph Anomaly Detection with Graph Neural Networks:
Current Status and Challenges
H. Kim et al.
IEEE Access
2022

GAD with GNNs, IEEEAccess, 22

Intro

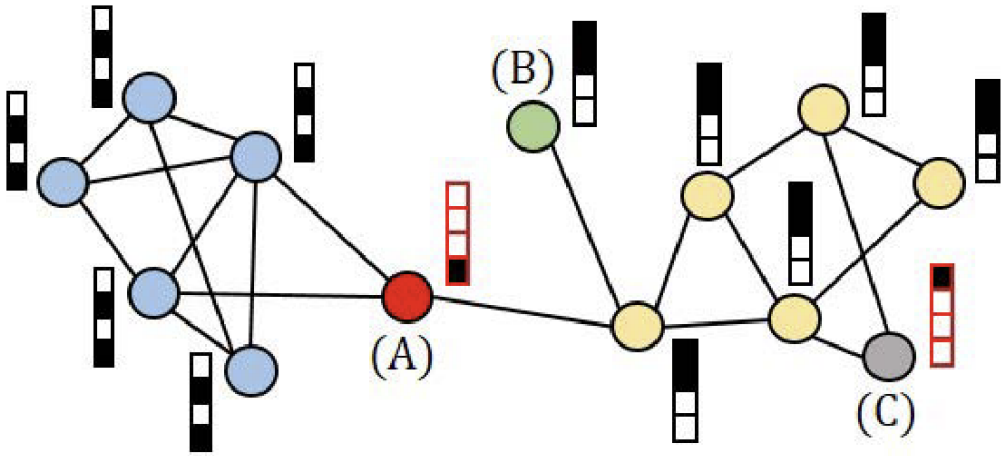


FIGURE 1. Example of graph anomaly detection. Nodes A and C are detected anomalous attribute-wise. Nodes A and B are detected anomalous structure-wise (as they do not belong to any community). Using GNN detects node A to be anomalous both attribute-wise and structure-wise.

On the basis of the structural information

- ✓ Global
  - are deviated node attributes in the graph.
- ✓ Structural
  - Are deviated structural information in the graph.
- ✓ Community
  - Deviated from both structural and node information in the community.

GAD with GNNs, IEEEAccess, 22

Intro

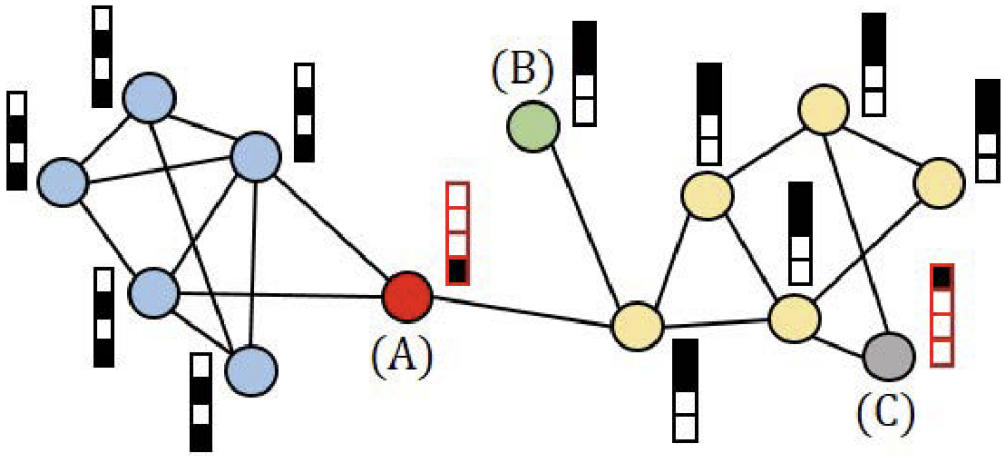
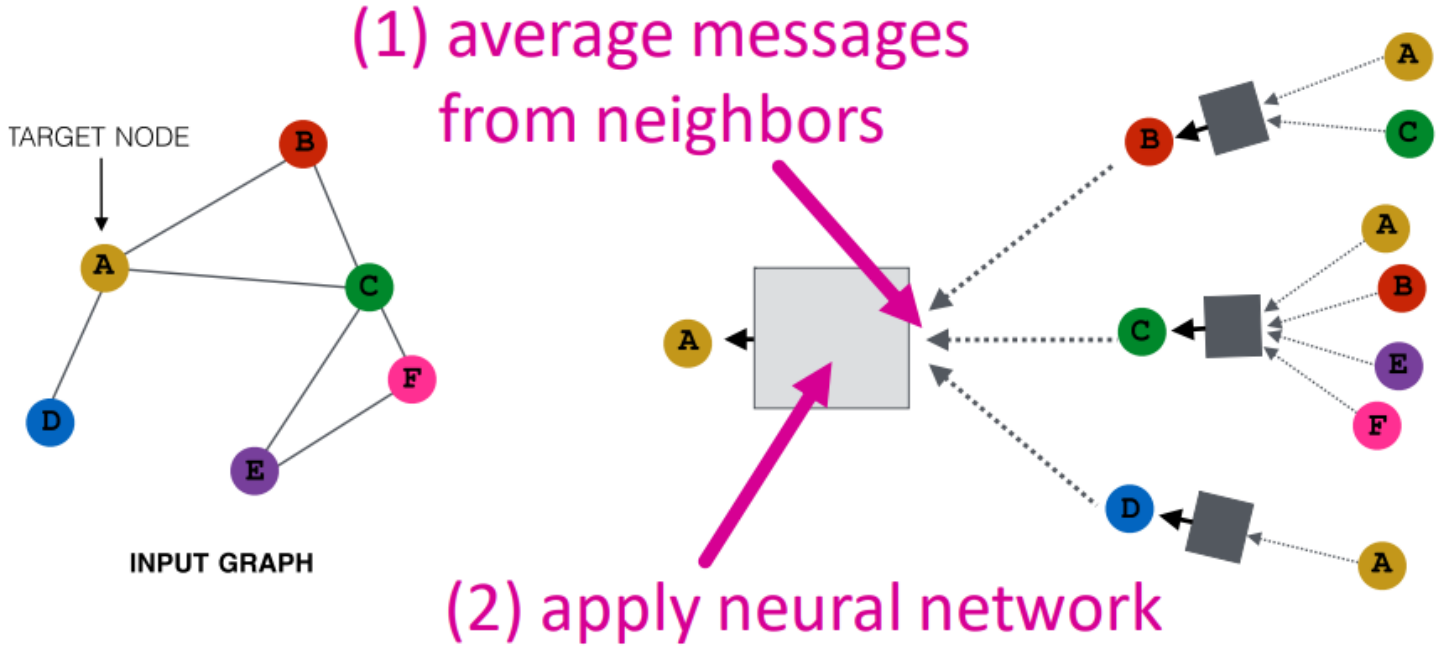


FIGURE 1. Example of graph anomaly detection. Nodes A and C are detected anomalous attribute-wise. Nodes A and B are detected anomalous structure-wise (as they do not belong to any community). Using GNN detects node A to be anomalous both attribute-wise and structure-wise.

- Early work on GAD dependent on domain knowledge and statical methods.
  - ✓ Handcrafted
    - Time-consuming, labor-intensive.
    - Real-world graphs contain a large #nodes and edges
      - Large scale and high-dimensional

GAD with GNNs, IEEEAccess, 22

- **Basic approach:** Average information from neighbors and apply a neural network



Cs224w, Stanford

GAD with GNNs, IEEEAccess, 22

Intro

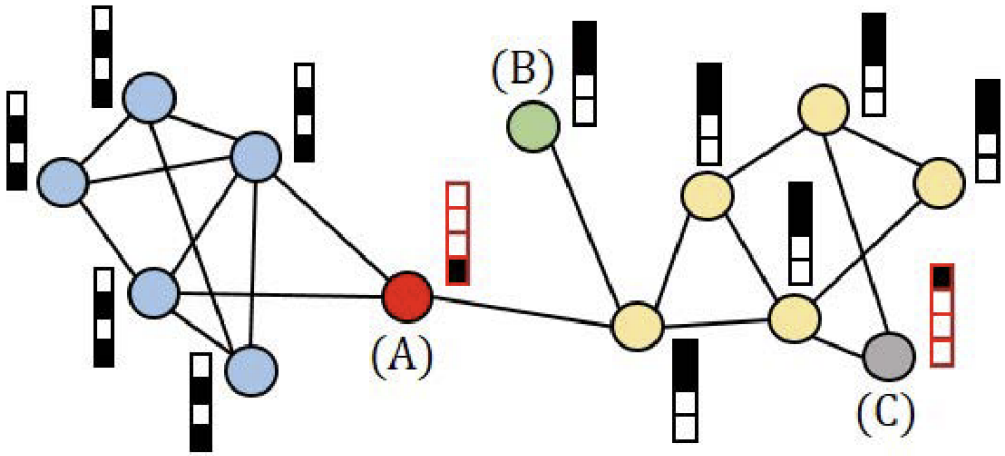


FIGURE 1. Example of graph anomaly detection. Nodes A and C are detected anomalous attribute-wise. Nodes A and B are detected anomalous structure-wise (as they do not belong to any community). Using GNN detects node A to be anomalous both attribute-wise and structure-wise.

- Nodes (A) and (C)
  - ✓ Detected anomalous of the node attributes.
- Nodes (A) and (B)
  - ✓ Detected in terms of graph topology
- Nodes (A)
  - ✓ Both node and topology.

GAD with GNNs, IEEEAccess, 22

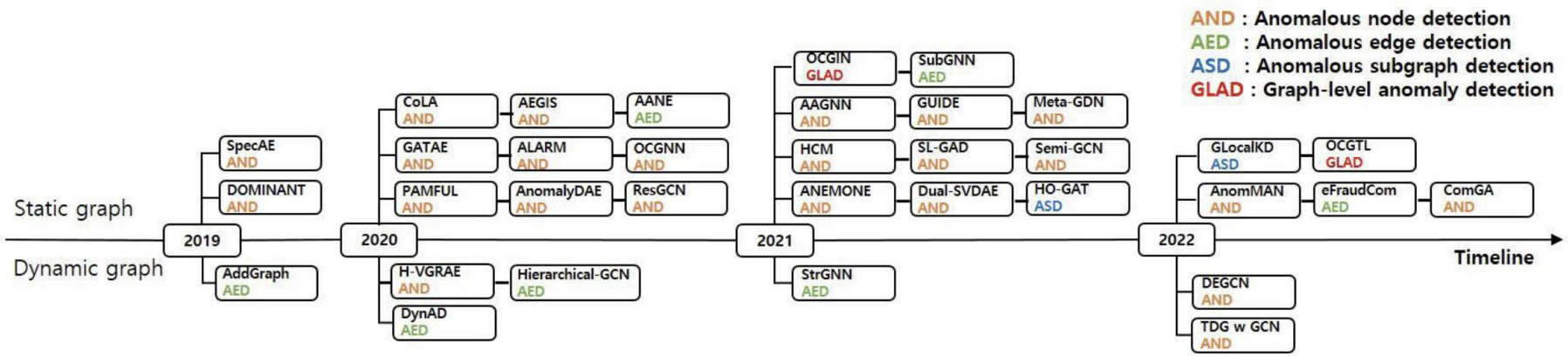


FIGURE 2. **Timeline of graph anomaly detection methods using GNN models. The methods above the timeline are for static graphs, and those below are for dynamic graphs.**

- Most research efforts on
  - ✓ static graph anomaly detection
  - ✓ and using gnn models

- Few addressed edge or subgraph anomalies.

CONTENTS

1. Introduction
   - Anomaly Detection
   - Graph anomaly detection
2. Related work
   - Generative based
     - Dominant (SDM, 2019)
     - MUSE(NIPS, 2024)
   - LLM
     - AnomalyLLM(IJCAI,2024)
   - LVLM(vision, language)
     - AnomalyGPT(AAAI, 2024)

CONTENTS

1. Introduction
   - Anomaly Detection
   - Graph anomaly detection
2. Related work
   - Generative based
     - Dominant (SDM, 2019)
     - MUSE(NIPS, 2024)
   - LLM
     - AnomlayLLM(IJCAI,2024)
   - LVLM (text, image)
     - AnomalyGPT(AAAI, 2024)

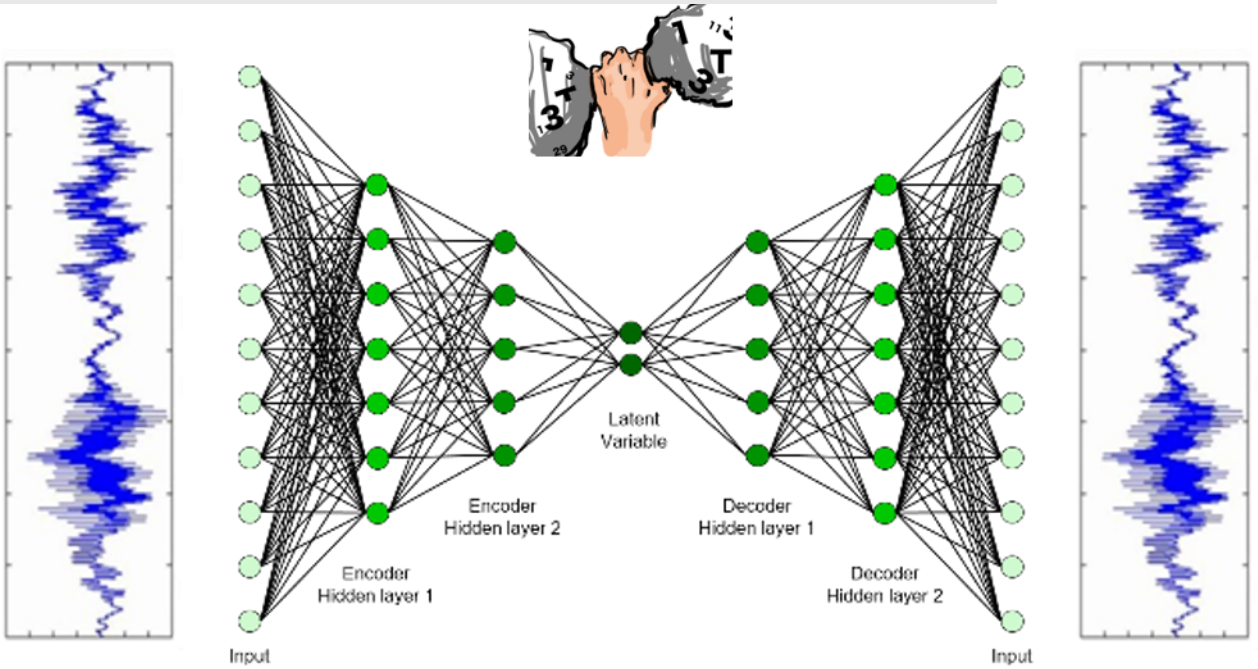
Related work – AE

Intro

- Autoencoders
  - ✓ Deep learning version of unsupervised learning
  - ✓ Network consists of two parts.
    - Encoder : z = f(x)
    - Decoder : x = g(z)
    - Learn to set g(f(x) = x
  - ✓ Network produce a reconstruction.

