Querying Videos Using DNN Generated Labels

Yifan Wu*
University of California, Berkeley
yifanwu@berkeley.edu

Steven Drucker
Microsoft Research
sdrucker@microsoft.com

Matthai Philipose
Microsoft Research
matthaip@microsoft.com

Lenin Ravindranath
Microsoft Research
lenin@microsoft.com

ABSTRACT

Massive amounts of videos are generated for entertainment, security, and science, powered by a growing supply of user-produced video hosting services. Unfortunately, searching for videos is difficult due to the lack of content annotations. Recent breakthroughs in image labeling with deep neural networks (DNNs) create a unique opportunity to address this problem. While many automated end-to-end solutions have been developed, such as natural language queries, we take on a different perspective: to leverage *both* the development of algorithms *and* human capabilities. To this end, we design a query language in tandem with a user interface to help users quickly identify segments of interest from the video based on labels and corresponding bounding boxes. We combine techniques from the database and information visualization communities to help the user make sense of the object labels in spite of errors and inconsistencies.

ACM Reference Format:
Yifan Wu, Steven Drucker, Matthai Philipose, and Lenin Ravindranath. 2018. Querying Videos Using DNN Generated Labels. In *HILDA'18: Workshop on Human-In-the-Loop Data Analytics, June 10, 2018, Houston, TX, USA.* ACM, New York, NY, USA, Article 4, 6 pages. https://doi.org/10.1145/3209900.3209909

1 INTRODUCTION

Deep learning has enabled a new wave of unprecedented semantic labeling and question answering capabilities in the past decade [2, 6, 11, 15, 17]. While increasingly these new algorithms are deployed in the wild, most techniques are still not at human level accuracy, which may pose new usability challenges. In this paper, we propose both a structured query language and corresponding user interface to leverage human knowledge. We bring in well-established techniques in information visualization and search interfaces to help provide affordances and context for the labels generated.

Traditional multi-media retrieval has relied on manual annotation [3, 9], low-level visual feature extraction [19], or a narrow subset of objects and actions (e.g., "zone", "equipment" [14]). These methods are challenging to scale to the quantity and diversity of current video repositories. Recent advances in computer vision have enabled semantic labeling of images, where deep neural networks (DNNs) can identify a diverse range of objects and their corresponding regions in the image [11, 17].

This new ability to work with spatial and temporal information for objects in videos creates new ways to interact with video streams. While traditional search techniques such as document retrieval exist, it is difficult to query for more complex events which inherently contain structure over the labels—structure in space and time.

While there has been much effort on direct question and answer support for video queries [5–7, 12, 15, 22] (to name a few), the algorithms used by these systems are more recent than that of object identification, and arguably more complex. Instead of trying to complete the whole task in an automated fashion, we hope to leverage the capabilities of users to combine the best of both the automated and human capabilities through a query language and user interface, designed in tandem. The goal is to enhance user capabilities and alleviate the complexity of the challenge to the computer systems. Specifically, we explore how to use training over basic images to create structured information via a query language.

To this end, we propose a **v**ideo **q**uery **l**anguage (VQL) over the labels obtained from state-of-the-art object recognition systems. VQL is designed to be intuitive to use, expressive, and compatible with a graphical user interface. VQL is executed by transpiling to SQL, using a small set of user-defined functions, leveraging existing database technologies, making the backend portable and efficient. A compelling addition to VQL notifies users if relevant new frames match their query using database `TRIGGER` operator.

Furthermore, we found that inconsistent and erroneous labeling of DNN algorithms require that queries be constructed iteratively. To help modify their query appropriately, users need to better contextualize the results and isolate parts of the query. We designed visualizations of the objects' temporal and spatial information to help the user navigate "unknown unknowns". Detailed annotations also help the user narrow down the parts of the query that need changes.

The project is an initial exploration in a large design space, and much more user experiments are needed to understand the human understanding of the results and errors. However, we hope to share some early results to start a conversation in the community.

