SVQ-ACT: Querying for Actions over Videos

Daren Chao
University of Toronto
drchao@cs.toronto.edu

Kaiwen Chen
University of Toronto
kckevin.chen@mail.utoronto.ca

Nick Koudas
University of Toronto
koudas@cs.toronto.edu

***Abstract*—We present SVQ-ACT, a system capable of evaluating declarative action and object queries over input videos. Our approach is independent of the underlying object and action detection models utilized. Users may issue queries involving action and specific objects (e.g., a human riding a bicycle, close to a traffic light and a car left of the bicycle) and identify video clips that satisfy query constraints. Our system is capable of operating in two main settings, namely *online* and *offline*. In the online setting, the user specifies a video source (e.g., a surveillance video) and a declarative query containing an action and object predicates. Our system will identify and label in real-time all frame sequences that match the query. In the offline mode, the system accepts a video repository as input, preprocesses all the video in an offline manner and extracts suitable metadata. Following this step, users can execute any query they wish interactively on the video repository (containing actions and objects supported by the underlying detection models) to identify sequences of frames from videos that satisfy the query. In this case, to limit the number of results produced, we introduce novel result ranking algorithms that can produce the k most relevant results efficiently.**

We demonstrate that SVQ-ACT can correctly capture the desired query semantics and execute queries efficiently and correctly, delivering a high degree of accuracy.

I. Introduction

Advances in deep learning (DL) enabled the deployment of relatively inexpensive models to process videos detecting objects present in frames [1], [2] as well as associated actions [3]–[7] present in frame sequences. Past work and associated systems [8]–[15] demonstrated how it is possible to include object detection models as first-class citizens as part of declarative query processing over videos. The $SVQ{-}ACT$ system introduces algorithms to execute declarative queries that contain both objects as well as actions over streaming videos or video repositories in a unified manner.

Action detection/classification models are typically trained end to end. As such, they require numerous data sets with suitable types of actions for training. The ML community over the years has developed a plethora of such data sets containing different types of actions to train the models. A significant challenge to utilizing action detection models as part of query processing lies in the interaction of the action detection model with other related query predicates. Consider, for example, a query seeking to detect a frame sequence that depicts a robot dancing while a car is visible in the frame sequence as depicted in Figure 1. From an action detection perspective, the typical models for detecting actions are trained on the actions themselves (e.g., Robot Dancing) and are not aware of other objects. Thus, an action detection module that is trained to recognize a robot dancing cannot be used to answer

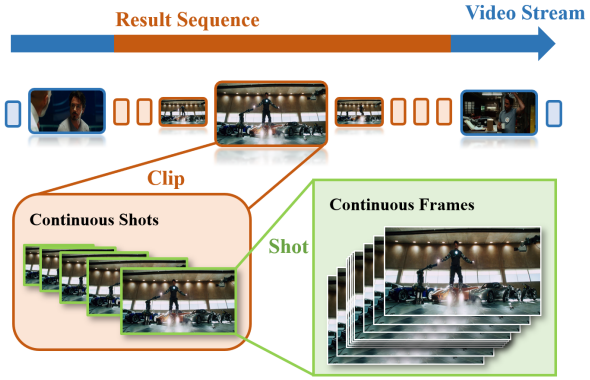

Fig. 1. An Example Video Stream.

such a query in an effective manner (as it necessitates a post-processing step). One could, in principle, train a model to recognize actions that also contain a car in the frame sequence. Such an approach is not scalable, however, as it would require a model for any possible combination of query predicates present in queries, which is clearly impractical.

A different approach would be to decouple the detection of the action from the detection of the other objects mentioned in a query. Namely, one could detect a sequence of frames containing the desirable action using an action detection model, then utilize an object detection algorithm to detect frame lists that contain the desired objects in the query and intersect them. Such an approach, however, requires several parameters/thresholds to be decided apriori. For example, for how many frames the action frame sequence should overlap the various frame lists containing objects in order to declare a query match? Object detection algorithms are typically noisy (yield false positives and negatives), so how would such noise affect the thresholds? Is it possible or even feasible to choose or decide such thresholds before a query executes or to tune such thresholds for each different query predicate?