Related work – AE

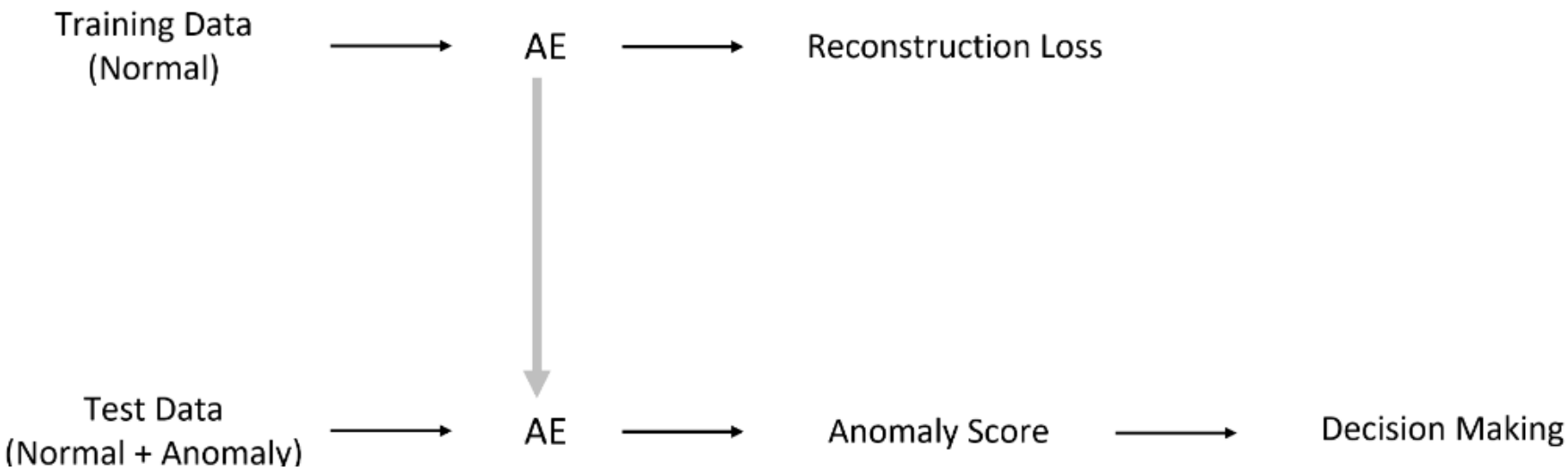
Intro



- Autoencoders
  - ✓ Deep learning version of unsupervised learning
  - ✓ Network consists of two parts.
    - Encoder : z = f(x)
    - Decoder : x = g(z)
    - Learn to set g(f(x) = x
  - ✓ Network produce a reconstruction.

Related work – AE

Intro

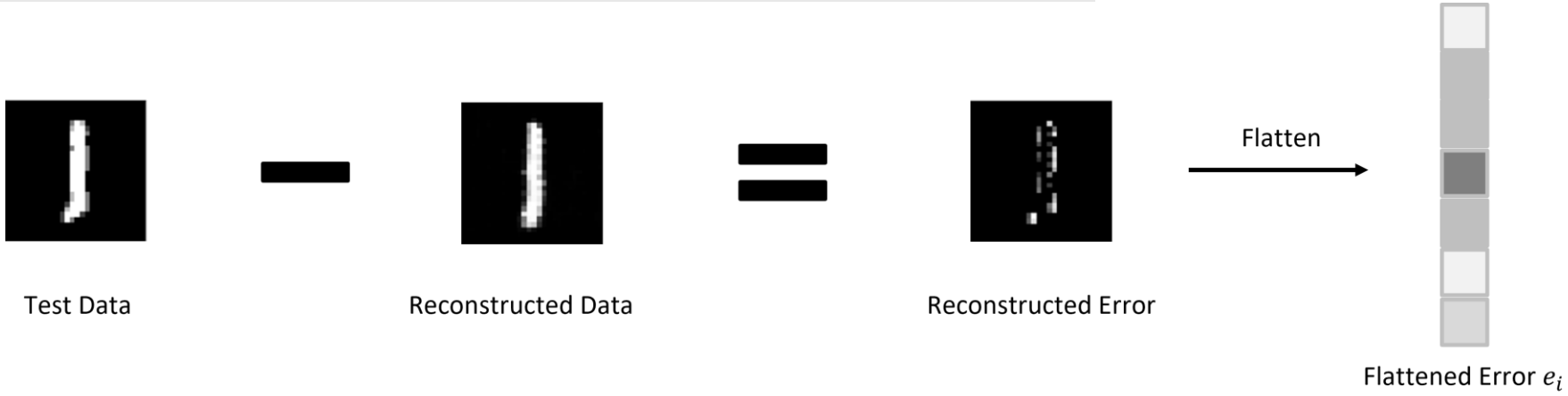


- Labeling anomalies
  - ✓ Labor-intensive
  - ✓ Costly

- Many efforts have focused on unsupervised learning.

Related work – AE

Intro



- Autoencoder learn latent representation
  - ✓ Aim to accurately reconstruct them.
  - ✓ Graph Auto Encoder expected to better reconstruct normal graph data.

Dominant, SDM, 19

Intro

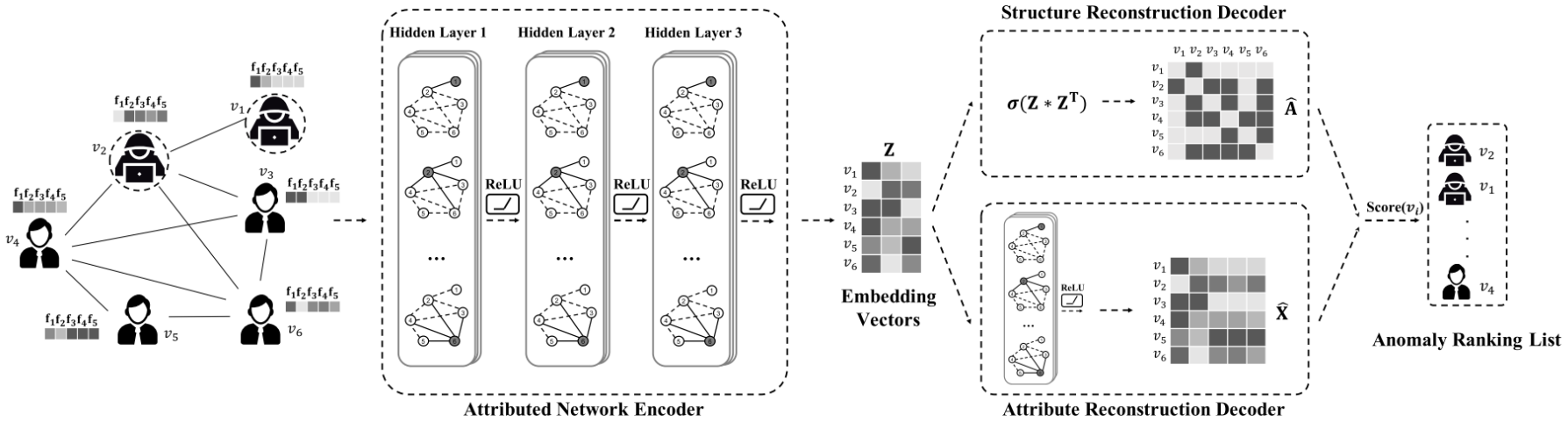


Figure 1: The overall framework of our proposed DOMINANT for deep anomaly detection on attributed networks.

- Shallow learning mechanisms
  - ✓ Capture node-to node interaction.
  - ✓ Often suffer from the network sparsity and data nonlinearity issues.
- Attributed graph
  - ✓ Node attributes complement the raw network structure.
  - ✓ With rich information.

Dominant, SDM, 19

Intro

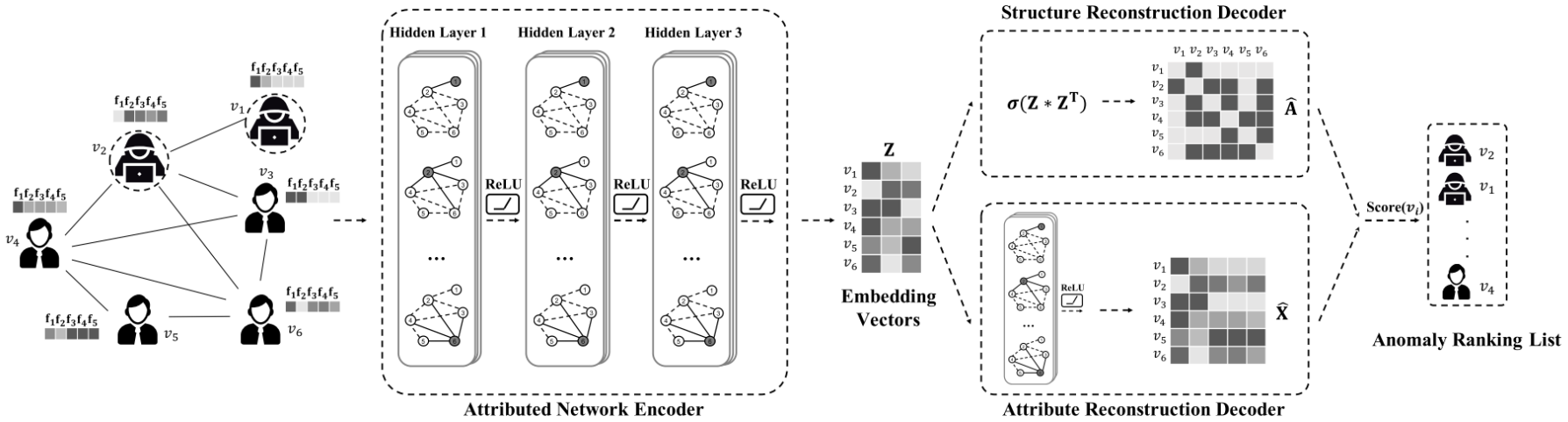


Figure 1: The overall framework of our proposed DOMINANT for deep anomaly detection on attributed networks.

- Using GCN
  - ✓ Models the topological structure and attributes.
- Autoencoder
  - ✓ Reconstruction nodes from both structure and attributes perspective.

Dominant, SDM, 19

Results

Precision@K												
	BlogCatalog				Flickr				ACM			
K	50	100	200	300	50	100	200	300	50	100	200	300
LOF	0.300	0.220	0.180	0.183	0.420	0.380	0.270	0.237	0.060	0.060	0.045	0.037
Radar	0.660	0.670	0.550	0.416	0.740	0.700	0.635	0.503	0.560	0.580	0.520	0.430
ANOMALOUS	0.640	0.650	0.515	0.417	**0.790**	0.710	0.650	0.510	0.600	0.570	0.510	0.410
DOMINANT	**0.760**	**0.710**	**0.590**	**0.470**	0.770	**0.730**	**0.685**	**0.593**	**0.620**	**0.590**	**0.540**	**0.497**
Recall@K												
	BlogCatalog				Flickr				ACM			
K	50	100	200	300	50	100	200	300	50	100	200	300
LOF	0.050	0.073	0.120	0.183	0.047	0.084	0.120	0.158	0.005	0.010	0.015	0.018
Radar	0.110	0.223	0.367	0.416	0.082	0.156	0.282	0.336	0.047	0.097	0.173	0.215
ANOMALOUS	0.107	0.217	0.343	0.417	**0.087**	0.158	0.289	0.340	0.050	0.095	0.170	0.205
DOMINANT	**0.127**	**0.237**	**0.393**	**0.470**	0.084	**0.162**	**0.304**	**0.396**	**0.052**	**0.098**	**0.180**	**0.248**

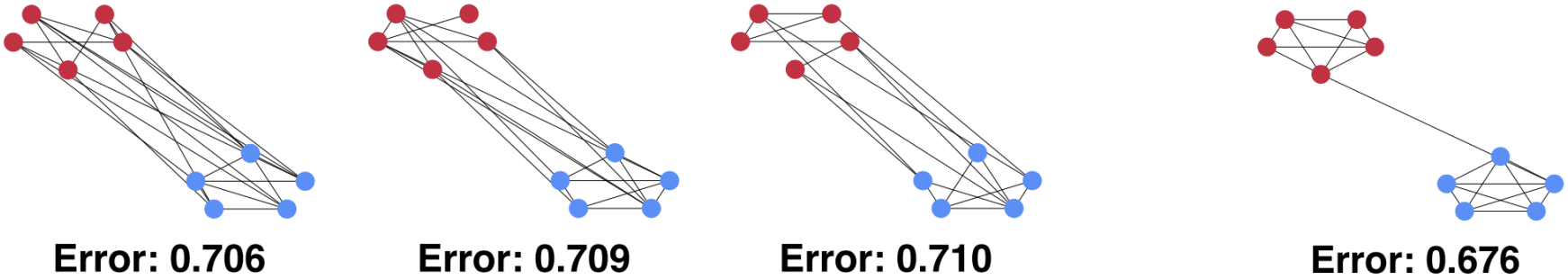
Table 2: Performance of different anomaly detection methods w.r.t. precision@K and recall@K.

CONTENTS

1. Introduction
   - Anomaly Detection
   - Graph anomaly detection
2. Related work
   - Generative based
     - Dominant (SDM, 2019)
     - **MUSE(NIPS, 2024)**
   - LLM
     - AnomalyLLM(IJCAI,2024)
   - LVLM(vision, language)
     - AnomalyGPT(AAAI, 2024)

MUSE, NIPS, 2024

Intro

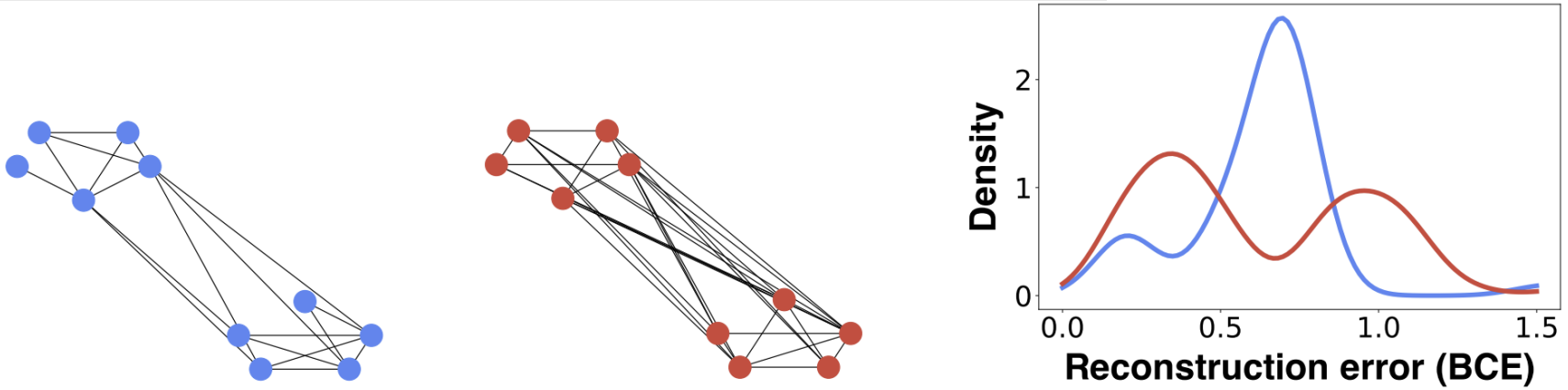


(a) Training graphs (b) Unseen graph

Figure 1: The training graphs in (a) and the unseen graph in (b) exhibit different structural characteristics, but a Graph-AE model reconstructs the graph in (b) more accurately than those in (a).

- Test reconstruction error is lower than training graphs
  - ✓ Which refer to performance flip
    - Unseen graphs share a primary structural pattern.

