Continuous GNN-based Anomaly Detection on Edge using Efficient Adaptive Knowledge Graph Learning

Sanggeon Yun*, Ryozo Masukawa*, William Youngwoo Chung*, Minhyoung Na†,
Nathaniel D. Bastian‡, and Mohsen Imani*⋆

*University of California, Irvine, CA, USA, †Kookmin University, Seoul, Republic of Korea,
‡United States Military Academy, West Point, NY, USA, ⋆Corresponding Author
Emails: {sanggeoy, rmasukaw, chungwy1}@uci.edu, minhyoung0724@kookmin.ac.kr,
nathaniel.bastian@westpoint.edu, m.imani@uci.edu

Abstract—The increasing demand for robust security solutions across various industries has made Video Anomaly Detection (VAD) a critical task in applications such as intelligent surveillance, evidence investigation, and violence detection. Traditional approaches to VAD often rely on finetuning large pre-trained models, which can be computationally expensive and impractical for real-time or resource-constrained environments. To address this, MissionGNN introduced a more efficient method by training a graph neural network (GNN) using a fixed knowledge graph (KG) derived from large language models (LLMs) like GPT-4. While this approach demonstrated significant efficiency in computational power and memory, it faces limitations in dynamic environments where frequent updates to the KG are necessary due to evolving behavior trends and shifting data patterns. These updates typically require cloud-based computation, posing challenges for edge computing applications. In this paper, we propose a novel framework that facilitates continuous KG adaptation directly on edge devices, overcoming the limitations of cloud dependency. Our method dynamically modifies the KG through a three-phase process: pruning, alternating, and creating nodes, enabling real-time adaptation to changing data trends. This continuous learning approach enhances the robustness of anomaly detection models, making them more suitable for deployment in dynamic and resource-constrained environments.

Index Terms—Video Anomaly Detection, Graph Neural Networks, Knowledge Graph, Edge Computing, Continuous Learning, Adaptive Knowledge Graphs

I. INTRODUCTION

The increasing concern for security across various domains has elevated the importance of Video Anomaly Detection (VAD). VAD plays a pivotal role in applications such as intelligent surveillance [1], [2], evidence analysis [3], and real-time violence detection [4], [5]. It aims to automatically identify and classify behaviors that deviate from normal patterns in video data [6]. Given its significant impact, researchers have dedicated considerable attention to developing VAD, from creating benchmarking datasets [7]–[9] to advancing anomaly detection models [10]–[17].

One promising approach in this area is the MissionGNN model [16], which efficiently addresses anomaly detection by leveraging a graph neural network (GNN) trained on a task-specific knowledge graph (KG). Unlike previous models that require fine-tuning large pre-trained models, MissionGNN constructs a KG using reasoning processes derived from a large language model (LLM) like GPT-4 [18]. This KG, tailored to the target anomaly, is generated in a one-time process
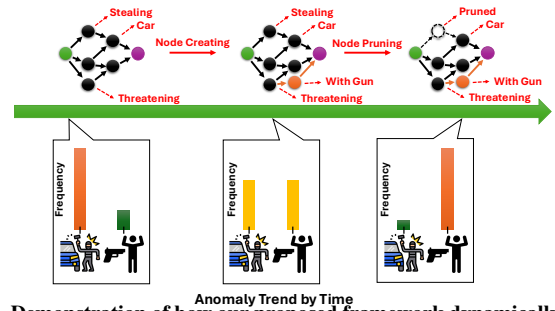

Fig. 1. **Demonstration of how our proposed framework dynamically adapts to anomaly trend shift in the real world**.

without the need for ongoing gradient computation or re-training of the LLM. This significantly reduces computational and memory overhead, making MissionGNN highly efficient for edge computing during the training phase.

However, real-world environments are dynamic, often necessitating updates to anomaly detection models due to evolving trends, shifts in behavior patterns, or changes in targets of interest. While MissionGNN is adept at efficiently handling minor updates, it eventually requires the regeneration of the KG to remain aligned with these changes. This process typically involves the use of cloud resources for new KG generation, imposing limitations on edge computing applications where cloud connectivity may be restricted or undesirable.

To address this challenge, we propose a novel framework that enables continuous knowledge graph adaptation directly on edge devices. Our approach employs a three-phase process—pruning, altering, and creating nodes—to dynamically modify the KG in response to evolving data trends as described in Figure 1. This allows for continuous learning and adaptation without relying on external cloud resources, ensuring that anomaly detection remains robust and relevant in changing environments.

In summary, our work represents a novel contribution to the field, offering the following key advancements:

- We introduce a framework that enables continuous adaptation of the knowledge graph directly on edge devices without the need for cloud connectivity, addressing the limitations of existing models in dynamic environments.
- Our approach utilizes pruning, altering, and creating nodes to modify the KG in response to new data trends, allowing the model to adapt to evolving anomalies effectively.

- By facilitating continuous learning and adaptation, our framework ensures that anomaly detection models remain robust and relevant, improving performance in real-world, ever-changing scenarios.

