Complex event processing on real-time video streams

Li, Ziyu; Katsifodimos, Asterios; Bozzon, Alessandro; Houben, Geert Jan

Publication date
2020

Document Version
Final published version

Published in
CEUR Workshop Proceedings

Complex Event Processing on Real-time Video Streams

Ziyu Li
supervised by Asterios Katsifodimos, Alessandro Bozzon and Geert-Jan Houben
Delft University of Technology
Z.Li-14@tudelft.nl

ABSTRACT

Cameras are ubiquitous nowadays and video analytic systems have been widely used in surveillance, traffic control, business intelligence and autonomous driving. Some applications, e.g., detecting road congestion in traffic monitoring, require continuous and timely reporting of complex patterns. However, conventional complex event processing (CEP) systems fail to support video processing, while the existing video query languages offer limited support for expressing advanced CEP queries, such as *iteration*, and *window*.

In this PhD research, we aim to develop systems and methods to alleviate these issues. In this paper, we first identify the need for an expressive CEP language which allows users to define queries over video streams, and receive fast, accurate results. To evaluate CEP queries on videos in real-time and with high accuracy, we explain how a streaming query engine can be designed to provide native support of machine learning (ML) models for fast and accurate inference on video streams. In addition, we describe a set of optimization problems that arise when ML models, with trade-offs in speed, accuracy, and cost, are part of a query plan. Finally, we describe how query plans on real-time videos can be optimized and deployed on edge devices with limited computational and network capabilities.

1. INTRODUCTION

Cameras installed in buildings, deployed on streets, and fitted on various devices generate vast amounts of video content on a daily basis. The video footage empowers a wide range of important applications such as in-door surveillance, traffic control, business intelligence and autonomous driving [8]. In real-world scenarios, we are generally interested in detecting complex events and receiving instant alerts, e.g., detecting road congestion in traffic monitoring and alerting to dangerous scenes in autonomous driving. Considering the urgent need for real-time response, it is infeasible to store videos in databases. Ideally, videos would be processed on the spot.

Proceedings of the VLDB 2020 PhD Workshop, August 31st, 2020. Tokyo, Japan.

Complex Event Processing (CEP) systems provide expressive languages to construct CEP queries for pattern detection over events. The constructs of an event algebra that fulfills the need of defining complex patterns was identified [3]. CEP query languages [4, 2, 1] define complex events to be detected and correlate them to more high-level meaningful information in data streams. However, less attention has been paid to content-based event detection on video streams in CEP systems, considering that conventional CEP systems accept structured data as input.

There have been many proposals for query languages on video [13, 12]. These languages provide high-level semantics, usually in an SQL-like format, allowing users to query video content. Compared to CEP languages, these SQL-like languages provide limited support for detecting complex patterns on video content, i.e., missing operations such as iteration and join, and restricted use of window (by counting number of frames), which leads to constrained queries. Table 1 shows a comparison of the current event query languages and existing video query languages. Operators listed in the table are the combination of the ones commonly used in streaming systems [3] and in video retrieval systems. From the table, we discover two research gaps, where 1) CEP systems lack support for video data while 2) multimedia retrieval languages fail to support well-rounded operators for CEP. This Ph.D. work aims to leverage these two gaps.

Video streams are difficult to process due to their low-level features (pixel value). The state-of-the-art object detection algorithm, Mask R-CNN [6] runs at 3 frames per second (fps), while in real-time the video frame rate is 25-30 fps. Nonetheless, techniques in ML, such as model specialization and model compression can be applied to expand the model search space [9, 8]. Models differ in shape, size, and the classes of object they can identify, and most importantly, in terms of accuracy and latency. Thus, it is essential to optimize these aspects in response to user preference, i.e., quality-oriented or speed-oriented.

