Multimodal Event Processing: A Neural-Symbolic Paradigm for the Internet of Multimedia Things

Edward Curry, Dhaval Salwala, Praneet Dhingra, Felipe Arruda Pontes, and Piyush Yadav

***Abstract*—With the Internet of Multimedia Things (IoMT) becoming a reality, new approaches are needed to process real-time multimodal event streams. Existing approaches to event processing have limited consideration for the challenges of multimodal events, including the need for complex content extraction, and increased computational and memory costs. This article explores event processing as a basis for processing real-time IoMT data. This article introduces the multimodal event processing (MEP) paradigm, which provides a formal basis for native approaches to neural multimodal content analysis (i.e., computer vision, linguistics, and audio) with symbolic event processing rules to support real-time queries over multimodal data streams using the multimodal event processing language to express single, primitive multimodal, and complex multimodal event patterns. The content of multimodal streams is represented using multimodal event knowledge graphs to capture the semantic, spatial, and temporal content of the multimodal streams. The approach is implemented and evaluated within a MEP engine using single and multimodal queries achieving near real-time performance with a throughput of ∼30 frames processed per second (fps) and subsecond latency of 0.075–0.30 s for video streams of 30 fps input rate. Support for high input stream rates (45 fps) is achieved through content-aware load-shedding techniques with a ∼127X latency improvement resulting in only a minor decrease in accuracy.**

***Index Terms*—Data management and analytics, event processing, Internet of Multimedia Things (IoMT), service middleware and platform.**

I. Introduction

With the rise of the Internet of Multimedia Things (IoMT) and smart environments, there has been a significant shift in the nature of data streams. Visual sensors, such as smartphones and CCTV cameras, are now pervasive and generate high volumes of unstructured data streams. The major chunk of these unstructured data are images, audio, and videos, termed multimedia data. There is a clear need to enable smart environments (including applications and objects within then) with the power of observing, sensing, and understanding the world through processing multimodal data efficiently and effectively. However, existing event processing methods have limited consideration for the challenges of multimedia processing [1].

Manuscript received 27 July 2021; revised 7 December 2021; accepted 7 January 2022. Date of publication 14 January 2022; date of current version 25 July 2022. This work was supported by the Science Foundation Ireland, co-funded by the European Regional Development Fund under Grant SFI/12/RC/2289_P2. *(Corresponding author: Edward Curry.)*

The authors are with the Insight SFI Research Centre for Data Analytics, Data Science Institute, National University of Ireland Galway, Galway, H91AEX4 Ireland (e-mail: edward.curry@nuigalway.ie).

Digital Object Identifier 10.1109/JIOT.2022.3143171

This article explores the use of the event processing paradigm for real-time IoMT data. It introduces the multimodal event processing (MEP) paradigm to meet the critical challenges for processing multimodal event streams. MEP provides a formal basis for native approaches to multimodal content analysis (i.e., computer vision, linguistics, and audio) within the event processing paradigm to support real-time queries over multimodal data streams.

MEP enables the user to process multimodal streams under the event-processing paradigm and simplifies the querying of multimodal event streams. Within the MEP paradigm, multimodal streams are processed to generate a knowledge graph representation, which can then be queried using the MEP language (MEPL). MEPL supports user-defined event operators, which can be developed using a neurosymbolic-based hybrid approach with statistical deep neural network (DNN) models and symbolically based spatial and temporal reasoning. MEP engines can then be implemented using hybrid processing, combining neural (DNNs) and symbolic (rules) models. The contributions of this article are as follows.

1) It motivates the need for MEP and the challenges that need to be overcome.
2) Introduces the MEP paradigm and establishes its foundations in event model, event representation, query language, multimodal event, and matching models.
3) Introduces the MEPL, an SQL-like declarative language where different operators identify patterns over multimodal streams.
4) Introduces the multimodal event knowledge graph (MEKG) to provide a knowledge representation for the multimodal event stream's semantic, spatial, and temporal content.
5) Details the first MEP engine (GNOSIS), providing details on its microservices-based architecture and implementation of the MEP concepts, including event model, query language, matching, and native content extraction.
6) Demonstrates the use of MEP to build real-time applications for IoMT-enabled smart environments in occupational health and safety (OHS).
7) A quantitative performance evaluation of the MEP engine in terms of latency, queries, single modal, and multimodal optimizations.

The work builds the previous single modal image-only event processing [14] to detect patterns over multimodal data streams, such as video and audio. The structure of this article is as follows. Section II details the motivation for new

TABLE I
COMPARISON OF IOT AND IOMT-BASED SYSTEMS (ADAPTED FROM [1])

Characteristics	IoT	IoMT
Resources	Low cost, size, and energy consumption	High cost, size, and energy consumption
Deployment	RFID tags, sensors	Video and audio sensors
Heterogeneity	Limited heterogeneity (Scalar data)	Heterogeneous (Unstructured data)
Modalities	Single (Structured)	Multiple (structured, audio, image, video, text)
Communication	WSN-based protocols (ZigBee, WLAN, Bluetooth, WIFI, UMTS etc.)	WMSN-based protocols (RMST, PSFQ, ESR, CODA, MRTP) (Non-standardized)
Quality of Service (QoS)	Low delay, packet loss, jitter, energy, and bandwidth	High delay, packet loss, jitter, accuracy, energy, and bandwidth
Storage, Searching and Processing	Data mining and analytics	Data-mining, feature extraction, and cloud-based multimedia storage
Query	Event processing languages	No available specialised language
Computational costs	Low	High (DNN + GPU)
Service Composition	SOA-based and event-based middleware	No available specialised middleware

forms of processing multimodal data within intelligent environments and identifies the key challenges. The coverage of these challenges within existing related works is then analyzed in Section III. Section IV introduces the MEP paradigm that includes the multimodal event model, the MEPL, and the multimodal event matching model. Section V details the fundamentals of the GNOSIS MEP engine. Section VI describes how MEP can create an application for smart environments, with Section VII presenting a quantitative evaluation of the MEP engine. Finally, the article concludes and Section VIII discusses future work.

II. FROM IOT TO IOMT

The IoMT is recently coined to represent multimedia communications using the Internet of Things (IoT). IoMT (sometimes referred to as MIoT) can be defined as an IoT-based paradigm that allows objects to connect and exchange structured and unstructured data to facilitate multimedia-based services and applications [2]. The result is smart environments producing large quantities of multimodal data streams (including structured, video, and audio) containing rich content on the environment. Furthermore, analytics can be performed over these unstructured multimedia streams to detect events of interest, which can power different applications, such as business intelligence, environmental surveillance, traffic monitoring [3], and optimization of agricultural practices [4].

There are significant differences in how we process structured and unstructured data. As detailed in Table I, IoMT introduces additional challenges over IoT in areas such as resourcing, networks, Quality of Service (QoS), and querying [2], [5], [6]. Furthermore, at the content-level, IoMT adds new challenges to event processing.

A. IoMT Challenges in Event Processing

1) Multimodal Events Representation: IoT data use a data model with a predefined schema from various sensors, including temperature, light, humidity, etc. Therefore, IoT data are structured with limited heterogeneity compared to unstructured multimodal (images, audio, and video) data. On the other hand, the fundamental difference with IoMT data is that it is unstructured without a predefined schema or data model. Therefore, to process unstructured IoMT data, there is a need to analyze the content to extract meaning (structure) to represent the content in the data.

2) Concept Space and Interpretability: Within IoT, the concept space is the predefined schema that can range from only a few classes to thousands of classes representing concepts [7], [8]. Thus, the structure of the data lends itself to a more straightforward interpretation. However, within IoMT, the number and size of the concept space can be significantly larger, with a greater range of human perception and interpretability of rich-content across different modalities, such as images, audio, and video (i.e., visual gnome has 75K unique image objects, and the open images data set has almost 20K classes. In addition, the variation in styles, textures, and colors of visual objects results in a large number of variations of a single concept that needs to be identified. This leads to a larger content space that must be supported within the query and processing of multimodal IoMT streams. Challenges include the semantic gap, interpretability, accuracy, and trust [1].

3) Data Volume and Computational Cost: IoMT streams are both network and computationally intensive. This leads to multiple challenges and hinders sustainable applications [9]–[11]. Stream producers within IoMT are often located at the edge of the network. Edge-based deployment using resource-constrained devices has limits on bandwidth and computation. The size of IoMT data can be significantly more than IoT data; MB versus KBs and video streaming can quickly saturate the available bandwidth within a smart environment [12]. The cost of content extraction from multimodal data is significantly more expensive than structured data. Content extraction methods for IoMT, such as deep learning methods, are computationally expensive and often require GPU for high performance.

4) Complex Processing Pipelines: Processing multimodal streams requires a complex pipeline topology that must be dynamic and adapt to changing query needs (see Fig. 3). This makes the cost of deploying and maintaining the pipelines very high when new events and concepts need to be detected.

For IoMT, the pipelines need to be suitable for real-time processing data [13] regarding scalability, latency, volume, bandwidth, and computation costs. In addition, other QoS dimensions, such as energy usage, are also critical when you consider the computational costs and the high energy consumption of GPUs.

We will now illustrate the challenges with processing events for IoMT with a motivational example.

B. Motivational Multimodal Event Processing Example

Consider an OHS scenario (see Fig. 1) where the safety of workers is of the utmost importance at construction and

Fig. 1. Occupational health and safety scenario for IoMT.

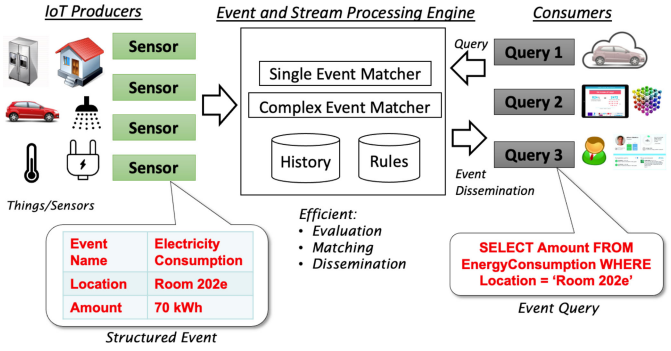

Fig. 2. Traditional approach for processing structured IoT events.

manufacturing sites. The health and safety regulatory authorities have issued safety guidelines regarding PPE to prevent mishaps, construction hazards, and accidents. Monitoring the safety of a workplace requires several conditions to be constantly checked, which can be monitored using closed-circuit cameras to ensure safety and compliance with the guidelines, for example.

1) Is everyone on the construction site wearing PPE?
2) Is the site at a safe temperature?
3) Is smoke detected?
4) Is the wind speed safe on the site?
5) Is there any unsafe behavior?
6) Is the site noise compliant?

As an OHS supervisor, performing manual inspection for safety compliances from each video camera is time consuming, tedious, and error prone (Fig. 1). Similarly, there can be other scenarios like counting the number of workers and visitors for a specific day. Custom solutions with specialized ML models are currently used, leading to multiple challenges in unstructured video representation, event querying, processing pipelines, and computation in a distributed deployment.

III. Current Approaches

This section provides a brief overview of event and stream processing techniques, and visual analytics platforms. The current state-of-the-art approach for event processing offers limited or no support for unstructured event types. On the other hand, computer vision libraries provide minimal or no support for large-scale processing of distributed streams and dissemination mechanisms.

A. Stream Processing and Continuous Queries

Quickly reacting to an event is a critical requirement within many real-world situations. Stream processing and tactile sensing systems [15] can provide reliable data processing capabilities suitable for real-time information. Stonebraker *et al.* [13] suggested eight requirements for an effective and efficient design for such systems. Many distributed applications have motivated the event processing paradigm that requires on-the-fly and low-latency processing of information items (see Fig. 2) [16]. Event processing has evolved from the works of several communities [16], including active databases, reactive middleware, event-based software engineering, event-based systems, and message-oriented middleware. Event processing is an essential technique in developing "smart" applications that target the analysis of events captured in real time (i.e., detecting patterns) where the processing of streams must be done "in-stream" without persisting the events [13]. Event processing platforms are a form of middleware that abstract the application developers from the underlying technologies providing event processing languages for users to express event patterns for their detection at runtime. The limitations of event and stream processing systems for IoMT include the following.

1) *Event Query Language:* Current event processing systems use SQL-like declarative languages [16] to express patterns over structured events. There is limited support to define queries over unstructured event types. Writing declarative queries for human-level concepts using low-level features (e.g., pixel values) is challenging. There is a need to support human-level multimodal queries that can express semantic, spatial, and temporal event patterns over multimodal events.
2) *Event Representation:* Current event processing engines assume events are produced with a well-defined fixed data model using an event model, key-value pairs, XML, or RDF triples [17]. They have little to no support for representing the features (i.e., low-level semantic features from computer vision) from multimodal data. Thus, there is a need to express low-level features into a high-level representation that the event processing engine can process to detect multimodal event patterns.
3) *Content Extraction:* Current event processing systems have no native support to analyze the unstructured multimodal event streams. Multimodal analysis (i.e., image, audio, and video) is computationally intensive, requiring machine learning methods for content extraction, including DNNs. Furthermore, extracting an accurate representation of the content from multimodal streams may require the content to be processed by multiple content extraction models within a pipeline (see Fig. 3).
4) *Event Matching:* Different event models, such as automata, column [18], semantic [19], declarative [20], and cognitive [21], have been proposed in the literature. Existing event processing systems use temporal pattern matching within single modality streams [16]. However, multimodal

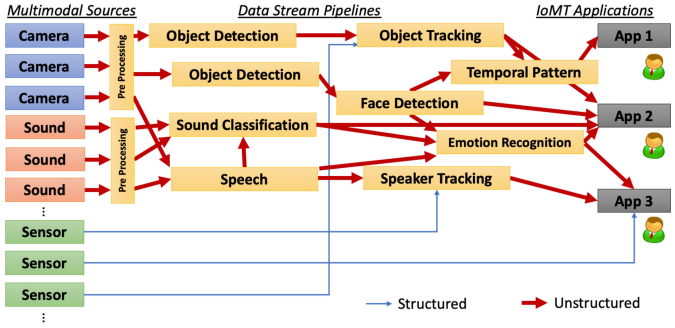

Fig. 3. Traditional processing flow for IoMT applications.

streams have complex and dynamic characteristics, which require sophisticated event matching across multiple streams and modalities. Therefore, pattern detection is needed across different modalities with semantic, spatial, and temporal constraints.