II. BACKGROUND AND RELATED WORKS

A. Video Anomaly Detection

State-of-the-art (SOTA) VAD models leverage advanced architectures like CNNs, LSTMs, and transformers to capture complex spatial-temporal patterns, enhancing anomaly detection accuracy [10], [12]. Recent approaches also integrate LLMs and vision models (LVMs) for improved contextual reasoning [14], [19]–[21]. However, LLM-based models are computationally expensive, limiting real-time applicability. MissionGNN addresses this by using a graph neural network (GNN) built on an LLM-generated knowledge graph, reducing computational costs while maintaining strong performance in real-time scenarios [16]. Yet, none of these models effectively handle continuous adaptation to evolving anomaly trends.

B. Edge Computing on VAD

Deploying VAD models on edge devices is critical for real-time surveillance in resource-constrained environments. Edge computing minimizes latency and dependence on cloud infrastructure, wich is essential for timely anomaly detection [22]–[25]. While lightweight models enable real-time processing, they often underperform compared to SOTA models [26], [27]. MissionGNN offers a solution by encoding knowledge from large models into a lightweight graph neural network (GNN), reducing resource demands and enabling efficient edge deployment [16]. However, MissionGNN still requires cloud access to update its knowledge graph when anomalies change, limiting its utility in disconnected environments. Our approach allows continuous knowledge graph updates directly on edge devices, ensuring models remain effective without cloud dependency.

C. Reasoning on Knowledge Graph via GNN

Integrating graph neural networks (GNNs) with knowledge graphs (KGs) offers a promising approach for VAD by capturing semantic relationships and enabling structured reasoning [10], [14], [28], [29]. MissionGNN exemplifies this potential, using an LLM-generated KG to detect anomalies efficiently with minimal resources [16]. However, in dynamic environments, frequent KG updates are necessary to maintain accuracy, typically requiring cloud connectivity [18]. Our method enables continuous KG adaptation on edge devices, allowing the VAD model to evolve with new data trends without relying on cloud resources, thus maintaining efficiency and effectiveness in real-time, dynamic settings.

III. METHODOLOGY

A. Overall Pipeline

Figure 2 illustrates the overall pipeline of our proposed framework, which consists of three main stages: Mission-Specific Knowledge Graph Generation, Light-weight GNN-based Decision Model Training, and Deployment and Continuous KG Adaptive Learning. Initially, we generate a mission-specific KG using the reasoning KG generation framework of MissionGNN, as depicted in Figure 2.(A). This step serves as prior knowledge extraction from a large language model (LLM), incorporating domain-specific insights into the KG. Next, we train a GNN-based decision model that predicts anomalous actions, as shown in Figure 2.(B). This model operates on the generated mission-specific KG, leveraging structured knowledge to enhance anomaly detection performance while being efficient enough to run on edge devices. Finally, we deploy the trained GNN-based decision model on edge computing devices and implement our proposed continuous KG adaptive learning mechanism over the deployed KG, as illustrated in Figure 2.(C). This approach allows the model to self-adjust to evolving anomaly trends even after deployment in environments where devices operate independently without cloud connectivity. Additionally, it provides interpretable KG retrieval, ensuring that the dynamically adapted KG remains human-readable and maintains the interpretability inherent in the MissionGNN model.

By integrating these stages, our framework enables efficient and adaptive anomaly detection on edge devices without reliance on continuous cloud access. The continuous KG adaptive learning mechanism ensures that the model remains up-to-date with new patterns of anomalous behavior, enhancing its robustness and applicability in dynamic real-world scenarios.

B. Mission-specific KG Generation

Figure 3 illustrates our mission-specific reasoning KG generation framework. Given a user-defined mission and predefined prompt formats, we utilize an LLM to generate a KG, which a GNN utilizes for reasoning to accomplish the specified mission.

We define a reasoning KG as a hierarchical directed acyclic graph (DAG) where each node represents a unique concept expressed in short text and is assigned to a specific level. Edges in the KG connect nodes from level i to nodes in level $i + 1$ only, enforcing a strict hierarchical structure.

