A Framework for Effective Known-item Search in Video

Jakub Lokoč, Gregor Kovalčík, Tomáš Souček, Jaroslav Moravec, Přemysl Čech
SIRET Research Group, Department of Software Engineering
Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic
[lokoc,cech]@ksi.mff.cuni.cz,[gregor.kovalcik,tomas.soucek1]@gmail.com,jaroslav.moravec@centrum.cz

ABSTRACT

Searching for one particular scene in a large video collection (known-item search) represents a challenging task for video retrieval systems. According to the recent results reached at evaluation campaigns, even respected approaches based on machine learning do not help to solve the task easily in many cases. Hence, in addition to effective automatic multimedia annotation and embedding, interactive search is recommended as well. This paper presents a comprehensive description of an interactive video retrieval framework VIRET that successfully participated at several recent evaluation campaigns. Utilized video analysis, feature extraction and retrieval models are detailed as well as several experiments evaluating effectiveness of selected system components. The results of the prototype at the Video Browser Showdown 2019 are highlighted in connection with an analysis of collected query logs. We conclude that the framework comprise a set of effective and efficient models for most of the evaluated known-item search tasks in 1000 hours of video and could serve as a baseline reference approach. The analysis also reveals that the result presentation interface needs improvements for better performance of future VIRET prototypes.

KEYWORDS

Interactive video retrieval, shot boundary detection, deep learning, similarity search, known-item search, video browser showdown

ACM Reference Format:
Jakub Lokoč, Gregor Kovalčík, Tomáš Souček, Jaroslav Moravec, Přemysl Čech. 2019. A Framework for Effective Known-item Search in Video. In *Proceedings of the 27th ACM International Conference on Multimedia (MM '19), October 21–25, 2019, Nice, France.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3343031.3351046

1 INTRODUCTION

Video recording, sharing and watching have become an ordinary activity in our daily life. With a high quantity of watched videos, users start to face problems to rediscover a memorized video scene in a large video archive (e.g., personal collection, channel, browser history). Let us emphasize that such information retrieval task inherently requires 100% recall as one particular scene has to be found. This paper investigates *known-item search* (KIS) tasks, where users search for a previously observed scene (visual KIS) or a described scene (textual KIS) in a given collection. A known-item search task can be solved relatively quickly by a sequential search for a small video collection. However, for hundreds of hours of videos users have to rely on features (search clues) that could be used to find a searched scene more quickly. An example of the ideal search feature is an available unique attribute remembered for the searched scene (e.g., date of observing the scene or video filename). However, once users do not have/remember these attributes, content-based features have to be considered. Generally speaking, a suitable content-based feature for known-item search satisfies jointly the following three properties:

- The feature can be easily recognized, memorized and reproduced by users in a query interface.
- The feature can be automatically detected in raw video data and precisely matched to a corresponding user query.
- The feature is distinct and unique in the dataset.

Thanks to significant improvements in deep learning during last years [15, 16, 20, 23, 37, 39, 42], the effectiveness of various content-based retrieval models for video data increased significantly. Nevertheless, trained feature detectors used by the retrieval models can still fail in some cases and also users usually do not know/remember all details of the searched (more complex) scene to provide a comprehensive query. Hence, the three properties are not always satisfied and so current known-item search video retrieval tools [1, 2, 8, 30, 31, 34] still rely also on interactive search [38] to boost their effectiveness. The tools integrate various information retrieval models [5] and informative visualizations in well-arranged responsive interfaces to let users inspect results and decide between various search strategies.

The current trend followed by the tools is to rely on deep neural networks to generate automatic annotations for keyword search and semantic deep features for modeling image similarity, while various custom models for sketch-based search are supported as well. For example, the *vitrivr* system [31] (winner of VBS 2019) considers a rich set of modalities for query by sketch/example/text search modes. Various deep learning-based detectors are supported, enabling users to search for known items with queries targeting detected concepts, captions, and OCR/ASR data. In the ranked result set, the system enables inspection of context and video playback. As the searched item can still receive a worse rank in a result set, several systems specialize also on effective browsing of larger result sets. For example, Barthel et al. [7, 8] investigated various hierarchical browsing structures using sorted image maps inspired by classical map-based browsing. Sorted image maps are implemented also in the *diveXplore* interactive video retrieval tool [34] that integrates also advanced search options like browsing autopilot or minimap. Both successful browsing systems support also query initialization (keywords, sketches) to initialize the search. For an

MM '19, October 21–25, 2019, Nice, France

ACM ISBN 978-1-4503-6889-6/19/10…$15.00
https://doi.org/10.1145/3343031.3351046

overview of other related tools, we refer the readers to recent Video Browser Showdown (VBS) summary papers [12, 25, 26].

Although interactive known-item search systems often share similar approaches, the VIRET framework [27, 28] detailed in this paper differs in several aspects from the systems that currently participated at VBS. For example, the framework uses an own temporal segmentation approach used for frame selection and filtering (see Section 2). An own limited set of pre-selected classes from ImageNet is considered for automatic annotation (see feature extraction in Section 3) and a convenient simple color/semantic sketch approach is used, enabling ALL/ANY specification. In addition, users can formulate multi-modal temporal queries enabling rich specification of searched scenes (see Section 4). According to our analysis, the ability to specify advanced queries with the selected set of supported models often enabled users to get the searched frame on the first page at VBS 2019.

The main contributions of the paper can be summarized as

- A comprehensive description of currently used features, models and building blocks of a known-item search framework VIRET that regularly and successfully participates at the Video Browser Showdown [12, 25] and Lifelog Search Challenge [17]. Hence, VIRET can serve as a reference approach.
- A new effective and efficient shot boundary detection deep network TransNet [36] for common shot transitions.
- The results of the most recent version of the VIRET prototype at the Video Browser Showdown 2019 accompanied with a detailed log analysis (discussed in Section 6).

2 REPRESENTATIVE FRAME SELECTION

The framework comprises retrieval models that operate just on a set $\mathcal{S}$ of selected representative frames from all videos. This section summarizes steps of the (key) frame selection process used for the last participation at the Video Browser Showdown 2019. The process follows standard steps, where the video is divided to shots and then representative frames are selected [22].

In the very first step, all frames from each video are extracted and videos that are inside a (single color) static border are additionally cropped. For further processing, all frames are resized to the resolution 48×27. Based on this much smaller representation, a frame selection method was designed as a combination of a simple shot boundary detection model based on 3D deep convolutional neural networks and a histogram based frame selection heuristic. The overall frame selection pipeline is presented in Figure 1.