5) *Neurosymbolic Event Rules:* Current event processing systems focus on rule-based approaches. Event rules are crafted to perform complex temporal and spatial reasoning over structured streams. As discussed above, unstructured multimodal event detection warrants novel event reasoning. Neurosymbolic approaches combine logical symbolic reasoning with inductive statistical methods to create high-level knowledge representation and formalism [22]–[24]. Therefore, a novel hybrid reasoning approach is required where neural-based inductive techniques can extract low-level content from unstructured multimodal streams following logic-based deductive approaches to detect complex event patterns.

B. Visual Analytics

Video streams are continuous sequences of images. The image understanding domain focuses on reasoning over image content and describes the image using high-level human-understandable concepts. In computer vision, these high-level visual concepts are termed objects. Objects are the basic building block of images, which are a collection of low-level features (pixels, intensity, color, and edges) and have been given a high-level semantic label, such as a car and bike.

Modern systems for querying images and video content (i.e., AWStream [25]) rely on extraction techniques based on deep convolutional neural networks (CNNs) due to their accuracy in common computer vision tasks, such as classification and object detection. Visual data are collected and persisted in stable storage in such systems, and then analyzed either by domain-specific learning models or accelerated by lightweight filters following a store and query processing model (i.e., BlazeIT [26], DeepLens [27], NoScope [28], Optasia [29], Sprocket [30], Tahoma [31], VideoChef [32], and VStore [33]). The most recent and competitive object detection models (Faster R-CNN [34], SSD, YOLOv3 [35], and RetinaNet) have proven to be suitable for image recognition in achieving high-performance results. Several works

TABLE II
COVERAGE OF MULTIMODAL EVENT PROCESSING REQUIREMENTS

Name	Multimodal Event Query Language	Event Representation	Content Extraction	Matching/ Filtering	Multimodal Streams
AWStream [25]	Video only	No	Yes	No	No
BlazeIT [26]	Video only	Yes	Yes	Yes	No
CloudSeg [41]	No	No	Yes	No	No
DeepLens [27]	No	Frame-level	Yes	No	No
MELINDA [42]	No	Yes	Yes	Topics	Yes
NoScope [28]	No	No	Yes	No	No
Optasia [29]	Video only	Yes	Yes	No	No
Sprocket [30]	No	No	Yes	No	No
Tahoma [31]	No	No	No	No	No
V-PRISM [43]	No	No	Yes	No	Yes
VidCEP [44]	Video only	Yes	Yes	Yes	No
VideoStorm [45]	No	No	Yes	No	No
VideoChef [32]	No	No	Yes	No	No
VStore [33]	No	Video-only	Yes	No	No
YOLO [35]	No	No	Yes	No	No
Faster R-CNN [34]	No	No	Yes	No	No
Bazhenov [38]	No	No	Yes	Yes	No
Reducto [39]	Video only	No	Yes	Yes	No
VID-WIN [40]	Video only	Video only	Yes	Yes	No
GNOSIS MEP	Yes	Yes	Yes	Yes	Yes

have focused on accelerating the extraction process by designing probabilistic and specialized models, like the conventional predicate pushdown optimization [36]. The intuition behind this is that queried semantic concepts often occur rarely in large amounts of visual content being processed by expensive deep CNNs. Thus, applying lightweight filters allows filtering out many irrelevant frames, minimizing the processing cost, and speeding up the query answering process. Edge-centric event analytics is another research area where data processing happens locally near the source [37]. Processing video data at IoT edge nodes can improve throughput, latency with reduced bandwidth consumption. Bazhenov and Korzun [38] proposed a camera-based edge-centric solution for personal monitoring in a smart manufacturing process. However, the work was limited to video data and lacked discussion on multimodal events. Edge-based solutions have their challenges in terms of limited CPU, memory, and low accuracy event detection [39], [40].

Table II details the limitation of current approaches to processing events within IoMT streams. It is clear from the analysis that existing approaches lack support for specialized middleware [1] to provide services required by multimodal IoMT applications. In many cases, the queries are complex and require preprocessing of the multiple heterogeneous streams, use multiple classifiers and feature extractors, be executed in a parallel manner, and produce synchronized results.

IV. MULTIMODAL-EVENT PROCESSING

The core contribution of this article is proposing the MEP paradigm to meet the critical challenges for processing

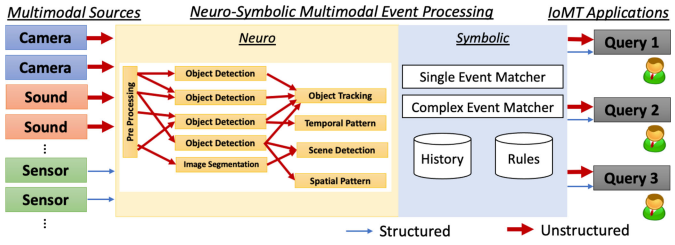

Fig. 4. High-level neurosymbolic MEP for IoMT.

multimodal event streams. MEP provides a formal basis for native approaches to multimodal content analysis (i.e., computer vision, linguistics, and audio) within the event processing paradigm to support real-time queries over multimodal data streams.

MEP enables the user to process multimodal streams under the event-processing paradigm and simplifies the querying of multimodal event streams. Within the MEP paradigm, multimodal streams are processed to generate a knowledge graph representation, which can then be queried using the MEPL. MEPL supports user-defined event operators, which can be developed using a neurosymbolic-based hybrid approach such as statistical DNN models and symbolically based spatial and temporal reasoning. MEP engines can then be implemented using hybrid processing, combining neural (DNNs) and symbolic (rules) models with the deployment and optimization of multiple pipelines as illustrated in Fig. 4.

The remainder of this section details the fundamentals of the MEP approach, including the event model, neural-symbolic event recognition, event representation, query language, and multimodal event matching models.

A. Multimodal Event Model Definition

Oxford British and World English online dictionary define an event as: "A thing that happens or takes place, especially one of importance." The definition of an event within the event processing community is provided by the event processing technical society (EPTS) glossary [46] "an object that represents, encodes, or records an event, generally for the purpose of computer processing." In MEP, we have extended the event processing concept of the event to define the multimodal event. We define a multimodal event as follows.

A multimodal event represents a state or change of state in several modalities that contain part of the description of the same event of interest.

MEP defines three categories of single modal events, primitive multimodal events, and complex multimodal events.

1) *Single Modal Event:* An atomic occurrence of an event in a single modality.
2) *Primitive Multimodal Event:* An event that occurs in several modalities that contain part of the description of the same things of interest and does not summarize or represent a set of other events.
3) *Complex Multimodal Event:* A complex multimodal event summarizes, represents, or denotes a set of single modal and/or primitive multimodal events.

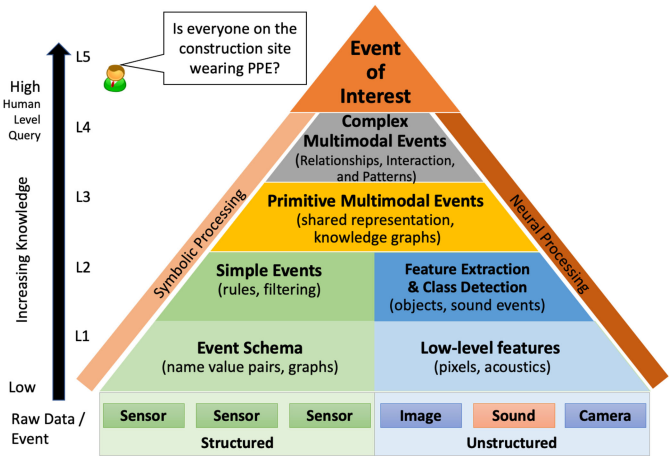

Fig. 5. Multimodal event hierarchy.

B. Neural-Symbolic Event Hierarchy

MEP detects events following the multimodal event hierarchy, as detailed in Fig. 5. The hierarchy is structured as follows.

1) *L1-Raw Data:* The foundation of the hierarchy contains the raw streams in their native formats; this includes both structured and unstructured streams.
2) *L2-Single Modal Events:* Individual streams are analyzed for the occurrence of single modal events. The content of unstructured streams is analyzed to extract any necessary features/classes needed for single modal events.
3) *L3-Primitive Multimodal Events:* A multimodal knowledge graph establishes a shared representation between the streams. The shared representation is used to detect primitive multimodal events across multiple streams.
4) *L4-Complex Multimodal Events:* Patterns and relationships between single modal and primitive multimodal events are analyzed to identify complex multimodal events.
5) *L5-Events of Interest:* Users express their queries within the MEPL. A notification message is generated and forwarded when the event stream matches a MEPL statement.

C. Multimodal Event Knowledge Graph Representation

Knowledge graphs (KG) provide a flexible knowledge representation structure that can describe entities and concepts from multiple systems and domains and at varying levels of granularity. As defined by Paulheim [47], a "knowledge graph: 1) mainly describes real-world entities and their interrelations, organized in a graph; 2) defines possible classes and relations of entities in a schema; 3) allows for potentially interrelating arbitrary entities with each other; and 4) covers various topical domains."

A knowledge graph is a set of entities (e.g., Person and Vest), a set of relations between those entities (e.g., "ownerOf" and "wearing"), and a set of facts. Facts combine the entities and relationships "Marie Curie, wasResidentOf, France." More formally, a knowledge graph is a tuple (E, R, G), where:

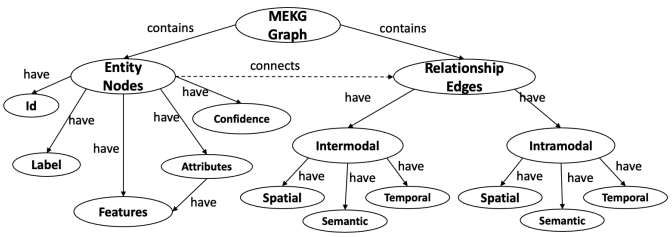

Fig. 6. Multimodal event knowledge graph schema.

1) E is a set of nodes, each representing an *entity* in the domain;
2) R is a set of edge labels, each representing a *predicate* or a semantic relation type;
3) G $\subseteq$ E $\times$ R $\times$ E is a set of $\langle$subject, predicate, object$\rangle$ *triples*, denoting *facts*.

A MEP engine converts the unstructured multimodal data into a formal structured format using the MEKG schema (see Fig. 6). MEKG models the incoming multimodal streams as a continuous evolving graph stream. The content of each stream is modeled using a schema with entities, objects, concepts, sounds, etc., represented as nodes. In a stream's MEKG graph, edges capture the interstream semantic, spatial, and temporal relationships. MEKG graphs from different streams (and modalities) can be merged to provide a shared representation across the different stream modalities. In the shared representation, edges capture intermodal relationships, including "sameAs" equivalence and spatial and temporal relationships.

Definition (MEKG Graph): For any multimodal stream event, the resulting MEKG is a labeled graph represented as MEKG = (**E, R, Ep, G**) where.

1) **E** is a set of nodes $\boldsymbol{O_i}$, each representing an *entity* in the domain;
2) **R** is a set of edge labels representing spatial, semantic, and temporal relation types;
3) **EP** is a set of properties mapped to each entity node such that $\boldsymbol{O_i}$ = (id, attributes, label, confidence, features);
4) $\mathbf{G} \subseteq \mathbf{E}_p \times \mathbf{R} \times \mathbf{E_p}$.

MEKG provides a top-level schema for shared representation across modalities. As MEKG is a knowledge graph, it can support schema that are designed to capture the specific features appropriate for the format of that modality. The video event knowledge graph (VEKG) [48] and the audio event knowledge graph (AEKG) are two such specializations for video and audio data, respectively. Fig. 7 shows how an MEKG can represent the content of a video and audio stream at different time instances. The content of the video stream is represented using VEKG, while the content of the audio stream is represented using the AEKG. These single modal graphs are then merged to create the MEKG to represent both modalities together. The entity nodes in MEKG graphs are connected using spatial, semantic, and temporal edges. The semantic relation edge between entity nodes is created by identifying the same entity nodes across modalities (i.e., entity linking), while the temporal relation edge between entity nodes is created by identifying the same entity nodes at different times using appropriate techniques (i.e., object tracking). Fig. 7 shows the MEKG construction example over a video and audio stream at four-time points, with the construction process showing the application of intramodal (t2), intermodal (t3), and temporal relations over time (t1–t4). The resulting MEKG can be used to get insights into the different facets (modalities) of an event.

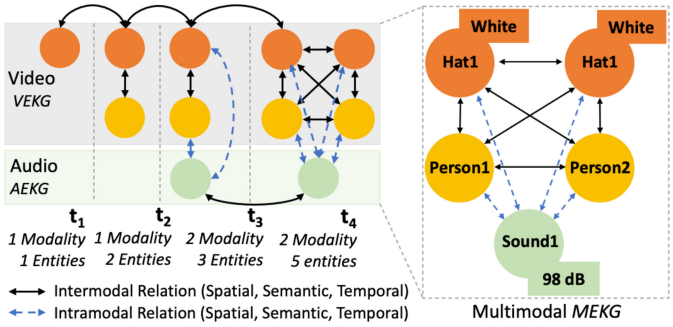

Fig. 7. Example of MEKG representing the content of a video and audio stream at different time instances.

```
REGISTER QUERY [query name]
OUTPUT [ANN_IMAGE_BBOX | ANN_IMAGE_QUERY_OUTPUT
| K_GRAPH_IMAGE | K_GRAPH_DICT]
CONTENT Service(s)Name
MATCH [openCypher Match query]
WHERE [predicates on Match clause]
FROM [publisher Id]
WITHIN [TUMBLING_COUNT_WINDOW|TUMBLING_TIME_WINDOW
| SLIDING_COUNT_WINDOW | SLIDING_TIME_WINDOW]
RETURN [reference id list, aggregation operators]
```

Fig. 8. Multimodal event processing language statement.