The KG generation process begins by creating initial reasoning nodes that form the first layer of the KG. Following this, we initiate a KG expansion loop based on the nodes generated in the previous level. In each iteration of this loop, the following steps are performed: Node Generation, Edge Generation, and Error Detection and Correction. In the node generation, the LLM is prompted to generate the next set of reasoning nodes that can be inferred from the nodes at the current level. Then, in the edge generation step, we request the LLM to establish direct connections (edges) between nodes in the current level and the newly generated nodes, adhering to the hierarchical structure of the KG. Finally, at the error detection and correction step, we check for errors in the expanded level, focusing on two types: Duplicated Concepts – nodes that represent concepts already present in previous levels – and Invalid Edges – connections that violate the rule that edges only connect nodes from level i to level $i + 1$. If no errors are detected, the process proceeds to generate the next layer. If errors are found, we enter an error correction loop where we prompt the LLM to correct the detected issues using predefined prompts that specify the problems. Acknowledging that the LLM might introduce new
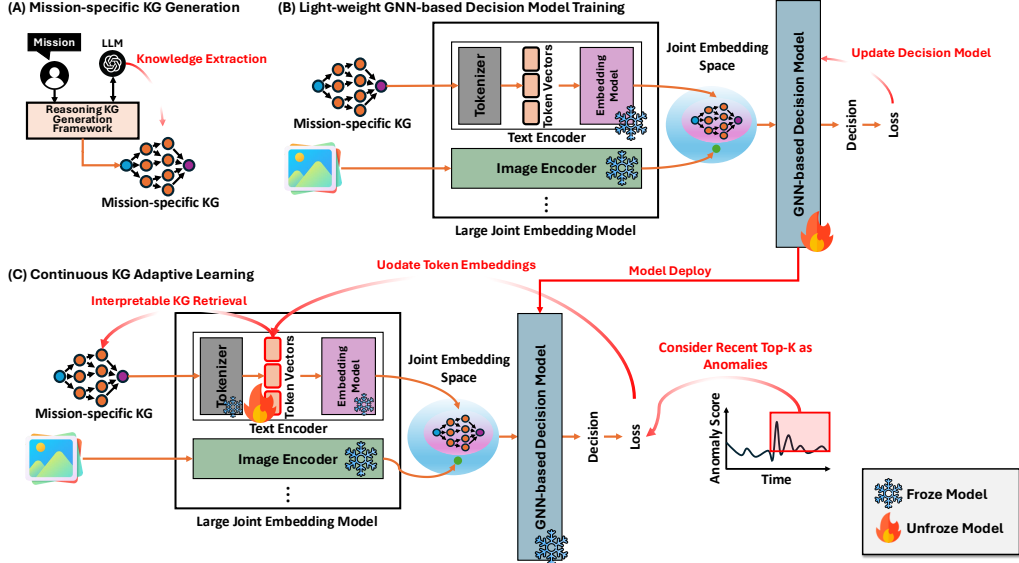
Fig. 2. **The overall framework for our proposed KG adaptive learning framework utilizing the novel concept of hierarchical graph neural network.**
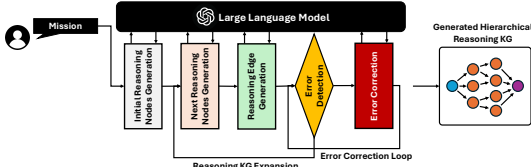

Fig. 3. **Detailed description of our mission-specific knowledge graph generation framework.**

errors during correction, this loop repeats until no errors are detected or a maximum number of iterations is reached. If the maximum number of iterations is reached without resolving all errors, we prune the problematic nodes or edges.

After generating the desired number of layers, we attach a sensor node and an embedding node to the KG, completing the reasoning KG generation procedure. This finalized KG is structured to facilitate efficient reasoning for the given mission, providing a solid foundation for subsequent tasks in our framework.

C. Light-weight GNN-based Decision Model Training

We follow the Hierarchical GNN framework proposed from MissionGNN to be our GNN-based decision model. Our GNN-based decision model receives a given frame F_t at the sensor node for each mission-specific KG G and reasons by propagating information from the sensor node to its embedding node.

Initially, the frame F_t is encoded by $\mathcal{E}_I(F_t)$ to create an embedding vector for the sensor node v^{snr}. Then, $d + 2$ GNN layers are applied in a hierarchical manner, allowing the reasoning process to propagate from the sensor node. The final embedding is located in the embedding node. Each GNN layer $\mathcal{G}_l(.)$ consists of five sub-layers: a dense layer, a hierarchical message passing layer, a hierarchical aggregate layer, a batch normalization layer, and an activation layer.

First, the dense layer ($\phi_l: \mathbb{R}^{|V| \times D_{l-1}} \to \mathbb{R}^{|V| \times D_l}$), which refines the embedding space to more accurately depict the relationships between knowledge embeddings, is formulated as follows:

$$\phi_l(X_{l-1}) = \mathbf{W}_l^{\phi} X_{l-1} + \mathbf{b}_l^{\phi} \tag{1}$$

where D_{l-1} indicates the dimensionality of the previous layer and D_l indicates the dimensionality of the current layer, $\mathbf{W}_l^{\phi}$ is a trainable parametric matrix, $\mathbf{b}_l^{\phi}$ is a trainable bias, and $X_{l-1} \in \mathbb{R}^{|V| \times D_{l-1}}$ represents the embeddings of the previous layer nodes.