Intro



(a) Graph $\mathcal{G}_1$ (b) Graph $\mathcal{G}_2$ (c) Gaussian KDE visualization of error distributions of $\mathcal{G}_1$ (blue) and $\mathcal{G}_2$ (red).

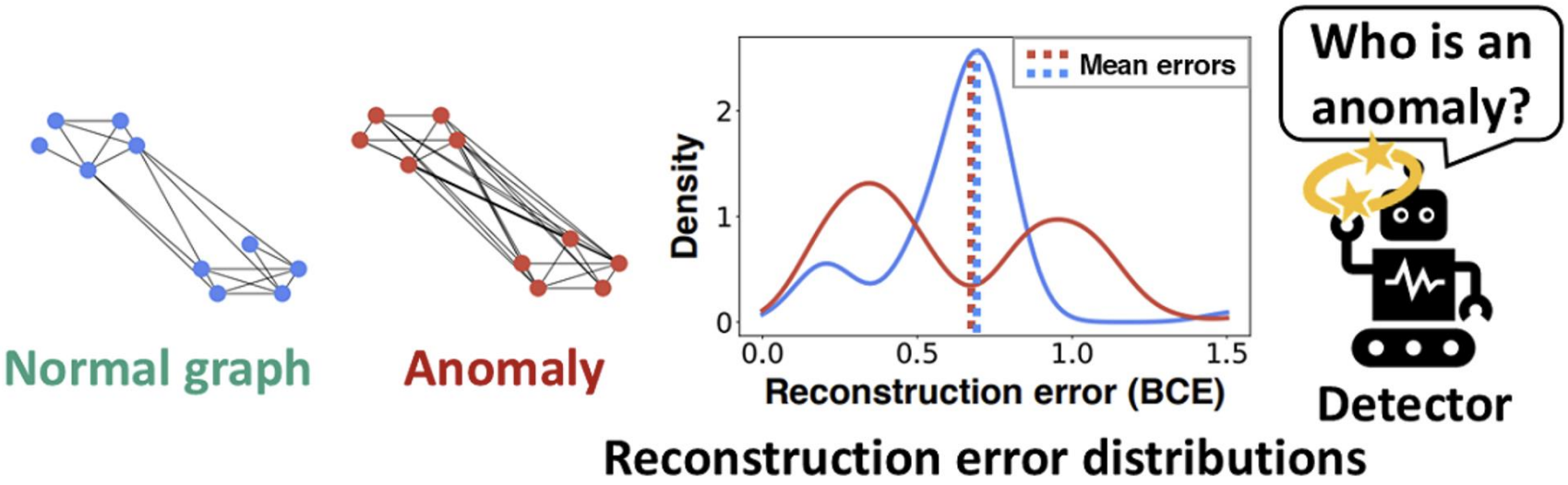
Figure 5: A case of Graph-AEs (specifically, GAE [22]) having similar mean reconstruction errors for two dissimilar graphs (specifically, $\mathcal{G}_1$ **has 0.6622** and $\mathcal{G}_2$ **has 0.6627**), while their error distributions differ significantly.

- Mean is not all you need.
  - ✓ Different #edges
    - their mean errors are similar.
      - Making difficult to distinguish.

MUSE, NIPS, 2024

Intro

- **[Detail]** Anomaly mean error ≈ Normal graph mean error



Reconstruction error distributions

The detector fails to distinguish which graph is an anomaly!

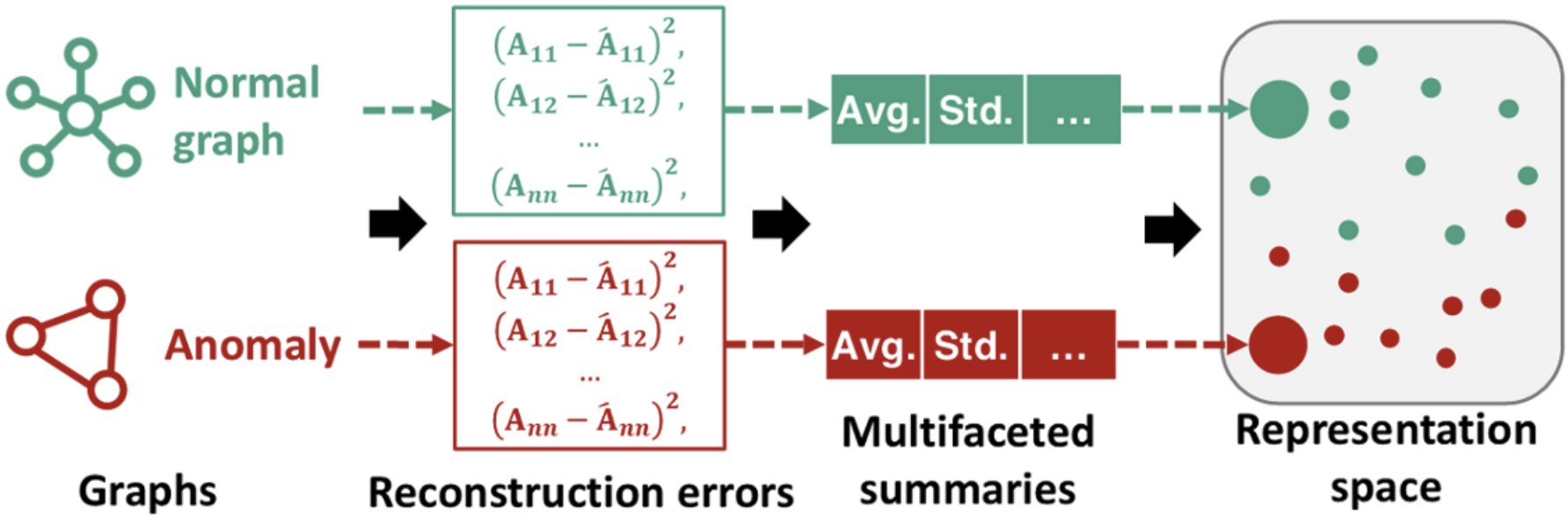
http://dmlab.kaist.ac.kr/~kijungs/museNeurIPS2024.pdf

- Mean is not all you need.
  - ✓ Different #edges
    - their mean errors are similar.
      - Making difficult to distinguish.

MUSE, NIPS, 2024

Method

- **[MUSE Overview]**

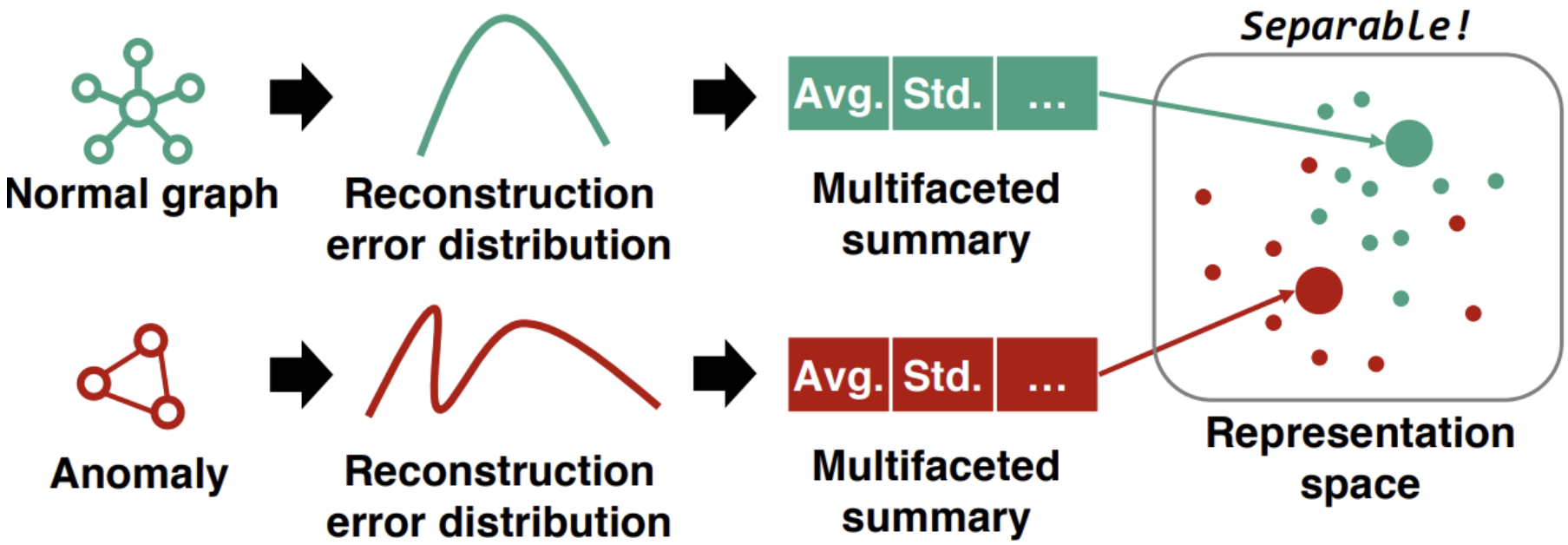


http://dmlab.kaist.ac.kr/~kijungs/museNeurIPS2024.pdf

- Using multifaceted summaries of errors.
  - ✓ To represent the error distribution.

MUSE, NIPS, 2024

Method



http://dmlab.kaist.ac.kr/~kijungs/museNeurIPS2024.pdf

- Mean is not all you need.
  - ✓ Different #edges
    - their mean errors are similar.
      - Making difficult to distinguish.

MUSE, NIPS, 2024

Method

Table 1: **GLAD performance**: Mean and standard deviation of test AUROC values (×100) in the GLAD task are reported. The best and second-best performances are highlighted in green and yellow. A.R. denotes average ranking. MUSE obtains the best average ranking among 18 methods.

	Method	DD	Protein	NCI1	AIDS	Reddit	IMDB	MUTAG	DHFR	BZR	ER	AR
GLAD methods	DOMINANT-G [6]	64.3 (4.4)	55.9 (9.7)	65.5 (6.1)	80.6 (4.0)	58.6 (5.3)	60.8 (6.7)	65.0 (4.2)	56.6 (9.2)	76.2 (7.8)	58.7 (5.5)	10.7
	OCGTL [39]	74.5 (5.1)	71.0 (8.7)	61.2 (5.5)	95.3 (3.7)	69.0 (4.0)	65.8 (5.8)	64.9 (4.9)	66.5 (9.9)	71.3 (17.1)	63.0 (3.6)	6.9
	GLocalKD [34]	47.8 (8.5)	50.7 (8.5)	51.6 (5.6)	51.2 (1.2)	49.8 (4.2)	58.5 (6.7)	55.1 (4.4)	54.1 (8.1)	55.8 (16.7)	54.4 (4.4)	17.0
	GLADC [33]	52.1 (5.2)	50.7 (5.6)	51.4 (3.6)	51.4 (1.0)	52.2 (2.6)	57.7 (5.2)	53.3 (4.5)	55.8 (4.1)	59.0 (14.5)	52.8 (4.2)	16.8
	GLAM [57]	61.6 (5.2)	60.3 (5.6)	58.1 (1.9)	93.6 (2.6)	75.6 (4.0)	65.1 (3.5)	63.0 (2.0)	57.2 (2.7)	72.6 (8.9)	55.2 (2.9)	9.8
	HIMNET [38]	52.1 (3.7)	56.9 (5.8)	53.6 (4.6)	64.3 (3.2)	65.7 (2.4)	61.8 (4.3)	57.5 (2.9)	63.6 (6.7)	72.0 (9.9)	55.7 (2.8)	12.3
	SIGNET [32]	64.2 (9.3)	56.4 (6.4)	63.1 (4.0)	97.2 (1.6)	78.0 (4.4)	48.2 (4.8)	67.5 (1.6)	40.2 (5.8)	66.6 (9.5)	56.2 (4.3)	10.4
SSL- based	GraphCL-1 [53]	64.5 (3.9)	60.7 (4.2)	55.8 (3.1)	71.2 (6.6)	57.7 (5.5)	54.2 (6.2)	53.6 (2.3)	57.8 (6.7)	60.5 (9.3)	55.5 (4.1)	14.2
	GAE-1 [22]	64.7 (5.2)	61.3 (7.0)	62.5 (2.2)	86.2 (1.4)	74.8 (3.2)	63.8 (7.4)	63.2 (3.3)	56.5 (9.6)	68.5 (13.7)	60.0 (3.9)	10.3
	GraphMAE-1 [15]	56.7 (7.3)	60.5 (4.9)	53.4 (3.2)	91.8 (5.3)	72.7 (3.2)	67.0 (5.0)	62.3 (2.6)	62.2 (9.6)	70.1 (7.6)	52.2 (3.6)	10.6
	GraphCL-2 [53]	66.1 (3.0)	59.1 (5.2)	60.3 (4.4)	91.8 (3.5)	77.3 (4.1)	66.3 (5.6)	67.4 (3.3)	59.1 (4.6)	71.9 10.4	67.3 (3.4)	7.2
	GAE-2 [22]	67.2 (3.4)	62.3 (5.0)	62.4 (3.9)	85.8 (1.6)	75.3 (5.7)	66.6 (7.6)	67.3 (3.3)	60.8 (5.6)	72.0 (8.8)	65.7 (2.0)	7.0
	GraphMAE-2 [15]	68.0 (4.3)	61.2 (4.0)	68.3 (3.6)	90.8 (3.6)	75.8 (4.8)	66.7 (5.8)	68.1 (2.4)	61.4 (6.0)	72.8 (6.4)	66.2 (6.4)	5.1
Variants	MUSE w/o $\mathbb{L}_{\mathbf{X}}$	79.4 (3.7)	75.6 (3.7)	69.2 (3.7)	99.6 (0.5)	72.2 (4.0)	65.8 (5.7)	65.8 (3.1)	60.4 (6.6)	65.6 (19.4)	66.3 (3.6)	5.8
	MUSE w/o $\mathbb{L}_{\mathbf{A}}$	61.8 (7.6)	64.7 (7.1)	63.1 (3.3)	89.3 (2.8)	72.0 (4.8)	56.9 (7.1)	57.0 (3.5)	58.1 (3.1)	68.7 (14.2)	60.7 (4.0)	11.0
	MUSE w/o AVG	78.6 (4.0)	68.1 (5.5)	68.0 (2.0)	95.0 (2.6)	73.2 (6.6)	66.2 (6.5)	60.9 (3.9)	60.1 (2.4)	66.3 (13.0)	62.0 (3.5)	7.7
	MUSE w/o STD	74.3 (5.4)	74.4 (5.2)	65.2 (3.6)	98.7 (0.5)	70.5 (4.3)	70.7 (3.7)	62.0 (2.4)	62.9 (6.4)	71.3 (11.5)	66.7 (2.4)	5.6
	MUSE	80.5 (2.3)	78.4 (2.2)	71.1 (2.0)	99.7 (0.5)	78.4 (5.7)	69.2 (3.5)	67.5 (3.4)	63.8 (8.6)	69.5 (12.6)	67.9 (3.6)	2.2

- Traditional approach
- SSL

CONTENTS

1. Introduction
   - Anomaly Detection
   - Graph anomaly detection
2. Related work
   - Generative based
     - Dominant (SDM, 2019)
     - MUSE(NIPS, 2024)
   - **LLM**
     - **AnomalyLLM(IJCAI,2024)**
   - LVLM(vision, language)
     - AnomalyGPT(AAAI, 2024)

AnomalyLLM

Intro

- Few-shot Anomaly Edge Detection for Dynamic Graphs using Large Language Models
  - ✓ Few shot을 위해 LLM의 일반화 능력을 이용 => In-Context Learning, graph-text align
  - ✓ Dynamic graph을 특성을 활용 => dynamic aware learning

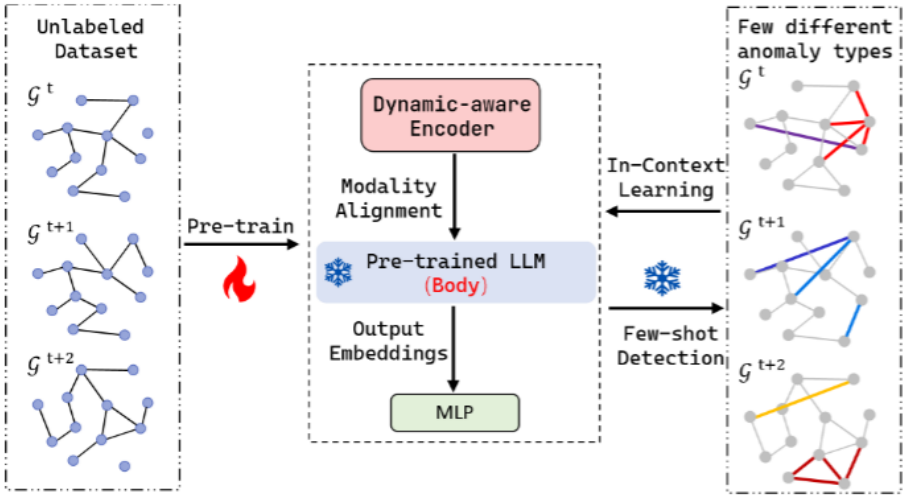


Figure 1: The motivation of AnomalyLLM. In the real world, edge anomaly types are diverse, evolving over time, and typically associated with limited labeled data.

AnomalyLLM

Intro

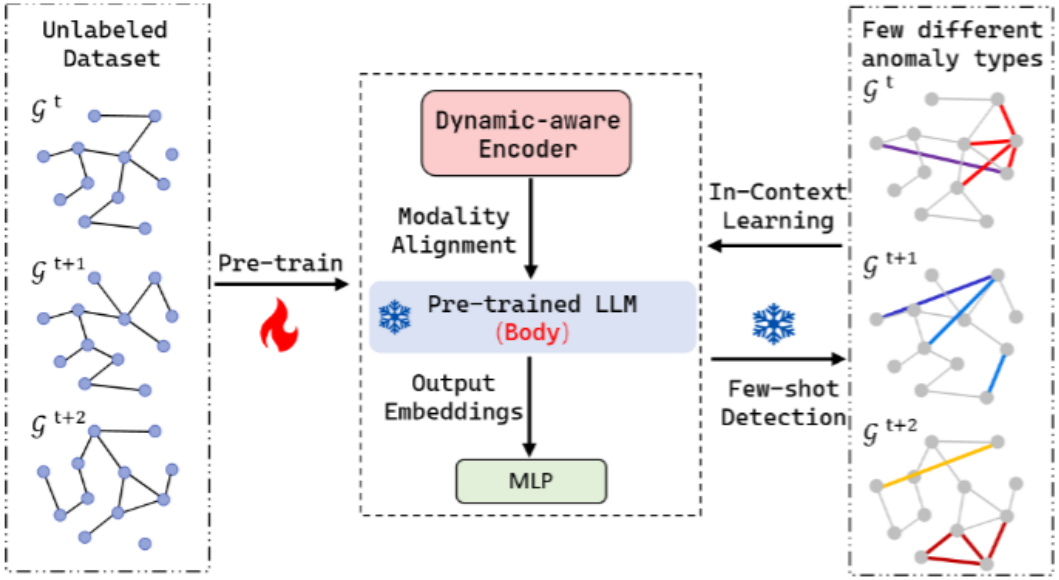


Figure 1: The motivation of AnomalyLLM. In the real world, edge anomaly types are diverse, evolving over time, and typically associated with limited labeled data.

- Dynamic graph
  - ✓ Evolving relation among the entities.
    - RECSYS, Social Networks, data center for DevOps

AnomalyLLM

Intro

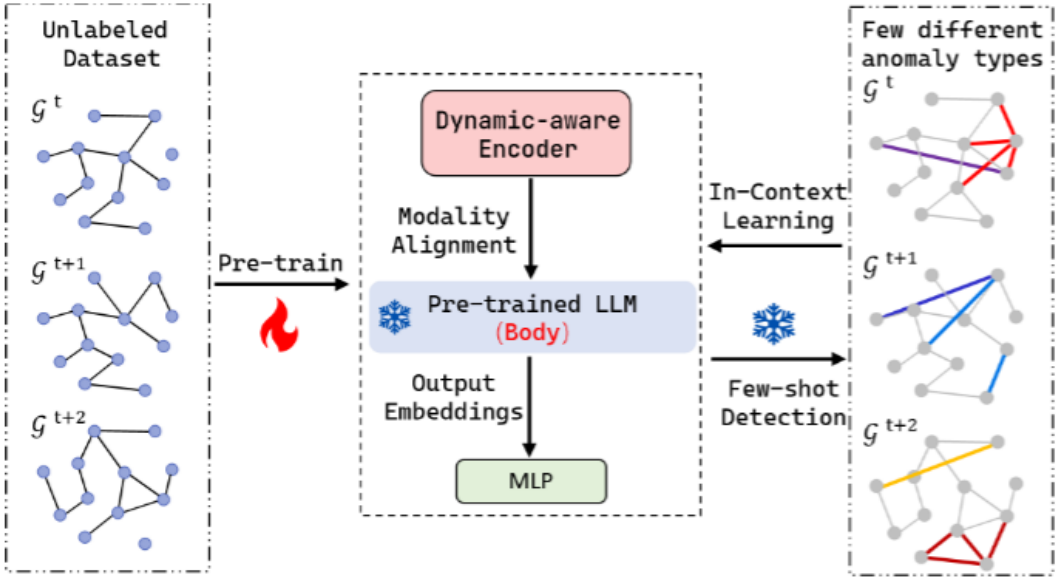


Figure 1: The motivation of AnomalyLLM. In the real world, edge anomaly types are diverse, evolving over time, and typically associated with limited labeled data.

- Anomaly Edge in dynamic graphs.
  - ✓ Abnormal interactions
    - Between fraudsters. Suspicious interactions.
  - ✓ Due to the temporary nature of dynamics -> label sparsity.

AnomalyLLM

Intro

- Technique in anomaly detection

  ✓ Supervised
  - Utilize labeled data samples
    - Limits their scalability for training.

  ✓ Unsupervised
  - To capture deviations from normal pattern.
    - Hard to extend for anomaly types.

  ✓ Semi Supervised
  - Supervised + unsupervised
    - Use hundreds of labels
      - » Not for a few anomaly type

AnomalyLLM

Intro

- LLM achieve few shot anomaly edge
  - ✓ Representation of dynamic graph
    - Chaining of the graph topology
      - Edge representation
        - » adjacent topology
        - » temporal dynamics
  - ✓ Alignment between graph and language.
  - ✓ Adaption few shot for anomaly task.

TEA-GLM, NeurIPS,24

Method

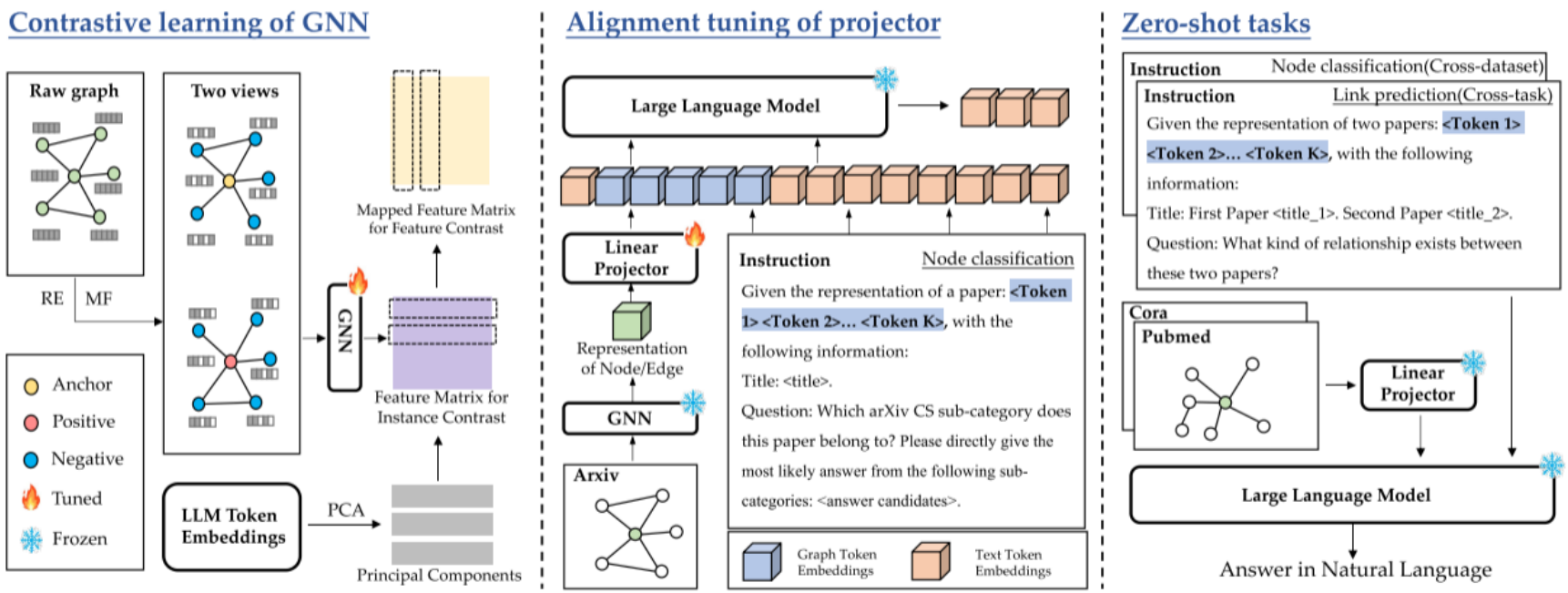


Figure 1: Framework of TEA-GLM

- Representation enhancer
- Graph text alignment
- Text to graph task

graphGPT, SIGIR, 24

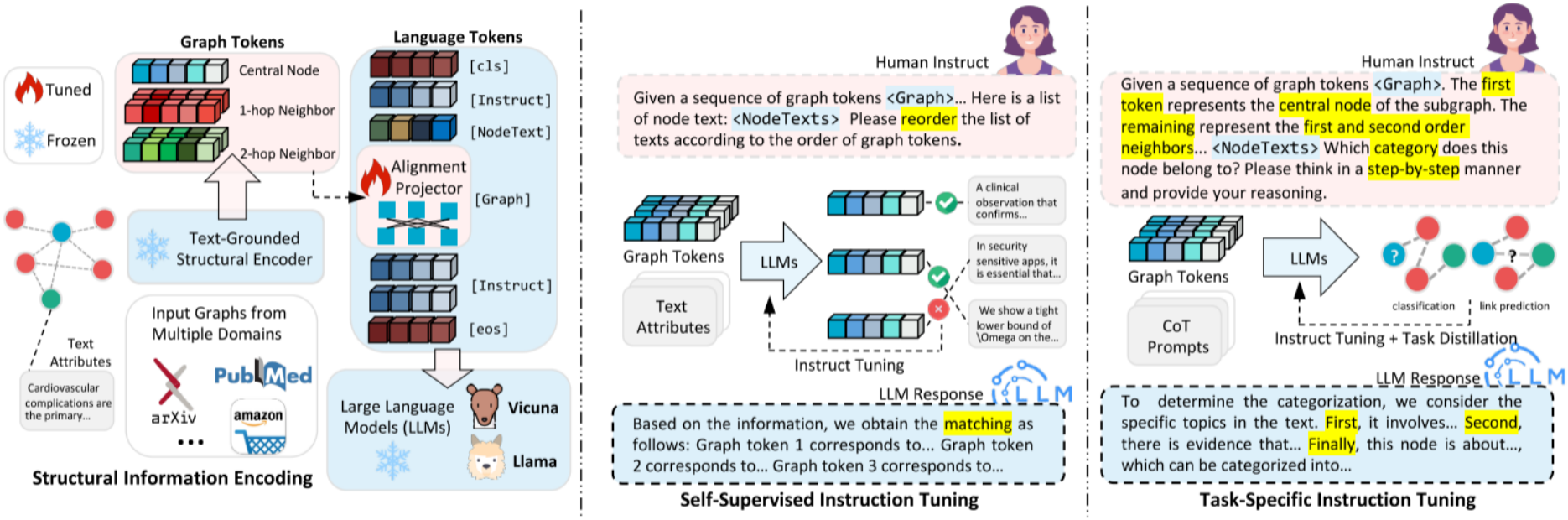


Figure 2: The overall architecture of our proposed GraphGPT with graph instruction tuning paradigm.

Alignment space between graph and text.

1. Enhancing graph representation : graph text alignment
   - Structural Information Encoding
2. For LLM input : graph to text
   - Self-sup Instruction Tuning
3. For output (task) : text to graph
   - Task-Specific Instruction Tuning

AnomalyLLM - Method

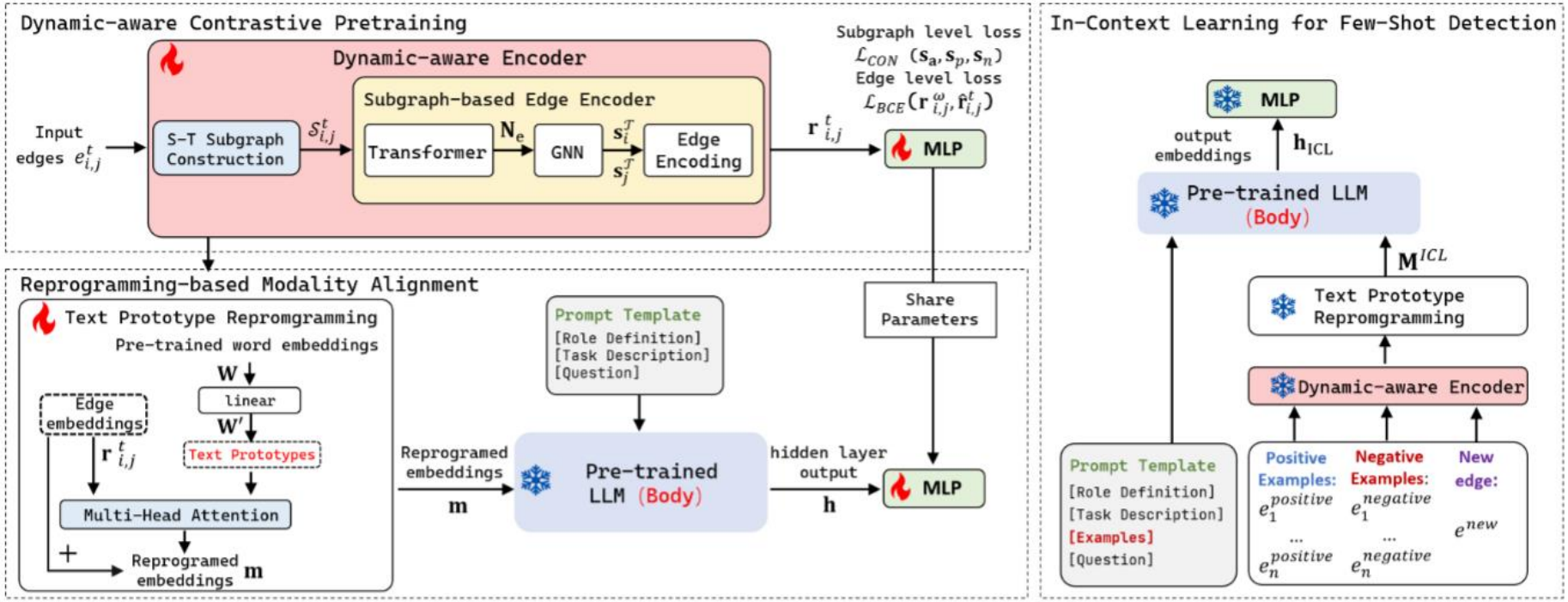
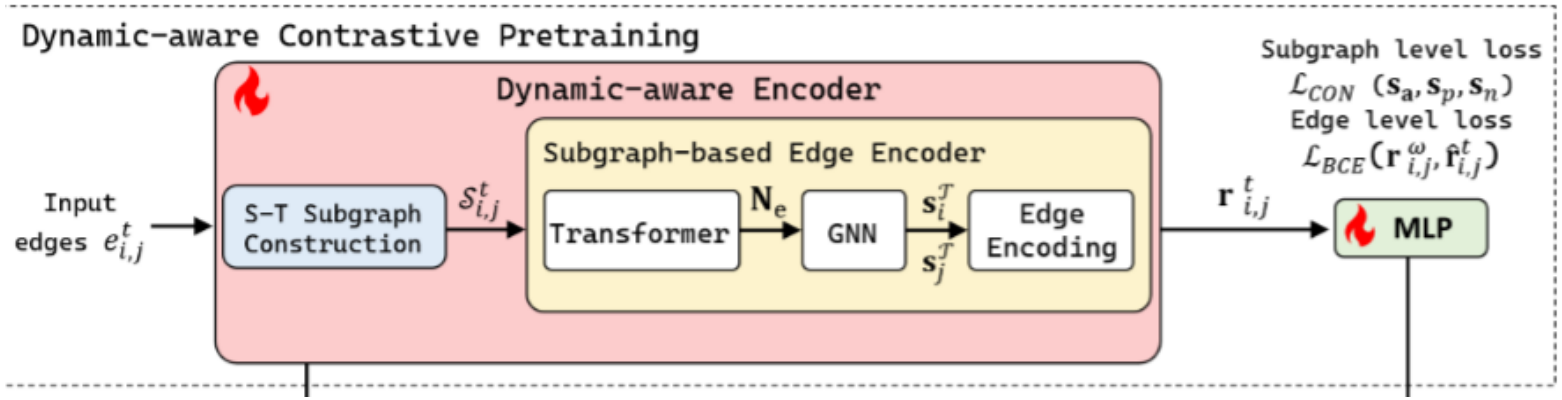


Figure 2: Overview of AnomalyLLM. AnomalyLLM comprises three modules: Dynamic-aware Contrastive Pretraining, Reprogramming-based Modality Alignment, and In-Context Learning for Few-Shot Detection.

- Dynamic Contrastivce Pretraining enhancer
- Reprograming-based Modality Alignment
- In-Context Learning for few-shot Detection

AnomalyLLM - Method



- S-T subgraph construction : $e_{i,j}^t$ to $S_{i,j}^t$
  - ✓ Select top K adjacent node from diffusion matrix.
  - ✓ Sliding window size t.

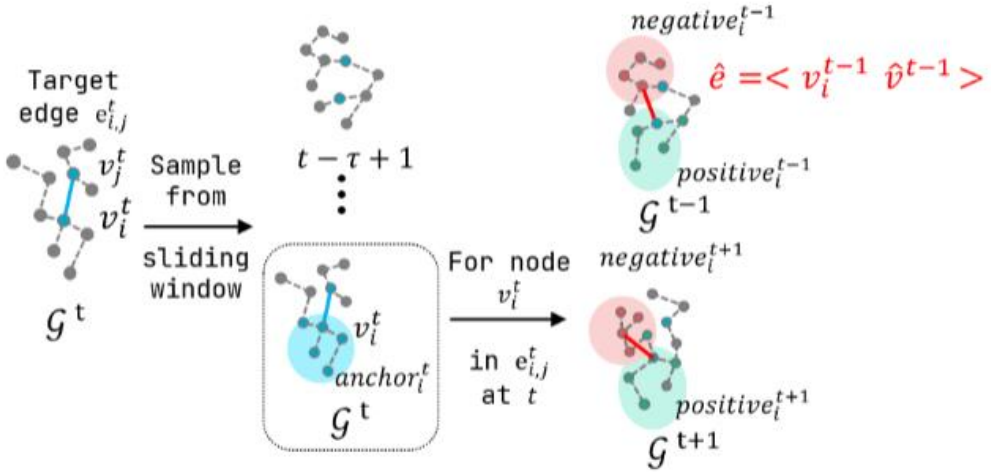
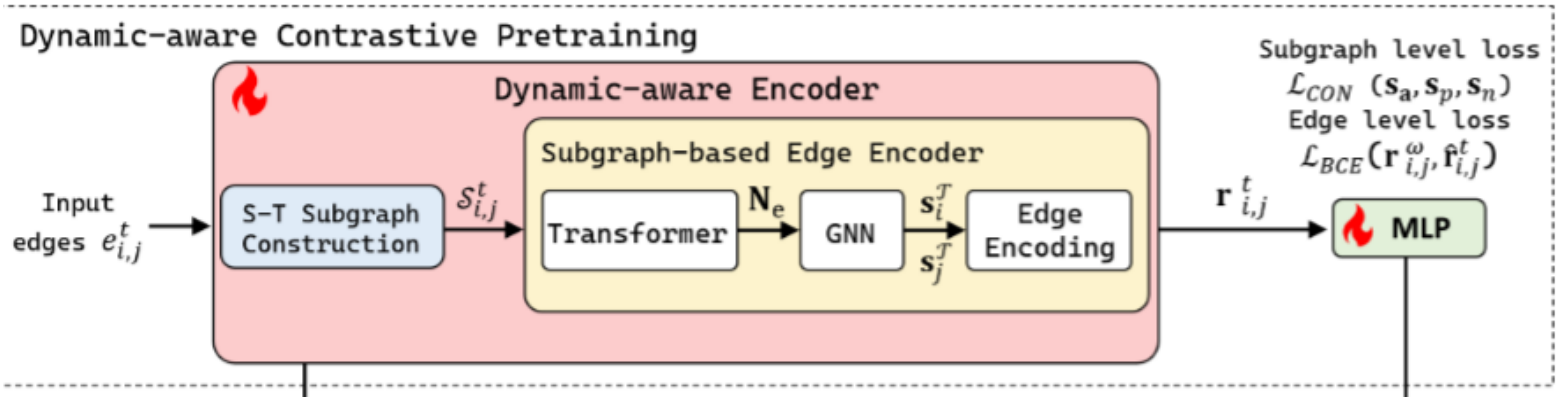


Figure 3: Sample process of contrastive training triplet

AnomalyLLM - Method



- S-T subgraph construction : $e_{i,j}^t$ to $\mathcal{S}_{i,j}^t$
  - ✓ Select top K adjacent node from diffusion matrix.
  - ✓ Sliding window size t.

- Subgraph-based Edge Encoder
  - ✓ Transformer
    - Temporal encoding sequence of node
  - ✓ GNN
    - Encoding within the subgraph
  - ✓ Edge Encoding
    - Average pooling of node within the subgraph
    - Concat the subgraph representation and pass the FC

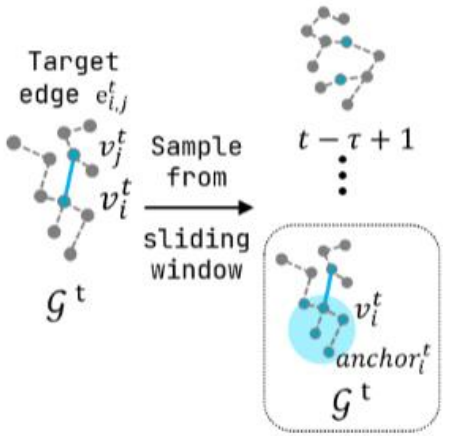
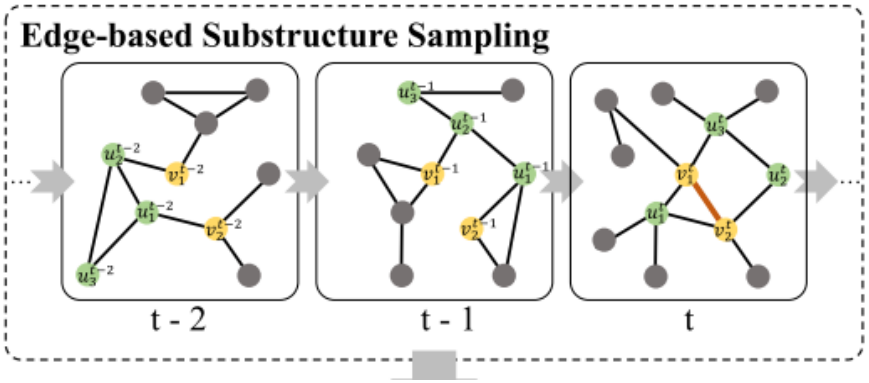


Figure 3: Sample process

TADDY (Transformer-based AD for DYnamic graph)

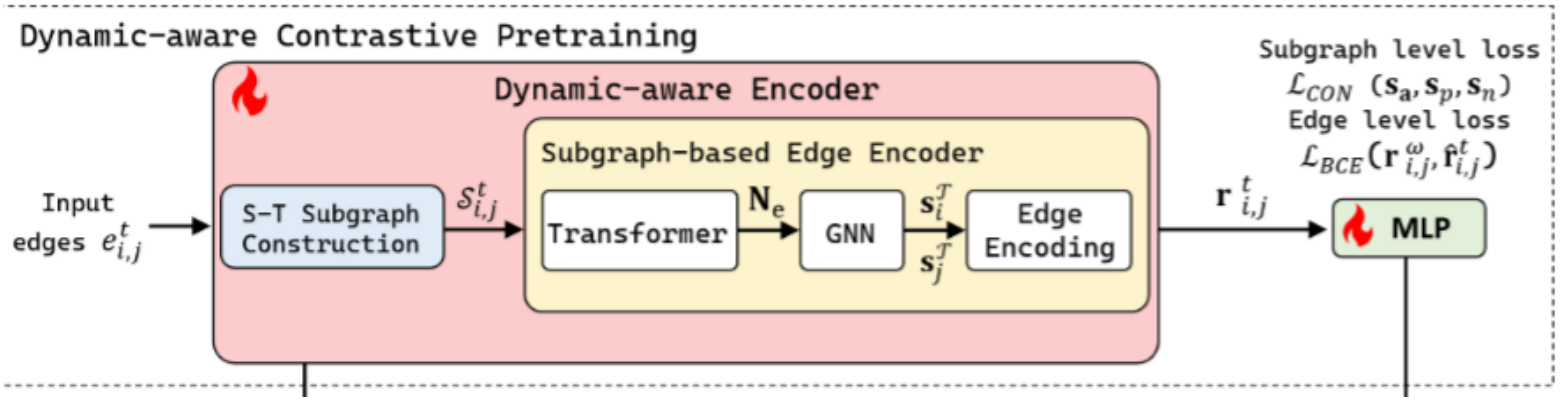
Method



- Anomalies often occur in local substructures.
  - ✓ Receptive field to local scale

- Sample importance-aware contextual node set.
  - ✓ Based on graph diffusion matrix, select to k nodes.
    - Global view of graph structure.
    - $S = \sum_{k=0}^{\infty} \theta_k T^k , where\ T = AD^{-1}, \quad \theta_k = \alpha(1-\alpha)^k\ or\ \frac{e^{-\beta}\beta^k}{k!}$
      - Degree of connectivity with continuous value
    - $s_{e_{tgt}} = s_{v1} + s_{v2}$
    - Select the top k nodes to form the contextual node set $\mathcal{U}(e_{tgt})$
    - Substructure $S(e_{tgt}) = \{v_1, v_2, \mathcal{U}(e_{tgt})\}$

AnomalyLLM - Method

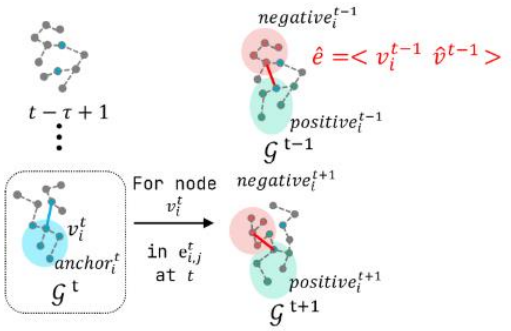


- Edges with different subgraph should dissimilar.

- Distinguishable between Edges and random sampled edge.

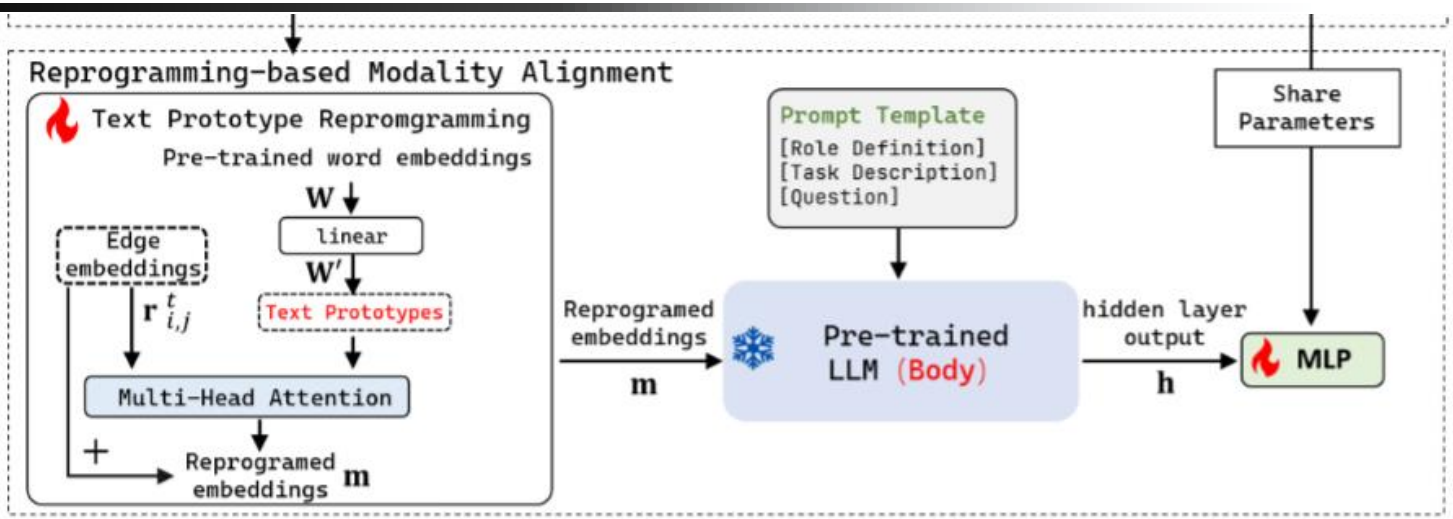
Negative Edge : not connected to e_{ij} in G_{t-1}, G_{t+1}

Negative Subgraph : Subgraph not connected node to e_{ij}



e process of contrastive training triplet

AnomalyLLM - Method



Edge embedding -> Query

W : word embedding

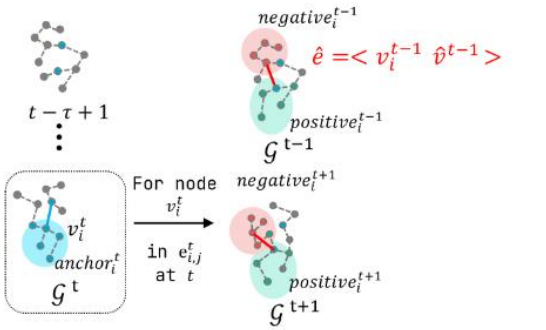
W' : prototype : transformed word embedding -> key value

Reprogrammed embedding

- ✓ soft-weighted word embedding
  - Soft-weighted : attention between word and edge embedding

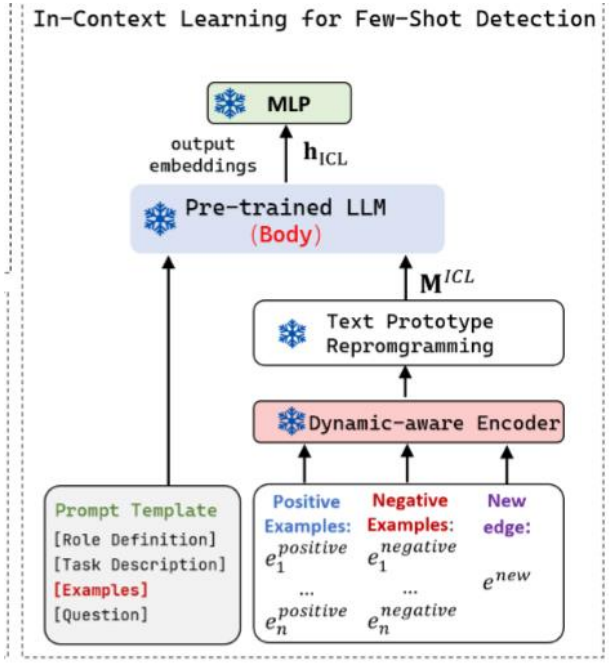
Anomaly Edge with text representation h

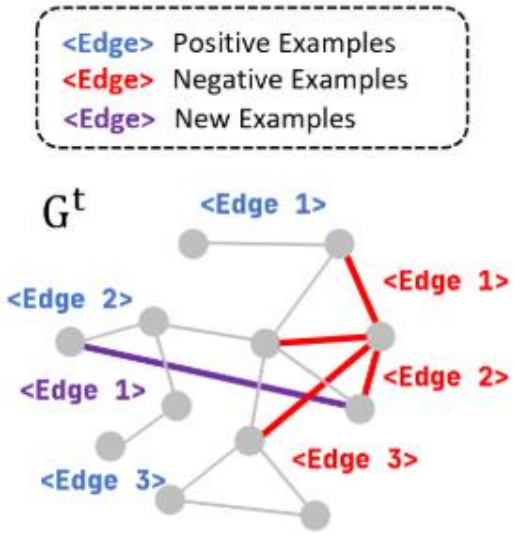
- ✓ Edges with different subgraph should dissimilar.



e process of contrastive training triplet

AnomalyLLM - Method





AnomalyLLM Instruction

① **Role Definition**:
As an AI few-shot learning approach

② **Task Description**:
The description of the anomaly type

③ **Examples:**
Positive Examples: Negative Examples:
Example 1:<Edge> Example 1:<Edge>
Label: Normal Label: Anomalous

④ **Question**:
Based on the pattern in the examples provided, classify new edge.
New Example :<Edge>
Label:?

Figure 4: The prompt of In-Context Learning

Instruction and edge embedding -> LLM

h_{ICL} = LLM()[:-1]

AnomalyLLM

Results

Table 1: Performance comparison results of few-shot anomaly detection on multiple anomaly types.

Dataset	Model	1-shot			5-shot			10-shot		
		CDA	LPL	HHL	CDA	LPL	HHL	CDA	LPL	HHL
BlogCataLog	StrGNN	0.5891	0.5756	0.5974	0.6018	0.6041	0.6122	0.6222	0.6329	0.6402
	AddGraph	0.5994	0.6023	0.5988	0.6097	0.6033	0.6104	0.6216	0.6238	0.6172
	Deep Walk	0.6102	0.6073	0.6202	0.6113	0.6122	0.6196	0.6155	0.6176	0.6154
	TGN	0.6732	0.6699	0.6919	0.7112	0.7023	0.7118	0.7263	0.7387	0.7311
	GDN	0.6733	0.6795	0.6609	0.6997	0.7051	0.7121	0.7321	0.7311	0.7319
	SAD	0.6841	0.6792	0.6411	0.7002	0.7018	0.6988	0.7342	0.7216	0.7265
	TADDY	0.6892	0.6983	0.6891	0.7148	0.7186	0.7177	0.7258	0.7326	0.7334
	AnomalyLLM	**0.8288**	**0.8334**	**0.8255**	**0.8331**	**0.8319**	**0.8407**	**0.8402**	**0.8456**	**0.8447**
UCI Message	StrGNN	0.6143	0.5956	0.5722	0.6113	0.7132	0.6512	0.6442	0.6724	0.6249
	AddGraph	0.5842	0.5466	0.5647	0.6018	0.6667	0.6321	0.4642	0.5728	0.7001
	Deep Walk	0.6198	0.6187	0.6142	0.6256	0.6263	0.6176	0.6255	0.6209	0.6197
	TGN	0.6521	0.6535	0.6643	0.7098	0.7193	0.7155	0.7335	0.7365	0.7324
	GDN	0.6577	0.6818	0.6611	0.7201	0.7289	0.7255	0.7493	0.7511	0.7546
	SAD	0.6703	0.6587	0.6693	0.7102	0.7146	0.7194	0.7416	0.7453	0.7406
	TADDY	0.6992	0.7078	0.6132	0.7204	0.7237	0.7218	0.7255	0.7278	0.7243
	AnomalyLLM	**0.8414**	**0.8358**	**0.8368**	**0.8446**	**0.8459**	**0.8424**	**0.8488**	**0.8546**	**0.8442**

Supervised (Deep Walk), Unsuprvised(StrGNN, AddGraph), Semi(GDN,TGN,SAD, TADDY)

1. Outperform existing method be attributed to the generization power of LLM
2. Semi-sup show improvement with increasing shot count.

AnomalyLLM

Results

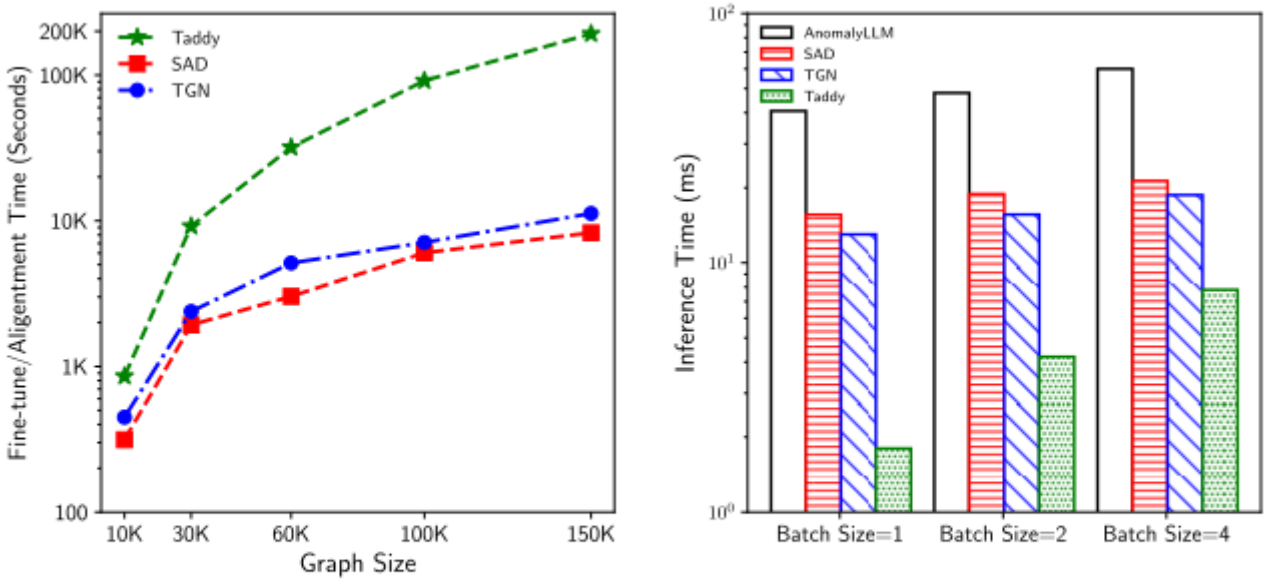


Figure 5: Inference time of AnomalyLLM

Poor efficiency

- Edge embedding = temporal + subgraph embedding
- transformer + GNN
- self attention reprogramming

Results

Table 3: Ablation Results

Dataset	Method	Anomaly Types		
		CDA	LPL	HHL
UCI Message	w/o ICL	0.7406	0.7465	0.7328
	w/o alignment	0.7849	0.7892	0.7994
	w/o encoder	0.7727	0.7883	0.7822
	AnomalyLLM	0.8402	0.8456	0.8447
BlogCatalog	w/o ICL	0.7398	0.7421	0.7396
	w/o alignment	0.7767	0.7812	0.7726
	w/o encoder	0.7821	0.7726	0.7732
	AnomalyLLM	0.8488	0.8546	0.8442

observe the edge construction
by focusing on subgraph embeddings
can extract useful information and capture the evolving properties of edges
In dynamic graphs

AnomalyLLM

Conclusion

- Integrate LLMs with dynamic graph anomaly detection
  - ✓ Addressing few-shot anomaly

- Introduce approach
  - ✓ Reprogram the edge embedding
    - To align semantics between graph and LLM

CONTENTS

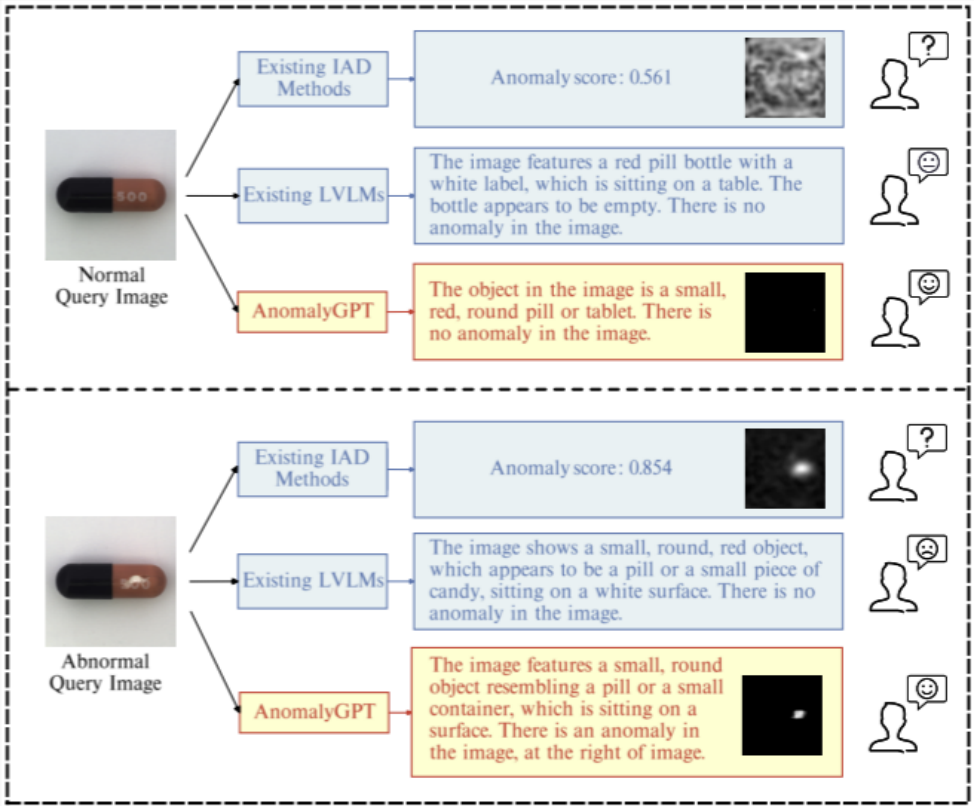
1. Introduction
   - Anomaly Detection
   - Graph anomaly detection
2. Related work
   - Generative based
     - Dominant (SDM, 2019)
     - MUSE(NIPS, 2024)
   - LLM
     - AnomalyLLM(IJCAI,2024)
   - **LVLM(vision, language)**
     - **AnomalyGPT(AAAI, 2024)**

AnomalyGPT (24, AAAI - oral)

Intro

- Detecting Industrial Anomalies Using Large Vison-Language Models
  - ✓ Using LVLM for Industrial Anomaly Detection(IAD)

Figure 1. Comparison between our AnomalyGPT, existing IAD methods and existing LVLMs.



Existing IAD methods can only provide anomaly scores and need manually threshold setting,
while existing LVLMs cannot detect anomalies in the image.

AnomalyGPT can not only provide information about the image but also indicate the presence and location of anomaly.

AnomalyGPT (24, AAAI - oral)

Intro

- Detecting Industrial Anomalies Using Large Vison-Language Models
  - ✓ The threshold varies considerably for each category of objects.
    - One-class-one-model.

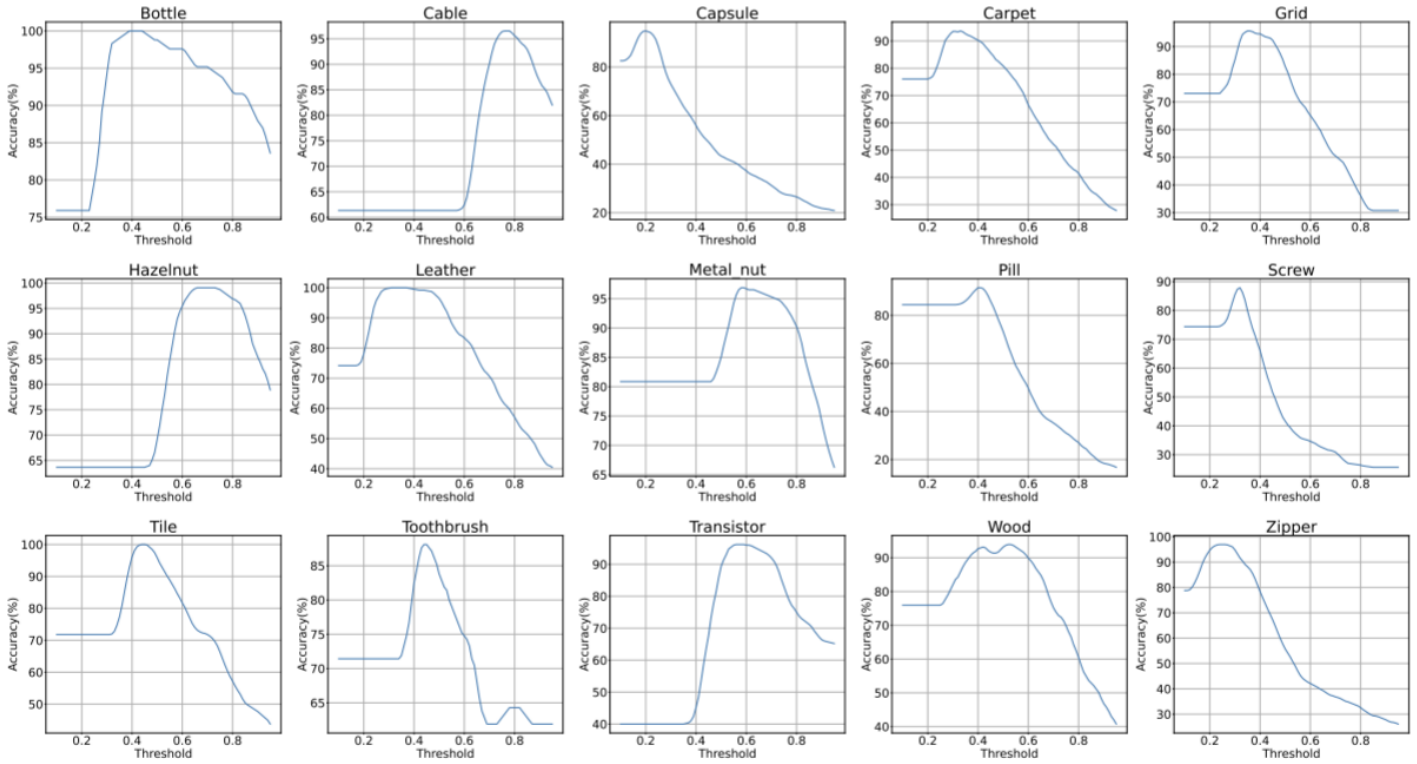
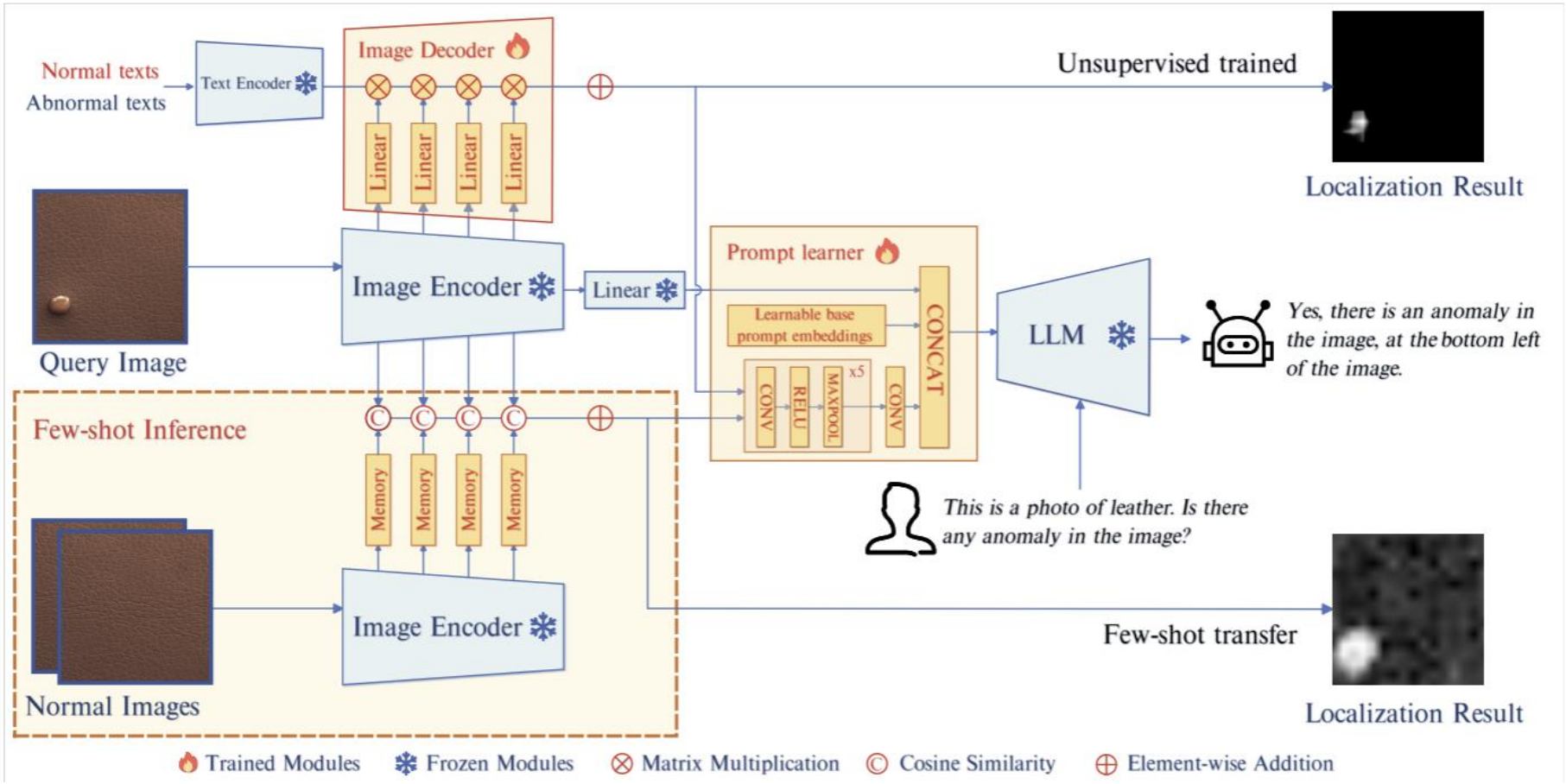


Figure 7. Experimental results of PatchCore [23] on the MVTec-AD [1] dataset across each category under different thresholds. The optimal threshold varies considerably for each category of objects.

AnomalyGPT

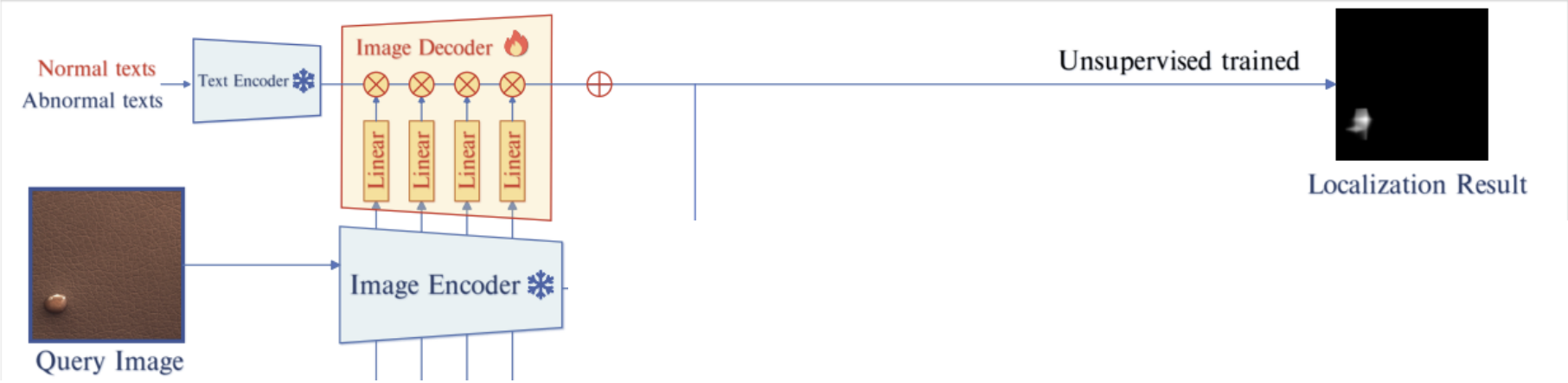
Method



- Image Encoder -> Decoder + Text -> Localization Results
- Prompt Learner -> LLM -> Answer
  - ✓ (Image + Localization Results + Learnable prompt)

AnomalyGPT

Method



- To achieve pixel-level anomaly localization
  - ✓ Compute the **similarity** between **texts** and intermediate **image** features.
  - ✓ Image encoder into 4 stage to obtain intermediate features
    - $M = \text{Upsample}\left(\sum_{i=1}^{4} \text{softmax}\left(\hat{F}_{patch}^{i} F_{text}^{T}\right)\right) \in \mathbb{R}^{H \times W}$
      - $\hat{F}_{patch}^{i} \in \mathbb{R}^{H_i \times W_i \times C_{text}}$: intermediate image feature into 4 stages in patch level.
      - $F_{text} \in \mathbb{R}^{2 \times C_{text}}$: normal and abnormal text features.
      - $\hat{F}_{patch}^{i} F_{text}^{T} \in \mathbb{R}^{H_i \times W_i \times 2}$: similarity between image path and text features.
      - Softmax : Map the similarity to probability indicating how a patch is abnormal..
      - Upsample : To original Image Size, $\sum_{i=1}^{4}$: summation over all feature hierarchal levels .

AnomalyGPT

Method

(a) *State*-level (normal)

- c := "[o]"
- c := "flawless [o]"
- c := "perfect [o]"
- c := "unblemished [o]"
- c := "[o] without flaw"
- c := "[o] without defect"
- c := "[o] without damage"

State-level (anomaly)

- c := "damaged [o]"
- c := "broken [o]"
- c := "[o] with flaw"
- c := "[o] with defect"
- c := "[o] with damage"

(b) *Template*-level

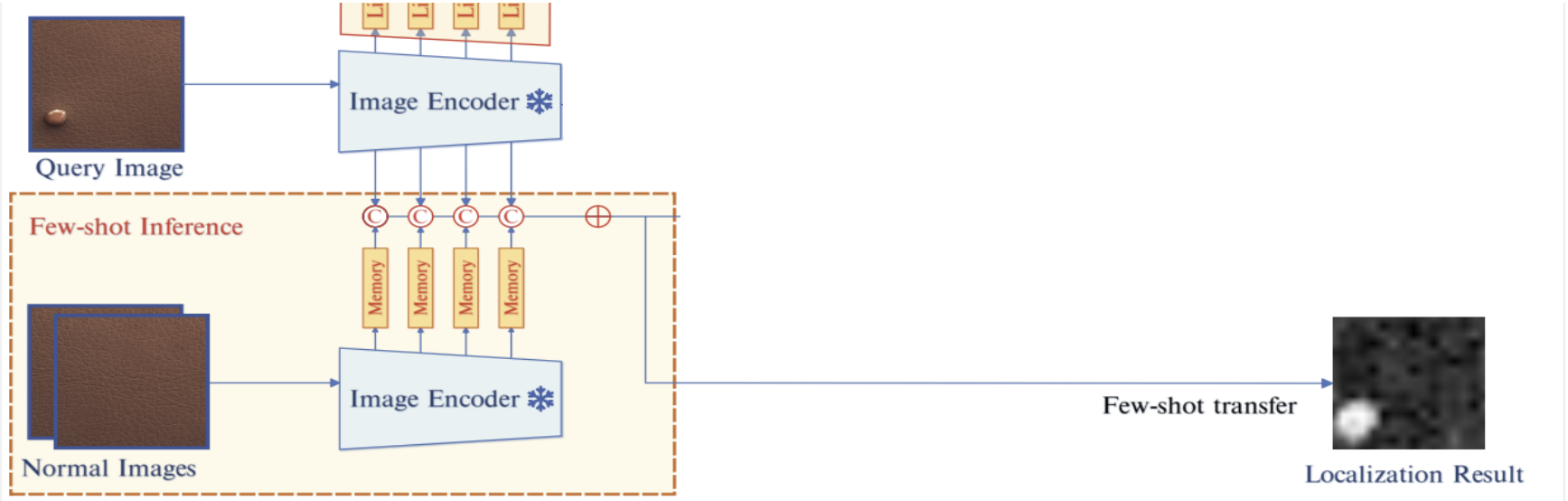
- "a cropped photo of the [c]."
- "a cropped photo of a [c]."
- "a close-up photo of a [c]."
- "a close-up photo of the [c]."
- "a bright photo of a [c]."
- "a bright photo of the [c]."
- "a dark photo of the [c]."
- "a dark photo of a [c]."
- "a jpeg corrupted photo of a [c]."
- "a jpeg corrupted photo of the [c]."
- "a blurry photo of the [c]."
- "a blurry photo of a [c]."
- "a photo of a [c]."
- "a photo of the [c]."
- "a photo of a small [c]."
- "a photo of the small [c]."
- "a photo of a large [c]."
- "a photo of the large [c]."
- "a photo of the [c] for visual inspection."
- "a photo of a [c] for visual inspection."
- "a photo of the [c] for anomaly detection."
- "a photo of a [c] for anomaly detection."

Table 5. Lists of multi-level texts considered in this paper to present normal and abnormal semantics.

- Normal and Abnormal text
  - ✓ State level
  - ✓ Template level

AnomalyGPT

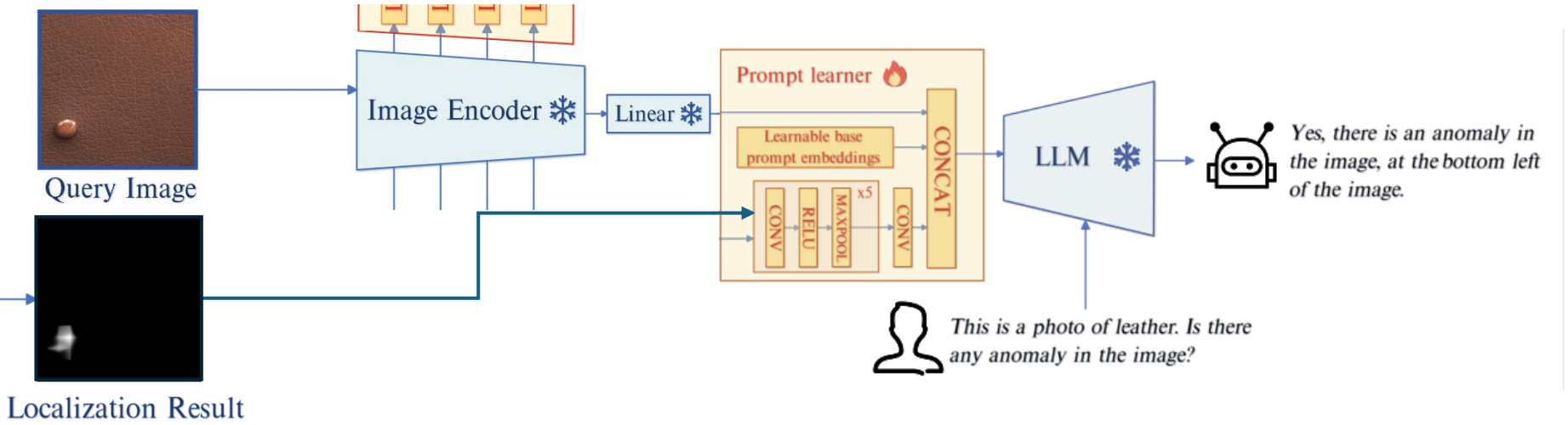
Method



- For few shot,
  - ✓ Similarity between normal samples and query.
    - $M = \text{Upsample}\left(\sum_{i=1}^{4}\left(1 - \max\left(\left(\hat{F}_{patch}^{i} B^{i^T}\right)\right)\right) \in \mathbb{R}^{H \times W}\right.$
      - $B^i \in \mathbb{R}^{N \times C_i}$: normal sample that image patch level feature in memory banks.
      - $\hat{F}_{patch}^{i} B^{i^T}$: similarity between normal samples and query Image.
      - $1 - \max(\text{sim}(x, y))$: most dissimilar normal sample -> abnormal

AnomalyGPT

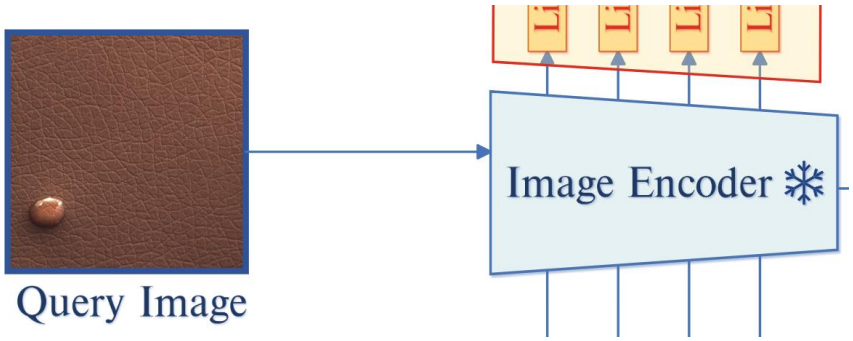
Method



- Prompt learner
  - ✓ To fine-tune
    - Encoder output -> linear : original image information. $\in \mathbb{R}^{H_i \times W_i \times C_{emb}}$
    - Learnable prompt embedding. $E_{base} \in \mathbb{R}^{n_1 \times C_{emb}}$
    - Localization Results $\in \mathbb{R}^{H \times W} \rightarrow E_{dec} \in \mathbb{R}^{n_2 \times C_{emb}}$

AnomalyGPT

Method



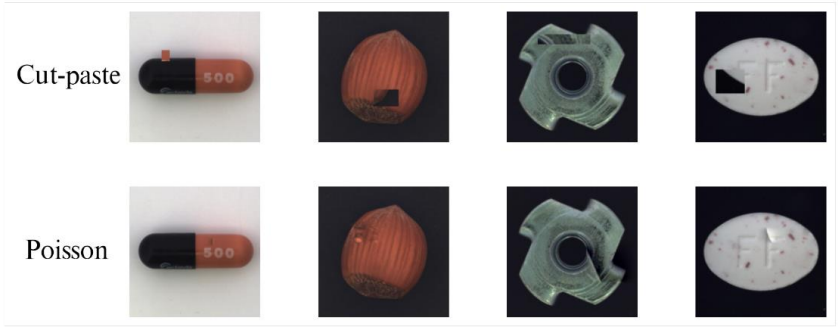


Figure 3. Illustration of the comparison between cut-paste and poisson image editing. The results of cut-paste exhibit evident discontinuities and the results of poisson image editing are more natural.

- Data Augmentation
  - ✓ To improve performance.
    - Cut-paste
      - Radom crop and past a block.
    - Poisson editing
      - Seamlessly crop and paste.

AnomalyGPT

Method

Prompts fed to the LLM typically follow the format:
Human: <Img>E_{img}</Img>E_{prompt}[Image Description]Is there any anomaly in the image?###Assistant:
$E_{img} \in \mathbb{R}^{C_{emb}}$ represents the image embedding being processed through the image encoder and linear layer, $E_{prompt} \in \mathbb{R}^{(n_1+n_2)\times C_{emb}}$ refers to the prompt embeddings generated by the prompt learner, and *[Image Description]* corresponds to the textual description of the image.

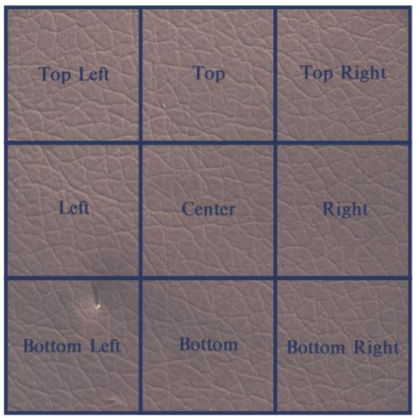


Figure 4. Illustration of the 3×3 grid of image, which is used to let LLM verbally indicate the abnormal position.

- Question Answer
  - ✓ Prompt tuning
    - Description of input image.
    - Queries the presence of anomaly
      - » Model continues to specify the location
  - ▪ Poisson editing
    - Seamlessly crop and paste.

AnomalyGPT

Method

Prompts fed to the LLM typically follow the format:
Human: <Img>E_{img}*</Img>*E_{prompt}*[Image Description]Is there any anomaly in the image?###Assistant:*

$E_{img} \in \mathbb{R}^{C_{emb}}$ represents the image embedding being processed through the image encoder and linear layer, $E_{prompt} \in \mathbb{R}^{(n_1+n_2)\times C_{emb}}$ refers to the prompt embeddings generated by the prompt learner, and *[Image Description]* corresponds to the textual description of the image.

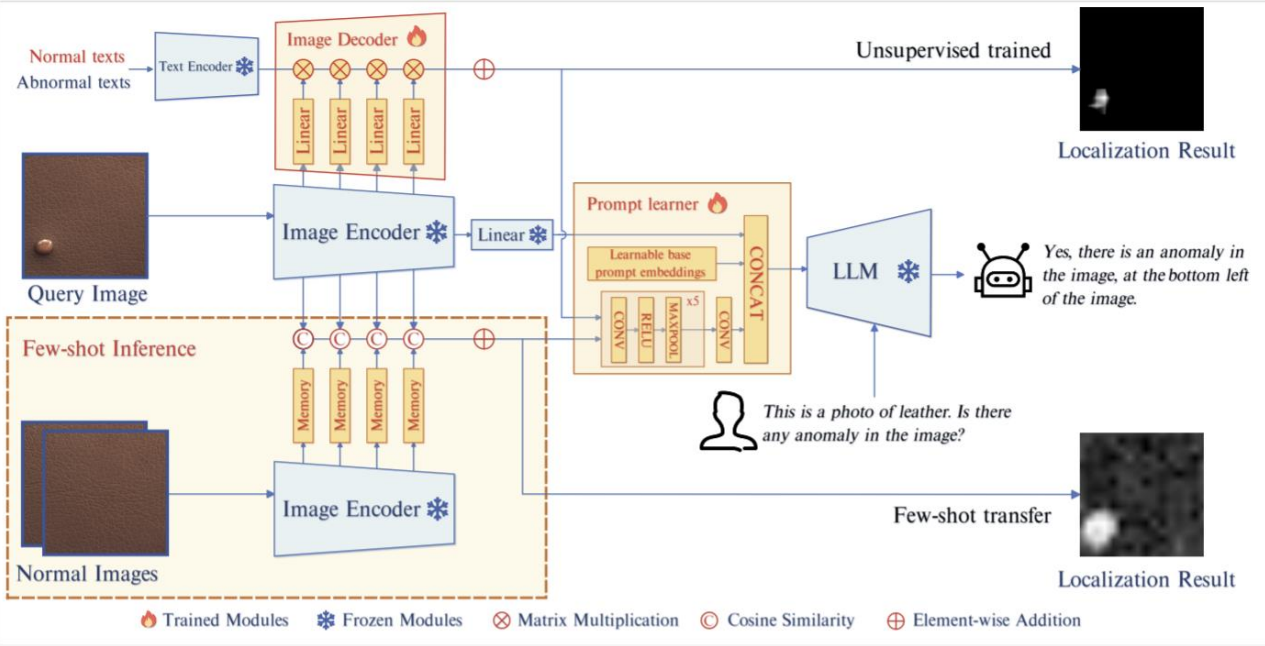
Class	Image description
Bottle	This is a photo of a bottle for anomaly detection, which should be round and without any damage, flaw, defect, scratch, hole or broken part.
Cable	This is a photo of three cables for anomaly detection, they are green, blue and grey, which cannot be missed or swapped and should be without any damage, flaw, defect, scratch, hole or broken part.
Capsule	This is a photo of a capsule for anomaly detection, which should be black and orange, with print '500' and without any damage, flaw, defect, scratch, hole or broken part.
Carpet	This is a photo of carpet for anomaly detection, which should be without any damage, flaw, defect, scratch, hole or broken part.
Grid	This is a photo of grid for anomaly detection, which should be without any damage, flaw, defect, scratch, hole or broken part.

AnomalyGPT

Method

- Overall Loss $L = \alpha L_{ce} + \beta L_{focal} + \gamma L_{dice}$

  - ✓ $L_{ce} = -\sum_{i=1}^{n} y_i \log(p_i)$
    - Ground Truth text and generated text
  - ✓ $L_{focal} = -\frac{1}{n}\sum_{i=1}^{n}(1-p_i)^{\gamma}\log(p_i)$
    - For image pixel
  - ✓ $L_{dice} = -\frac{\sum_{i=1}^{n} y_i \hat{y}_i}{\sum_{i=1}^{n} y_i^2 + \sum_{i=1}^{n} \hat{y}_i^2}$
    - For image mask



AnomalyGPT

Results

Setup	Method	MVTec-AD			VisA		
		Image-AUC	Pixel-AUC	Accuracy	Image-AUC	Pixel-AUC	Accuracy
1-shot	SPADE	81.0 ± 2.0	91.2 ± 0.4	-	79.5 ± 4.0	95.6 ± 0.4	-
	PaDiM	76.6 ± 3.1	89.3 ± 0.9	-	62.8 ± 5.4	89.9 ± 0.8	-
	PatchCore	83.4 ± 3.0	92.0 ± 1.0	-	79.9 ± 2.9	95.4 ± 0.6	-
	WinCLIP	93.1 ± 2.0	95.2 ± 0.5	-	83.8 ± 4.0	**96.4 ± 0.4**	-
	AnomalyGPT (ours)	**94.1 ± 1.1**	**95.3 ± 0.1**	**86.1 ± 1.1**	**87.4 ± 0.8**	96.2 ± 0.1	**77.4 ± 1.0**
2-shot	SPADE	82.9 ± 2.6	92.0 ± 0.3	-	80.7 ± 5.0	96.2 ± 0.4	-
	PaDiM	78.9 ± 3.1	91.3 ± 0.7	-	67.4 ± 5.1	92.0 ± 0.7	-
	PatchCore	86.3 ± 3.3	93.3 ± 0.6	-	81.6 ± 4.0	96.1 ± 0.5	-
	WinCLIP	94.4 ± 1.3	**96.0 ± 0.3**	-	84.6 ± 2.4	**96.8 ± 0.3**	-
	AnomalyGPT (ours)	**95.5 ± 0.8**	95.6 ± 0.2	**84.8 ± 0.8**	**88.6 ± 0.7**	96.4 ± 0.1	**77.5 ± 0.3**
4-shot	SPADE	84.8 ± 2.5	92.7 ± 0.3	-	81.7 ± 3.4	96.6 ± 0.3	-
	PaDiM	80.4 ± 2.5	92.6 ± 0.7	-	72.8 ± 2.9	93.2 ± 0.5	-
	PatchCore	88.8 ± 2.6	94.3 ± 0.5	-	85.3 ± 2.1	96.8 ± 0.3	-
	WinCLIP	95.2 ± 1.3	96.2 ± 0.3	-	87.3 ± 1.8	**97.2 ± 0.2**	-
	AnomalyGPT (ours)	**96.3 ± 0.3**	**96.2 ± 0.1**	**85.0 ± 0.3**	**90.6 ± 0.7**	96.7 ± 0.1	**77.7 ± 0.4**

Table 2. Few-shot IAD results on MVTec-AD and VisA datasets. Results are listed as the average of 5 runs and the best-performing method is in **bold**. The results for SPADE, PaDiM, PatchCore and WinCLIP are reported from [11].

- Our method outperform pervious approaches
  - ✓ Competitive Pixel level AUC

AnomalyGPT

Results

Decoder	Prompt learner	LLM	LoRA	MVTec-AD (unsupervised)			VisA (1-shot)		
				Image-AUC	Pixel-AUC	Accuracy	Image-AUC	Pixel-AUC	Accuracy
		✓		-	-	72.2	-	-	56.5
	✓	✓		-	-	73.4	-	-	56.6
		✓	✓	-	-	79.8	-	-	63.4
✓		✓		97.1	90.9	72.2	85.8	96.2	56.5
✓		✓	✓	97.1	90.9	84.2	85.8	96.2	64.7
✓	✓	✓	✓	96.0	88.1	83.9	85.8	**96.5**	72.7
✓				97.1	90.9	90.3	85.8	96.2	75.4
✓	✓	✓		**97.4**	**93.1**	**93.3**	**87.4**	96.2	**77.4**

Table 4. Results of ablation studies. The ✓ in "Decoder" and "Prompt learner" columns indicate module inclusion. The ✓ in "LLM" column denotes whether use LLM for inference and the ✓ in "LoRA" column denotes whether use LoRA to fine-tune LLM. In settings without LLM, the maximum anomaly score from normal samples is used as the classification threshold. In settings without decoder, due to the sole textual output from the LLM, we cannot compute image-level and pixel-level AUC.

- Decoder impressive pixel-level AD performance
  - ✓ Provide additional functionality
- Prompt learner outperforms Lora
  - ✓ Provide transferability.

AnomalyGPT

Results

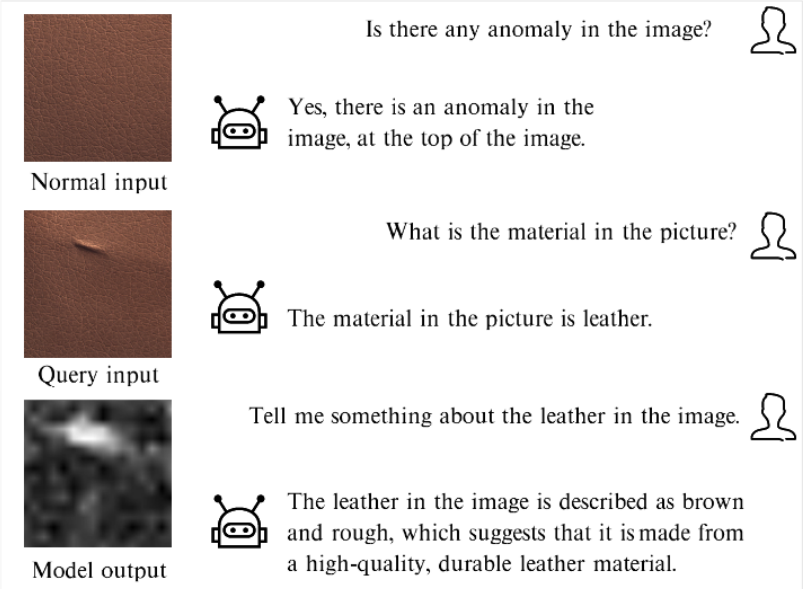


Figure 6. Qualitative example of AnomalyGPT in the **one-normal-shot** setting. The localization performance is slightly lower compared to the unsupervised setting due to the absence of parameter training.

- Presence of anomalies.
- Pinpointing anomaly location presenting pixel-level.
- Can engage in multi-turn dialogues

AnomalyGPT

Conclusion

- Conversational IAD vision language model

  - ✓ Whether contain anomalies
  - ✓ Pinpointing location without specified threshold

  - ✓ Remarkable performance
    - Few shot in-context learning.

Thank you ☺

Q & A