*Work done while interning at Microsoft Research

HILDA'18, June 10, 2018, Houston, TX, USA

ACM ISBN 978-1-4503-5827-9/18/06…$15.00
https://doi.org/10.1145/3209900.3209909

2 BACKGROUND AND MOTIVATION

Since the 90's, several languages for querying videos have been proposed: Huang et al. developed a language, VIQS, to query human labeled objects and activities to find video segments like "where a party occurs" [9]. To help eliminate tedious, manual labeling, Chang et al. developed a video search system, VideoQ, where users can draw the shape, specify color, or draw the trajectory of an object [3]. More recently, Saykol et al. presented VSQL, which is restricted to a small set of events for surveillance, such as objects entering a scene, items being picked up, and four types of objects [19]. Similarly, Le at al. proposed SVSQL for querying surveillance videos with objects and events, e.g., `SELECT <Output > FROM < Database > WHERE <Condition >` [14]. These previous efforts were limited by the technologies at the time—they either require intensive human labor or expose only low-level details of the video. The past decade saw a rapid rise of deep neural networks, and we are now able to identify objects in images with high accuracy [17, 18]. However, we cannot directly apply the previous query techniques because they are not designed with the characteristics of the new capabilities. For instance, VIQS assumes a notion of activity be labeled [9], and SVSQL events [14]. Neither events nor activities are available with basic image classification. In addition, none of the query systems mentioned deal with the cases of inconsistent or incorrect labels, the effect of which we will discuss in more detail in Section 4.

Beyond object identification, there are many recent ML research on segmenting and captioning events and actions [2, 11, 13, 16, 20], but the accuracies tend to be lower than those of object classifications—all state of art labeling algorithms have a top-5 accuracy of over 90% on ImageNet dataset [17, 18]. Recently systems researchers have developed fast systems to answer basic (binary classification queries) [12] or predefined queries [22]. These research tend to have a high "cold start" cost for a new query and requires much more instrumentation than identifying objects in videos. Machine learning researchers also have worked on natural language querying [5–7, 15]. While these approaches can eventually offer experiences much closer to human to human interactions, the field is still under active research for arguably simple cases (e.g., 3D shapes [6]) and require non-trivial processing per query [2]. *Our goal is orthogonal with those of the state of the art query answering systems*—we do not intend to compete in the benchmarks for automatic identification or Q&A tasks—our goal is to create design techniques to leverage human capabilities in the query process to simplify the task complexity and accuracy required by the machine.

To this end, there have been some efforts to integrate search interfaces with that of ML labeled results. Zhang et al. developed an object and scene faceted search interface over segmented videos [23]. However the interface there is limited to binary toggles of objects and scene keywords. We hope to make more detailed use of the labels with their bounding boxes using a query language to capture both temporal and spatial semantics.

3 QUERY LANGUAGE

To support the user in making effective use of the object labels, we need first a formalism to express and compose the predicates on individual frames across time. This simple query language is designed in tandem with the user interface, described in the following section—the query is not intended to be used standalone but rather as a driver for the UI.

We first describe the overall goal of the language and user interface, the high-level relationship of VQL and prior art, a running example to explain and motivate design choices, the data model, and lastly the operators.

The goal of VQL is to help users transform labels and bounding boxes of each frame in a video into predicates that locate segments of videos. While the goal is unique to the needs of querying videos, the query approach shares many similarities with geospatial queries such as PostGIS [1] and time series queries [4, 10]. While we could describe VQL in terms of these specialized operators, we chose to describe VQL operations in terms of SQL for simplicity. Again, our goal is not to innovate on spatial or temporal queries, but rather to use these techniques from databases with techniques from information visualization and search to create a novel lens at an emerging use case.

As a driving example, take a homeowner analyzing the video of her front porch, trying to figure out why a package delivered sometime during the day was opened and refer to that throughout the paper.

3.1 Data Model

The data model for VQL is a table with the following columns: `label, region, frame, confidence`, where each row is the output of a DNN. There is an additional table with columns `frame, time, previous, next`, used for processing a sequence of frames into time periods. More concretely, we discuss the entities in this data model:

- **label**: a string that describes an object, labeled by DNN algorithms. Labels are not *objects* since the same object could be labeled differently across frames. For instance, in the porch video, the person who opened the package has been labeled as "man", "woman", "traveler", and "living thing", all of which are somewhat correct but are difficult to use. This "inconsistency" is reasonable given that the DNN algorithm we deploy focus on a single frame and are agnostic to the fact that all the frames are from the same video.
- **grid**: a *fixed* rectangular partitioning of the frame of the video into rows and columns. Physical distances are also not currently supported by image recognition DNNs, and grids are easier to specify and more efficient to compute. A **cell** in a grid is a unit rectangle in the grid, represented by a tuple of row (left to right) and column (top to down) indices. E.g., `(1,2)`.
- **region**: a rectangle in the video frame. Represented as two opposing cells of the rectangle. E.g., `((1,2) to (3,5))`. This is called a "bounding box" in computer vision.
- **frame**: a single image, a point in time, represented as an integer that is the sequence id of the frame in the video. A **segment** is a set of contiguous frames represented by a tuple with the starting frame and the ending frame. A segment represents a period and serves as a proxy for an event. A **segment set** is a set of non-overlapping segments.
- **confidence**: a number between 0 and 1, yielded by the DNN about its confidence that the label is correct.