The hierarchical message passing layer $\mathcal{M}_l^h: \mathbb{R}^{|V| \times D_l} \to \mathbb{R}^{|E^{(l)}| \times D_l}$ then computes messages corresponding to each node, where $E^{(l)} \subset E$ is a set of edges that connect nodes at depth $l - 1$ with nodes at depth l. The message passing is defined as:

$$\mathcal{M}_l^h(X) = \{X_s \cdot X_d\}_{(s,d) \in E^{(l)}} \tag{2}$$

where s and d denote source and destination nodes respectively, and $X \in \mathbb{R}^{|V| \times D_l}$ represents the node embeddings. The hierarchical aggregate layer $\mathcal{A}_l^h: \{\mathbb{R}^{|V| \times D_l}, \mathbb{R}^{|E^{(l)}| \times D_l}\} \to \mathbb{R}^{|V| \times D_l}$ consolidates messages from source nodes into a unified vector, while preserving the embeddings of nodes that do not receive any messages:

$$\mathcal{A}_l^h(X, M) =$$
$$\left\{ X_d \mathbf{1}(d \notin V^{(l)}) + \frac{\sum_{(s,d) \in E^{(l)}} M_{s,d}}{|\{s \in V | (s,d) \in E^{(l)}\}|} \mathbf{1}(d \in V^{(l)}) \right\}_{d \in V} \tag{3}$$

where $X \in \mathbb{R}^{|V| \times D_l}$ indicates node embeddings, $M \in \mathbb{R}^{|E^{(l)}| \times D_l}$ indicates messages from $\mathcal{A}_l^h$, and $\mathbf{1}(.)$ is an indicator function.

The final form of the GNN layer at layer l is:

$$\mathcal{G}_l(X_{l-1}) = \mathrm{ELU}\left(\mathrm{BatchNorm}\left(\mathcal{A}_l^h(X, M)\right)\right) \tag{4}$$

During this procedure, embeddings representing knowledge at the current layer $X_l = \mathcal{G}_l(X_{l-1}) \in \mathbb{R}^{|V| \times D_l}$ are derived based on embeddings from the previous layer $X_{l-1} \in \mathbb{R}^{|V| \times D_{l-1}}$.

Unlike $\mathcal{M}_l^h$ and $\mathcal{A}_l^h$, which operate exclusively on nodes in $V^{(l)}$ and edges in $E^{(l)}$, other components apply to all nodes V and edges E. This ensures that embeddings for every node can be integrated into the same embedding space for subsequent reasoning propagation. The final embedding $\vec{r}_{T_i}$
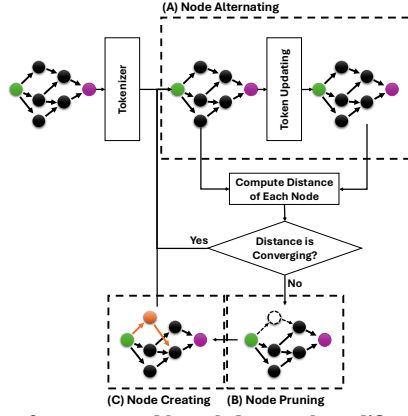
Fig. 4. **Pipeline of our proposed knowledge graph modification mechanism**.

for KG reasoning is extracted from v^{ecd} of X_{d+2}, the final embedding layer of KG G. Lastly, the reasoning embedding $\vec{f}_t$ for the frame encoding $\mathcal{E}_I(F_t)$ is formed by concatenating all $\vec{r}_{T_i}$ from each KG: $\vec{f}_t = \vec{r}_{T_1} \frown \vec{r}_{T_2} \frown \cdots \frown \vec{r}_{T_n} \in \mathbb{R}^D$, where $D = \sum_i D_{d+2}$.

Our method employs hierarchical GNNs to reason over multiple knowledge graphs using a single frame. To extract richer video information, we use a transformer-based temporal model $\mathcal{T} \colon \mathbb{R}^{T \times D} \rightarrow \mathbb{R}^D$, processing sequences of embeddings from previous $T - 1$ consecutive frames $\vec{F}_t = \{\vec{f}_{t-T+1}, \vec{f}_{t-T+2}, \cdots, \vec{f}_t\} \in \mathbb{R}^{T \times D}$ to the transformer and only takes the last output embedding $\vec{f}_t' = \mathcal{T}(\vec{f}_t) \in \mathbb{R}^D$ which is corresponding to the last input $\vec{f}_t$. Focusing on short-term relationships, the model assumes anomalies are detectable in brief intervals.

The output of the short-term temporal model $\vec{f}_t'$ is then processed by a decision model $f^{dec} \colon \mathbb{R}^D \rightarrow \mathbb{R}^{n+1}$ which is consists of a simple linear layer and softmax function:

$$f^{dec}(\vec{f}_t') = \text{Softmax}(\mathbf{W}^{dec} \vec{f}_t' + \mathbf{b}^{dec}) \qquad (5)$$

where the matrix $\mathbf{W}^{dec}$ and the bias $\mathbf{b}^{dec}$ are both trainable parameters. The decision model's output, $\vec{s}_t = f^{dec}(\vec{f}_t')$, provides the estimated probabilities. Specifically, the probability that frame F_t is normal is given by $p_N(F_t) = \vec{s}_{t,0}$, while the probabilities that the frame belongs to each type of anomaly $i \in \{1, 2, \ldots, n\}$ are given by $p_{A,i}(F_t) = p_A(F_t)p_{i|A}(F_t) = \vec{s}_{t,i}$. Here, $p_A(F_t) = 1 - p_N(F_t)$ represents the probability that the frame is abnormal, and $p_{i|A}(F_t) = \frac{\vec{s}_{t,i}}{1 - p_N(F_t)}$ represents the conditional probability of anomaly type i given that the frame is abnormal.