$SVQ{-}ACT$ addresses these exact problems and produces a solution in a principled manner. The system can process queries both in an *online* setting in which query results have to be reported as the video streams and an *offline* setting in which queries are issued against a specified repository of videos that have been suitably pre-processed. To eliminate the burden of setting thresholds per query manually (an impossible task) and cope with the inaccuracies inherent to action and object detection models (which are impossible to control), our query processing approach utilizes a theoretically

grounded methodology based on scan statistics [16]–[18]. This approach first estimates the distribution of predictions made by each individual model involved in the query when the query predicates are not satisfied. Then, for each query predicate, it computes what constitutes a significantly large number (a critical value, k_{crit}) of positive predictions (the query predicate is satisfied) conducted by each individual model in a sequence of frames. These are utilized to synthesize an answer and determine whether a query is satisfied in a sequence of frames. Intuitively, if the number of positive predictions across the models utilized in the query exceeds k_{crit} in a sequence of frames, then this frame sequence has a higher probability of satisfying the query. An additional innovation the system introduces is the adaptive computation of k_{crit} for each query, so this threshold is not treated as a constant that has to be specified apriori.

We next outline the main architecture of our system and present the demonstration experience.

II. SVQ-ACT Architecture

A video is defined by a sequence of frames $V = \{v_1, \dots v_n\}$. The length of a video (number of frames) n can be finite or infinite. If n is unbounded, we refer to V as a video stream. The video repository is a collection of videos where each of them has a finite length. A shot is defined by a group of continuous frames of fixed length. Action classification models will accept a shot as input and yield their predictions for the action the shot includes. Typically the length of shots is defined by the action detection algorithms (e.g., 10 frames). A continuous collection of shots of a set length is called a clip. There might be multiple actions and object predictions in a clip as it contains several shots. These are depicted in Figure 1. The clip length is a parameter in our setting. Smaller clip lengths may fragment a long result sequence into multiple sequences, while a larger clip length may not. $SVQ{-}ACT$ enables visualization of the effects of the length of clips, and this is part of the demo experience.

Generally, a query specification encompasses several predicates, which can be specific actions (e.g., robot dancing from Kinetics-700 [19]), presence of objects in frames (e.g., human, car), or relationships between objects in frames (e.g., human left of the car). $SVQ{-}ACT$ considers queries consisting of conjunctions of query predicates. A query q is: $\{p_1; ...; p_I; a \in A\}$, where I is the number of query predicates p_i involving object types, a represents the query action, and A is the set of all actions the action detection model is trained on. Predicates p_i may request the presence of a specific object type o_i on frames or evaluate the existence of a constraint (e.g., spatial relationship) among object types on frames [10]. In either case, the evaluation of each p_i on a frame has a binary output (true/false). The techniques used to evaluate each p_i on a frame are orthogonal to our approach. For example, the existence of an object type can be conducted by any object detection model [2]. The evaluation of spatial relationships among objects can be conducted by evaluating spatial predicates on the detected objects [10].

Results reported in $SVQ{-}ACT$ are presented as sequences of frames, which is a continuous collection of clips. It is represented as a pair of start and end clip identifiers and can be of any length as determined by our algorithms. Our system can adapt any model for action detection and any model for object detection, and both are treated as black boxes. The system operates in two modes that we detail below.

A. Online Mode

In the online mode, a user can select a video source that provides frames in real-time (e.g., a surveillance video) and provide a query in an SQL-like language at the UI that encompasses the action and the objects that should be present for the query to be true. The system will process each object predicate and the action predicate independently, producing sequences of frames (for object predicates) and sequences of shots (for the action predicate) that satisfy the corresponding predicate. These sequences are produced by our query processing methodology that is grounded on scan statistics requiring no additional parameter settings other than the query provided. Any parameter required is estimated adaptively (as the video streams), relieving the user from the burden of setting and/or adjusting parameters. This mode encompasses algorithms we propose that operate in real-time to identify the start and end frames that the query is satisfied. The proposed algorithms deploy models inspired by Scan Statistics in an online query processing framework.