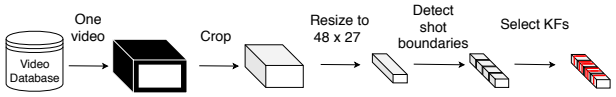

Figure 1: Representative frame selection pipeline.

2.1 Employed shot boundary detection

In order to detect shot boundaries, we present a simple 3D convolutional neural network called *TransNet*[1] (Figure 2). The TransNet

[1]The source code is available at https://github.com/soCzech/TransNet.

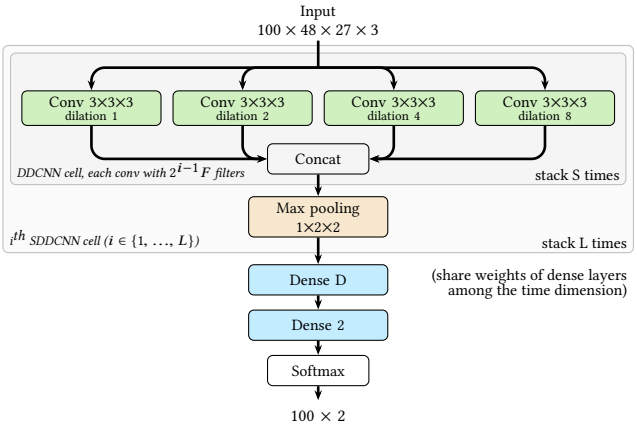

Figure 2: A schema outlining the TransNet shot boundary detection network architecture for $S = 1$ and $L = 1$.

architecture considers a sequence of N resized video frames on input and employs 3D convolutions as recently investigated by Gygli [18]. The network provides a prediction for each input frame, estimating how likely a given frame forms a shot boundary. The content of this section was adapted from our arxiv version [36], where additional details are available.

The main building block of the proposed model (Dilated DCNN cell) comprise four 3D 3×3×3 convolutional operations. The convolution operations differ in dilation rates for the time dimension and their outputs are concatenated in the channel dimension. Multiple DDCNN cells on top of each other followed by spatial max pooling then form a more complex Stacked DDCNN block. The TransNet can consist of multiple SDDCNN blocks, every next block operating on smaller spatial resolution but a greater channel dimension. Please note that using multiple SDDCNN blocks further increases depth of the network and the receptive field used for predictions. Compared to the work of Gygli, which uses only standard 3D convolutions, our approach significantly increases field of view of the convolutional layers while maintaining relatively low number of trainable parameters.

In addition, two dense layers refine the features extracted by the convolutional layers and predict a possible shot boundary for every frame representation independently (layers' weights are shared). Except the last dense layer with softmax output, ReLU activation function is used in all remaining layers. All convolutional layers use stride 1 and the 'same' padding.

For training, TRECVID IACC.3 dataset [4] was used since it has a predefined master shot boundary reference, which can be used to automatically generate transitions by joining different shots with a transition type. Specifically, 3000 videos were randomly selected from the IACC.3 dataset and for each video, shots were selected as follows:

(1) too short shots with less than 5 frames were removed
(2) and then only odd shots were selected to exclude a potentially very similar consecutive shot.

Overall, 54884 shots were selected. Generating transitions between two frame sequences from randomly selected shots, inspired

by [18], enables to generate training data for a network on demand. Two common types of transitions were considered for training: hard cuts and dissolves. The length of each training sequence was set to $N = 100$ frames and the spatial resolution of input frames was set to 48×27 pixels. For validation, 100 videos from the IACC.3 dataset were chosen and about 3800 shots were manually labeled. The RAI dataset [6] was used as a test set. Please note that during validation and testing only predictions for frames 25-75 are used due to incomplete temporal information for the first/last frames.

A grid search was performed over the following architecture meta-parameters:

(1) S, the number of DDCNN cells in a SDDCNN layer,
(2) L, the number of SDDCNN layers,
(3) F, the number of filters in the first set of DDCNN layers (doubled in each following SDDCNN layer),
(4) D, the number of neurons in the dense layer.

All investigated network variants were trained with batch size of 20 and for only 30 epochs, each with 300 batches, to prevent overfitting to the training dataset. We experimented with dropout and saw no improvement in results, however maybe more advanced forms of regularization or further dataset augmentation could improve network's generalization abilities. Adam optimizer [24] with the default learning rate 0.001 and cross entropy loss function was used for the training.

The average F1 score of inspected models on the validation dataset (top epoch was selected for each configuration) and the result of this selected network on test RAI dataset are presented in Figure 3. During inference, a frame is marked as a shot boundary if the network predicts the boundary with at least 10% confidence. Based on the results, the model that has 16 filters in the first layer, two stacked DDCNN cells in three layers and with 256 neurons in the dense layer (F=16, L=3 S=2, D=256) was selected and used in the frame selection pipeline. As presented in Table 1, the score 0.94 of the selected model on RAI dataset corresponds to the state-of-the-art value reported by Hassanien et al. [19], who proposed an order of magnitude larger network with more than 40 times as many parameters trained for a rich set of transition types.

The lower F1 score 0.73 for the validation dataset corresponds in part to our observation that the videos contain candidates for transitions that are ambiguous even for human annotators (e.g., changing subtitles). However, also the new network relying on

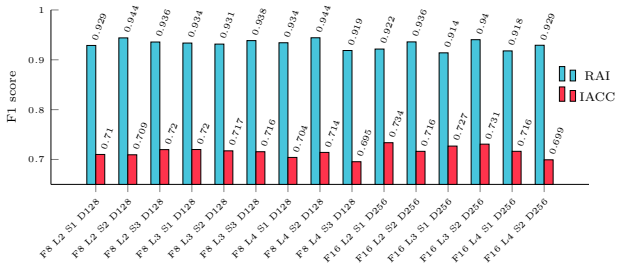

Figure 3: Observed average F1 scores of tested networks for the validation and test datasets. Shot detection pipeline did not use the cropping step for this evaluation.

Baraldi et al.	Gygli	Hassanien et al.	ours
0.84 [6]	0.88 [18]	0.94 [19]	0.91 - 0.94

Table 1: Average F1 scores for the RAI test dataset.

simple features still does not generalize well in specific situations. After an inspection of several videos, we have identified both false positives (e.g., in highly dynamic shots) and false negatives (e.g., in gradual transitions). Nevertheless, the simple fast deep detection approach[2] is mostly used as a frame selection preprocessing step complementary to the color-based features used in the last step of frame selection. The approximate knowledge of shot boundaries was used also for a presentation filter limiting the number of top ranked selected frames from one shot. This might be a problem with many false shot boundary negatives, or for too long shots with variable contents. However, the filters were effective in most searches for the V3C dataset according to our observations presented in Section 6.