D. Continuous KG Adaptive Learning

As illustrated in Figure 2.(C), after training the lightweight GNN-based decision model, we deploy both the decision model and the generated mission-specific KG to an edge computing device. These devices often have limited computational power and lack internet access, making cloud computing infeasible. Once deployed, the system continuously monitors the anomaly score distribution over time. It keeps track of this distribution and identifies the top K data points with the highest anomaly scores within the most recent N data points, considering them
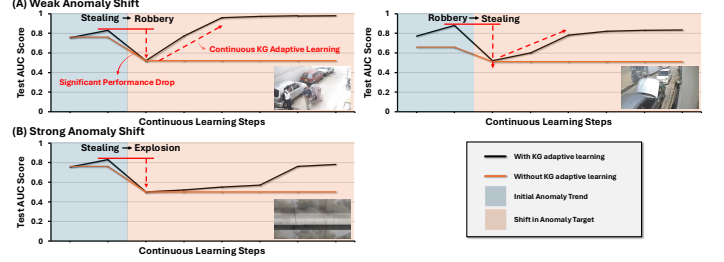

Fig. 5. **Test AUC scores measured across shifts in the anomaly target, comparing different types of anomalies.**

as anomalies. These selected data points are then used to compute loss functions, and backpropagation is performed to update the token embeddings of the mission-specific KG. Importantly, only the embeddings of the KG tokens are updated; the weights of other models, including the large joint embedding model and the GNN-based decision model, remain unchanged.

The parameter K is determined based on the score distribution. Specifically, given the change in the mean of the anomaly score distribution, defined as $\Delta m = m_t - m_{t'} < 0$, where m_t represents the mean value at the current time t, K is calculated as $K = |\Delta m| \times N$. This approach helps in identifying new types of data points that are similar to the initially trained anomalous actions, allowing the model to adjust to shifts in anomaly patterns. The hyperparameters t' and N should be tuned using a validation set to optimize performance.

Figure 4 provides a detailed view of our continuous KG adaptive learning mechanism, focusing specifically on KG structure modification. After tokenizing the initially generated KG, the system continues to update the token embeddings of each node, as shown in Figure 4.(A), using the mechanism explained in Figure 3. After each token update, it computes the distance between the old and updated token embeddings of a node using the L2 distance metric.

If the distance does not increase, we consider the node to be converging towards a certain concept, and no action is taken. However, if the distance increases, indicating divergence, we initiate a node pruning process as depicted in Figure 4.(B). The node and its connected edges are removed from the KG. Subsequently, we perform a node creation procedure, illustrated in Figure 4.(C), where a new node with a random token embedding is created at the same level as the pruned node, along with random edge connections.

This adaptive learning mechanism allows the KG to evolve over time, adjusting to new anomaly patterns without requiring cloud connectivity or retraining the entire model. It ensures that the system remains effective in detecting anomalies in dynamic environments, even when operating solely on edge devices.

E. Interpretable KG Retrieval

The core idea behind Interpretable KG Retrieval is to translate the new token embeddings—which capture semantic knowledge generated by our real-time KG adaptation—into interpretable concept words. We base our retrieval procedure on the work from CoOp [30] and extend it to be compatible with ImageBind [31] instead of CLIP [32]. Given the adaptively learned embeddings from our framework, the proposed retrieval algorithm searches for similar token embeddings based on a
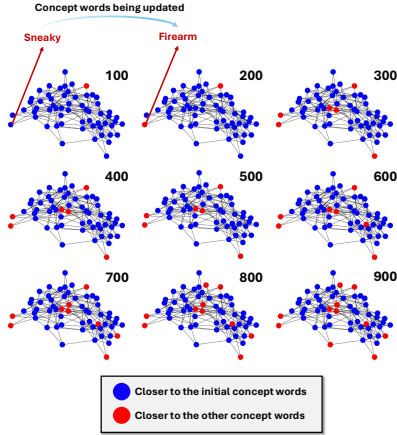
Fig. 6. **Qualitative evaluation of knowledge updates within the proposed knowledge graph update loops, with each number representing the iteration count**.

similarity metric and retrieves the corresponding interpretable words.

While Interpretable KG Retrieval could support any token vocabulary, we utilize the original simple byte-pair encoding (BPE) vocabulary [33] used in ImageBind for tokenization. Each token corresponds to an index in the pre-trained BPE vocabulary, allowing us to decode any given token back to its original text form.

For each learned embedding from our framework, we perform a similarity search within the token embedding space to identify the indices of the nearest tokens. Multiple similarity metrics for retrieving the nearest words were tested, including dot product and cosine similarity. Although these metrics were successful, we opted to use Euclidean distance as the metric, as it outperformed the others in our experiments. This process results in a distance metric where smaller values indicate token embeddings that are more similar to the given learned embeddings.