B. Offline Mode

In the offline mode, our system accepts any video repository as input. This is typically a collection (of any size) of videos of varying lengths. The idea is to first preprocess these videos in a manner, making them available for interactive query processing. The queries supported are exactly the same as in the online case. However, since the number of results could be large across numerous videos, we enhance the query syntax that limits the result size to a user-specified number k. Since we would like to produce the k results that are most relevant to the query, we introduce a generic monotone ranking mechanism to quantify result relevance. $SVQ{-}ACT$ can accept any ranking mechanism as long as it satisfies some generic monotonicity criteria.

The ingestion phase is executed only once when videos are added to the repository. We will process each video utilizing object detection and action classification models. Since the queries are not known in advance, we extract metadata for all possible object types recognized by the deployed object detection model (assuming that is maybe part of some query in the future) and for all possible actions (similarly assuming it could be part of some query in the future), supported by the action classification model utilized. For each possible action and object, we can adopt a similar methodology as the online case to generate tables containing sequences of frames (for objects) and shots (for actions) that independently satisfy the corresponding object and action predicates.

During the query phase, we are given the query and the specified number of results required, K, as the input. By utilizing the metadata extracted during the ingestion phase, our algorithms first identify all results that satisfy each query predicate independently and then merge them in an efficient

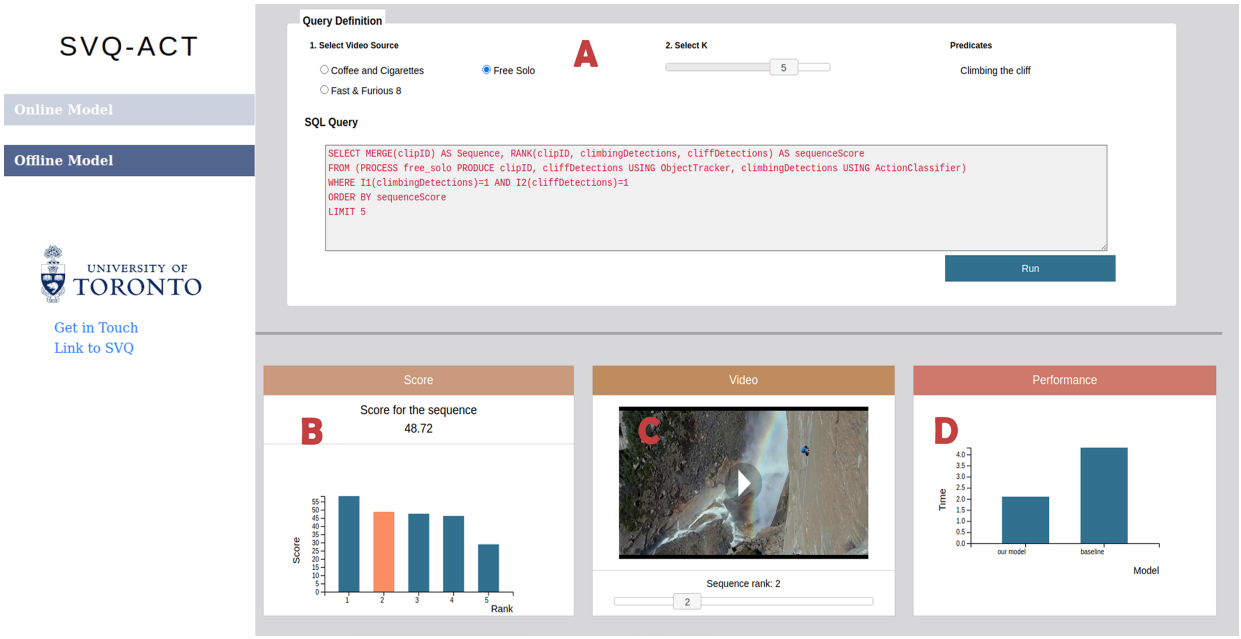

Fig. 2. A sample front end for the offline case.

way to identify the K most relevant (highest ranking) results in an effective manner. This is doable via novel top-K query processing algorithms that we propose. Our techniques utilize a very different evaluation strategy than popular top-k query processing algorithms, like TA [20], [21] and are tailored to the specifics of video sequences.

III. Demonstration Experience

The emphasis of the demonstration is on usability and demonstrating the accuracy and usefulness of the system for specific tasks. Details on the underlying algorithms can be provided as well if requested but are not part of the demonstration. Interested users will have the capability to choose video sources, define the queries in a declarative style interface, observe the query results interactively, view their execution progress, and observe the accuracy of the system compared to annotated ground truth for select queries.