The objective of this PhD research is to leverage the expressiveness of streaming languages in order to construct complex event patterns over real-time video streams, and to optimize the process aiming for efficient and sound results. To achieve the goal, in this paper we describe the PhD plan, split in three main research lines. First, we identify the motivating query types that cannot be supported adequately by current video query languages. Second, we show how to exploit the operators in the context of video processing by leveraging the existing streaming operators and state-of-the-art computer vision techniques. Since traditional query

Table 1: Comparison between CEP and video query languages. (*SIO*: Single-item Operators, *LO*: Logic Operators, *FMO*: Flow Management Operators; *TR*: Temporal Relationship; *SR*: Spatial Relationship)

	SIO	*LO*	*Iterations*	*Windows*	*FMO*	*Aggregates*	*TR*	*SR*	*Data*
CEP Languages									
SASE+ [4]	✓	Only negation	✓	✓			✓		Structured
Flink [1]	✓	✓	✓	✓	✓	✓	✓		Structured
TESLA [2]	✓	Only negation	✓	✓			✓		Structured
VidCEP [17]	✓	✓		✓			✓	✓	Video
Video Query Languages									
CVQL [12]	✓	✓	✓					✓	Video
SVQL [13]	✓	✓					✓	✓	Video
FrameQL [8]	✓	✓					✓		Video
SVQ [16, 10]	✓	✓				✓	✓	✓	Video

optimizers typically do not consider trade-offs of accuracy, speed and cost, we demonstrate the need to construct a ML model search space and to navigate the search space applying Pareto frontier and an AND-OR graph. Finally we consider deploying CEP query plans over video data closer to the device where they are generated and outline the challenges for optimizing such plans to execute at the edge, reducing communication and computation costs.

2. MOTIVATING EXAMPLE QUERIES

We first identify the use cases and scenarios that users may be interested in when processing real-time video streams.

Q1-Identifying of an object with attributes. The detection on a single object class requests on the occurrence of a specific object. This query type, as the simplest event pattern, covers a wide range of applications. For instance, a road surveillance system can monitor whether a car is parked at a non-stopping lane. Also, it is an essential task in autonomous driving to detect a traffic light along with its color. However, recent works, such as [9, 16, 17], do not either optimize the model in different scenarios, or provide support for additional attributes detection.

Q2- Identifying multiple classes of objects. What differentiates Q2 to Q1 is the detection on more than one class of object. In this query type, users are interested to perceive multiple objects. For example, in autonomous driving, the system should be alerted when multiple pedestrians are walking in the cross road with a yellow light on [8]. Still, there is deficiency in [17, 10], where the model is fixed and the optimization cannot solve the problem of model drift.

Q3-Temporal Relation. In this type of query, the time factor is involved in the complex event patterns. For example, detect the sequence of events within a time interval. the proposal of this query type distinguishes itself by the high expressiveness of window operators and the availability to detect iterated patterns. By providing a broader functionality of window, it is feasible to analyse the data streams in a timely manner, i.e., report aggregates over time, which has never been seen in previous works [9, 10, 8, 16, 12, 13]. In addition, few existing works in video query language mentioned iteration. This operator serves as an important role in defining the sequence of events by indicating the occurrence of an event multiple times in sequence.

Q4-Spatial Relation. This query type perceives the spatial relation between objects, i.e., to detect a car on the right of a lamp post. The application of this type of query can be applied to identify human-object interaction [16] and serves as a fundamental support for autonomous driving. To combine with *Q3* or other query types, it is feasible to construct more expressive queries, e.g detect human behaviors or track an object throughout time.

3. RESEARCH OVERVIEW

In this section, we would like to highlight the challenges of CEP over real-time video streams.

To process video streams in a CEP system, the challenges are reflected from three aspects: *query language*, *query planning with ML models*, and *query optimization on inference*, which will be illustrated below and explained in the rest of the paper.

Defining a CEP language for video processing. In CEP systems, a user can define specific queries using CEP language or streaming operators to detect occurrences of particular patterns of (low-level) events. Then the event streams are fed into the user-defined patterns. If all the conditions are met, the user will receive an outcome or alarm.