Using this similarity metric, we select the top K similar embeddings for each learned embedding. We collect the indices of these top K similar token embeddings and convert them back into interpretable words using the tokenizer's decoder. By doing so, we translate the adaptive representations optimized by our framework into human-readable text.

As a result of this process, we retrieve an interpretable knowledge graph where the embeddings have been learned and optimized based on the given input data. This approach allows for the exploration of new semantic connections related to the data and provides an interpretation of the model's optimized representations in dynamic environments. However, similar to CoOp, since the embeddings are optimized in a continuous space, the retrieved words may not always make perfect sense, as the embeddings may capture semantic knowledge beyond the existing vocabulary.

IV. EXPERIMENTS

A. Experimental Settings

1) Implementation Details: We implemented our multi-modal embedding model based on the MissionGNN approach [16], utilizing the pre-trained ImageBind Huge model.

GPT-4 and ConceptNet 5 were used to generate the mission-specific knowledge graph. The loss balance coefficients, λ_{spa} and λ_{smt}, were both set to 0.001. We optimized the model using the AdamW optimizer [34] with a learning rate of 10^{-5}, weight decay of 1.0, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. The decaying threshold α_d was set to 0.9999. For our GNN models, we maintained a consistent dimensionality of $D_{m_i,l} = 8$ across all layers. The short-term temporal model used an inner dimensionality of 128 with 8 attention heads. Training was conducted for 3,000 steps with a mini-batch size of 128.

2) Dataset: We evaluated our model using the UCF-Crime dataset [7], a benchmark for Video Anomaly Detection (VAD). UCF-Crime comprises 1,900 untrimmed surveillance videos depicting 13 types of real-world anomalies with significant public safety relevance. The dataset is divided into a training set with 800 normal and 810 anomalous videos, and a testing set with 150 normal and 140 anomalous videos.

B. Evaluation of Adaptation to Anomaly Trend Shifts

In this section, we assess the adaptability of our proposed framework in response to shifts in anomaly trends. The evaluation begins by setting an initial target anomaly for generating a mission-specific knowledge graph (KG), followed by training a GNN-based decision model on this KG. After completing the initial training, the trained GNN model is deployed, and KG adaptive learning is performed using samples of the initially selected target anomaly type and corresponding non-anomalous samples from the training set.

Following several iterations with the initial anomaly type, the target anomaly is shifted to a different type, while keeping the same non-anomalous samples in the training set. KG adaptation continues under the new anomaly type, and we measure the test AUC scores at each adaptation step.

Figure 5 presents the evaluation results, comparing the performance of our KG adaptation method against a static KG approach. We selected three types of anomalies from the UCF-Crime dataset: Stealing, Robbery, and Explosion. Two scenarios were considered: a weak anomaly shift, where the new anomaly is closely related to the previous one, and a strong anomaly shift, where there is a significant difference between the new and previous anomalies.

In the weak anomaly shift scenario, shown in Figure 5.(A), the initial performance drops noticeably when the anomaly shift occurs, but the model quickly adapts, with the AUC scores converging to higher levels, demonstrating effective adaptation. In contrast, the strong anomaly shift scenario, depicted in Figure 5.(B), exhibits a slower improvement in performance after introducing a new, more distantly related anomaly, reflecting the greater challenge in adapting to more significant shifts in anomaly type.

C. Evaluation of KG Adaptation using Interpretable KG Retrieval

In this evaluation, we assess how the knowledge graph is updated over time to qualitatively determine whether meaningful updates occur. For this analysis, we extracted learned token embeddings from the experiment detailed in subsection IV-B,

TABLE I

DETAILED COMPUTATIONAL AND PERFORMANCE COMPARISON BETWEEN BASELINE AND PROPOSED METHODS

Metric	Baseline Method (Cloud-based KG Updates)	Proposed Method (Edge-based KG Adaptation)
Initial Setup		
Human Intervention	Yes	Yes
Initial KG Generation Time (minutes)	1	1
Initial KG Generation Computational Cost (FLOPs)	1×10^{15}	1×10^{15}
Memory Usage for KG (GB)	0.5	0.5
Memory Usage for GPT-4 during Initial KG Generation (GB)	200	200
Edge Device Storage Requirements (GB)	1	1
Monthly Updates and Maintenance		
Human Intervention	Yes	No
KG Update Frequency (per month)	4	0
KG Update Time per Update (minutes)	1	0
Total KG Update Time (minutes/month)	$4 \times 1 = 4$	0
GPT-4 Computational Cost per KG Update (FLOPs/update)	1×10^{15}	0
Total GPT-4 Computational Cost (FLOPs/month)	$4 \times 1 \times 10^{15} = 4 \times 10^{15}$	0
Edge Device Computational Cost per Adaptation (FLOPs/day)	N/A	1×10^{9}
Total Edge Device Computational Cost (FLOPs/month)	N/A	$30 \times 1 \times 10^{9} = 3 \times 10^{10}$
Memory Usage for GPT-4 during Updates (GB)	200	0
Network Bandwidth Usage for KG Updates (GB/month)	High (Approx. 2 GB)	Zero
Edge Device Energy Consumption per Update (Joules)	N/A	Minimal (Approx. 5 J)
Operational Performance		
Average AUC score	0.93	0.91
Latency for KG Update (seconds)	High (Cloud-dependent)	Low (Real-time)
Scalability (Number of Edge Devices Supported)	Limited by Cloud Resources	High (Independent)