2.2 Employed frame selection

Once a video is divided into (approximate) shots, a simple classical frame selection heuristic can be considered to select representative frames (for a survey of frame selection methods see [22]). In the current version, only uniformly sampled shot frames are picked and then a similarity threshold approach is used to cluster the sampled frames in each shot. As a similarity model, the cosine similarity is used for quantized color histograms computed from resized sampled frames ($48 \times 27 \times 3$), resulting in $6 \times 6 \times 6 \times 12$ dimensions for the uniform grid of twelve (4×3) regions in the frames. The employed hierarchical clustering process consisted of the following steps:

(1) every frame is initially a cluster, where clusters are temporally ordered according to frame numbers
(2) while there are some consecutive clusters represented by histograms that have cosine similarity higher than a given threshold (0.78 was used), find two such clusters with the highest cosine similarity, merge them and compute the new mean histogram representing the new cluster
(3) uniformly subdivide every cluster with more than 15 frames, so finally each cluster has less than 15 frames (seconds)

As a result, the middle frame of each cluster was selected as a representative frame.

For VBS 2019, the implementation of the frame selection process skipped some very short shots. The approximation was caused by employed global uniform sampling of video frames picking just one frame per second. As an undesired consequence, a few salient frames were missing for the task V_4 starting with a dynamic sequence of very short shots. Hence, for future prototypes we plan to guarantee at least one selected frame from each detected shot. Let us note that VIRET users solved the task V_4 anyway (see Section 6).

[2]It took just 50s to detect shot boundaries of preprocessed frames from the whole RAI dataset (about 98 minutes of video) using Tesla V100 GPU.

3 FEATURE EXTRACTION

This section presents an overview of considered feature extraction approaches $f_{extract} : \mathcal{S} \rightarrow \mathbb{R}^n$ for the set of selected frames. The extraction pipeline employs a convolutional neural network (CNN) architecture for image annotation and deep feature extraction. So far, the VIRET framework uses state-of-the-art image classification networks that can be easily trained with available large training datasets [13, 33]. Let us note that the framework supports also several simple features for filtering (e.g., for grayscale images) that are not detailed in this paper.

3.1 Automatic annotation

Commonly, the image classification networks are trained using the ILSVRC competition dataset. However, the dataset provides just a limited dictionary for known-item search purposes. Therefore, VIRET considers an own set created from 6641 image classes selected from ImageNet database [13]. First, the selected image classes were mapped to a vector space using $\mathcal{M} : class \rightarrow \mathbb{R}^n$. Then, k-means clustering algorithm was used to create clusters of semantically similar classes and representative classes of each cluster were manually selected to be part of the new train dataset. The mapping $\mathcal{M}(c)$ of a class c is defined as

$$\mathcal{M}(c) = \mathbb{E}_{\boldsymbol{x} \sim p_c} f(\boldsymbol{x}; \boldsymbol{\theta}) \approx \frac{1}{|S_c|} \sum_{\boldsymbol{x} \in S_c} f(\boldsymbol{x}; \boldsymbol{\theta}),$$

where $\boldsymbol{x}$ are images from a distribution of the given class p_c and $f(\cdot, \boldsymbol{\theta})$ is an arbitrary mapping of an image to $\mathbb{R}^n$ parameterized by $\boldsymbol{\theta}$. An image-classification CNN with the last layer removed was used as the function f. The expected value is approximated by a set S_c of images belonging to the class c from ImageNet database. Given the described method, 1150 classes[3] were selected, later enriched by another distinct 93 classes taken from Places2 dataset.

A standard procedure is used to retrain the state-of-the-art NasNet convolutional neural network [42]. First, publicly available weights[4] are used to initialize the network and only the last layer is retrained to prevent destroying learned features with initially random gradients. Further, the whole network is fine-tuned yielding Top-1/Top-5 accuracy 62.5/88.1 for the NasNet Mobile version and 68.5/92.2 for the NasNet Large version.

Using the retrained network, each $o \in \mathcal{S}$ was assigned confidence scores $\xi\ (label_j, o)$ from the computed network output (each $class_j$ has assigned $label_j$). In order to decrease storage/memory costs, only scores higher than a small constant were stored. In addition, a WORDNET [29] based hierarchy of hypernyms of the labels was utilized to further extend the supported vocabulary and also provide additional relations between the basic 1243 labels. For relevance scoring (see Section 4.1), each hypernym is transformed to the set of descendant class labels.

3.2 Deep features

In order to search by an example image for semantically similar images, an embedding from image space to a vector space is frequently used. For example, neuron activations of a deep neural network layer represent a popular generic representation for similarity search [14], because similar representations (e.g., considering cosine similarity) often correspond to semantically related concepts. For the current version of the VIRET prototype, the vector of activation features from the last pooling layer of a retrained NasNet Mobile network is used. For the training, the classes for automatic annotation were used.

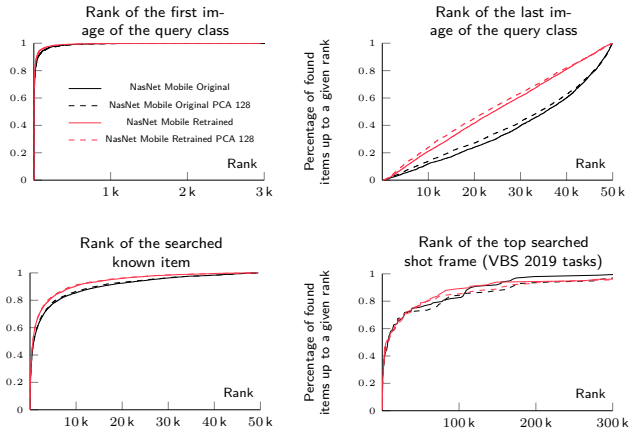

Figure 4: Simulated searches for the first/last image of a query class and known-item. A real test is presented in the bottom right graph with X-axis limited at 30% |DB|.