This raises some questions: How to define a complex event pattern over video streams given the available operators? Are there any additional operators required that can express the content of videos?

Query planning with ML models. Unlike conventional CEP systems, the video streams generated from devices can not be directly applied to match patterns, but instead are first processed by cascading models, where the video frames are transformed into events, i.e., object class, location, object attributes.

Given a complex event pattern, how can we decompose it and solve the components, i.e., identifying color, detecting objects? How many models can be applied to address the issue? Do the state-of-the-art models fulfill the need? If not, how can we extend the model search space?

Optimizing inference. To achieve high efficiency and effectiveness, an optimization mechanism plays a role in automatically assigning fast models for a given query and accuracy target. Is there an optimal solution to each query that can achieve low latency and high accuracy? Is there any method that can present the model selection process? Can we further optimize the inference by pushing more computation to the edge closer to the data source?

4. A CEP LANGUAGE FOR VIDEOS

As discussed previously, the conventional CEP systems lack support for video data, while video query languages do

not cater for advanced CEP needs. To leverage both, it is feasible to solve the problem and design a CEP language for videos. We will showcase the operators that are available in terms of the query types described above.

The most basic operator is that of *selection* [5]. This operator applies to events and filters out those that do not satisfy the predicates. *Projection* is another operator that belongs to *single-item operators*, together with selection. This type of operator transforms the attribute values of an event, and thus can be applied to generate video events by transforming the video frames into streams of events, including a set of attributes, i.e., timestamp, object class and color.

In terms of *Q3*–manipulating with the temporal relation, the time factor plays a pivotal role here. Existing video query languages [8, 17, 10] measure time by counting frames and doing the math using the speed of incoming frames measured in frames per second. Such operation is limited and constrained to answer requests, e.g., report on a regular basis. Our notion of the *window* operator, incorporated from stream processing, assigns windows to events with respect to different notions of time opted by application developers. One important notion of time supported is event time, which signifies the time a frame was generated at the source device. This provides us with a powerful abstraction for matching temporal relationships. Notably, different types of windows can be expressed, namely tumbling windows, sliding windows, session windows and global windows[1]. The high expressiveness of window operators, hence, enables timely report and flexible manipulation on queries. Also, it is worthy to note that *iteration*, as an important feature to define the repeated occurrences of a match event, is not covered in previous works [13, 12, 8, 17, 10]. In the proposal, we will extend the temporal patterns that have been seen in previous works, by incorporating the rich definition of window operators and introducing the iteration to define sequence of events.

To detect the spatial relation between objects is an important task in video retrieval, which is different from the conventional stream processing where the events do not reveal any spatial relationship. New operators should be defined to detect such relations on video contents. State-of-the-art object detection provide information that helps us to identify the class of object as well as their location within a frame, which enables to answer queries of *Q4*.

In addition, modern streaming systems scale-out by running multiple operator instances that process disjoint parts of the data in shared-nothing commodity hardware. This scalable distributed architecture is suitable for joining streams from distributed sources or distributing data to various nodes. It lays the foundation for edge computing, where a task may be divided and distributed to various computational nodes.