focusing on the scenario where the anomaly shifts from Stealing to Robbery. We then applied our proposed interpretable KG retrieval method. As shown in Figure 6, the KG demonstrates a gradual update process, converging toward new concept word embeddings that align with the new anomaly actions. For example, the concept word "Sneaky" gradually converges to a token embedding associated with "Firearm," which is consistent with the shift in anomaly from Stealing to Robbery.

D. Computational Efficiency

In this section, we evaluate the computational efficiency of our proposed framework compared to existing methods that rely on cloud-based KG updates using LLMs such as GPT-4. We focus on key metrics such as computational cost, memory usage, and anomaly detection performance.

Table I presents a comparison between the baseline method—which relies on frequent KG updates using GPT-4 in the cloud—and our proposed method, which performs continuous KG adaptation directly on edge devices without further use of LLMs. We conducted our measurements under an assumed scenario where the anomaly trend alternates between *Stealing* and *Robbery* four times per month. For the baseline method, we generated a new KG each time the anomaly trend changed, resulting in four KG updates per month. In contrast, our proposed edge-based KG adaptation model performed a single loop of KG modification once per day. While assessing computational costs, we also measured the test AUC score, averaging the daily AUC scores over the evaluation period.

Our proposed method eliminates the need for frequent KG updates using GPT-4, significantly reducing computational cost and memory requirements. By performing continuous KG adaptation on edge devices, we avoid the overhead of cloud-based updates and associated latency. Despite these reductions, our method maintains high anomaly detection performance, with an AUC score only slightly lower than the baseline method.

The results demonstrate that our framework offers a computationally efficient alternative to cloud-dependent methods. By enabling continuous KG adaptation on edge devices without the need for further LLM use, we achieve significant savings in computational cost and memory usage while maintaining high anomaly detection accuracy. This makes our approach suitable for deployment in resource-constrained environments where cloud connectivity is limited or undesirable.

V. CONCLUSIONS

We introduced a framework for VAD that allows continuous KG adaptation on edge devices, removing the need for cloud updates. Building on the MissionGNN framework, our approach adapts to evolving anomalies in real time, reducing computational costs while maintaining high detection accuracy. This makes it ideal for resource-constrained environments, offering an efficient, scalable alternative to cloud-dependent VAD solutions.

REFERENCES

[1] Q. Bao, F. Liu, Y. Liu, L. Jiao, X. Liu, and L. Li, "Hierarchical scene normality-binding modeling for anomaly detection in surveillance videos," in *Proceedings of the 30th ACM International Conference on Multimedia*, pp. 6103–6112, 2022.

[2] X. Feng, D. Song, Y. Chen, Z. Chen, J. Ni, and H. Chen, "Convolutional transformer based dual discriminator generative adversarial networks for video anomaly detection," in *Proceedings of the 29th ACM International Conference on Multimedia*, pp. 5546–5554, 2021.

[3] S. K. Nanda, D. Ghai, P. Ingole, and S. Pande, "Soft computing techniques-based digital video forensics for fraud medical anomaly detection," *Computer Assisted Methods in Engineering and Science*, vol. 30, no. 2, pp. 111–130, 2022.

[4] K. B. Sahay, B. Balachander, B. Jagadeesh, G. A. Kumar, R. Kumar, and L. R. Parvathy, "A real time crime scene intelligent video surveillance systems in violence detection framework using deep learning techniques," *Computers and Electrical Engineering*, vol. 103, p. 108319, 2022.

[5] M. Islam, A. S. Dukyil, S. Alyahya, and S. Habib, "An iot enable anomaly detection system for smart city surveillance," *Sensors*, vol. 23, no. 4, p. 2358, 2023.

[6] J. J. P. Suarez and P. C. Naval Jr, "A survey on deep learning techniques for video anomaly detection," *arXiv preprint arXiv:2009.14146*, 2020.

[7] W. Sultani, C. Chen, and M. Shah, "Real-world anomaly detection in surveillance videos," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[8] P. Wu, j. Liu, Y. Shi, Y. Sun, F. Shao, Z. Wu, and Z. Yang, "Not only look, but also listen: Learning multimodal violence detection under weak supervision," in *European Conference on Computer Vision (ECCV)*, 2020.

[9] W. Liu, D. L. W. Luo, and S. Gao, "Future frame prediction for anomaly detection – a new baseline," in *2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[10] G. Li, G. Cai, X. Zeng, and R. Zhao, "Scale-aware spatio-temporal relation learning for video anomaly detection," in *European Conference on Computer Vision*, pp. 333–350, Springer, 2022.