During the demo, users will have the ability to observe two different workflows. The first corresponds to the offline query experience. Figure 2 presents an overview of the demo layout for the offline case. In area A, a user is presented with a choice of video sources to query. Once a video repository is selected, a number of applicable predicates are presented representing objects and actions present in the repository. Once these are selected, the user specifies the number of top results required. Then the system is presenting an auto-generated SQL query that will dynamically update when the query predicates change. A user can edit the query manually if required. For example, in figure 2, the user is interested in identifying the top 5 sequence results that contain a person climbing the cliff in the repository Free Solo.

Once the query is submitted, it will be processed against all metadata applicable to the repository. The top k sequence results, as well as their corresponding rank generated by our algorithm, will be presented in area C. Users can view each sequence results with a specific rank using the slider bar. Area B displays the actual score of each result as calculated for each sequence and highlights the score for the currently selected sequence. The example in figure 2 shows that the score for the current sequence is 48.72, and it ranks second among all k results. Users can move the slider bar, change results and observe them and judge for themselves whether the ranking makes sense. Area D will compare the processing time of our approach with baseline approaches. In the above example, our algorithm is at least twice as fast as the baseline.

The second workflow corresponds to the online case. Figure 3 presents the layout of the demo for this case. Similarly, in area A, users can select the video source to issue queries against. Once the source is selected, all applicable predicates are revealed, and the user can make suitable choices. Subsequently, the query is generated dynamically (Section C). One of the main innovations in our system is the ability to conduct query processing utilizing a statistical approach that eliminates the need to set thresholds [1]. In order to demonstrate the benefits of our parameter-free approach, in this case, users have the option to set thresholds manually and compare them with our automated way to generate applicable thresholds per query. This is available in Section B, where users can experiment with parameters of baseline approaches and set thresholds. Once the query executes, the results will be displayed in real-time, along with the accuracy of our approach compared with manually annotated results (ground truth) for select queries. The results can be reviewed in section D as they are produced along with associated accuracy (versus ground

[1]See Section I. Such thresholds correspond to a critical value k_{crit} utilized by our models to quantify what constitutes a sufficiently high number of detections for a specific predicate.