5. QUERY PLANNING WITH ML MODELS

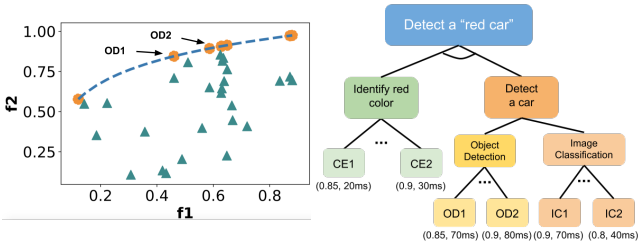
In order to run queries in the CEP language, we first need to identify the models that can be applied to solve the tasks in each query, e.g., detecting color or identifying a specific object. For each task, there are various models and solutions available. For example, to detect an object, both object detection and image classification can be applied. The state-of-the-art object detection models include but are not limited to Mask R-CNN [6], YOLOv2 [14]. On the other hand, AlexNet [11] and VGG-16 [15] are state-of-the-art deep image classification models. Despite all the above-mentioned models, techniques in ML, such as specialized model and model compression, can also be applied to enlarge the model search space. For example, a full and complex algorithm, e.g., YOLO9000 [14], can classify or detect thousands of classes (tasks). While a specialized model is a smaller model (i.e., with fewer layers and neurons) that mimics the behavior of a full NN model on a particular task. The sacrifice of generalization of inference models on restricted tasks can substantially reduce inference cost and latency.

To compile the query, we will construct a model search space by showcasing all the models available to solve a particular task. Accordingly, there are multiple options that can be selected in terms of various outcomes, i.e., accuracy and inference latency. These models differ in complexity, i.e., layers, number of neurons in dense layers and the ability to generalize.

6. OPTIMIZING VIDEO CEP QUERIES

As discussed above, the solutions to solve a task, e.g., detecting object class, are various. And there is no single model that can outperform all the others in terms of accuracy and speed. Let alone, the performance of the models is data-dependent [9]. And thus, we should optimize on the inference models for each query and consider the trade-off between accuracy and speed when assigning the models.

Manipulating the trade-off between accuracy and inference throughput can be regarded as a multi-objective optimization problem (MOP). Pareto-optimal solutions are applied in order to select pre-optimal solutions for each task. In this case, the aim is to process the video fast and accurately, and thus the objectives are accuracy and speed. To give an example, as shown in Figure 1a, every symbol represents a specific model, varying in shape, size and set of task, but fulfill the same goal, i.e., identifying a specific object. The blue line represents the *Pareto frontier*. Suppose that f1 is inference time and f2 is accuracy. We expect f1 to be lower and f2 to be higher. In the Pareto frontier, no solutions in the search space are superior to the others in the line in terms of both objectives. Only the models that lie in the Pareto frontier (such as *OD1* and *OD2*) are considered for further comparison.



(a) Pareto frontier (b) AND-OR graph

Figure 1: Optimization approaches

After we obtain the sub-optimal models for each task, the next step is to select and assign the optimal solution. In the previous step, the models are coupled with the outcomes, including the inference throughput and accuracy. If a user is quality-oriented, then the model with the highest accuracy is selected, and vice versa. To represent the solution of the

task, we will apply the AND-OR graph, as shown in Figure 1b. The query is decomposed into a set of smaller problems, i.e., identify red color and detect a car. The leaves of the AND-OR graph represent unique sub-optimal models in the Pareto frontier, and the optimal option will be sent to their parents for further analysis. The process goes on until the query is solved.

7. EDGE/FOG COMPUTING

To process the video streams in real-time, it is feasible to reduce transmission of data from one edge to another, by offloading tasks to the devices closer to the data source. But due to the limited computation power available at edge devices, the interactions with remote clusters or public clouds are inevitable. The models that are offloaded to the edge devices may thus sacrifice accuracy for inference and the complex models deployed remotely should compensate for it. In this phase, the challenge is to decide what task shall be offloaded and what information will be transmitted given the dynamic circumstances in real-world deployment.

In the edge computing paradigm, the objectives that need to be considered and optimized are not restricted to inference latency and accuracy anymore, but include wireless bandwidth, processing capacity and energy consumption [7], which increase the difficulty level of assigning optimal configurations in real-time.

8. FUTURE PLAN

Building prototype. As the aim of this PhD project is to provide highly expressive semantics for users to define queries over real-time video steams, we intend to first develop a prototype that is available for simple queries, e.g., $Q1$ and $Q2$. In this phase, the focus lies in the optimization module, where trade-off between latency and accuracy is exploited. Other queries will be implemented thereafter.

Video decoding. The characteristics of video may influence the model performance and should also be considered, i.e., frame resolution, frame sampling rate. For example, a crowded road scene compared to a quiet neighborhood require a more frequent frame sampling rate so that no event is missed, and a higher resolution to retain the details of content. The video characteristics will serve as important factors and will be taken into account in the configuration plan.

Adaptive configuration. Given the scene captured by the camera changes overtime, even by a static-angle camera, model drift is a main issue that affects the performance of the video processing. Dynamic configuration is complex and challenging if we want to maintain real-time performance, since dynamic configuration may hinder the inference procedure. In the future, we will investigate adaptive configurations given the trade-off between the inference speed and the quality of the results.

Edge computing. To bring the computation closer to the edge, we will investigate how to apply an edge computing paradigm into the deployment. In this phase, the design of the architecture of edge-cloud, the measure of real-time performance, and the decision to offload tasks and commit data will be important problems going forward.

9. ACKNOWLEDGMENTS

The author would like to thank Cognizant for their supports in this project. And many thanks to Marios Fragkoulis, Christos Koutras, Agathe Balayn, Andra Ionescu and Georgios Siachamis for their valuable feedback on this work.

10. REFERENCES

[1] P. Carbone, A. Katsifodimos, S. Ewen, V. Markl, S. Haridi, and K. Tzoumas. Apache flink: Stream and batch processing in a single engine. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 36(4), 2015.

[2] G. Cugola and A. Margara. Tesla: a formally defined event specification language. In *Proceedings of the Fourth ACM International Conference on Distributed Event-Based Systems*, pages 50–61, 2010.

[3] G. Cugola and A. Margara. Processing flows of information: From data stream to complex event processing. *ACM Computing Surveys (CSUR)*, 44(3):1–62, 2012.

[4] Y. Diao, N. Immerman, and D. Gyllstrom. Sase+: An agile language for kleene closure over event streams. *UMass Technical Report*, 2007.

[5] N. Giatrakos, E. Alevizos, A. Artikis, A. Deligiannakis, and M. Garofalakis. Complex event recognition in the big data era: a survey. *The VLDB Journal*, 29(1):313–352, 2020.

[6] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[7] C.-C. Hung, G. Ananthanarayanan, P. Bodik, L. Golubchik, M. Yu, P. Bahl, and M. Philipose. Videoedge: Processing camera streams using hierarchical clusters. In *2018 IEEE/ACM Symposium on Edge Computing (SEC)*, pages 115–131. IEEE, 2018.

[8] D. Kang, P. Bailis, and M. Zaharia. Blazeit: Optimizing declarative aggregation and limit queries for neural network-based video analytics. *arXiv preprint arXiv:1805.01046*, 2018.

[9] D. Kang, J. Emmons, F. Abuzaid, P. Bailis, and M. Zaharia. Noscope: optimizing neural network queries over video at scale. *arXiv preprint arXiv:1703.02529*, 2017.

[10] N. Koudas, R. Li, and I. Xarchakos. Video monitoring queries. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, pages 1285–1296. IEEE, 2020.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[12] T. C. Kuo and A. L. Chen. Content-based query processing for video databases. *IEEE Transactions on Multimedia*, 2(1):1–13, 2000.

[13] C. Lu, M. Liu, and Z. Wu. Svql: A sql extended query language for video databases. *International Journal of Database Theory and Application*, 8(3):235–248, 2015.

[14] J. Redmon and A. Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

[15] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[16] I. Xarchakos and N. Koudas. Svq: Streaming video queries. In *Proceedings of the 2019 International Conference on Management of Data*, pages 2013–2016, 2019.

[17] P. Yadav and E. Curry. Vidcep: Complex event processing framework to detect spatiotemporal patterns in video streams. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 2513–2522. IEEE, 2019.