D. Multimodal Event Processing Language

MEP provides an event processing language for querying event detection over multimodal streams. The user can subscribe to different queries using the MEPL to fetch multimodal patterns. The MEPL enables users to query multimodal events in SQL-like declarative syntax that use graph matching clauses based on openCypher. In addition, MEPL provides various operator suites enabling a wide range of multimodal event detection capabilities, such as identifying objects, attributes, and complex spatiotemporal relationships.

Fig. 8 shows the MEPL syntax with the clauses explained in Table III. In addition, examples of queries are presented in Section VI for an OHS use case.

E. Multimodal Event Detection

An MEP engine converts the incoming stream as a structured graph stream using content analysis pipelines and treats multimodal event detection as a graph matching problem. MEP can perform both stateless and stateful event processing. Fig. 9 shows the high-level two-step process used within an MEP engine for MEKG: 1) construction and 2) matching.

1) Event Graph Construction: The construction algorithm creates a knowledge graph for each unstructured data stream by extracting entities, attributes, and relations using content analysis. Every single modal MEKG (or the specializations such as VEKG or AEKG) can then be merged into a single graph to provide a share representation across the streams.

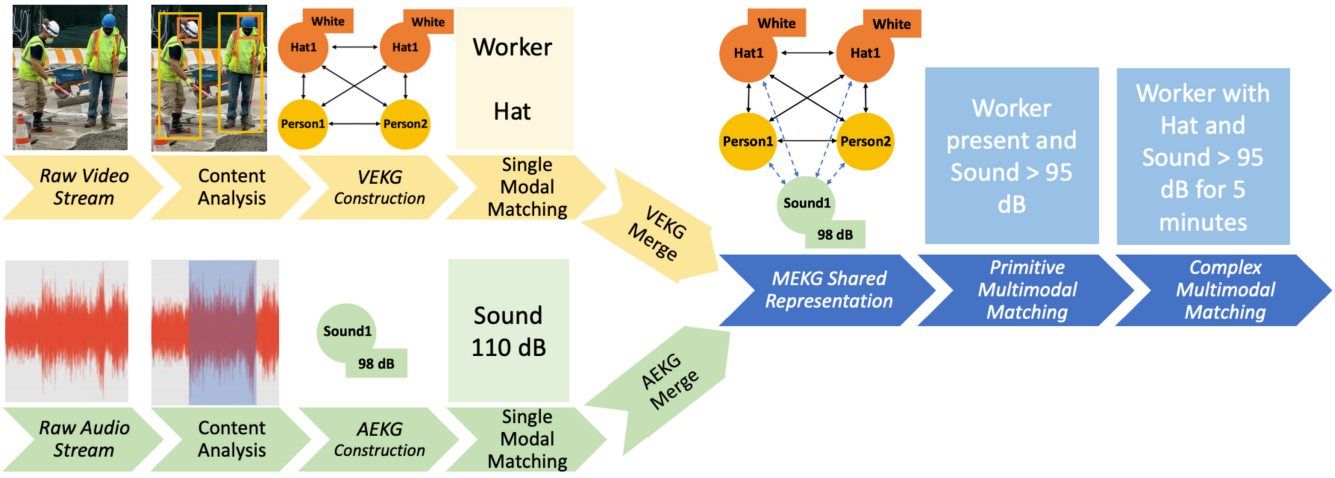

Fig. 9. Multimodal knowledge graph construction and event matching.

TABLE III
QUERY CLAUSES DEFINITION

Query Statements	Description
REGISTER QUERY SomeQuery	query name, unique for each sub
OUTPUT K_GRAPH_JSON	any of (ANN_IMAGE_BBOX | ANN_IMAGE_QUERY_OUTPUT | K_GRAPH_IMAGE | K_GRAPH_JSON)
CONTENT ObjectDetection, ColorDetection, ObjectDistanceDetection	Optional statement. List of Services, separated by comma
MATCH (p:person)-[r:close_to {distance: 20}]-(b:bus)	Valid openCypher query Match statement.
OPTIONAL MATCH (c)-[:on_top_of]->(p:person)	Valid openCypher optional MATCH statement.
WHERE c:car AND c.color = "blue"	Valid openCypher WHERE statement.
FROM web_cam1, webcam2	publisher id separated by comma or "*" for all
WITHIN TUMBLING_COUNT_WINDOW(10)	Any one of the available functions (with their respective arguments): (TUMBLING_COUNT_WINDOW|TUMBLING_TIME_WINDOW | SLIDING_COUNT_WINDOW | SLIDING_TIME_WINDOW)
WITH_QOS accuracy = "high", latency = "low"	List of QoS policies to be used by the query, separated by comma
RETURN p, r.distance, COUNT(b) as closeBuses, COUNT(x) as blueCars	A valid openCypher return statement

Algorithm 1 Multimodal Event Matching

Input: **MultiModalStreamSet** $(MS_{set}) = \{V_1, A_1, \ldots, V_i, A_n\}$
where $V_i \in$ *video stream*, $A_n \in$ *audio stream*
Query Q
where $Q \in (V_1,\ A_1)$
Output: MultiModalEvent (*ME*)

CreateMultiModalEventKnowledgeGraph(Q, V_1, A_1)
$win_{audio} \longleftarrow initiatewindow(Q, A_1)$
$win_{video} \longleftarrow initiatewindow(Q, V_1)$
while V_1, A_1 *not null* **do**
  $audiosignal_i \longleftarrow getaudio(A_1)$
  $frame_i \longleftarrow getvideoframe(V_1)$
  $(object_{list},\ attribute_{list},\ bbox_{list}) \longleftarrow DNNCascade(frame_i)$
  $VEKG_i \longleftarrow createVEKG(object_{list}, attribute_{list}, bbox_{list})$
  if $(win_{audio}, win_{video}).size < trigger - time$ **then**
    $win_{video} \longleftarrow addtowindow(VEKG_i)$
    $win_{video} \longleftarrow updateVEKG(Q)$
    $win_{audio} \longleftarrow addtowindow(audiosignal_i)$
    continue
  if $(win_{audio}, win_{video}).size >= trigger - time$ **then**
    $(audiot_{label},\ audio_{decibel}) \longleftarrow classifyaudio(win_{audio})$
    $AEKG \longleftarrow createAEKG(audiot_{label},\ audio_{decibel})$
    $MEKG_i \longleftarrow createMEKG(win_{video}(VEKG), AEKG)$
    $sendtoMultimodalEventMatcher(MEKG_i)$
    $resetwindow(win_{video}, win_{audio})$
    Continue

MultiModalEventMatching(Q, $MEKG_i$)
$MEKG_i \longleftarrow getMEKG()$
$ME \longleftarrow MultimodalCypherMATCH\ (Q,\ MEKG_i)$
$notify(ME)$

A key challenge in combining the graphs is linking entities and creating relationships between the different modalities. Entities can be linked using "sameAs" relationships, and semantic, spatial, and temporal links can be established using suitable reasoning techniques. The matching models then reason over the constructed graph to identify patterns

2) Event Matching Models: MEP can handle complex and straightforward event patterns across multiple modalities in semantic, spatial, and temporal dimensions. MEP constitutes matching models and window operators (such as sliding and tumbling time windows) to handle stateless and stateful multimodal event analytics (See Algorithm 1). The following matching models are supported.

1) *Single Modal Event Matching:* Stateless event matching on a single modality primarily focusing on detecting entities and attributes.
2) *Primitive Multimodal Event Matching:* Stateless event matching over multiple modalities primarily focusing on detecting entities and attributes.

TABLE IV
OUTPUT CLAUSES FOR MEPL STATEMENTS

Notification Format	Description
ANN_IMAGE_BBOX	Displays output as a stream of images encoded with Object and Relationships information. The result is an image annotated with the object's class label and bounding box (s).
ANN_IMAGE_QUERY_OUTPUT	Displays output as a stream of images encoded with Object and Relationships information. The result is an image annotated with the Return Clause output (e.g., Visitor Count: 2, this information embedded on the image).
K_GRAPH_IMAGE	Displays output as a Knowledge Graph image generated from Objects and Relationships running over the length of a source video.
K_GRAPH_JSON	Displays output as a Knowledge Graph JSON generated from Objects and Relationships running over the length of a source video.

3) *Complex Multimodal Event Matching:* In the complex event matching engine, data streams are considered an unbounded timestamped data sequence [44]. Windows are used to capture the stream state by taking the input streams and producing a substream of finite length, which can be persisted in storage if needed. Next, the complex matcher analyzes the window to identify the queries' specified temporal and spatial patterns.

F. Notifications

When an event stream matches a MEPL statement, a notification message is generated and forwarded to the subscriber. Many event processing engines use matching event messages as notifications. However, MEPL supports both the definition of the format and the results contained in the notification. This simplifies the notifications delivered to subscribers and improves efficiency by eliminating unnecessary information. The exact format of the notification can be defined in the MEPL statement using the *OUTPUT* clause (see Table IV). The results of the statement can also be expressed using the *RETURN* clause. The result may include a subset of attributes in matching graphs indicated in the *MATCH* clause and aggregations or operators defined in the *RETURN* clause.

Algorithm 1 explains the multimodal event matching process. The inputs are MultiModalStreamSet (constitute video and audio data streams) and query Q. The output is a multimodal event *ME*, which is detected based on query Q. The process of creating MEKG is described in CreateMultiModalEventKnowledgeGraph function, which takes query, video, and audio stream as input. Based on the query input stream, the audio (win_{audio}) and video (win_{video}) windows are initiated. Next it reads the audio signals and video frames ($frame_i$) using an encoder. Then, it sends the video frames ($frame_i$) to DNN models cascade to extract metadata, such as objects ($object_{list}$), attributes ($attribute_{list}$), and bounding box ($bbox_{list}$) information from the frames. Using extracted metadata information, the VEKG graph is created that is then added to the video window (win_{video}), and its edge relations are updated using Q. The audio signal is directly added to the audio window as audio classification over a given time slice. Both audio and video windows (possibly with multiple instances in a multiquery scenario) continuously collect the VEKG and audio signals and trigger when the duration is completed. win_{audio} will send the collected audio signals to an audio classifier to create AEKG. A combined multimodal MEKG graph is constructed using AEKG and VEKG and sent to for event matching.

V. GNOSIS: A MULTIMODAL EVENT PROCESSING ENGINE

GNOSIS is a MEP engine (gnosis-mep.org) using that processing IoMT multimodal streams using hybrid processing, which combines neural (DNNs) and symbolic (rules) models within the event-based paradigm. GNOSIS simplifies the querying of multimodal event streams and the deployment and optimization of multiple pipelines. GNOSIS provides native processing support of multimodal event/streams, queried using the MEPL. Once GNOSIS receives a multimodal stream, it is processed to generate a knowledge graph representation using query-aware edge-cloud processing pipelines. GNOSIS follows the following design principles.

1) *Microservice Architecture:* Open, incremental deploy ability, synchronization, parallelism, scalability.
2) *Query-Driven:* In GNOSIS, users can register continuous multimodal queries for a spatiotemporal event matching over multimodal streams.
3) *Self-Adaptive and Autonomic:* In ever-changing deployments, it is vital to promptly adapt to changes in the environment, workload, and the system's users. This can be done with the use of self-adaptive (autonomic) systems.
4) *Neurosymbolic Approaches:* MEPL supports user-defined event operators, which can be developed using neurosymbolic-based hybrid approach, such as statistical DNN models and symbolically based spatial and temporal reasoning.

A. GNOSIS Architecture

GNOSIS is implemented as a distributed microservice-based event processing system and communicate via configurable messaging services. Fig. 10 shows the high-level GNOSIS architecture, which is divided into five major components: 1) query manager; 2) steam manager; 3) content extractor; 4) matching engine; and 5) adaptation engine. GNOSIS follows a serverless approach where these components act as independent microservices and can be deployed over the cloud or local computing nodes. The microservices pipeline is broken down into four distinct stages: 1) querying; 2) preprocessing; 3) content extraction; and 4) matching.

The GNOSIS service components are explained below in detail.

1) Query Manager: The query manager constitutes MEPL Query Engine, where all EPL queries from different subscribers are stored and indexed. It extracts the EPL query predicates and stores them as a configuration profile. The query engine sends the configuration values to the *Stream Manager* (model pipeline information and publisher information) and *Matching Engine* (event rules, windows information, and query

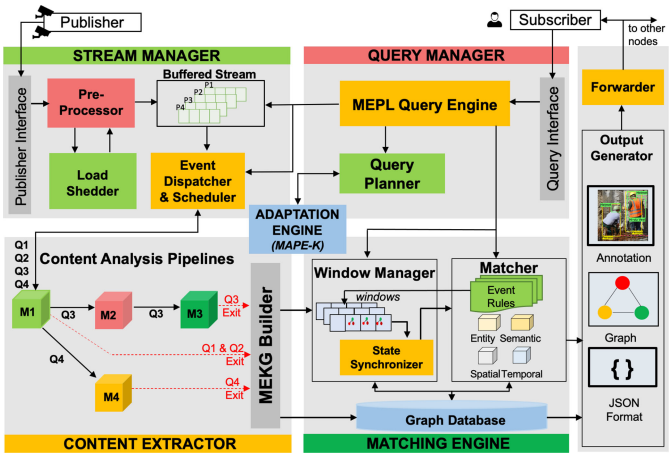

Fig. 10. Overview of GNOSIS system architecture.

graph) components to initiate the instances for handling different queries. As shown in Fig. 10, the *Query Manager* component stores, parses, and instantiates other GNOSIS components (like windows, matcher, and event dispatcher) as per query metrics.