Table 1: Example Queries

Description	Query
more than three cats and two dogs	`FIND ((COUNT "cat") > 3) AND (COUNT "dog") > 2);`
a car stopped in the middle of the street	`FIND ("car" IN (3,3) to (5,5)) FOR 30s;`
a person was near a car for longer than 1 minute.	`FIND ("person" NEAR "car" BY 1) FOR 60s;`
a person quickly moved close to the camera	`FIND (("person" IN (1,1) to (8,8) > 0.6) AFTER ("person" IN (1,1) to (8,8) < 0.2)) < 5;`

3.2 Spatial Operators

In the example scenario, the homeowner may want to verify that the package was indeed delivered in the first place. Because her neighbors are moving, there are other boxes in the video. To disambiguate, she can specify the package to be *in a region*. To achieve this, in VQL, one could write `FIND <label> IN <region>`.

The operator `IN` is an instance of a *spatial operator*. `IN` returns true for a frame if the frame contains a label whose bounding box is in the specified region. One extension to the `IN` operator is to limit by how much a labeled object fills the input region. We introduce two *threshold operators*, > and < to specify by how much the object fills in the region, evaluated as $\frac{area(label)}{area(region)} > threshold$.

Physical relationships between an object and space can be more than containment; we list them below. Due to the relative simplicity of the operators, we do not describe more detailed implementation mechanism here:

- `OVERLAP <region>`: if *parts* of the labeled region is in the specified region.
- `NOTIN <region>`: applying existence, `OVERLAP` to the union of the rectangles in the inverse of `<region>`.
- `NEAR <region> BY <number of cells>`: expanding the region that the label is in by a certain amount, parametrized by the user, by number of cells, starting from the boundary of the region.

Often, the *object* of interest is composed of multiple labels, maybe because the DNN labeled the same object with different words, or if the user cares about different objects. We, thus, extend `<label>` to `<label> (OR <label>)*` (regular expression syntax) to support this feature.

Lastly, we also support spatial queries between objects, such as `'man' NEAR 'CAR' BY 1`, where the "1" is the number of cell distances apart. We take the same query semantics as the label region query but cast the label to its bounding box for every frame evaluated. This is particularly useful when the video is taken by nonstationary cameras. If there are multiple matching objects, the query is evaluated on all pairs and returns true if there is a match.

3.3 Temporal Operators

An atom is evaluated on a video by applying it to each frame. This yields a *segment set* which becomes the input to *temporal operators*. Since segments correspond to (parts of) events, there should be predicates applied on the properties of segment sets to help users further narrow down their target. For instance, the homeowner may have located when the package was delivered, but since the DNN offers no label for *open* versus *closed* package, she needs to find people that have come close *after* the package is delivered.

To achieve the functionality to query segment sets, we introduce the following operators:

- **Duration**: filters segments that are true longer or shorter than some period. Syntax: `<atom> [>, <] <duration>`.
- **Smooth**: merges disconnected segments that are within a certain distance. Syntax: `<atom> SMOOTH <duration>`
- **Logical composite**: intersect or union two segment sets. Syntax: `<atom1> [AND, OR] <atom2>`.
- **Order composite**: filter out segments in one segment set that occurs after or before segments in another segment set, with no other segments in between. Syntax: `<atom1> AFTER <atom2>`. This is may be useful for events with sequential dependency.

3.4 Non-Segment Queries

Whereas all previous queries return segment sets, we now talk about queries that return regions and objects.

- All the **objects** in region: `FIND LABELS IN <region>`—useful for users discovering labels. For instance, the homeowner was initially confused why a search for package did not show the earliest frame when the package was present, but then upon viewing the list of objects realized that the DNN had previously used the label "box".
- All the **regions** in which an object has existed `FIND REGIONS OF <label>`—useful for creating the heat map, which we describe in the following section. For instance, the homeowner is trying to narrow down when or if someone got close to the package. It would be helpful to see the places the label "person" may have been, for her to then specify the location in the query.
- The **counts** of satisfying labels of an atom in the video per frame: `FIND COUNTS OF <atom>`. For instance, users might identify outliers in histograms, which are often indications of interesting events.