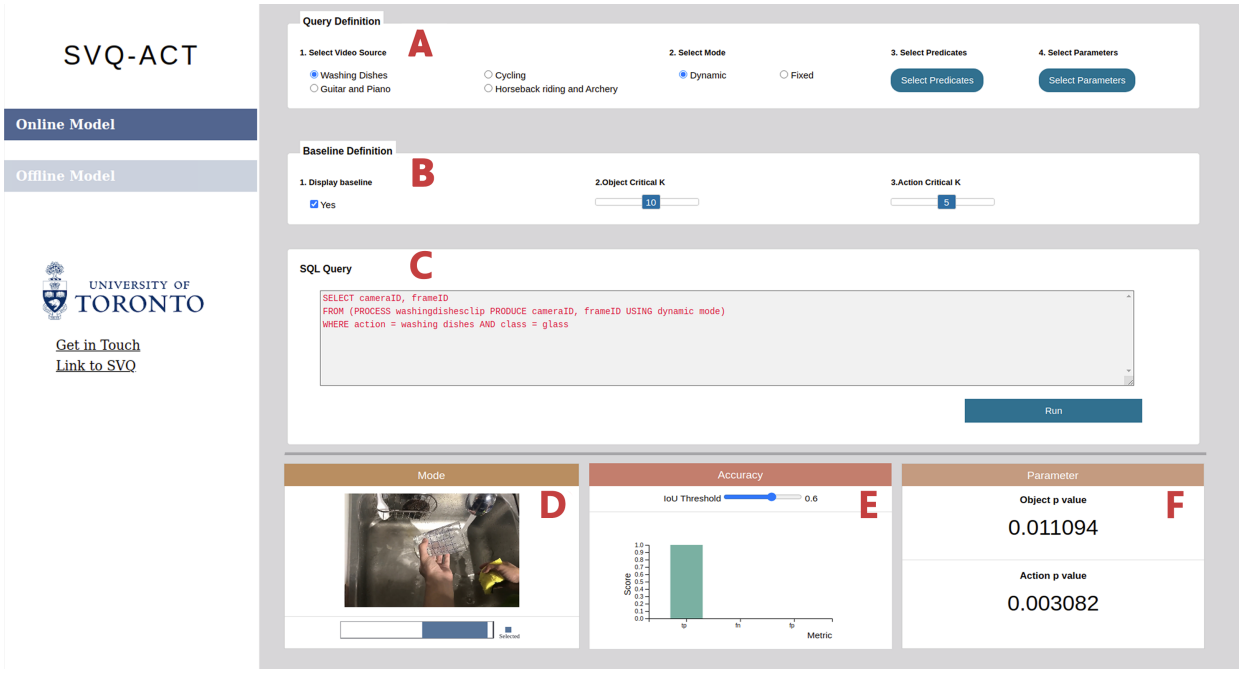

Fig. 3. A sample front end for the online case.

truth for manually annotated data) (Section E) and visualize the adaptive parameters computation our algorithms conduct (Section F).

IV. Conclusion

We presented $SVQ{-}ACT$, a system capable of processing declarative style queries involving objects and actions in an online and offline setting. We demonstrate that our algorithms for the online case produce more robust and accurate results compared to other applicable methods. In the offline case, our techniques significantly improve query time performance. We believe that research in this area is very fruitful and pressing given the prevalence of videos and the rapid advances in DL algorithms, offering unique opportunities for the data management community.

References

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

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

[3] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in neural information processing systems*, 2014, pp. 568–576.

[4] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6299–6308.

[5] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A closer look at spatiotemporal convolutions for action recognition," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 6450–6459.

[6] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7794–7803.

[7] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 6202–6211.

[8] D. Kang, J. Emmons, F. Abuzaid, P. Bailis, and M. Zaharia, "Noscope: optimizing neural network queries over video at scale," *Proceedings of the VLDB Endowment*, vol. 10, no. 11, pp. 1586–1597, 2017.

[9] 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.

[10] N. Koudas, R. Li, and I. Xarchakos, "Video monitoring queries," in *2020 IEEE 36th International Conference on Data Engineering*, 2020, pp. 1285–1296.

[11] I. Xarchakos and N. Koudas, "Svq: Streaming video queries," in *Proceedings of the 2019 International Conference on Management of Data*, 2019, pp. 2013–2016.

[12] D. Chao, N. Koudas, and I. Xarchakos, "Svq++: Querying for object interactions in video streams," in *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, 2020, pp. 2769–2772.

[13] Y. Chen, X. Yu, N. Koudas, and Z. Yu, "Evaluating temporal queries over video feeds," in *Proceedings of the 2021 International Conference on Management of Data*, 2021, pp. 287–299.

[14] D. Chao, N. Koudas, and X. Yu, "Marshalling model inference in video streams," in *2023 IEEE 39th International Conference on Data Engineering*, 2023.

[15] D. Chao, Y. Chen, N. Koudas, and X. Yu, "Track merging for effective video query processing," in *2023 IEEE 39th International Conference on Data Engineering*, 2023.

[16] J. I. Naus, "Approximations for distributions of scan statistics," *Journal of the American Statistical Association*, vol. 77, no. 377, pp. 177–183, 1982.

[17] J. Glaz, J. Naus, and S. Wallenstein, *Scan Statistics*, 1st ed. Springer, 2001.

[18] R. Turner, Z. Ghahramani, and S. Bottone, "Fast online anomaly detection using scan statistics," in *2010 IEEE International Workshop on Machine Learning for Signal Processing*, 2010, pp. 385–390.

[19] J. Carreira, E. Noland, C. Hillier, and A. Zisserman, "A short note on the kinetics-700 human action dataset," *arXiv preprint arXiv:1907.06987*, 2019.

[20] R. Fagin, "Combining fuzzy information: an overview," *ACM SIGMOD Record*, vol. 31, no. 2, pp. 109–118, 2002.

[21] R. Fagin, A. Lotem, and M. Naor, "Optimal aggregation algorithms for middleware," *Journal of computer and system sciences*, vol. 66, no. 4, pp. 614–656, 2003.