2) Stream Manager: The stream manager preprocesses the incoming multimodal streams using libraries for encodings, such as FFmpeg and OpenCV3, for preprocessing video and audio streams. It then forwards them to the *Content Extractor*. Next, the *Event Dispatcher and Scheduler* component creates a control flow of a model pipeline using the CONTENT clause of the GNOSIS EPL. It then sends the received streams to the content analysis pipelines.

3) Content Extractor: GNOSIS generates flexible DNN and ML content analysis pipelines (or models cascade) at runtime, pretrained on specific data sets. Content Extractor microservices can be implemented using various multimodal techniques (see Table V). The *Content Extractor* component feeds the pipeline information as a directed acyclic graph (DAG). ML models act as a node, and the edges refer to the input and output flow of data from one node to another node. Fig. 10 shows a DAG for a multimodal content analysis pipeline for the four MEPL queries described in Section VI. Queries 1, 2, and 3 use a pipeline that consists of deep learning-based microservices M1(PPE), M2(Object/Person), and M3(Color), while Query 4 uses a pipeline of M1(PPE) and M4(Audio). Depending on the number of streams that needs to be processed, separate content extraction pipeline instances are created dynamically to process each stream in parallel. The extracted metadata from the content analysis pipeline are then passed through *MEKG Builder* to create timestamped MEKG. The MEKG model is detailed in Section IV. The MEKG data are stored in a graph database, and the references are sent to the *Matching Engine* service for event pattern matching based on the MEPL query.

4) Matching Engine: The matching engine is the core of the GNOSIS framework that handles state management and multimodal event matching process. The *Windows Manager* assigns windows to different multimodal streams as per the information received via MEPL query engine. The window captures unstructured streams as MEKG graphs into a fixed bucket size, i.e., state. Inside the windows, the MEKG graph relations are updated using *Event Rules,* and the graph database is updated with the new relations. The completed windows states are sent to *the State Synchronizer,* which waits for all the incoming states from different streams. On receiving all the stream states for a given window instance, the state synchronizer then sends the captured states to the *Event Matcher* for further processing.

The event matcher executes the query matching over the graph database against the registered EPL queries. The matcher performs the entity, semantic, spatial, and temporal matching for the queries received by the query manager, and then outputs the matches to the forwarder to send notifications to the subscriber. Based on the OUTPUT EPL clause, the results are fed back to the *Output Generator* event pipeline to visualize the results in formats, such as JSON, graphs, and

TABLE V
CURRENT CONTENT ANALYSIS SERVICES

Content Service	Model Type	Content	Description
ObjectDetection	TF / SSDImageNet	The common object from MS COCO	The model can detect 80 classes of Microsoft COCO
PPEDetection	TF / YOLOv3	wearing_hat, not_wearing_hat	The model can detect whether a person is wearing a hard hat.
ColorDetection	OpenCV	Red, yellow, white	The model can detect the presence of different colours in a given bounding box.
Face Detection	PyTorch/RetinaNet	Detect faces	Model detect faces as bounding boxes.
Audio Detection	Keras/ Tensorflow	Detect jackhammer and drilling sound	Models detect classes with decibels
DeepBee	Keras/ Tensorflow	Capped, Larva, Egg, Honey, Nectar, Pollen, Other	Model is used to analyse honeycombs in beehives
WheatRust	PyTorch	Healthy, lear rust, stem rust	Detects wheat rust
Plant Disease Identification	PyTorch/ Densenet/Resnet	39 different diseases that affect plants	Identifies plant diseases
Weed Identification	PyTorch/ResNet	Black-grass, Charlock, Cleavers, Common Chickweed, etc	Identification of common weeds within crops.
Orchard Harvest	PyTorch	Apples	Yield estimation for orchards
Animal Identification	Tensorflow	MS-COCO animal classes	Animal counting on farms

image annotations. The *Forwarder* component then forwards the result to the query subscriber or route the results to other nodes for further processing.

B. Content-Aware Adaptivity

GNOSIS is capable of managing itself to achieve and maintain predefined goals in specific environments. A typical architecture for adaptive systems is the MAPE-K, which stands for monitor, analyze, plan, and execute, using a knowledge base. It allows the system to monitor its components and analyze their behavior to plan and execute the necessary changes, such that the adaptation's goals are met. In GNOSIS, these adaptations were based on the user queries and the available services to reduce the latency with a slight tradeoff in accuracy.

One example of content-aware adaptation is state-aware load-shedding across data streams. In our experience of developing GNOSIS, we observed that it is very easy for an IoMT stream processing system to get overloaded due to high input data rates. Furthermore, the performance worsens when data-intensive streams (such as video) are processed over costly DNN-enabled operators. Thus, a state-aware load-shedding strategy has been devised to improve the overall system performance in throughput and latency while maintaining acceptable levels of accuracy. The load-shedding process is decoupled across the multimodal streams. Decoupling enables the system to assign different load-shedding policies to different streams.

For example, to improve performance in a use case with video and audio streams, load-shedding is only applied to the video stream. In contrast, the audio stream is processed normally. However, such differential load-shedding policies can lead to synchronization issues and inaccurate results. To avoid synchronization and out-of-order event problems [49], [50], a state-based window policy is proposed [40], where states of the data streams are preserved. In addition, histogram-based frame similarity is used to drop similar frames across time. Fig. 11 shows the latency distribution of different image similarity algorithms. The selection of the histogram approach is based on its real-time and low-latency ($\sim$2 ms) frame similarity matching performance. For further details on this state-based filtering approach, refer to VID-WIN [40].

The following section discusses the use cases to show the GNOSIS potential usage in real-world applications.

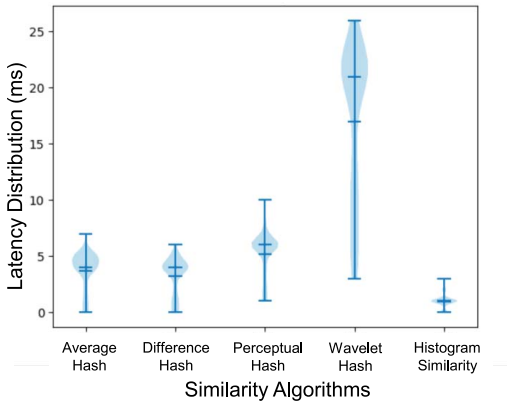

Fig. 11. Latency distribution of different frame similarity algorithms [40].

VI. Application Use Case

A. Occupational Health and Safety

The OHS use case demonstrates four safety compliance queries using different content analysis pipelines via GNOSIS to generate hard hat related events and generate an alert to the OHS supervisor. The queries are listed in the increasing order of their complexity.

1) Q1—Count the Number of Workers Wearing Hard Hat (Primitive Event and Count):

```
Q1: REGISTER QUERY CountHardHatWorkers
OUTPUT ANN_IMAGE_BBOX, ANN_IMAGE_QUERY_OUTPUT
CONTENT PPEDetection
MATCH (worker_hat:HAT)
FROM video.mp4
RETURN
COUNT(worker_hat) as WorkerCount
```

Query 1 (Q1) uses a single content service PPEDetection to count the number of workers wearing a hard hat using the CONTENT. The COUNT operator counts the number of MEKG nodes with the label worker_hat. Fig. 12(a) shows the worker count message (ANN_IMAGE_QUERY_OUTPUT) and bounding boxes (ANN_IMAGE_BBOX), which are generated due to the OUTPUT clause.

2) Q2—Detect Hard Hat Compliance Event (Single Modal and Spatial):

```
Q2: REGISTER QUERY HardHatCompliance
OUTPUT ANN_IMAGE_BBOX, ANN_IMAGE_QUERY_OUTPUT
CONTENT PPEDetection
MATCH (worker_hat:HAT)OR(non_worker_hat:NOT_HAT)
FROM video.mp4
RETURN COUNT(DISTINCT worker_hat)>0 AND
COUNT(DISTINCT non_worker_hat)=0 as
ComplianceStatus
```

Query 2 (Q2) detects a complex high-level HardHatCompliance event using a single PPEDetection model. The RETURN clause limits the required output and sets the *ComplianceStatus* as TRUE [if there are workers and everyone is wearing a hat, Fig. 12(b)] or False [Fig. 12(c)].

Fig. 12. Screenshot (a) worker count for Q1; (b) hard hat compliance and (c) hard hat compliance violation for Q2; and (d) safety compliance for Q4.

3) Q3—Person Classification (Single Modal and Spatial):

```
Q3: REGISTER QUERY CountWorkerVisitorCompliance
OUTPUT ANN_IMAGE_BBOX, ANN_IMAGE_QUERY_OUTPUT
CONTENT ObjectDetection, HardHatDetection,
ColorDetection
MATCH (worker:Person)<-[SPATIAL:OVERLAP_TOP]-
>(worker_hat:Hat {colour: 'WHITE'})
WHERE OVERLAP_TOP=60%
FROM video.mp4
RETURN COUNT(DISTINCT SPATIAL) as VisitorCount
```

Safety regulations may specify different color hats for different stakeholders [51], [52]. For example, the site managers and workers can be classified based on white and blue color hats, respectively. Query 3 (Q3) uses three DNN-based content analysis services—1) ObjectDetection (to detect person); 2) HardHatDetection (to detect hat); and 3) ColorDetection (to detect the hat color). The OVERLAP_TOP spatial operator is used to find the spatial alignment of "hat" and "person" objects (Fig. 13). Q3 shows the expressive potential of MEPL and GNOSIS, where events can be detected using a combination of content analysis pipelines and spatiotemporal operators.

4) Detect Hard Hat PPE and Noise Compliance (Multimodal Event With Rule):

```
Q4: REGISTER QUERY SafetyComplianceTest
OUTPUT ANN_IMAGE_QUERY_OUTPUT
CONTENT PPEDetection, AudioDetection
MATCH (s:DRILLING) AND (worker_hat:HAT) OR
(non_worker_hat:HAT)
WITH s.decibel > 95 AS s_count, worker_hat,
non_worker_hat
FROM Pub01
WITHIN TUMBLING_TIME_WINDOW(3)
RETURN COUNT(s_count) == 0 AS AudioCompliance,
COUNT(worker_hat)>0 AND COUNT(non_worker_hat)==0
AS PPECompliance, AudioCompliance AND
PPECompliance AS OverallCompliance
```

Query 4 (Q4) is a multimodal query identifying safety compliance at a construction site if the workers are PPE compliant and the instruments (such as drilling machines and jackhammers) are noise compliant. The query uses two content services PPEDetection for hard hat identification and AudioDetection for classifying audio stream (drilling machine and jackhammer). The query is executed over a time window of 3 s. A decibel operator is used to measure the instrument

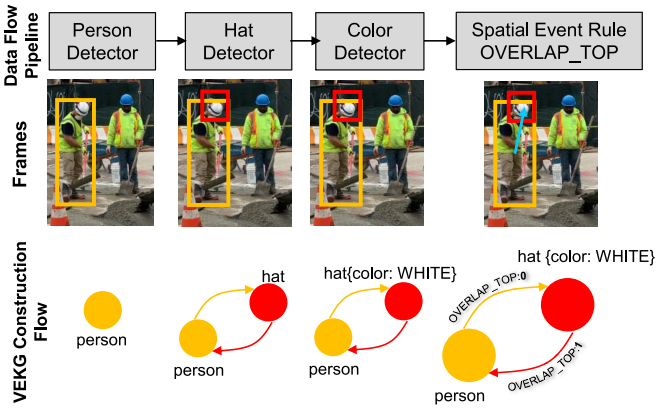

Fig. 13. VEKG graph construction for query 3.

sound in decibels. In Query 4, an MEKG is created using PPEDetection and AudioDetection content analysis services across two modalities. Fig. 12(d) shows an OverallCompliance status as false as the system detects the instrument (drilling machine) is noise compliant, but the person is not PPE compliant (not wearing a hard hat).

The following section focuses on the experimental design and evaluation performed over the GNOSIS platform using different data sets.

VII. Evaluation

The experiments are performed on a Linux machine running with a 3.1-GHz processor, 64-GB RAM and Nvidia RTX 2080 Ti GPU. The services talk via Nvidia Docker Runtime to perform complex DNN operations on the CUDA enabled GPU. The supported CUDA version is 10.1 with Nvidia Driver version 440. GNOSIS services can run on the CPU in the absence of a GPU device.

A. Data Sets

The experiments were performed with OHS-related events over three video streams (V1, V2, and V3). Table VI enlists the key characteristics of the selected data. First, queries Q1, Q2, and Q3 have been evaluated on video data V1 and V2, respectively. Next, query 4 (Q4) is executed on data stream V3, which constitutes audio and visual data.

Data Set Ground-Truth Preparation: The ground-truth data are manually annotated using the *similarity* technique. The

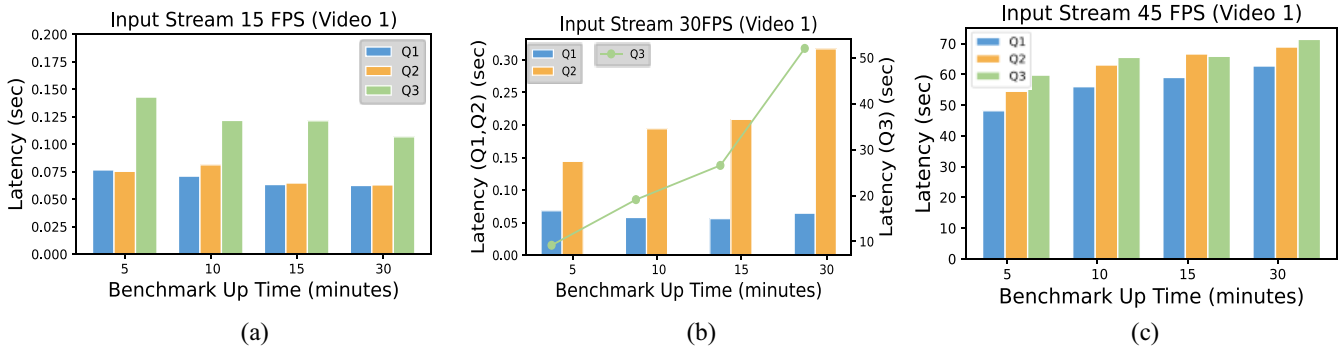

Fig. 14. End-to-end system latency for Q1, Q2, and Q3 for Video 1 at (a) 15 fps; (b) 30 fps; and (c) 45 fps.

TABLE VI
DATA SET DESCRIPTION

Data Stream	Stream Nature	Event of Interest	Other Info
Video 1 (V1)	Unstructured, Single Modal	A single person with a white hat, less object motion	Frame Resolution: 416*416
Video 2 (V2)	Unstructured, Single Modal	Multiple persons with different colour hats, significant object motion	Frame Resolution: 416*416
Video 3 + Audio (V3)	Unstructured, Multimodal	A single person with an electric drill, drill audio signal, significant object motion	Resolution: 416*416 Audio Bitrate: 128 KB, Sampling rate: 44100

TABLE VII
BENCHMARK CONFIGURATIONS FOR EVALUATIONS

Video	Query	Input rate	Load Shedding Policies	Benchmark Up Time
Video1 (V1)	Q1, Q2, Q3	15 fps, 30 fps, 45 fps	Non-Aggressive 45fps + 0.999 (Sim. Score) Mild-Aggressive 45fps + 0.99 (Sim. Score) Aggressive 45fps + 0.98 (Sim. Score)	5 mins, 10 mins, 15 mins, 30 mins
Video2 (V2)	Q1, Q2, Q3	15 fps, 30 fps, 45 fps	Non-Aggressive 45fps + 0.999 (Sim. Score) Mild-Aggressive 45fps + 0.99 (Sim. Score) Aggressive 45fps + 0.98 (Sim. Score)	5 mins, 10 mins, 15 mins, 30 mins
Video3 +Audio (V3)	Q4	15 fps, 30 fps, 45 fps	Non-Aggressive 45fps + 0.999 (Sim. Score) Mild-Aggressive 45fps + 0.99 (Sim. Score) Aggressive 45fps + 0.98 (Sim. Score)	5 mins, 10 mins, 15 mins, 30 mins

video frames are temporally correlated, and the content changes gradually across frames. A histogram-based similarity measure is used to create the batches of similar frames. The first frame of each batch is then manually annotated, and the same label is assigned to other frames of the batch. The goal is to create generic ground-truth data, which can be used across different queries. The labels were assigned from atomic events to high-level, complex events, including objects, attributes, and spatial relations. For query 4 (Q4), the audio signal is divided into smaller chunks using a time window and are annotated manually.

Benchmark Configuration: For each query, the performance is evaluated based on throughput, latency, accuracy, and memory consumption. To stress-test the system, performance evaluations are performed at different input streaming loads by varying frames processed per second (fps) at low (15 fps), standard (30 fps), and high (45 fps) rates. Similarly, the benchmark results are evaluated for different durations, i.e., 5, 10, 15, and 30 min. Table VII details the benchmark configurations set to evaluate various performance metrics. Single modal queries, such as Q1, Q2, and Q3, are evaluated for video V1 and V2, while multimodal query Q4 is evaluated against video V3. Three load-shedding policies are devised to evaluate queries based on different histogram similarity (Sim.) scores. The load-shedding is performed for the highest input rate (45 fps). Three histogram similarity scores, 0.999 (nonaggressive filtering), 0.99 (mild-aggressive filtering), and 0.98 (aggressive filtering), are identified empirically for video V1, V2, and V3. The histogram similarity scores are highly sensitive to color pixels, changing rapidly due to object motion and lighting conditions. Therefore, the above-identified score can be different for different video streams. The detailed single and multimodal evaluations are now explained in the following sections.

B. Single Modal Evaluation

1) End-to-End System Latency: The end-to-end latency includes the total time an event takes from GNOSIS input to its sink. As per (1), the system latency is defined as

$$t_{\text{system-latency}} = t_{\text{stream-manager}} + t_{\text{contentextractor}} + t_{\text{matchingengine}} + t_{\text{forwarder}} + t_{\text{delay}}. \quad (1)$$

In (1), $t_{\text{system-latency}}$ is the average latency of all the events across the GNOSIS components. The end-to-end latency also consists of delays (t_{delay}) in processes, such as tracing, messaging, network [53], [54], and graph databased input/output (I/O). The GNOSIS experiments were conducted on the same machine where the input source was present. Since all the microservices were running on the same machine, there was no network communication delay overhead (only internal communication). Figs. 14 and 15 show the system latency of

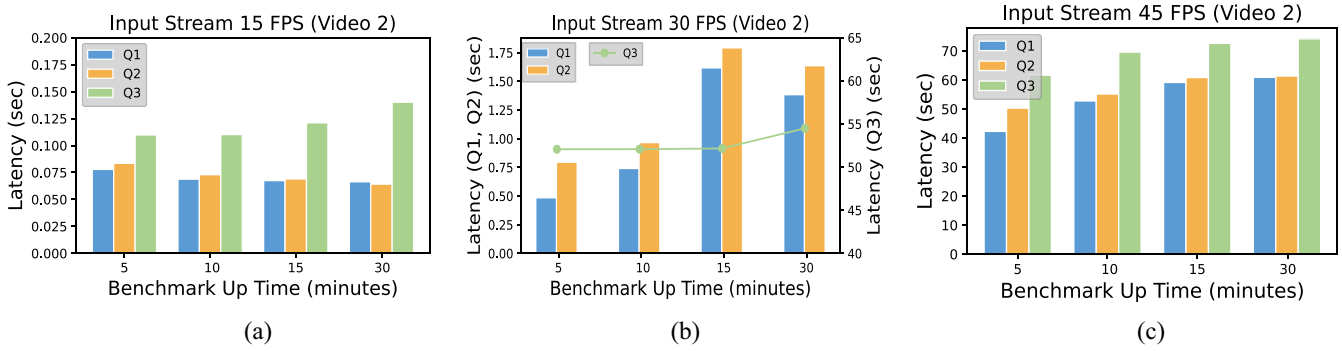

Fig. 15. End-to-end system latency for Q1, Q2, and Q3 for Video 2 at (a) 15 fps; (b) 30 fps; and (c) 45 fps.

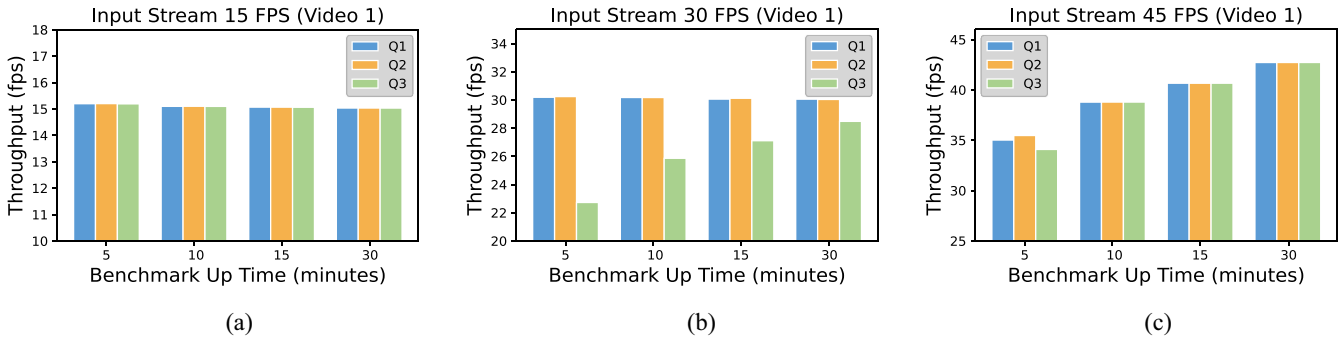

Fig. 16. System throughput for Q1, Q2, and Q3 for Video 1 at (a) 15 fps; (b) 30 fps; and (c) 45 fps.

query Q1, Q2, and Q3 for video V1 and V2, respectively. In Fig. 14(a), the system latency of V1 for 15 fps rate ranges between 0.069 and 0.150 s for Q1, Q2, and Q3. The latency of Q1 and Q2 is nearly the same across different benchmark times, while Q3 (0.150 s) is ∼2.1X of Q1 and Q2. This is due to Q3 complexity that requires three content analysis services (ObjectDetection, HardHatDetection, and ColorDetection) to detect the multimodal complex event pattern. The Q2 latency ranges between 0.14 s (5 min) and 0.31 s (30 min) for 30 fps video stream [Fig. 14(b)]. The Q2 latency is more than Q1 due to increased matching time despite having the same content analysis pipeline. As shown in Fig. 14(b) and (c), the Q3 performance deteriorates with an increase in input rate and benchmark time. The Q3 latency increases from 9.17 (5 min) to 52.01 s (30 min) for 30 fps [Fig. 14(b)] and 61.61 (5 min) to 74.16 s (30 min) for 45 fps input rate [Fig. 14(c)].

The above results show that latency increases for Q1–Q3 due to increased matching complexity and the number of models in the content analysis pipeline. Similarly, the latency increases with an increase in input rate and benchmark time. This is due to the increase in backpressure [55], [56], where an event gets buffered in an internal messaging queue to get processed due to the high input rate and fixed ML model cost. The cost performance of content analysis services is explained in Section V-B related to load-shedding operations. The other reason for increased latency over benchmark time is memory constraints. The current evaluations have been performed in limited experimental settings (like 64-GB RAM, 1 GPU). GNOSIS latency performance will improve with large memory and multiple GPU instances. Fig. 15 shows the system latency of Q1, Q2, and Q3 for Video 2 and follows the same pattern discussed in Fig. 14. For 15 fps, the Q1 latency is around 0.075 s and for Q3 is between 0.10 and 0.14 s [Fig. 15(a)]. For 30 fps, the Q3 latency ranges between 52.06 and 54.49 s [Fig. 15(b)] and is higher than Video 1 due to the increased number of objects and their motion over time. The system performs worst for 45 fps [Fig. 15(c)] with a minimum latency of 41.8 s (Q1—5 min) and maximum latency of 72.38 s (Q3—30 min).

2) System Throughput: System throughput is the number of event messages that the system processes within a specified period. In GNOSIS, the throughput is measured as the number of fps. As per (2), system throughput is defined as

$$t = \frac{\text{total number of event messages}}{\text{total time}}. \tag{2}$$

Figs. 16 and 17 show the system throughput of Video 1 and Video 2 for Q1, Q2, and Q3. For a 15 fps video stream, the throughput is ∼15 fps for all three queries [Figs. 16(a) and 17(a)]. Therefore, the system gracefully processes the incoming video data (Video 1 and Video 2) in real time with no backpressure. Fig. 17(a) shows that the throughput for Q1, Q2, and Q3 is slightly higher than the input stream rate. This is due to fluctuations in the ffmpeg streaming. The encoder, when set to a specific input rate, does not exactly stream at the same rate as the *vsync* parameter,[1,2] by default, and it is set at −1. Thus, continuous minor fluctuations depend on muxer capabilities (e.g., 14.8, 15.6, 15.8, and 16 fps), leading to minor observations. For 30 fps, queries Q1 and Q2 achieve nearly

[1]https://itectec.com/superuser/ffmpeg-libx264-how-to-specify-a-variable-frame-rate-but-with-a-maximum/

[2]https://superuser.com/a/883951

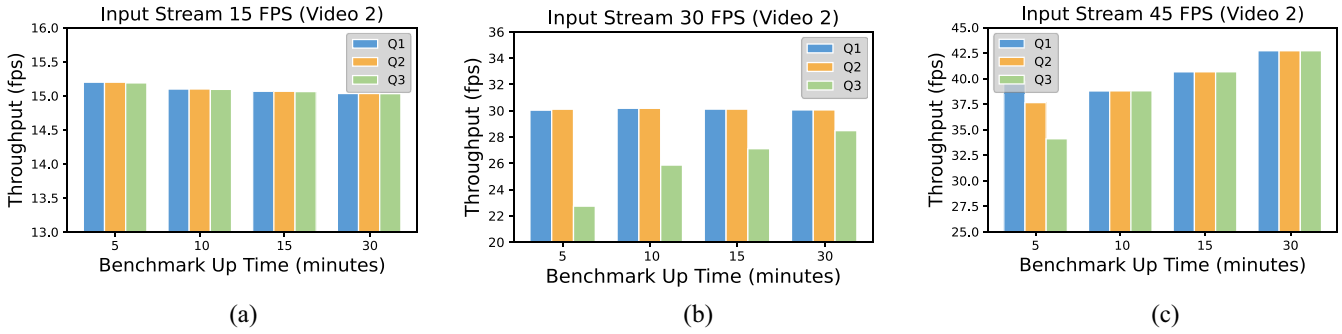

Fig. 17. System throughput for Q1, Q2, and Q3 for Video 2 at (a) 15 fps; (b) 30 fps; and (c) 45 fps.