The choice was motivated by simulations performed on a test collection of 499 randomly sampled ImageNet classes[5], each containing 100 sampled images. The objective of the comparison was to inspect the performance at known-item search or ad-hoc search tasks evaluated for example at VBS [25]. For the experiment, pairs of images were randomly selected from each class. The first object from the pair represents the searched "known" image, while the second represents an actually available query object (selected to target the searched item). Considering the dataset sorted with respect to a query, the rank/position of the first/last image of the query class and the searched known image are collected for 4990 queries (10 from each class). The cumulative percentage of found images up to a given rank on the X axis is presented in Figure 4. In the simulations, the retrained models provided better results for similarity search. However, known-item search based comparison of the models for the real 45 query example images (extended by additional 54 images selected afterwards) used to search for the VBS 2019 tasks did not confirm the findings from known-item search simulations. In other words, there was no clear difference between features from the original and retrained NasNet Mobile and so the results of simulations might signalize a specific property of ImageNet classes and the distribution of considered features.

3.3 Representations for sketch-based models

For color based sketch retrieval, the VIRET prototype employs small thumbnails (resolution 26×15), where the colors in the thumbnails

[3] The list of the selected classes is provided in [35].
[4] Available on github.com/tensorflow.
[5] The set of 6641 classes, that was considered in selection process described in Section 3.1, was excluded.

are transformed to the CIE Lab color space. The resulting representation vector O_S consists of 390 uniformly positioned colored points.

The sketch canvas can be used also for localization of face or text bounding boxes extracted by state-of-the-art networks for text [41] and face [21] detection, both of which have implementations available online. For each selected frame and type (face or text), a bit mask was created and stored.

4 RETRIEVAL MODELS

This section summarizes implemented models for ranking of selected frames based on a query. After the feature extraction, each selected frame $o \in \mathcal{S}$ has a precomputed representation for keyword search, color-sketch search, query by example image and filtering by localized face/text in the sketch. For each modality, users can issue queries to compute ranking scores for each frame. We remember employed relevance score functions presented already for the Lifelog Search Challenge version [28] (except a minor modification) and summarize also fusion approaches. Please note that the set of employed models was selected to enable intuitive means of query formulation, highly efficient and easy implementation of query processing for one million frames (desired property of a reference system), and to consider a limited working memory of users (e.g., relying just on simple sketches). Although the currently used set performed well at the Video Browser Showdown (see Section 6), we plan further fine-tuning of the models and additional comparative evaluations with advanced simulations and real users.

4.1 Employed relevance score models

For keyword search, VIRET provides a prompting text box to construct sets of supported labels $N_i \subseteq \{label_1, \ldots, label_{|L|}\}$. The overall query is represented as $Q_K = \{N_i\}_{i=1}^{k}$ for k provided sets. The relevance score $r\_keyword(Q_K, o)$ is computed for each frame $o \in \mathcal{S}$ as

$$\prod_{\forall N_i \in Q_K} \left(\sum_{\forall label_j \in N_i} \xi\left(label_j, o\right) \right).$$

The color-sketch retrieval option is provided by an interactive color-sketch canvas, where each sketch can be constructed/edited as a finite set of colored ellipses $Q_S = \{(q_i, a_i, b_i, t_i)\}_{i=1}^{k}$, where k is the number of positioned colored centers $q_i \in R^5$ and $a_i, b_i \in R_0^+$ specify major/minor axes of the ellipse-shaped query region. An additional parameter $t_i \in \{ALL, ANY\}$ specifies whether ALL pixels inside the query region are required to have a selected color or just the pixel with the most similar color is considered in the region. The parameter provides users an option to search for color regions with certain or uncertain position[6]. Given that each database frame is represented as a set $O_S = \{o_j\}_{j=1}^{26 \cdot 15}$ of uniformly positioned colored points $o_j \in R^5$, the relevance score $r\_sketch(Q_S, O_S)$ of a database object O_S with respect to a query sketch Q_S is computed as

$$-\Bigg(\sum_{\substack{\forall (q_i, a_i, b_i, t_i) \in Q_S \\ t_i = ANY}} MIN_{\forall o_j \in O_i} \left(L_2^c\left(q_i, o_j\right)\right) + \sum_{\substack{\forall (q_i, a_i, b_i, t_i) \in Q_S \\ t_i = ALL}} AVG_{\forall o_j \in O_i} \left(L_2^c\left(q_i, o_j\right)\right) \Bigg),$$

where the Euclidean distance $L_2^c(q_i, o_j)$ is evaluated just for the color dimensions of q_i, o_j and $O_i = \{o | o \in O_S \wedge (q_i.x - o.x)^2/a_i^2 + (q_i.y - o.y)^2/b_i^2 \leq 1\}$. To prefer larger color regions, more ellipses of the same color can be used.

Considering browsing in the result set of a previous query or an external search engine, users can pick a suitable image as a query example (or can pick a set of example images). Given the corresponding set of k query feature vectors $Q_E = \{v_i^q\}_{i=1}^{k}$ and a database object feature vector v^o, the relevance score for v^o is defined as

$$r\_example\left(Q_E, v^o\right) = \sum_{\forall v_i^q \in Q_E} \sigma\left(v_i^q, v^o\right),$$

where the Cosine similarity is used as effective and efficient similarity function σ.

In a similar way as users can flexibly target colors using the ALL/ANY specification, the sketching canvas enables also to insert ellipse queries localizing/targeting an appearance of face bounding boxes in a searched frame. For ALL option, the corresponding score function for one query ellipse returns 1 only if all pixels in the ellipse belong to a bounding box of a face; otherwise the score is 0. For ANY option, the corresponding score function for one query ellipse returns 1 only if at least one pixel in the ellipse belongs to a bounding box; otherwise the score is 0. The overall score used for filtering is computed as the logical *AND* for all query ellipses. Addressing regions without faces can be easily implemented by inverting bounding box bit masks. Searching for text bounding boxes is supported in the same way.

4.2 Employed fusion strategies

Multi-modal fusion strategies [3, 11] represent a popular option to boost retrieval effectiveness. Depending on the current query, various fusion approaches can be considered by the VIRET framework.

In order to describe/target content changes in a temporal sequence of frames, users can enter a temporal query (temporally ordered inputs) for a particular modality (e.g., as used for sketch-based search at VBS [10, 40]). VIRET currently supports a sequence of two queries. For example, users can enter $Q_{K_1} = \{\{CANYON\}\}$ and $Q_{K_2} = \{\{BRIDGE\}\}$ to address a frame showing a canyon followed by a frame with a bridge. The employed temporal fusion model for keyword search evaluates $score_{o_i}^{K_1} = r\_keyword(Q_{K_1}, o_i)$ and $score_{o_i}^{K_2} = r\_keyword(Q_{K_2}, o_i)$ for each selected frame. Then, the overall score after fusion is computed as

$$score_{o_i}^{K_1, K_2} = score_{o_i}^{K_1} \cdot \max_{j=i+1 \ldots i+c} (score_{o_j}^{K_2}),$$

where c determines the size of the temporal context and is flexibly adapted at the end of each video. Frames without any available temporal context (either the context frames were previously filtered or there is no additional frame at the end of a video) are still included in the result set, but their $score_{o_i}^{K_1}$ is multiplied with a small constant. In addition, users can select which query from the sequence is primary (i.e., frames of which query are displayed). The presented

[6] For example, for moving objects users do not know which frames are selected.

formula is for the case where Q_{K_1} is the primary query. In case Q_{K_2} is chosen as the primary query, the fusion formula is defined as

$$score_{o_i}^{K_2,K_1} = score_{o_i}^{K_2} \cdot \max_{j=i-c\ldots i-1}(score_{o_j}^{K_1}),$$

where selected frames at the beginning of each video are treated analogically to the last frames for the Q_{K_1} primary query.

Temporal models are defined in a similar way for other modalities, considering different combination functions (instead of multiplication). Specifically, addition is used for color-based and query by example image models, while logical AND is used for localized object models (face/text bounding boxes). For frames without temporal context and addition combination function, $score_{o_i}^{Q_2}$ is currently used as the second term. For the logical AND combination function, value 0 is used as the second multiplier if a temporal context frame is not available (e.g., is filtered). Let us also note that the aggregation should work even if one of the two queries is empty for a potential late fusion phase presented in the following paragraph. Hence, each relevance score function has a proper implicit value for an empty/undefined query.

Given relevance scores of frames for a query modality (including temporal fusion), a configurable threshold is used to obtain just top ranked frames for each modality. For a multi-modal query, the intersection of such sets is computed and the result is sorted by a selected modality (asymmetric late fusion suggested in [11]). This phase includes also other utilized filtering options. Finally, presentation filters limiting the number of displayed frames from one video and shot are applied.

5 PROTOTYPE INTERFACE

The VIRET tool prototype interface (see Figure 2 in [27]) conforms to a classical image based retrieval system. The left panel consists of query formulation input components for employed retrieval models, while the right grid presents the result of a (combined) query sorted by relevance. The temporal aspects of videos are considered in query formulation, where users can describe features of two consecutive frames. In addition, each frame in the result set enables fast inspection of its temporal video context using mouse wheel over the frame. With every rotation, the sequence of frames is played (forward or backward) and at the same time, on the right, a stripe panel presents selected frames from a more distant neighborhood. For each frame, it is also possible to perform query-by-example image action or quick jump to other grid panels.

Each frame in the result set represents a direct link to its video source using a "Video" button. The button enables users to display a new form with all temporally ordered representative frames from the video, with focus on the part containing the selected frame. More specifically, the form shows two grids. The left grid presents all representative frames from the displayed video, while the right grid presents a video summary for fast insight and also quick navigation. The "Map" button presents the very same form, but instead of video frames, the grid contains a reorganized prefix of the current result set (inspired by maps investigated by Barthel et al. [7, 9]).

The VIRET tool can be also integrated with a web-based external image search engine. In order to face an initial ideal query gap, users can browse an external web image search engine and intuitively drag&drop a selected image to the sketch panel. This action starts corresponding feature extraction of the query object for the underlying $r_{example}$ relevance score function. According to our observations, the workflow is easy and intuitive also for novice users, especially given two screens (one dedicated for the external image search engine).

6 VIRET FRAMEWORK AT VBS 2019

Recently, the VIRET framework prototype tool participated at the Video Browser Showdown at MMM 2019 in Thessaloniki. The competition used a subset of the new V3C dataset [32] comprising 1000 hours of video content. At the competition, teams compete in known-item and ad-hoc search tasks in the same room and highly competitive way [12, 25]. Regarding known-item search, 10 visual KIS tasks and 8 textual KIS tasks were evaluated by expert users (authors of the tools), while 5 visual KIS tasks (a subset of the expert tasks) were evaluated by novice users randomly selected from the audience. Each expert team consisted of two users controlling the tool and one assistant, while novice teams consisted just of two users. Since the organizers of the competition published the collected database of logs from the VBS server, we present also a performance analysis of the VIRET tool at the competition. Following, observed submission times are presented in connection with the analysis of VIRET tool performance during the live evaluation of known-item search tasks. Please note that a thorough comparison analyzing also the logs of other participating tools is out of the scope of this paper and is planned for a VBS 2019 analysis paper.

6.1 Performance at KIS tasks

Figure 5 presents all evaluated KIS tasks as vertical dashed lines. The tasks are presented in the same order as they were evaluated at the competition. The bottom horizontal line represents the start of each task (i.e., the server counter was activated), while the top horizontal line the time limit of each task. The time of correct submissions of participating teams is marked on the line with respect to the start of each task. At VBS 2019, other five systems participated – vitrivr [31], VIREO [30], VISIONE [1], diveXplore [34], VERGE [2].

Considering the number of successfully solved tasks, the expert VIRET users were able to solve 100% of the visual KIS tasks within five minutes and 75% of textual KIS tasks within eight minutes. Altogether, the expert VIRET users solved sixteen KIS tasks[7] out of eighteen (89% !) in 1000 hours of video. In the novice visual KIS session, however, only two tasks were solved despite the fact that the searched items appeared several times on the first page (see the log analysis in the following section). Please note that the novice teams did not have a third member, who could consult the search strategy and observe displayed results. We conclude that the involved video analysis, selected features and retrieval models enable users to effectively target known-items purely based on visual information, but the presentation interface has to be analyzed and improved for future installments of VBS.

Considering the time to solve a task by the VIRET prototype tool, frequent browsing interactions and (re)constructions of a multimodal query often led to a longer average time for solved KIS

[7] For the presentation settings of KIS tasks please see the recent VBS survey [25].

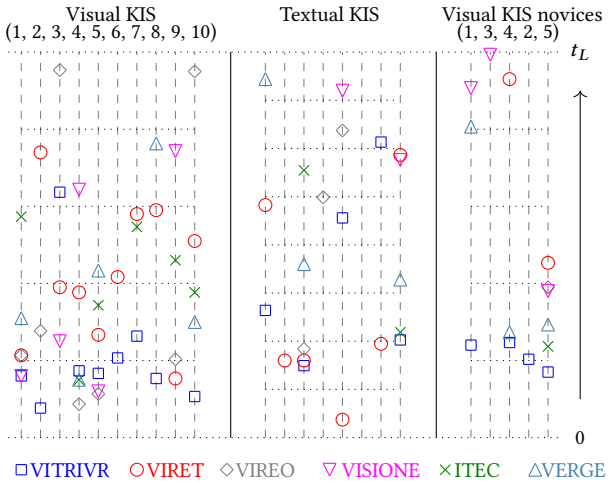

Figure 5: The time elapsed until a correct submission was received from a tool at VBS 2019 in visual and textual KIS tasks. The time limit t_L was set to 5 minutes for visual KIS tasks and 8 minutes for textual KIS tasks. The visual KIS tasks for novices are the same (except their order) as the first five visual KIS tasks for experts.

tasks[8], except cases when an ideal query object was found at the very beginning of the search and users registered the searched frame in the display. A higher average search time was observed also for other teams and their solved KIS tasks, except the overall winning vitrivr team. According to the authors of the tool, extracted OCR/ASR data[9] were often used to solve visual KIS tasks. In the current VBS settings for visual KIS tasks (the scene is played in the loop with blurring [25]), both expert and novice users were often able to recognize English spoken words or texts in presented scenes, and type them as an ideal query object to target the searched scene. The observed high frequency of speech/text clues in the V3C dataset and the current effectiveness of the ASR/OCR models used by the vitrivr team [31] highly motivate to include these features to future versions of the VIRET framework.

6.2 Log analysis

Provided the knowledge of a task scene (including task start timestamp) and also collected query logs by our tool, it is possible to reconstruct the position of the top ranked selected frame from the 20 second long task scene. If all the selected frames from the searched scene were filtered, at least the top ranked frame from the task video can be tracked. For both participating VIRET tool instances[10] PC1 and PC2, Table 2 presents detected first occurrences of a searched scene frame on the first page during VBS task evaluation. Please note that the appearance on the first page does not mean that the user registers the frame. The detected rank is presented in connection with its search time, while the task submission time is presented if the user submitted a correct frame. In addition, the table shows also the top achieved rank of a searched scene frame if it did not appear on the first page. If all the searched scene frames were filtered during the search, the top ranked frame position from the searched video is presented, marked by "(vid)".

We may observe that VIRET tool users were able to pull up a frame from the searched scene to the first page in more than 50% searches, provided that the whole team stopped the search once the task was solved by one of the tool instances PC1 or PC2. Both instances were successful, each reaching about 50% of solved tasks. Please note that PC2 relied solely on automatically extracted annotations and features, while PC1 often relied also on external images found on Google Images. Altogether, only in 7 cases out of 46, the selected frames of the searched scene were filtered out during the team search time (which does not always equal to the whole task time limit). Nevertheless, in two cases (V_3^n PC1 and V_3 PC2) the found frame from the correct video was at rank 0/1 which has led to inspection of the video.

TID	$rank_1$	t_1	t_1^s		$rank_2$	t_2	t_2^s
V_1^n	44	96s			3	92s	
V_3^n	(vid) 0	299s			1804	53s	
V_4^n	1	243s	279s		2259	3s	
V_2^n	336	70s			46	107s	
V_5^n	42	120s	136s		13	10s	
V_1	(vid) 5	28s			62	16s	64s
V_2	168	215s			33	165s	222s
V_3	(vid) 125	24s			(vid) 1	106s	117s
V_4	25	95s	113s		434	6s	
V_5	2	58s	80s		377	24s	
V_6	160	110s	125s		471	119s	
V_7	237	20s			37	20s	174s
V_8	135	167s	177s		1798	118s	
V_9	51	27s	46s		84	23s	
V_{10}	88	110s			84	141s	153s
T_1	49	35s			49	19s	290s
T_2	5	14s	96s		5	6s	
T_3	35	22s			18	10s	96s
T_4	(vid) 93	18s			7800	417s	
T_5	(vid) 78	12s			8	11s	23s
T_6	38	249s			2334	267s	
T_7	44	22s	117s		139	34s	
T_8	(vid) 8	252s			69	19s	352s
	PC1, 140 frames / page				PC2, 88 frames / page		

Table 2: Detected first occurrences of a searched scene frame on the first page during VBS task evaluation, its rank, corresponding search time t and submission time (t^s) if a tool instance submitted a correct frame. Gray color is used to present detected best ranks of searched frames in searches, where users did not achieve to get the frames on the first page. If all searched scene frames were filtered, gray color depicts the top reached position of a searched video frame (the frame could reveal the correct video).

Table 2 reveals interesting "overlook gaps" $og = t^s - t$ between the first occurrence of a searched scene frame on the first page in time t and the time of the correct submission t^s. We may observe long gaps in textual KIS tasks (e.g., T_8 PC2, $og = 333s$) and even in visual KIS tasks (e.g., V_7 PC2, $og = 154s$).

[8]VIRET expert users needed on average 127s for solved visual KIS tasks and 162s for solved textual KIS tasks.
[9]https://cloud.google.com/speech-to-text/ and https://cloud.google.com/vision/
[10]Each VBS team could use two notebooks with an instance of their tool.

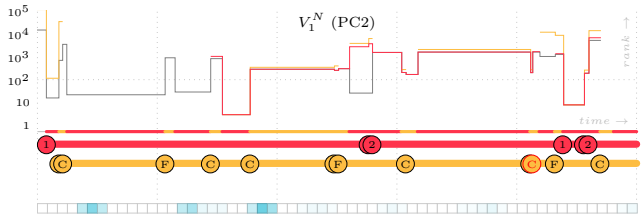

Figure 6: Actions and actual position of searched video/scene frames for one known-item search task.

In Figure 6, an augmented graph-based visualization of performed known-item search actions is presented to detail the V_1^N task at PC2. The x-axis presents the time from the task start, while y-axis (log scale is used) shows the actual rank of the top ranked frame from the searched video (gray line) and the searched scene (red line), considering the presentation filter. For a searched frame, the potential top ranked position without the presentation filter is depicted too (orange line). In addition, currently used queries are presented under the x-axis in the form of intervals, including the keyword model (red color), used color/semantic sketch (orange color, C = colors, F = faces), and also temporal variants (presented as the second line with the same color). Example images were not used in this task by the user. Below, a heatmap is presented for logged browsing interactions. For the lack of space, additional query options (e.g., change of filters) are not visualized and so it may happen that the current displayed rank of the searched video/scene "suddenly" changes. The presented lines do not have to be continuous in cases the searched video/scene frame is filtered out or the rank becomes higher than the Y-axis limit.

We may observe that in task V_1^N the novice user of PC2 combined keyword search and color/face sketches. A frame from the searched scene was twice on the first page (first time even at position 3), yet the frame was not registered and the task was not solved[11]. According to our analysis [27], in about 80% of cases when the searched frame appeared on the first page, it was based on a temporal and/or multi-modal query and additional filters. The VIRET tool users often employed also various browsing features (especially temporal context inspection). We may conclude that iterative query (re)formulation of more complex queries and interactive browsing represent a promising strategy for effective known-item search.

6.3 Summary

The VIRET framework prototype achieved a competitive performance at the Video Browser Showdown 2019, given a limited set of retrieval models. Even though the system enabled to solve a promising number of known-item search tasks in 1000 hours of video, the log analysis also shown that there is a room for improvements. The following list summarizes lessons learned.

- The presentation of results in a grid is a significant factor for known-item search process. Since human perception is limited, about 20-50 images are usually considered for one page [7]. On the other hand, a grid with more images enables to observe a larger portion of the result set at once, which was the strategy of the VIRET tool at VBS 2019. However, as many searched items were obviously overlooked, the results presentation needs to be more thoroughly analyzed.
- An optimal known-item search workflow requires more investigation, as the users can miss searched frames even during the query construction phase. Even though the users can start with an ideal query item, they may promptly continue to enter additional query items without an inspection of the result set. Since the new items could be noisy for the overall ranking, the search process could be significantly prolonged.
- It is still necessary to search for additional complementary models to extend KIS frameworks, following the three properties of suitable content-based features mentioned in the introduction. For example, proposed visual concepts/features are still not sufficient in terms of known-item search speed, once ASR/OCR clues are present and effectively detected in the searched scene and recognized/memorized by users.
- The currently used retrieval models need further investigation to improve their effectiveness. For example, for keyword search we would like to compare currently used simple automatic annotation method based on selected classes with more advanced approaches for automatic annotation. We also plan to investigate different transformation functions for the output of employed deep classification networks.

[11] Please note that it was for the first time the novice users tried the system and that this setting could be changed for future VBS events.

7 CONCLUSIONS

The paper summarizes a frame-based video retrieval framework VIRET that successfully participated at recent interactive video retrieval competitions. The effectiveness of currently considered models was presented in the context of known-item search tasks evaluated at the Video Browser Showdown 2019. Based on the results and comparison with participating systems, we conclude that the presented set of models is competitive and effective for known-item search tasks. As such, the set of models could represent a baseline approach for comparative known-item search evaluations. In the future, we plan to extend the framework by ASR/OCR approaches and more effective result visualization methods. We also plan to focus on more effective search models, workflows and further inspect shot detection and frame selection approaches [22].

ACKNOWLEDGMENTS

This paper has been supported by Czech Science Foundation (GAČR) project Nr. 19-22071Y, Charles University project PROGRES Q48, and SVV-260451. We also thank the VBS organizers, all participating teams, and Dr. Bernd Muenzer for his support of the VBS server.

REFERENCES

[1] Giuseppe Amato, Paolo Bolettieri, Fabio Carrara, Franca Debole, Fabrizio Falchi, Claudio Gennaro, Lucia Vadicamo, and Claudio Vairo. 2019. VISIONE at VBS2019. In *MultiMedia Modeling - 25th International Conference, MMM 2019, Thessaloniki, Greece, January 8-11, 2019, Proceedings, Part II*. 591–596. https://doi.org/10.1007/978-3-030-05716-9_51

[2] Stelios Andreadis, Anastasia Moumtzidou, Damianos Galanopoulos, Foteini Markatopoulou, Konstantinos Apostolidis, Thanassis Mavropoulos, Ilias Gialampoukidis, Stefanos Vrochidis, Vasileios Mezaris, Ioannis Kompatsiaris, and Ioannis Patras. 2019. VERGE in VBS 2019. In *MultiMedia Modeling - 25th International Conference, MMM 2019, Thessaloniki, Greece, January 8-11, 2019, Proceedings, Part II*. 602–608. https://doi.org/10.1007/978-3-030-05716-9_53

[3] Pradeep K. Atrey, M. Anwar Hossain, Abdulmotaleb El Saddik, and Mohan S. Kankanhalli. 2010. Multimodal fusion for multimedia analysis: a survey. *Multimedia Systems* 16, 6 (01 Nov 2010), 345–379. https://doi.org/10.1007/s00530-010-0182-0

[4] George Awad, Asad Butt, Jonathan Fiscus, Martial Michel, David Joy, Wessel Kraaij, Alan F. Smeaton, Georges Quénot, Maria Eskevich, Roeland Ordelman, Gareth J. F. Jones, and Benoit Huet. 2017. TRECVID 2017: Evaluating Ad-hoc and Instance Video Search, Events Detection, Video Captioning and Hyperlinking. In *Proceedings of TRECVID 2017*. NIST, USA.

[5] Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. 2011. *Modern Information Retrieval - the concepts and technology behind search, Second edition*. Pearson Education Ltd., Harlow, England.

[6] Lorenzo Baraldi, Costantino Grana, and Rita Cucchiara. 2015. Shot and Scene Detection via Hierarchical Clustering for Re-using Broadcast Video. In *Computer Analysis of Images and Patterns*, George Azzopardi and Nicolai Petkov (Eds.). Springer International Publishing, Cham, 801–811.

[7] Kai Uwe Barthel and Nico Hezel. 2019. Visually exploring millions of images using image maps and graphs. In *Big Data Analytics for Large-scale Multimedia Search*, Benoit Huet, Stefanos Vrochidis, and Edward Chang (Eds.). John Wiley and Sons Inc., 251–275.

[8] Kai Uwe Barthel, Nico Hezel, and Klaus Jung. 2018. Fusing Keyword Search and Visual Exploration for Untagged Videos. In *MultiMedia Modeling*, Klaus Schoeffmann, Thanarat H. Chalidabhongse, Chong Wah Ngo, Supavadee Aramvith, Noel E. O'Connor, Yo-Sung Ho, Moncef Gabbouj, and Ahmed Elgammal (Eds.). Springer International Publishing, Cham, 413–418.

[9] Kai Uwe Barthel, Nico Hezel, and Radek Mackowiak. 2015. ImageMap - Visually Browsing Millions of Images. In *MultiMedia Modeling - 21st International Conference, MMM 2015, Sydney, NSW, Australia, January 5-7, 2015, Proceedings, Part II*. 287–290. https://doi.org/10.1007/978-3-319-14442-9_30

[10] Adam Blažek, Jakub Lokoč, and Tomáš Skopal. 2014. Video Retrieval with Feature Signature Sketches. In *Similarity Search and Applications*, Agma Juci Machado Traina, Caetano Traina, and Robson Leonardo Ferreira Cordeiro (Eds.). Springer International Publishing, Cham, 25–36.

[11] Petra Budíková, Michal Batko, and Pavel Zezula. 2017. Fusion Strategies for Large-Scale Multi-modal Image Retrieval. *T. Large-Scale Data- and Knowledge-Centered Systems* 33 (2017), 146–184. https://doi.org/10.1007/978-3-662-55696-2_5

[12] Claudiu Cobârzan, Klaus Schoeffmann, Werner Bailer, Wolfgang Hürst, Adam Blazek, Jakub Lokoc, Stefanos Vrochidis, Kai Uwe Barthel, and Luca Rossetto. 2017. Interactive video search tools: a detailed analysis of the video browser showdown 2015. *Multimedia Tools Appl.* 76, 4 (2017), 5539–5571. https://doi.org/10.1007/s11042-016-3661-2

[13] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 248–255. https://doi.org/10.1109/CVPR.2009.5206848

[14] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. 2014. DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32 (ICML'14)*. JMLR.org, 647–655.

[15] R. Girdhar, D. Ramanan, A. Gupta, J. Sivic, and B. Russell. 2017. ActionVLAD: Learning Spatio-Temporal Aggregation for Action Classification. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 3165–3174. https://doi.org/10.1109/CVPR.2017.337

[16] Albert Gordo, Jon Almazán, Jerome Revaud, and Diane Larlus. 2016. Deep Image Retrieval: Learning Global Representations for Image Search. In *Computer Vision – ECCV 2016*, Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling (Eds.). Springer International Publishing, Cham, 241–257.

[17] Cathal Gurrin, Klaus Schoeffmann, Hideo Joho, Andreas Leibetseder, Liting Zhou, Aaron Duane, Duc-Tien Dang-Nguyen, Michael Riegler, Luca Piras, Minh-Triet Tran, Jakub Lokoč, and Wolfgang Hürst. 2019. [Invited papers] Comparing Approaches to Interactive Lifelog Search at the Lifelog Search Challenge (LSC2018). *ITE Transactions on Media Technology and Applications* 7, 2 (2019), 46–59. https://doi.org/10.3169/mta.7.46

[18] Michael Gygli. 2018. Ridiculously Fast Shot Boundary Detection with Fully Convolutional Neural Networks. In *2018 International Conference on Content-Based Multimedia Indexing, CBMI 2018, La Rochelle, France, September 4-6, 2018*. 1–4. https://doi.org/10.1109/CBMI.2018.8516556

[19] Ahmed Hassanien, Mohamed A. Elgharib, Ahmed Selim, Mohamed Hefeeda, and Wojciech Matusik. 2017. Large-scale, Fast and Accurate Shot Boundary Detection through Spatio-temporal Convolutional Neural Networks. *CoRR* abs/1705.03281 (2017). arXiv:1705.03281 http://arxiv.org/abs/1705.03281

[20] K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 770–778. https://doi.org/10.1109/CVPR.2016.90

[21] Peiyun Hu and Deva Ramanan. 2016. Finding Tiny Faces. *CoRR* abs/1612.04402 (2016). arXiv:1612.04402 http://arxiv.org/abs/1612.04402

[22] Weiming Hu, Nianhua Xie, Li Li, Xianglin Zeng, and S. Maybank. 2011. A Survey on Visual Content-Based Video Indexing and Retrieval. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* 41, 6 (Nov 2011), 797–819. https://doi.org/10.1109/TSMCC.2011.2109710

[23] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*. 448–456.

[24] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014). arXiv:1412.6980 http://arxiv.org/abs/1412.6980

[25] Jakub Lokoč, Werner Bailer, Klaus Schoeffmann, Bernd Münzer, and George Awad. 2018. On Influential Trends in Interactive Video Retrieval: Video Browser Showdown 2015-2017. *IEEE Trans. Multimedia* 20, 12 (2018), 3361–3376. https://doi.org/10.1109/TMM.2018.2830110

[26] Jakub Lokoč, Gregor Kovalčík, Bernd Münzer, Klaus Schöffmann, Werner Bailer, Ralph Gasser, Stefanos Vrochidis, Phuong Anh Nguyen, Sitapa Rujikietgumjorn, and Kai Uwe Barthel. 2019. Interactive Search or Sequential Browsing? A Detailed Analysis of the Video Browser Showdown 2018. *ACM Trans. Multimedia Comput. Commun. Appl.* 15, 1, Article 29 (Feb. 2019), 18 pages. https://doi.org/10.1145/3295663

[27] Jakub Lokoč, Gregor Kovalčík, Tomáš Souček, Jaroslav Moravec, and Přemysl Čech. 2019. VIRET: A Video Retrieval Tool for Interactive Known-item Search. In *Proceedings of the 2019 on International Conference on Multimedia Retrieval (ICMR '19)*. ACM, New York, NY, USA, 177–181. https://doi.org/10.1145/3323873.3325034

[28] Jakub Lokoč, Tomáš Souček, and Gregor Kovalčík. 2018. Using an Interactive Video Retrieval Tool for LifeLog Data. In *Proceedings of the 2018 ACM Workshop on The Lifelog Search Challenge, LSC@ICMR 2018, Yokohama, Japan, June 11, 2018*. 15–19. https://doi.org/10.1145/3210539.3210543

[29] George A. Miller. 1995. WordNet: A Lexical Database for English. *Commun. ACM* 38, 11 (Nov. 1995), 39–41. https://doi.org/10.1145/219717.219748

[30] Phuong Anh Nguyen, Chong-Wah Ngo, Danny Francis, and Benoit Huet. 2019. VIREO @ Video Browser Showdown 2019. In *MultiMedia Modeling - 25th International Conference, MMM 2019, Thessaloniki, Greece, January 8-11, 2019, Proceedings, Part II*. 609–615. https://doi.org/10.1007/978-3-030-05716-9_54

[31] Luca Rossetto, Mahnaz Amiri Parian, Ralph Gasser, Ivan Giangreco, Silvan Heller, and Heiko Schuldt. 2019. Deep Learning-Based Concept Detection in vitrivr. In *MultiMedia Modeling - 25th International Conference, MMM 2019, Thessaloniki, Greece, January 8-11, 2019, Proceedings, Part II*. 616–621. https://doi.org/10.1007/978-3-030-05716-9_55

[32] Luca Rossetto, Heiko Schuldt, George Awad, and Asad A. Butt. 2019. V3C - A Research Video Collection. In *MultiMedia Modeling - 25th International Conference, MMM 2019, