Supplementary Materials of "MissionGNN: Hierarchical Multimodal GNN-based Weakly Supervised Video Anomaly Recognition with Mission-Specific Knowledge Graph Generation"

Sanggeon Yun
University of California, Irvine
Irvine, CA, 92617
sanggeoy@uci.edu

Ryozo Masukawa
University of California, Irvine
Irvine, CA, 92617
rmasukaw@uci.edu

Minhyoung Na
Kookmin University
Seoul, South Korea, 02707
minhyoung0724@kookmin.ac.kr

Mohsen Imani*
University of California, Irvine
Irvine, CA, 92617
m.imani@uci.edu

Table 1. Ablation study results on different size of KGs on XD-Violence dataset.

Case	# of Key Concept Nodes	Depth of Subgraphs	AP
-	30	1	**98.42**
Wide KG	60	1	97.66
Deep KG	30	2	97.06

A. Ablation Study on KG Size Variations

Table 1 presents the results of our ablation study examining the impact of KG size variations on anomaly detection across different anomaly types. We explored two distinct scenarios: one where the KG is expanded by adding more key concept nodes, effectively widening the KG, and another where the KG is deepened through the incorporation of additional subgraphs from ConceptNet.

Our findings suggest that merely increasing the size of the KG can lead to overfitting, resulting in diminished performance. This underscores the importance of developing tailored, mission-specific KGs, as emphasized in our main paper, rather than simply enlarging existing KGs.

B. Ablation Study on Decaying Threshold Method

Table 2 presents the results of our ablation study investigating the impact of various decay rates on the performance of our proposed decaying threshold-based anomaly localization method, specifically within the UCF-Crime dataset context. The findings demonstrate that our decaying threshold approach significantly enhances performance, evidenced by a 2.37% increase in the AUC score.

Table 2. Ablation study results on different decay rate of decaying threshold on UCF-Crime dataset.

Deacay Rate	AUC
0.5	63.72
0.6	79.23
0.7	77.16
0.8	81.75
0.9	81.07
0.99	**84.48**
0.999	81.57
0.9999	83.57
1.0 (w/o Decaying Threshold)	82.11

Moreover, the study highlights the critical importance of selecting an appropriate decay rate to maximize performance, with optimal results observed for decay rates $\alpha_d \geq 0.9$. This emphasizes the necessity of fine-tuning the decaying rate for achieving high performance in anomaly detection as we discuss in the main paper.

C. Ablation Study on Loss Terms

Table 3 presents an ablation study evaluating the impact of each loss term in our proposed framework. The evaluation was conducted on the UCF-Crime dataset, measuring AUC scores after removing each loss term from the training loop. The results show that using all loss terms together yields the highest performance, with a significant score drop of at least 5.23% when any individual loss term is omitted. These findings highlight the critical importance of the proposed loss terms in achieving optimal model performance.

*Corresponding Author

Table 3. Ablation study results on loss terms.

Case	AUC
Without $\mathcal{L}_N$	60.49
Without $\mathcal{L}_{spa}$	79.25
Without $\mathcal{L}_{smt}$	77.40
Full Model	**84.48**

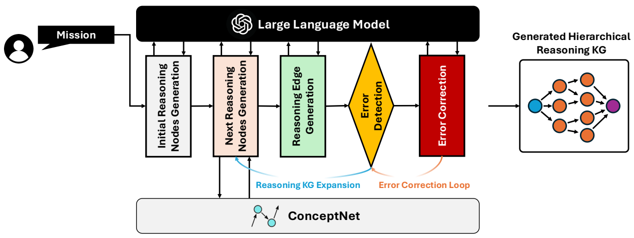

Figure 1. **Detailed process of mission-specific knowledge graph generation**.

D. Ablation Study on Joint Embedding Models

Table 5 shows the results of our ablation study comparing different joint embedding models. We conducted the evaluation on the UCF-Crime dataset, measuring AUC scores using two joint embedding models: OpenCLIP and ImageBind. Aside from the embedding model and its dimensionality, all other hyperparameters were kept constant. The results demonstrate that our proposed framework performs consistently across both models, with only a 0.36% difference in scores. This suggests that the reasoning capability of our Hierarchical GNN framework is robust, regardless of the specific joint embedding model used.

E. Ablation Study on Local Context Size

Table 5 presents the results of our ablation study on different local context sizes. Using the UCF-Crime dataset, we measured AUC scores by varying the number of consecutive frames used during inference. The results show a trade-off between performance and resource usage, including GPU memory and runtime. As the local context size increases, the AUC scores improve, but at the cost of slightly higher memory and runtime demands. However, the performance gains tend to level off, with a smaller improvement between 20 and 30 frames compared to the larger gap between 10 and 20 frames. This suggests that after a certain context size, the model's ability to benefit from additional frames diminishes, likely indicating diminishing returns on performance versus resource consumption.