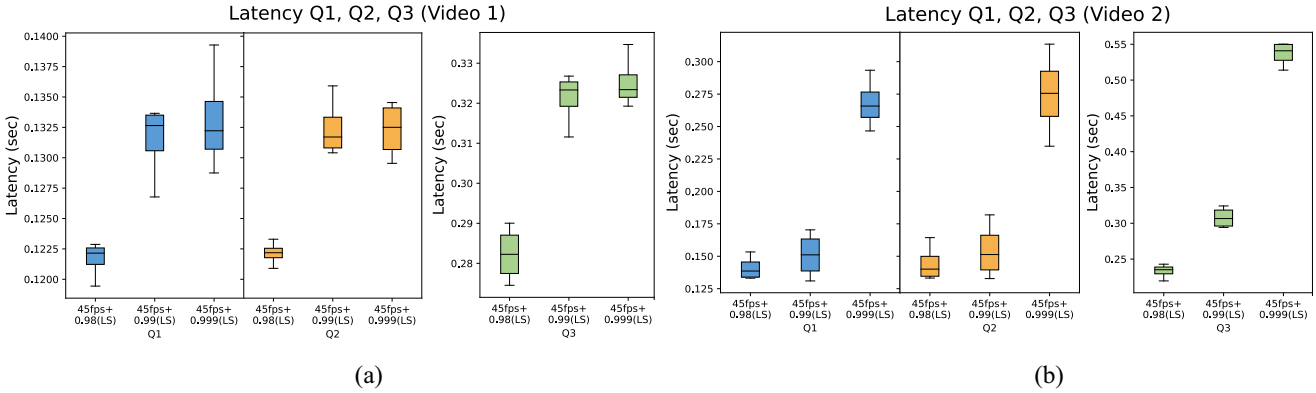

Fig. 18. System latency for Q1, Q2, and Q3 for different load-shedding policies for (a) 45 fps Video 1 and (b) 45 fps Video 2.

∼30 fps throughput, but Q3 throughput ranges between 22.7 and 28.4 fps [Figs. 16(b) and 17(b)]. The low throughput of Q3 is due to increased content analysis pipeline complexity having three models for processing the data. For 45 fps, the throughput ranges between 34.08 (Q3—5 min) and 42.72 fps (Q1—30 min) for Video 1 [Fig. 16(c)] and 34.10 (Q3—5 min) and 42.70 fps (Q1—30 min) for Video 2 [Fig. 17(c)], respectively. Compared to the input rate, the low throughput is due to the backpressure built as the number of input events (here, video frames) is higher than the cost of deployed models. For 45 fps, there is an increment in the throughput with an increase in benchmark time [Figs. 16(c) and 17(c)]. This is due to an increase in the number of events (video frames) over time. Therefore, a higher number of input events led to higher throughput. Still, it may have a cost burden like increased latency [Figs. 14(c) and 15(c)] that affect the overall system performance.

We tackle these issues using optimization techniques, including load-shedding and filtering to improve overall performance, i.e., reduced latency with increased throughput. The following section evaluates the content-aware load-shedding described in Section V-B to assess the overall performance.

C. Content-Aware Load-Shedding Optimization

As stated in benchmark configuration (Table VII), three load-shedding policies are used to evaluate the performance for 45 fps video stream: 1) nonaggressive (Sim. score-0.999); 2) mild aggressive (Sim. score-0.99); and 3) aggressive policies (Sim. score-0.98). The evaluations are performed on latency and relative throughput (RT) and are discussed in the following sections.

1) System Latency and Filtering Performance: Fig. 18 shows the latency distribution of Q1, Q2, and Q3 for Video 1 and Video 2 at different filtering rates. In Fig. 18(a), for Q1, the median latency is around ∼0.122 s for aggressive filtering (0.98 Sim. score) and increases to ∼0.132 s for nonaggressive filtering (0.999 Sim. score). A similar latency pattern is for query Q2. Compared to Q1 and Q2, query Q3 results in higher latency of ∼0.281, 0.323, and 0.324 s for aggressive, mildly aggressive, and nonaggressive filtering policies. For V2 [Fig. 18(b)], Q3 has the highest latency distribution of ∼0.543 s for nonaggressive filtering due to the increased complexity of video content. Overall, from Fig. 18, the two key takeaways are: 1) Q1 and Q2 have similar latency distribution while Q3 has the highest latency across videos due to its complexity and 2) the latency distribution increases from aggressive filtering to nonaggressive filtering. This makes sense as the system needs to process more data in a nonaggressive approach, leading to increased latency. On the other hand, the filtering approach over 45 fps input rate achieves near real-time latency, reducing ∼127 X latency even in the worst case scenario (Q3).

Fig. 19 shows the filtering percentage of events across video V1 and V2 for different load-shedding policies. For V1, the similarity algorithm filters 92.88%, 98.47%, and 98.50% of frames for nonaggressive, mildly aggressive, and aggressive filtering policies. The V2 filtering percentage is 33.65%,

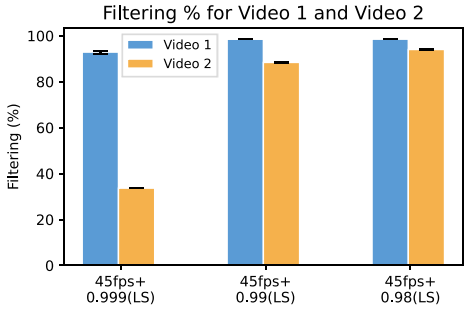

Fig. 19. Filtering percentage of Video 1 and Video 2 for different load-shedding policies.

88.39%, and 94.02% for the same filtering policies. The filtering rate of V2 is lower than V1 due to higher object motion leading to a rapid content change in V2. The current filtering policy is query agnostic and only depends on frame similarity metrics. The intent is to send only nonredundant video frames to the system to prevent backpressure. More complex query-based filtering techniques [40] can be deployed to improve system performance further.

2) Relative Throughput: The system processes less data due to filtering leading to an overall drop in system throughput. Thus, (2) does not hold in load-shedding as it does not give a clear picture of processing events within the system. To handle such a situation, RT [57], [58] is devised to understand system performance. Equation (3) defines RT as

$$\text{Relative Throughput (RT)} = \frac{\text{Throughput}_{\text{achieved}}}{\text{Operator}_{\text{cost}}}. \tag{3}$$

In (3), $\text{Throughput}_{\text{achieved}}$ is the current throughput achieved and $\text{Operator}_{\text{cost}}$ is the maximum throughput that an operator (here, content extractor service) can perform without any backpressure. For example, if the $\text{Operator}_{\text{cost}}$ is 30 fps, then for an input video rate of 30 fps, RT will be 1. This is the maximum input rate that a system can process without any backpressure. RT less than 1 represents a *no backpressure zone* and vice versa. Fig. 20 shows RT of V1 and V2 for Q1, Q2, and Q3 for different input rates, including load-shedding policies. In Fig. 20(a) and (b), the blue area represents the *no backpressure zone*. For 30 fps input rate, the RT is ∼1 suggesting the GNOSIS system operators can process Q1 and Q2 with no backpressure and Q3 with little backpressure. For 45 fps in both videos, RT is greater than 1 (∼1.3), suggesting a high backpressure scenario. The situation can be validated from Figs. 14(c) and 15(c), where the system has very high latency due to backpressure. For nonaggressive filtering (45 fps + 0.999 score), the RT drops to nearly 0.1 for Q1, Q2, and Q3 with about 13X improvement [Fig. 20(a)]. The performance improves further for mild and aggressive filtering where the system performs faster data processing with no backpressure. V2 [Fig. 20(b)] follows a similar pattern where RT for filtering policies is higher than V1 due to increased complexity in video content and more frames processing. The results can also be validated from Fig. 19, where the V2 filtering percentage is lower than V1, processing more frames.

3) Memory Consumption: Fig. 21 shows the memory usage for different input rates, including load-shedding policies of Q1, Q2, and Q3 for video V1. The memory usage increases from ∼10% to ∼23.1% over a given video duration. The 45 fps input rate has the highest memory consumption across queries (Q1, Q2, and Q3), followed by 30 fps. Q3 has the highest memory consumption across different input rates. Aggressive load-shedding improves memory usage by ∼1.9X, ranging from 10% to 12.16%. A similar pattern is followed for other videos and thus, not reported in this article.

D. Multimodal Query Evaluation With Load-Shedding Optimization

This section evaluates multimodal query Q4 on V3, including video and audio streams. The evaluation is performed using the metrics discussed above in the single modal queries Q1, Q2, and Q3.

1) End-to-End System Latency: As per (4), the multimodal latency is defined as

$$t_{\text{multimodal-latency}} = t_{\text{system-latency}}(\text{video, audio}) + t_{\text{synchronization-delay}}. \tag{4}$$

In (4), $t_{\text{synchronization-delay}}$ is an additional time that GNOSIS takes to synchronize the states of multiple streams (in this case, video and audio). The system waits for the incoming stream states and sends the data to the matcher when it receives all the data of a given window. For example, in Q4, the system processes the data only when it receives 3-s state window information for the audio and video streams. Since these streams are processed with different content analysis services with varying operators' costs, the system waits for the complete information. Fig. 22(a) shows the system latency for Q4 over Video 3 across different benchmark timings. The 45 fps stream has the highest latency ranging from 28.9 (5 min) to 129.75 s (30 min). Q4 is evaluated over a time window of 3 s, which adds up a synchronization delay; thus, the overall latency increases. We apply load-shedding over 45 fps, latency nearly equivalent to 15 and 30 fps input stream. The aggressive load-shedding latency ranges between 1.53 and 3.15 s. Currently, the load-shedding is only applied to video frames due to their computationally intensive nature. The audio streams are kept intact without any sampling.

2) System Throughput, Relative Throughput, and Filtering Performance: Fig. 22(b) shows the system throughput of Q4 for Video 3 at different benchmark times and input rates. The throughput for the multimodal query is also reported in fps. The reason is that the audio stream of the same duration is added to the MEKG and later processed for event matching. Thus, fps is a better metric to represent the throughput for Q4. However, the throughput metric can be generalized like events per second depending on the query context. For example, for the 15 fps input rate, the GNOSIS throughput ranges between 14.9 and 15.18 fps. The throughput increases with the increase in the input rate. For 45 fps, the maximum throughput rate is 38.94 fps (5 min), but increased latency due to backpressure is visualized in Fig. 22(a).

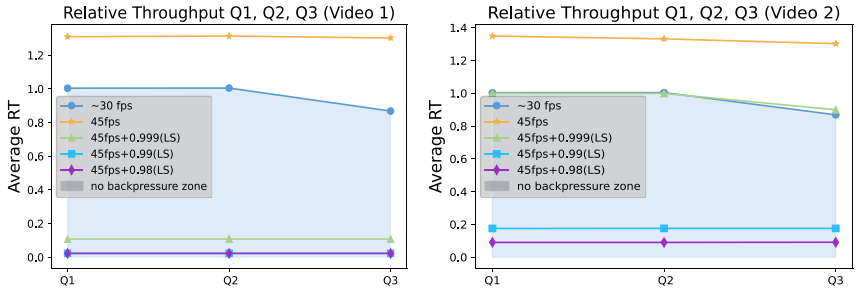

Fig. 20. RT for Q1, Q2, and Q3 for different input rates and load-shedding policies for (a) Video 1 and (b) Video 2.

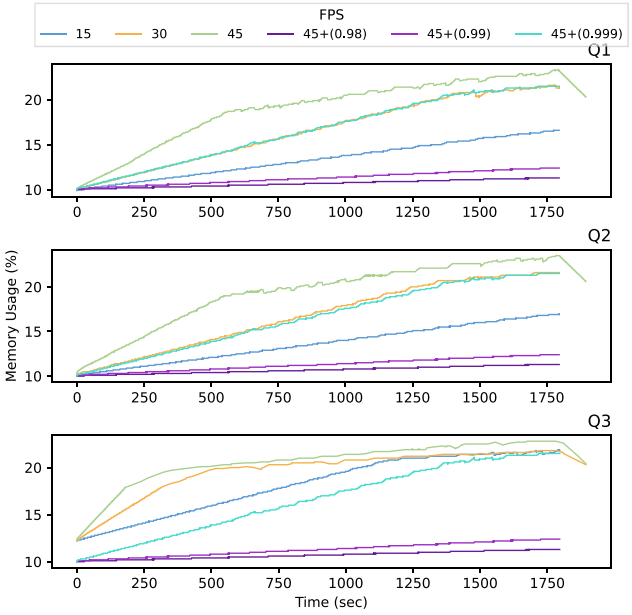

Fig. 21. Effect of load-shedding on memory usage for query Q1, Q2, and Q3 for Video 1.

Fig. 22(c) shows the RT for Q4. The pattern is like single modal queries Q1, Q2, and Q3 where 45 fps led to an RT of 1.29, suggesting increased backpressure in the system. The audio classifier has less cost than the video classification model. Therefore, $\text{Operator}_{\text{cost}}$ of 30 fps is considered based on the costliest model pipeline. In Fig. 22(c), all the filtering policies achieved RT less than 1 in no backpressure zone.

Fig. 23 shows the filtering performance of devised load-shedding techniques for Q4 over V3. Since load-shedding is performed over video data, the performance of the different policies is reported. The nonaggressive (0.999) filters out ∼81% of the video frames. On the other hand, the aggressive filtering (0.99) technique shed nearly 1.16 X more frames (94.2%) than nonaggressive filtering. Overall, the filtering percentage of V3 is lower than V1 and V2 due to increased object motion leading to less redundancy of frames.

E. Event Accuracy

The accuracy is evaluated by comparing pipeline results against the ground-truth data prepared beforehand. As per (5), the average (Av.) accuracy is defined as

$$\text{Av. Accuracy} = \frac{\sum_{i=1}^{n} \left(\frac{\text{event detected}}{\text{total events}} \right)}{n}. \tag{5}$$

The task gives the aggregated accuracy of the RETURN statement. The accuracy is the combined evaluation of all the content services output and the matcher engine outcome. For example, in query Q2, the return clause sends a Boolean ComplianceStatus event in true or false format. These Boolean events are then matched against the ground-truth data to evaluate the accuracy.