[11] W. Sultani, C. Chen, and M. Shah, "Real-world anomaly detection in surveillance videos," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6479–6488, 2018.

[12] Y. Tian, G. Pang, Y. Chen, R. Singh, J. W. Verjans, and G. Carneiro, "Weakly-supervised video anomaly detection with robust temporal feature magnitude learning," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4975–4986, 2021.

[13] P. Wu and J. Liu, "Learning causal temporal relation and feature discrimination for anomaly detection," *IEEE Transactions on Image Processing*, vol. 30, pp. 3513–3527, 2021.

[14] L. Zanella, B. Liberatori, W. Menapace, F. Poiesi, Y. Wang, and E. Ricci, "Delving into clip latent space for video anomaly recognition," *arXiv preprint arXiv:2310.02835*, 2023.

[15] Y. Zhou, Y. Qu, X. Xu, F. Shen, J. Song, and H. Shen, "Batchnorm-based weakly supervised video anomaly detection," *arXiv preprint arXiv:2311.15367*, 2023.

[16] S. Yun, R. Masukawa, M. Na, and M. Imani, "Missiongnn: Hierarchical multimodal gnn-based weakly supervised video anomaly recognition with mission-specific knowledge graph generation," *arXiv preprint arXiv:2406.18815*, 2024.

[17] R. Masukawa, S. Yun, Y. Yamaguchi, and M. Imani, "Pv-vtt: A privacy-centric dataset for mission-specific anomaly detection and natural language interpretation," *arXiv preprint arXiv:2410.22623*, 2024.

[18] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[19] G. Zara, S. Roy, P. Rota, and E. Ricci, "Autolabel: Clip-based framework for open-set video domain adaptation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11504–11513, 2023.

[20] H. Chen, W. Huang, Y. Ni, S. Yun, Y. Liu, F. Wen, A. Velasquez, H. Latapie, and M. Imani, "Taskclip: Extend large vision-language model for task oriented object detection," *arXiv preprint arXiv:2403.08108*, 2024.

[21] S. Jeong, H. Chen, S. Yun, S. Cho, W. Huang, X. Liu, and M. Imani, "Expanding event modality applications through a robust clip-based encoder," *arXiv preprint arXiv:2412.03093*, 2024.

[22] M. Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 50, no. 1, pp. 30–39, 2017.

[23] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE internet of things journal*, vol. 3, no. 5, pp. 637–646, 2016.

[24] S. Yun, H. Chen, R. Masukawa, H. Errahmouni Barkam, A. Ding, W. Huang, A. Rezvani, S. Angizi, and M. Imani, "Hypersense: Hyperdimensional intelligent sensing for energy-efficient sparse data processing," *Advanced Intelligent Systems*, p. 2400228, 2024.

[25] S. Jeong, H. E. Barkam, S. Yun, Y. Kim, S. Angizi, and M. Imani, "Exploiting boosting in hyperdimensional computing for enhanced reliability in healthcare," *arXiv preprint arXiv:2411.14612*, 2024.

[26] L. Wang, J. Zhang, J. Chuan, R. Ma, and A. Fei, "Edge intelligence for mission cognitive wireless emergency networks," *IEEE Wireless Communications*, vol. 27, no. 4, pp. 103–109, 2020.

[27] A. C. Cob-Parro, C. Losada-Gutiérrez, M. Marrón-Romera, A. Gardel-Vicente, and I. Bravo-Muñoz, "Smart video surveillance system based on edge computing," *Sensors*, vol. 21, no. 9, p. 2958, 2021.

[28] H. Chen, Y. Ni, A. Zakeri, Z. Zou, S. Yun, F. Wen, B. Khaleghi, N. Srinivasa, H. Latapie, and M. Imani, "Hdreason: Algorithm-hardware codesign for hyperdimensional knowledge graph reasoning," *arXiv preprint arXiv:2403.05763*, 2024.

[29] H. E. Barkam, S. Yun, H. Chen, P. Gensler, A. Mema, A. Ding, G. Michelogiannakis, H. Amrouch, and M. Imani, "Reliable hyperdimensional reasoning on unreliable emerging technologies," in *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, pp. 1–9, IEEE, 2023.

[30] K. Zhou, J. Yang, C. C. Loy, and Z. Liu, "Learning to prompt for vision-language models," *International Journal of Computer Vision*, vol. 130, no. 9, pp. 2337–2348, 2022.

[31] R. Girdhar, A. El-Nouby, Z. Liu, M. Singh, K. V. Alwala, A. Joulin, and I. Misra, "Imagebind: One embedding space to bind them all," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15180–15190, 2023.

[32] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*, pp. 8748–8763, PMLR, 2021.

[33] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (K. Erk and N. A. Smith, eds.), (Berlin, Germany), pp. 1715–1725, Association for Computational Linguistics, Aug. 2016.

[34] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.