F. Knowledge Graph Generation

Figure 1 illustrates the detailed pipeline for mission-specific knowledge graph (KG) generation. The process begins by taking the user's input and initiating the Initial Reasoning Nodes Generation step. In this phase, initial keywords are generated using a large language model (LLM)—GPT-4 was employed in our evaluation—by utilizing pre-formatted prompts. Table 6 provides the prompts used in this step, where SUBJECT represents the user-defined mission, and DUP_NODES refers to keywords that have appeared in previous layers.

Following the generation of the initial reasoning nodes, the process moves to the Next Reasoning Nodes Generation step, where related keywords are inferred from the previously generated keywords. This step combines the LLM with ConceptNet to extract related keywords. ConceptNet's output is used to guide the LLM in generating the next set of reasoning nodes. Table 7 presents the prompts used in this step, with COMMA_SEPARATED_LIST representing the previous layer's keywords, and SUGGESTED_KEYWORDS containing related keywords extracted from ConceptNet.

After generating the next set of reasoning nodes, the system identifies logical relations between the nodes from the previous and current layers during the Reasoning Edge Generation step. Table 8 shows the prompts for this step, where SUBJECT is replaced with a newly generated keyword, COMMA_SEPARATED_LIST contains the keywords from the previous layer, and NOT_APPEARED_NODES refers to selected nodes that do not appear in the previous layer, as identified by the LLM.

Once the new layer and corresponding edges have been generated, the framework performs an Error Detection step to validate the correctness of each node and edge according to the definition of our reasoning KG. If any invalid nodes or edges are detected, the Error Correction step is triggered. During this step, errors are corrected using error correction prompts, with a maximum of three correction attempts allowed. If errors persist beyond the third attempt, the erroneous nodes or edges are pruned from the KG. After the error detection and correction loop, the framework returns to the Next Reasoning Nodes Generation step to expand the reasoning KG until the desired number of layers is achieved.

To evaluate the effectiveness of our error correction mechanism, we generated 190 reasoning KGs using our proposed method and tracked error rates by layer and by number of correction attempts, as illustrated in Figure 2. Across all layers, the number of errors significantly decreases with each correction attempt. For node generation (shown in Figure 2.(A)), the process results in a maximum of only 1% pruned nodes by the second layer. Similarly, edge generation (shown in Figure 2.(B)) results in a maximum of 0.02% pruned edges. These results demonstrate that the prompts used in our KG generation framework successfully facilitate knowledge extraction from both the LLM and ConceptNet. Figure 3 presents one KG we gener-

Table 4. Ablation study results on different pre-trained joint embedding models.

Case	Number of Supporting Modalities	Joint Space Dimensionality	AUC
OpenCLIP (ViT-B-32)	2	512	84.12
ImageBind (imagebind_huge)	6	1024	**84.48**

Table 5. Ablation study results on a different number of frames.

Number of Frames	AUC	Inference GPU Memory Usage	Inference Runtime
10	80.78	**13,468 MiB**	**39 ms**
20	83.47	13,478 MiB	43 ms
30	**84.48**	13,540 MiB	46 ms

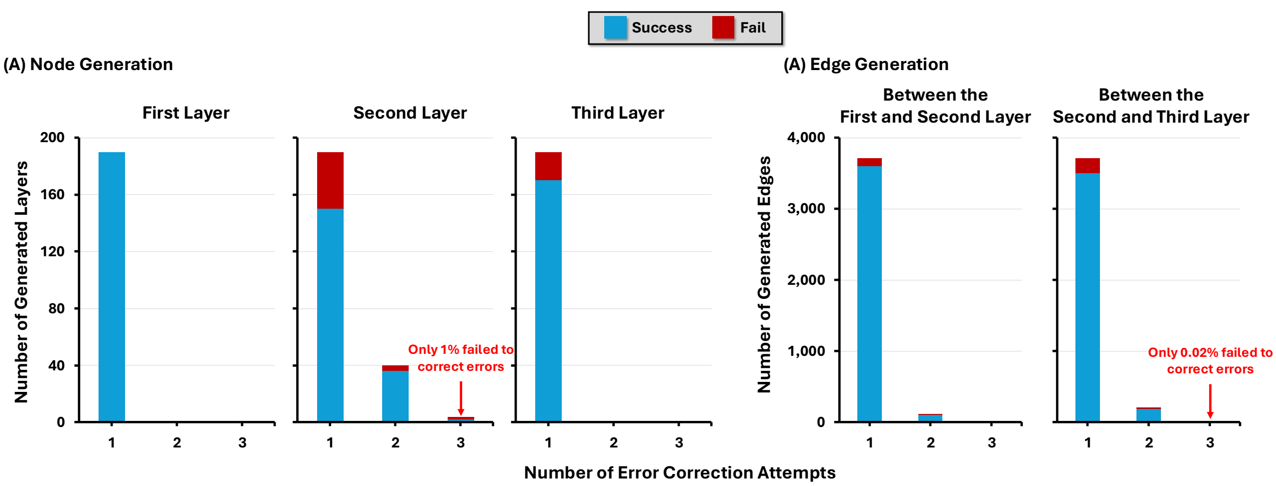

Figure 2. **Portion of fails on generating valid (A) nodes and (B) edges on each layer by the number of attempts**.