Since load-shedding is performed over incoming data, it impacts the overall accuracy. As more data are dropped, thus the overall accuracy will be reduced [40]. A new event-centric query accuracy is formulated to calculate the event accuracy in a filtering scenario. The metric is motivated from [59] and [60] to calculate the accuracy. For a given query Q, the event accuracy is calculated as

$$\begin{aligned} &\text{Event Occurence (EO)} = \begin{cases} 1, & \text{if an event is detected in win} \\ 0, & \text{if no event is detected in win} \end{cases} \\ &\text{Event Extra (EX)} = \frac{|E| \cap |G|}{|G|} \\ &\text{where} \begin{cases} E, & \epsilon \text{ extra events detected in win} \\ G, & \epsilon \text{ groundtruth events in win} \end{cases} \\ &\text{Event Accuracy} = \alpha * \text{EO} + \beta * \text{EX} \end{aligned} \tag{6}$$

$$\text{Av.Accuracy} = \frac{\sum_{i=1}^{n} \text{Event Accuracy}}{n}. \tag{7}$$

There can be the possibility of multiple events occurring in the same window (*win*) as the overall aim is to detect events over a given window length. Thus, detecting one event will suffice the query, i.e., Event Occurence (EO). Equation (6) works on the above assumption and set the value of α and β as 0.9 and 0.1, respectively. Finally, the event accuracies are averaged over n windows. For example, in query Q2, suppose for a given window of 1 s (30 frames), the ground truth (G) of OverallCompliance event is 30 (all frames comply with the query). After filtering, suppose the total events detected are 5. The EO = 1 and EX = 4/30. As per (6), the event accuracy will be $0.9 \times 1 + 0.1 \times 0.13 = 0.913$. This event accuracy will then be averaged across the given video length for n number of windows (7).

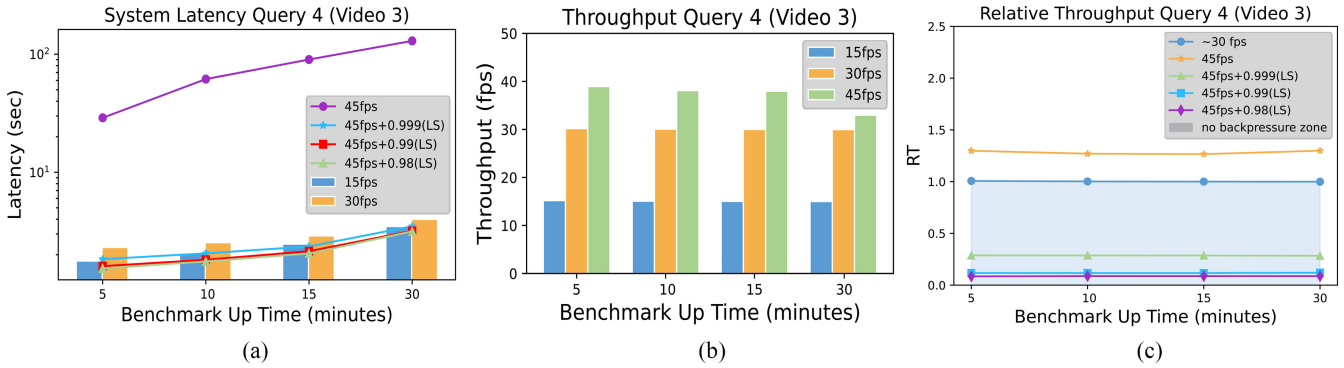

Fig. 22. Evaluation of multimodal query Q4 for Video 3 for (a) system multimodal latency; (b) system throughput; and (c) RT.

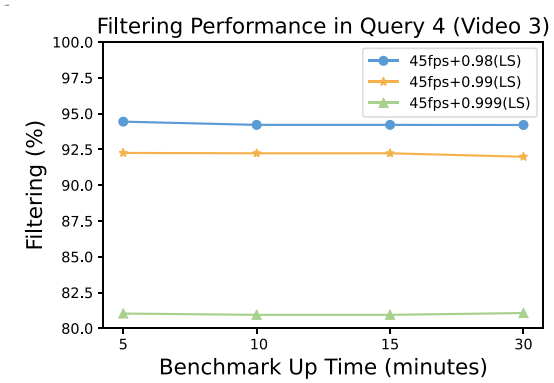

Fig. 23. Filtering performance for different load-shedding over Query 4 for Video 3.

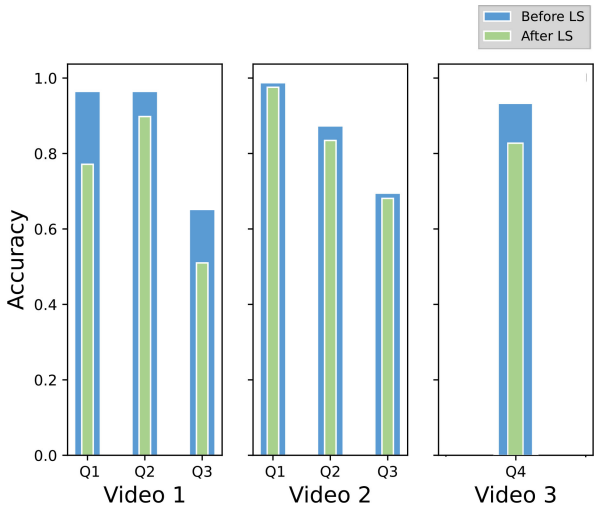

Fig. 24. Query accuracy of Q23, Q2, Q3, and Q4 for Video 1, Video 2, and Video 3 with (0.99 similarity score) and without load-shedding (L.S.).

Fig. 24 shows the accuracy of queries Q1, Q2, and Q3 for V1 and V2, and Q4 for V3 with and without load-shedding. Before load-shedding for V1, the Q1 and Q2 accuracy are around 0.96. The accuracy drops to 0.77 (Q1) and 0.89 (Q2) after load-shedding (0.99 similarity score). Q3 has the lowest accuracy of 0.65 (before LS) and 0.51 (after LS). Such a low accuracy of Q3 on V1 is due to query complexity where the false detection in the initial stage of the pipeline is carried further downstream. Another issue is that V1 has a minor content change because it has the least object motion. This filters out most of the frames leading to further accuracy drop. A similar pattern is found across queries in V2, where the accuracy of Q1, Q2, and Q3 are 0.98, 0.87, and 0.69, respectively, before load-shedding. The accuracy drops to 0.97 (Q1), 0.83 (Q2), and 0.68 (Q3) after load-shedding. Overall, V2 has better accuracy than V1 as the content analysis service could detect most objects correctly. Another reason is that V2 has a high change in content due to objects motion; thus, lower frames were filtered, resulting in better accuracy than V1 across queries. For multimodal Q4 (V3), the accuracy is shown for safety compliance only. The accuracy for Q4 is 0.89 (before LS) and 0.79 (after LS), respectively. Video 3 has the highest object motion resulting in a lower frame and accuracy drop.

VIII. Conclusion and Future Work

This article proposes the MEP paradigm as a formal basis for native approaches to multimodal content analysis (i.e., computer vision, linguistics, and audio) within the event processing paradigm to support real-time queries over multimodal data streams. Multimodal streams are processed to generate a knowledge graph representation, which can then be queried using the MEPL. MEPL supports user-defined event operators, which can be developed using a neurosymbolic-based hybrid approach, with statistical DNN models and symbolically-based spatial and temporal reasoning. MEP engines can then be implemented using hybrid processing, combining neural (DNNs) and symbolic (rules) models.

This article details an initial MEP implementation (GNOSIS) with near real-time performance with a throughput of ∼30 fps and subsecond latency of 0.075–0.30 s for 30 fps input rate video streams. The inclusion of a content-aware load-shedding for high input stream rates (45 fps) can significantly improve performance with only a minor drop in accuracy.

As a new paradigm, MEP has significant potential for future work, including the following.

1) *Adaptivity:* Adaptive techniques are a rich area to optimize the performance of MEP engines. In particular, the need for optimizations of pipelines for multiple queries, scheduling and workload placement across cloud-edge deployments, QoS, and Quality of

Experience, and the creation of query-specific deep learning models will be explored.

2) *Neurosymbolic Visual Reasoning:* The integration of visual and commonsense reasoning in MEP could improve and enhance the expressivity of event rules and queries. This can be achieved by carefully integrating knowledge about entities, relations, and rules from rich knowledge bases via reasoning over multimodal streams [61].
3) *Subjectivity:* Traditional event processing engines answer objective queries using the objective attributes of events. However, within MEP, we can have subjectivity as the content analysis process can produce a subjective representation of the event. Therefore, MEP engines will need to support subjectivity.

REFERENCES

[1] A. Aslam and E. Curry, "A survey on object detection for the Internet of Multimedia Things (IoMT) using deep learning and event-based middleware: Approaches, challenges, and future directions," *Image Vis. Comput.*, vol. 106, Feb. 2021, Art. no. 104095, doi: 10.1016/j.imavis.2020.104095.

[2] S. A. Alvi, B. Afzal, G. A. Shah, L. Atzori, and W. Mahmood, "Internet of multimedia things: Vision and challenges," *Ad Hoc Netw.*, vol. 33, pp. 87–111, Oct. 2015, doi: 10.1016/j.adhoc.2015.04.006.

[3] P. Yadav, D. Sarkar, D. Salwala, and E. Curry, "Traffic prediction framework for openstreetmap using deep learning based complex event processing and open traffic cameras," 2020, *arXiv:2008.00928*.

[4] A. García Pereira, A. Ojo, E. Curry, and L. Porwol, "Data acquisition and processing for GeoAI models to support sustainable agricultural practices," in *Proc. 53rd Hawaii Int. Conf. Syst. Sci. (HICSS)*, 2020, pp. 1–10. [Online]. Available: http://hdl.handle.net/10125/63854

[5] A. Floris and L. Atzori, "Quality of experience in the multimedia Internet of Things: Definition and practical use-cases," 2015, pp. 1747–1752, doi: 10.1109/ICCW.2015.7247433.

[6] L. Zhou and H.-C. Chao, "Multimedia traffic security architecture for the Internet of Things," *IEEE Netw.*, vol. 25, no. 3, pp. 35–40, May/Jun. 2011, doi: 10.1109/MNET.2011.5772059.

[7] M. L. Brodie and J. T. Liu, "The power and limits of relational technology in the age of information ecosystems," in *Proc. Seminar Digit. Enterprise Res. Inst.*, Apr. 2011.

[8] E. Curry, "Real-time linked dataspaces: A data platform for intelligent systems within Internet of Things-based smart environments," in *Real-Time Linked Dataspaces*. Cham, Switzerland: Springer Int., 2020, pp. 3–14, doi: 10.1007/978-3-030-29665-0.

[9] R. Atat, L. Liu, J. Wu, G. Li, C. Ye, and Y. Yang, "Big data meet cyber-physical systems: A panoramic survey," *IEEE Access*, vol. 6, pp. 73603–73636, 2018, doi: 10.1109/ACCESS.2018.2878681.

[10] J. Wu, S. Guo, H. Huang, W. Liu, and Y. Xiang, "Information and communications technologies for sustainable development goals: State-of-the-art, needs and perspectives," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 2389–2406, 3rd Quart., 2018, doi: 10.1109/COMST.2018.2812301.

[11] J. Wu, S. Guo, J. Li, and D. Zeng, "Big data meet green challenges: Big data toward green applications," *IEEE Syst. J.*, vol. 10, no. 3, pp. 888–900, Sep. 2016, doi: 10.1109/JSYST.2016.2550530.

[12] J. Wang *et al.*, "Bandwidth-efficient live video analytics for drones via edge computing," in *Proc. IEEE/ACM Symp. Edge Comput. (SEC)*, 2018, pp. 159–173, doi: 10.1109/SEC.2018.00019.

[13] M. Stonebraker, U. Çetintemel, and S. Zdonik, "The 8 requirements of real-time stream processing," *ACM SIGMOD Rec.*, vol. 34, no. 4, pp. 42–47, 2005, doi: 10.1145/1107499.1107504.

[14] P. Yadav, D. Salwala, F. Pontes, P. Dhingra, and E. Curry, "Query-driven video event processing for the Internet of Multimedia Things (Demo)," *Proc. VLDB Endowment*, vol. 14, no. 12, pp. 2847–2850, 2021. [Online]. Available: http://www.edwardcurry.org/publications/p2847-yadav.pdf

[15] S. S. Kamenov, "Experimental monitoring on network based tactile sensing system," in *Proc. IEEE XXVIII Int. Sci. Conf. Electron. (ET)*, Sep. 2019, pp. 1–4, doi: 10.1109/ET.2019.8878661.

[16] G. Cugola and A. Margara, "Processing flows of information: From data stream to complex event processing," *ACM Comput. Surveys*, vol. 44, no. 3, pp. 1–62, 2012, doi: 10.1145/2187671.2187677.

[17] F. Gao, M. I. Ali, E. Curry, and A. Mileo, "Automated discovery and integration of semantic urban data streams: The ACEIS middleware," *Future Gener. Comput. Syst.*, vol. 76, pp. 561–581, Nov. 2017, doi: 10.1016/j.future.2017.03.002.

[18] G. Cugola and A. Margara, "The complex event processing paradigm," in *Data Management in Pervasive Systems. Data-Centric Systems and Applications*. Cham, Switzerland: Springer, 2015, pp. 113–133, doi: 10.1007/978-3-319-20062-0_6.

[19] S. Hasan, S. O'Riain, and E. Curry, "Approximate semantic matching of heterogeneous events," in *Proc. 6th ACM Int. Conf. Distrib. Event-Based Syst. (DEBS)*, 2012, pp. 252–263. [Online]. Available: http://dx.doi.org/10.1145/2335484.2335512

[20] N. Navarin, M. Cambiaso, A. Burattin, F. M. Maggi, L. Oneto, and A. Sperduti, "Towards online discovery of data-aware declarative process models from event streams," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2020, pp. 1–8, doi: 10.1109/IJCNN48605.2020.9207500.

[21] A. ALDabbas and Z. Gal, "On the complex event identification based on cognitive classification process," in *Proc. 10th IEEE Int. Conf. Cogn. Infocommun. (CogInfoCom)*, Oct. 2019, pp. 29–34, doi: 10.1109/CogInfoCom47531.2019.9089989.

[22] M. Hilario, "An overview of strategies for neurosymbolic integration," in *Connectionist-Symbolic Integration: From Unified to Hybrid Approaches*. Mahwah, NJ, USA: Lawrence Erlbaum Assoc., 1997, pp. 13–36.

[23] I. Hatzilygeroudis and J. Prentzas, "Neuro-symbolic approaches for knowledge representation in expert systems," *Int. J. Hybrid Intell. Syst.*, vol. 1, nos. 3–4, pp. 111–126, Jan. 2005, doi: 10.3233/HIS-2004-13-401.

[24] A. S. d'Avila Garcez, K. B. Broda, and D. M. Gabbay, *Neural-Symbolic Learning Systems*. London, U.K.: Springer, 2002, doi: 10.1007/978-1-4471-0211-3.

[25] B. Zhang, X. Jin, S. Ratnasamy, J. Wawrzynek, and E. A. Lee, "AWStream," in *Proc. Conf. ACM Special Interest Group Data Commun.*, Aug. 2018, pp. 236–252, doi: 10.1145/3230543.3230554.

[26] D. Kang, P. Bailis, and M. Zaharia, "BlazeIt," *Proc. VLDB Endowment*, vol. 13, no. 4, pp. 533–546, Dec. 2019, doi: 10.14778/3372716.3372725.

[27] S. Krishnan, A. Dziedzic, and A. J. Elmore, "DeepLens: Towards a visual data management system," in *Proc. Conf. Innov. Data Syst. Res.*, 2019, pp. 1–10. [Online]. Available: http://cidrdb.org/cidr2019/papers/p40-krishnan-cidr19.pdf

[28] D. Kang, J. Emmons, F. Abuzaid, P. Bailis, and M. Zaharia, "NoScope: Optimizing neural network queries over video at scale," *Proc. VLDB Endowment*, vol. 10, no. 11, pp. 1586–1597, Aug. 2017, doi: 10.14778/3137628.3137664.

[29] Y. Lu, A. Chowdhery, and S. Kandula, "Optasia: A relational platform for efficient large-scale video analytics," in *Proc. 7th ACM Symp. Cloud Comput.*, Oct. 2016, pp. 57–70, doi: 10.1145/2987550.2987564.

[30] L. Ao, L. Izhikevich, G. M. Voelker, and G. Porter, "Sprocket: A serverless video processing framework," in *Proc. ACM Symp. Cloud Comput.*, Oct. 2018, pp. 263–274, doi: 10.1145/3267809.3267815.

[31] M. R. Anderson, M. J. Cafarella, G. Ros, and T. F. Wenisch, "Physical representation-based predicate optimisation for a visual analytics database," in *Proc. 35th IEEE Int. Conf. Data Eng.*, Macao, China, Apr. 2019, pp. 1466–1477, doi: 10.1109/ICDE.2019.00132.

[32] R. Xu *et al.*, "VideoChef: Efficient approximation for streaming video processing pipelines," in *Proc. USENIX Annu. Tech. Conf.*, Jul. 2018, pp. 43–56. [Online]. Available: https://www.usenix.org/conference/atc18/presentation/xu-ran

[33] T. Xu, L. M. Botelho, and F. X. Lin, "VStore: A data store for analytics on large videos," in *Proc. 14nth EuroSys Conf.*, Mar. 2019, pp. 1–17, doi: 10.1145/3302424.3303971.

[34] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017, doi: 10.1109/TPAMI.2016.2577031.

[35] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 2016, *arXiv:1506.02640*.

[36] Y. Lu, A. Chowdhery, S. Kandula, and S. Chaudhuri, "Accelerating machine learning inference with probabilistic predicates," in *Proc. Int. Conf. Manage. Data*, May 2018, pp. 1493–1508, doi: 10.1145/3183713.3183751.

[37] D. Korzun, A. Varfolomeyev, A. Shabaev, and V. Kuznetsov, "On dependability of smart applications within edge-centric and fog computing paradigms," in *Proc. IEEE 9th Int. Conf. Dependable Syst. Services Technol. (DESSERT)*, May 2018, pp. 502–507, doi: 10.1109/DESSERT.2018.8409185.

[38] N. Bazhenov and D. Korzun, "Event-driven video services for monitoring in edge-centric Internet of Things environments," in *Proc. 25th Conf. Open Innov. Assoc. (FRUCT)*, Nov. 2019, pp. 47–56, doi: 10.23919/FRUCT48121.2019.8981505.

[39] Y. Li, A. Padmanabhan, P. Zhao, Y. Wang, G. H. Xu, and R. Netravali, "Reducto: On-camera filtering for resource-efficient real-time video analytics," in *Proc. Annu. conf. ACM Special Interest Group Data Commun. Appl. Technol. Archit. Protocols Comput. Commun.*, Jul. 2020, pp. 359–376, doi: 10.1145/3387514.3405874.

[40] P. Yadav, D. Salwala, and E. Curry, "VID-WIN: Fast Video event matching with query-aware windowing at the edge for the Internet of Multimedia Things," *IEEE Internet Things J.*, vol. 8, no. 13, pp. 10367–10389, Jul. 2021, doi: 10.1109/JIOT.2021.3075336.

[41] Y. Wang, W. Wang, J. Zhang, J. Jiang, and K. Chen, "Bridging the edge-cloud barrier for real-time advanced vision analytics," in *Proc. 11th USENIX Workshop Hot Topics Cloud Comput.*, Jul. 2019, pp. 1–7. [Online]. Available: https://www.usenix.org/conference/hotcloud19/presentation/wang

[42] A. R. Neto, T. P. Silva, T. Batista, F. C. Delicato, P. F. Pires, and F. Lopes, "Leveraging edge intelligence for video analytics in smart city applications," *Information*, vol. 12, no. 1, p. 14, Dec. 2020, doi: 10.3390/info12010014.

[43] A. L. E. Battisti, D. C. Muchaluat-Saade, and F. C. Delicato, "Enabling Internet of Media Things With edge-based virtual multimedia sensors," *IEEE Access*, vol. 9, pp. 59255–59269, 2021, doi: 10.1109/ACCESS.2021.3073240.

[44] P. Yadav and E. Curry, "VidCEP: Complex event processing framework to detect spatiotemporal patterns in video streams," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2019, pp. 2513–2522, doi: 10.1109/BigData47090.2019.9006018.

[45] H. Zhang, G. Ananthanarayanan, P. Bodik, M. Philipose, P. Bahl, and M. J. Freedman, "Live video analytics at scale with approximation and delay-tolerance," in *Proc. NSDI*, 2017, pp. 377–392.

[46] D. Luckham, "Glossary of terminology: The event processing technical society: (EPTS) Glossary of terms—Version 2.0," in *Event Processing for Business*. Hoboken, NJ, USA: Wiley, 2015, pp. 237–258, doi: 10.1002/9781119198697.app1.

[47] H. Paulheim, "Knowledge graph refinement: A survey of approaches and evaluation methods," *Semantic Web*, vol. 8, no. 3, pp. 489–508, 2017, doi: 10.3233/SW-160218.

[48] P. Yadav and E. Curry, "VEKG: Video event knowledge graph to represent video streams for complex event pattern matching," in *Proc. 1st Int. Conf. Graph Comput.*, Sep. 2019, pp. 13–20, doi: 10.1109/GC46384.2019.00011.

[49] M. Liu, M. Li, D. Golovnya, E. A. Rundensteiner, and K. Claypool, "Sequence pattern query processing over out-of-order event streams," in *Proc. IEEE 25th Int. Conf. Data Eng.*, Mar. 2009, pp. 784–795, doi: 10.1109/ICDE.2009.95.

[50] B. Chandramouli, J. Goldstein, and D. Maier, "High-performance dynamic pattern matching over disordered streams," *Proc. VLDB Endowment*, vol. 3, nos. 1–2, pp. 220–231, 2010, doi: 10.14778/1920841.1920873.

[51] "What Colours are Safety Helmets (Hard Hats) on Construction Sites?" NBS. [Online]. Available: https://www.thenbs.com/knowledge/what-colours-are-safety-helmets-hard-hats-on-construction-sites (accessed Jun. 25, 2021).

[52] "Hard Hat Colour Codes—Meanings & Importance." [Online]. Available: https://www.highspeedtraining.co.uk/hub/hard-hat-colour-codes-in-construction/ (accessed Jun. 25, 2021).

[53] F. Guo, H. Lu, B. Li, D. Li, and C. W. Chen, "NOMA-assisted multi-MEC offloading for IoVT networks," *IEEE Wireless Commun.*, vol. 28, no. 4, pp. 26–33, Aug. 2021, doi: 10.1109/MWC.311.2000511.

[54] Y. Xiao, L. Xiao, Z. Lv, G. Niu, Y. Ding, and W. Xu, "Learning-based low-latency viot video streaming against jamming and interference," *IEEE Wireless Commun.*, vol. 28, no. 4, pp. 12–18, Aug. 2021, doi: 10.1109/MWC.101.2000474.

[55] S. Kulkarni *et al.*, "Twitter heron: Stream processing at scale," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2015, pp. 239–250, doi: 10.1145/2723372.2742788.

[56] R. L. Collins and L. P. Carloni, "Flexible filters: Load balancing through backpressure for stream programs," in *Proc. EMSOFT*, 2009, pp. 205–214, doi: 10.1145/1629335.1629363.

[57] J. Søberg, K. H. Hernes, M. Siekkinen, V. Goebel, and T. Plagemann, "A practical evaluation of load shedding in data stream management systems for network monitoring," in *Proc. Workshop Data Stream Anal.*, vol. 1, 2007, pp. 1–10.

[58] N. Tatbul and S. Zdonik, "Window-aware load shedding for aggregation queries over data streams," in *Proc. VLDB*, 2006, vol. 6, pp. 799–810.

[59] C. Canel *et al.*, "Scaling video analytics on constrained edge nodes," 2019, *arXiv:1905.13536*.

[60] T. J. Lee, M. J. Gottschlich, N. Tatbul, E. Metcalf, and S. Zdonik, "Precision and recall for range-based anomaly detection," 2018, *arxiv:1801.03168*.

[61] M. J. Khan and E. Curry, "Neuro-symbolic visual reasoning for multimedia event processing: Overview, prospects and challenges," in *Proc. Workshop Combining Symbolic Sub-symbolic Methods Appl. (CSSA)*, 2020, pp. 1–6.

Edward Curry received the Ph.D. degree in computer science from National University of Ireland, Galway, Ireland.

He is an Established Professor of Data Science and the Director of the Insight SFI Research Centre for Data Analytics and the Data Science Institute, National University of Ireland Galway, Galway, Ireland. He has made substantial contributions to semantic technologies, incremental data management, event processing middleware, software engineering, as well as distributed systems and information systems. He combines strong theoretical results with high-impact practical applications. The excellence and impact of his research have been acknowledged by numerous awards, including best paper awards and the NUI Galway President's Award for Societal Impact in 2017. His team's technology enables intelligent systems for innovative environments in collaboration with his industrial partners.

Dr. Curry is the Co-Founder and the Elected Vice President of the Big Data Value Association, an Industry-Led European Big Data Community, has built consensus on a joint European Big Data Research and Innovation Agenda, and influenced European Data Innovation Policy to deliver on the agenda.

Dhaval Salwala received the B.E. degree in IT from Gujarat Technological University, Ahmedabad, India, in 2021, and the M.Sc. degree in computer science from the National University of Ireland Galway (NUIG), Galway, Ireland, in 2019.

He has been a Research Assistant with the Insight Centre for Data Analytics, Data Science Institute, NUIG since October 2019. He is in the Smart Cities and Sustainable IT Group and researching in the field of real-time analytics and pattern detection for multimedia data streams. Before this, he has worked as a Software Professional for over five years in the Finance Sector with Tata Consultancy Services, Mumbai, India. He has comprehensive experience in designing, developing, and deploying E2E architecture for complex business process, and data processing pipeline.

Praneet Dhingra received the Bachelor of Technology degree in electronics and economics from Shiv Nadar University, Greater Noida, India, in 2017, and the M.Sc. degree in data science and analytics from the University College Cork, Cork, Ireland, in 2020.

He has been working as a Research Assistant with the Insight Centre for Data Analytics, NUI Galway, Galway, Ireland, since October 2020. He is currently with the ODS Group for Smart Cities and Sustainable IT researching and developing use cases in different smart domains. Before his master's, he was worked as a Data Analyst at a startup in New Delhi for almost an year where he gained experience with ETL tools, predictive analytics, and reporting. His research interests lie mainly in optimization algorithms and machine vision and also has a predilection for smart environments in sports.

Felipe Arruda Pontes received the M.Sc. (research) degree from the National University of Ireland Galway, Galway, Ireland, in 2021, where he is currently pursuing the Ph.D. degree with the Insight Centre, Data Science Institute.

He has worked for several years in the industry as a Software Engineer, participating in projects ranging from: online solutions for oil and gas exploration, to large scale online marketplace systems with distributed systems over the Cloud. He has participated in multiple patented research projects of fuzzy logic-based decision support systems. He is in the Open Distributed Systems Unit and researching in the field of distributed systems, sustainable IoMT applications, machine learning, and fuzzy logic.

Mr. Pontes has been a presenter and a coach on multiple conferences and social activities on development technologies, and is a contributor to many projects in the open-source community.

Piyush Yadav received the Ph.D. degree in computer science from the Lero SFI Research Centre for Software, National University of Ireland Galway (NUIG), Galway, Ireland, in 2021, and the M.Tech. (CSE) degree in information security from the Indraprastha Institute of Information Technology Delhi, New Delhi, India, in 2013.

He is a Senior Postdoctoral Researcher with the Insight SFI Research Center for Data Analytics, NUIG. Before joining NUIG, he was a Researcher with System Robotics Lab, Tata Research Development and Design Centre, Pune, India, where he worked in the field of smart cities, urban modeling, and privacy-preserving systems. He is in the Open Distributed Systems Unit and researching in the field of real time analytics and pattern detection for Multimodal Data Streams. His research interests are in complex event processing, multimedia analytics, computer vision, Internet of Things, machine learning, smart cities, and geospatial analytics.