3.5 Execution

Table 2 illustrate how VQL can be mapped to simple SQL queries with a UDF, `predicate`, to evaluate the spatial operators per frame. We implemented region operators as bit vectors to improve performance. The detailed transpilation is out of the scope of this paper. The temporal operators are all executed on the client, as the visualizations require more detailed frame- and segment-level information. It is future work to investigate limits to this approach.

Table 2: VQL to SQL

VQL	SQL	Output
`FIND "package" in ((6,6) to (7,7))`	`SELECT frame FROM labels WHERE label = "package" AND predicate(region, ((6,6) to (7,7)));`	set of segments
`FIND LABELS IN ((4,5) to (7,7))`	`SELECT label FROM labels WHERE predicate(region, ((4,5) to (7,7)));`	set of labels
`FIND COUNTS OF "person" IN ((4,5) to (7,7))`	`SELECT COUNT(*), frame FROM labels WHERE predicate(region, ((4,5) to (7,7))) GROUP BY frame;`	set of (frame, count)

Another optimization technique we use is skipping frames. Often a video does not change much, and one could save computation by only running DNN evaluation on frames that have changed by some threshold, measured by the percentage of pixels that are no longer of the same value. We currently use a heuristic value of 10%—there is a tradeoff between potentially missing important details with higher change-threshold, and creating too many noisy labels/consuming computation resources with lower change-threshold. Additionally, we use a heuristic for filtering out low confidence labels.

4 QUERY INTERFACE

The query language alone is difficult for the user to use, since for any given item, there may be several reasonable labels, and the exact locations of the objects are hard to specify without inspecting the frames. To facilitate discovery and context, we developed a user interface in tandem with the VQL. This section first describes the design goals of the UI, relevant literature from information visualization and search interfaces, functionalities to help users locate segments of interest, implementation, and evaluation considerations.

The design goal is to provide context and visualize the video segment results, and the key challenge is dealing with labeling errors and inconsistencies. We look to literature in the information visualization community and search community to address these two aspects. Direct manipulation is a common technique for UI developement [8], and well suited for our use case given the visual nature of videos and images to bridge the "gulf of specification", in addition, we provide visualizations of the search result to bridge the "gulf of evaluation".

To provide context for the labels in relation with the images, we use the concept of *cross-filter* [21] combined with VQL to define how interacting with the label list, frames, and segments update the other components, respectively, as illustrated in Figure 1. A cross-filter allows data to be filtered using a selection made in a separate part of the interface. We will discuss how the filtering works, as well as the components of the UI, integrated with the running example of the homeowner.

First, the user needs to gain a sense of what labels are available for query in the system. For instance, while the homeowner might query for "package", the DNN algorithm may have labeled the package as "box", which is not technically incorrect, but the user needs a way to reconciliation their conception with that of the systems. In other cases, the DNN algorithm could just be incorrect, for example, in one frame, we had the same package mistakenly labeled as a "book". Here, seeing that there is a mysterious "book"

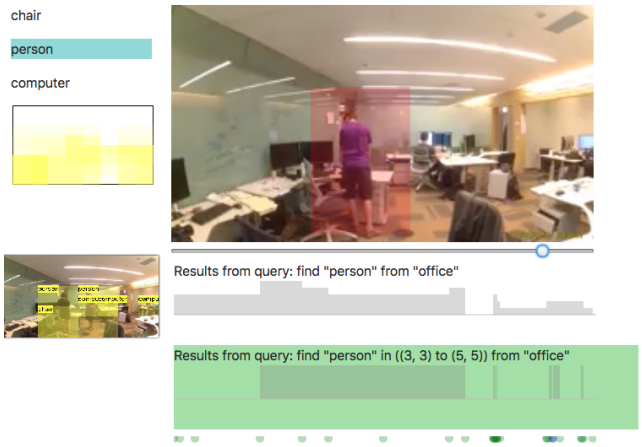

Figure 1: Snapshot of the query interface. Users typically start by seeing a list of labels on the left side to show what labels have been detected in the video. The different regions of the UI are all crosslinked so that when selecting a label, a preview heat map in yellow shows relevant regions in the video where the selected label is found. Timelines are also updated to show the presence of the label over time. The user can subsequently interact either spatially (by drawing a region of interest in the main video shown as a red-transparent rectangle) or with the timelines to show a portion of time as an interesting segment. Each successive interaction can be applied to all the related views helping the user refine the query to achieve the desired outcome.

label is the first step required for diagnosing problems in their query result.

To help users discover unexpected labels, the VQL UI presents **list of labels**, so that they can get an immediate sense of different labels used by having all labels listed. Users can further *contextualize* what a label is by clicking on the label, which queries for segments where the label is present, in components called "timelines" to be discussed.

Now that the homeowner has found the labels, "box" and "book", they need to specify the region. This can be done directly via brushing over the **video player**, which is snapped to the grid. The UI automatically detects the currently active label selected in the list of labels, and forms the query atom with the spatial operator, which

is defaulted to `OVERLAP` with a 0.8 threshold, to allow "fuzzy" specification. We provide additional controls to change the operator.

Sometimes, it is difficult to specify the region if the user does not know where the object will be ahead of time. For instance, the homeowner wanted to query for people being near the package, but her queries have returned no results. To help the user contextualize spatial information of the objects, we introduce **spatial visualization using heat maps** that show the regions where a label has been through the video. Here the homeowner may realize that their selection was too far left. Additionally, in the case the algorithm incorrectly evaluated the bounding box of the label, the heat map can help the user identify the cause of the incorrect query result.

In addition to refining the spatial queries, VQL UI also provides **timelines**, which are **temporal visualization** of the segment sets of each atom and frequency of labels per frame via a histogram, shown in Figure 1. We chose to separate the visualizations by atom because we found that often the query result is incorrect due to the inconsistency or minor errors of the object labels, and seeing the break down of the results help identify the issue faster.

In addition, across interactions, the query results are also pushed to the top of the list, maintaining a full history of the results that the user can drag and drop to reorder. The user can also choose to remove all or individual timelines. The history of user interaction could help users quickly navigate and compare results. Users can perform visually the temporal operators—e.g., compare across timelines to find intersections, unions, ordering, leveraging better domain knowledge such as the duration of events. The homeowner can use the histogram of the count of packages/boxes to discover when the count has increased (delivery), and when the count is decreased (moved or taken), instead of needing to write a more complex temporal query. Similarly, the user can intuitively group segments that are a jagged sequence, not needing to use the `SMOOTH` operator to reapply the query, thus simplifying the interaction.

Timelines are also annotated with the frames labeled by the DNN, filtered by a minimum confidence rate controlled by the user. Hovering over the timeline reveals the corresponding frames, clicking on the timeline starts the video, clicking on the frames show labeled objects, and clicking on a labeled object reveals on the timeline all the other components where the object was labeled, as well as samples of the instances of the images in small multiples.

4.1 Implementation

To implement the described interface, every interaction takes the current state of the UI through a finite state machine, each of whose state then transduces the corresponding query in VQL. As an example, clicking on a label "box" enters a state of the FSM, which generates the query, `FIND "box";`, whose result reactively updates the timeline UI components. Then if the user brushes on a region in the image, the query is now `FIND "box" in (2,3) to (4,7)`, whose result is pushed the top of the list of query results.

4.2 Limitations

One key objective of the project is to help users efficiently make sense of the labels with some error. Therefore it is important to evaluate *how people perceive the error or inconsistencies* with the underlying labeled data. Currently, we only have anecdotal positive feedback about the general usability of the interface. However, a formal user evaluation is needed to understand the effectiveness of the interface at different levels of accuracies of the labeling algorithm.

Furthermore, there are still many other aspects of the design space that is plausible, as well as the space of accuracies—how do user behavior and perception change as the accuracy change from 100% to 90% and 60%? These questions will be important for practitioners who may need to trade off between accuracy and latency/amount of resources.

5 CONCLUSION

This paper proposes both a query language and user interface for querying DNN generated object labels for videos. We identified the challenge of users working with directly machine generated labels due to inaccuracies and inconsistencies. We incorporated techniques from database and information visualization communities to help users make sense of the result.

Future areas of exploration include (1) evaluate how users use this language, and graphical interface and other variations in the design space, (2) characterize the broader implications of errors and inconsistencies produced by DNNs, and (3) bring techniques from data cleaning to further alleviate the challenge of labeling errors.

We hope this project contributes to the broader discussion of how best to leverage the capabilities of users while exploiting emerging technologies.

REFERENCES

[1] 2018. Spatial and Geographic objects for PostgreSQL. (2018). https://postgis.net/ Accessed: 2018-05-01.

[2] Hakan Bilen, Basura Fernando, Efstratios Gavves, and Andrea Vedaldi. 2017. Action recognition with dynamic image networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2017).

[3] Shih-Fu Chang, William Chen, Horace J Meng, Hari Sundaram, and Di Zhong. 1997. VideoQ: an automated content based video search system using visual cues. In *Proceedings of the fifth ACM international conference on Multimedia*. ACM, 313–324.

[4] Luca Deri, Simone Mainardi, and Francesco Fusco. 2012. tsdb: A compressed database for time series. In *International Workshop on Traffic Monitoring and Analysis*. Springer, 143–156.

[5] Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan Russell. 2017. Localizing Moments in Video with Natural Language. *arXiv preprint arXiv:1708.01641* (2017).

[6] Ronghang Hu, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko. 2017. Learning to reason: End-to-end module networks for visual question answering. *CoRR, abs/1704.05526* 3 (2017).

[7] Ronghang Hu, Huazhe Xu, Marcus Rohrbach, Jiashi Feng, Kate Saenko, and Trevor Darrell. 2016. Natural language object retrieval. In *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*. IEEE, 4555–4564.

[8] Edwin L Hutchins, James D Hollan, and Donald A Norman. 1985. Direct manipulation interfaces. *Human-computer interaction* 1, 4 (1985), 311–338.

[9] Eenjun Hwang and VS Subrahmanian. 1996. Querying video libraries. *journal of visual communication and image representation* 7, 1 (1996), 44–60.

[10] influxdata. 2018. The modern engine for Metrics and Events. (2018). https://www.influxdata.com/ Online; Accessed: 2018-05-01.

[11] Justin Johnson, Andrej Karpathy, and Li Fei-Fei. 2016. Densecap: Fully convolutional localization networks for dense captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4565–4574.

[12] Daniel Kang, John Emmons, Firas Abuzaid, Peter Bailis, and Matei Zaharia. 2017. NoScope: Optimizing Neural Network Queries over Video at Scale. *arXiv preprint arXiv:1703.02529* (2017).

[13] Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. 2017. Dense-captioning events in videos. In *Proceedings of the IEEE International Conference on Computer Vision*, Vol. 1. 6.

[14] Thi-Lan Le, Monique Thonnat, Alain Boucher, and François Brémond. 2008. A query language combining object features and semantic events for surveillance

video retrieval. In *International Conference on Multimedia Modeling*. Springer, 307–317.

[15] Zhenyang Li, Ran Tao, Efstratios Gavves, Cees GM Snoek, AW Smeulders, et al. 2017. Tracking by natural language specification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 6495–6503.

[16] Chih-Yao Ma, Asim Kadav, Iain Melvin, Zsolt Kira, Ghassan AlRegib, and Hans Peter Graf. 2017. Attend and Interact: Higher-Order Object Interactions for Video Understanding. *arXiv preprint arXiv:1711.06330* (2017).

[17] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 779–788.

[18] Joseph Redmon and Ali Farhadi. 2017. YOLO9000: better, faster, stronger. *arXiv preprint* (2017).

[19] Ediz Saykol, Ugur Gudukbay, and Ozgur Ulusoy. 2005. A database model for querying visual surveillance videos by integrating semantic and low-level features. *Lecture notes in computer science* 3665 (2005), 163.

[20] Fadime Sener and Angela Yao. 2018. Unsupervised Learning and Segmentation of Complex Activities from Video. *arXiv preprint arXiv:1803.09490* (2018).

[21] Chris Weaver. 2010. Cross-filtered views for multidimensional visual analysis. *IEEE Transactions on Visualization and Computer Graphics* 16, 2 (2010), 192–204.

[22] Haoyu Zhang, Ganesh Ananthanarayanan, Peter Bodik, Matthai Philipose, Paramvir Bahl, and Michael J Freedman. 2017. Live Video Analytics at Scale with Approximation and Delay-Tolerance.. In *NSDI*, Vol. 9. 1.

[23] Zhenxing Zhang, Wei Li, Cathal Gurrin, and Alan F Smeaton. 2016. Faceted navigation for browsing large video collection. In *International Conference on Multimedia Modeling*. Springer, 412–417.