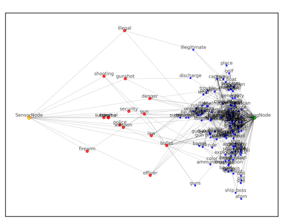

Figure 3. Example of KG for detecting the "Shooting" category in the UCF-Crime dataset. Each color represents: Yellow: Sensor Node, Red: Key Concept Nodes, Blue: Sub-graph Nodes, Green: Encoding Node.

ated.

Table 6. Persona: Initial Reasoning Node Generation

Type of Prompt	**Prompt Format**
System	Reference: A knowledge graph have hierarchical levels starting from naive observations to final prediction. Each level has the following inference words directly connected only with the previous level words. There are NOT the same words in different levels. Persona: You are a knowledge graph engineer who generates knowledge graph that will help to classify images. Objective: You will be provided a subject. Follow these steps to answer the user queries. Step 1. Observe 20 important words from a image which is related to the provided subject. Do not respond anything for this step. Step 2. Create a comma-separated list of the words that you observed. The comma-separated list you just created is first level of the knowledge graph. Keep in mind. Do not respond anything for this step. Step 3. Print first level of the knowledge graph on the first line. No extraneous text or characters other than the comma-separated list.
User	Subject: {SUBJECT}
Error Correction	The following concepts already appear in previous levels: {DUP_NODES} You must generate new concepts that can be inferred from previous level concepts. Correct this error and give a corrected answer. No extraneous text or characters other than the comma-separated list. Subject: {SUBJECT}

Table 7. Persona: Next Reasoning Nodes Generation

Type of Prompt	Prompt Format
System	Reference: A knowledge graph have hierarchical levels starting from naive observations to final prediction. Each level has the following inference words directly connected only with the previous level words. There are NOT the same words in different levels. Persona: You are a knowledge graph engineer who generates knowledge graph that will help to classify images. Objective: You will be provided a subject. And you will be provided comma-separated list which is the previous level of the knowledge graph. And you will be provided suggested keywords. Follow these steps to answer the user queries. Step 1. Create words related to the provided subject which can be explained from combination of several words from previous level. Reference suggested keywords for this step. If you have better keywords, suggest them. Do not respond anything for this step. Step 2. Create a comma-separated list of the words that you just created in step 1. The length of comma-seperated list must be 20. The comma-separated list you just created is next level of the knowledge graph. Keep in mind. Do not respond anything for this step. Print next level of the knowledge graph on the first line. No extraneous text or characters other than the comma-separated list.
User	Subject: {SUBJECT} Comma-separated list: {COMMA_SEPARATED_LIST} Suggested keywords: {SUGGESTED_KEYWORDS}
Error Correction	The following concepts already appear in previous levels: {DUP_NODES} You must generate new concepts that can be inferred from previous level concepts. Correct this error and give a corrected answer. No extraneous text or characters other than the comma-separated list. Subject: {SUBJECT} Comma-separated list: {COMMA_SEPARATED_LIST} Suggested keywords: {SUGGESTED_KEYWORDS}

Table 8. Persona: Reasoning Edge Generation

<table>
<tr><th>Type of Prompt</th><th>Prompt Format</th></tr>
<tr><td>System</td><td>Reference:
A knowledge graph have hierarchical levels starting from naive observations to final prediction.
Each level has the following inference words directly connected only with the previous level words.
There are NOT the same words in different levels.

Persona:
You are a knowledge graph engineer who generates knowledge graph that will help to classify images.

Objective:
You will be provided a subject and a comma-separated list.
Follow these steps to answer the user queries.

Step 1.
Select maximum 5 words from provided comma-separated list which are related to inferring provided subject.
Do not respond anything for this step.

Step 2.
Create a comma-separated list of the selected words that you observed.
Do not respond anything for this step.

Step 3.
Print the comma-separated list.
No extraneous text or characters other than the comma-separated list.</td></tr>
<tr><td>User</td><td>Subject: {SUBJECT}
Comma-separated list: {COMMA_SEPARATED_LIST}</td></tr>
<tr><td>Error Correction</td><td>The following concepts do not appear in the previous level nodes: {NOT_APPEARED_NODES}
You must select concepts from the previous level concepts that can be important clues to infer the new concept.
Correct this error and give a corrected answer.
No extraneous text or characters other than the comma-separated list.
Subject: {SUBJECT}
Comma-separated list: {COMMA_SEPARATED_LIST}</td></tr>
</table>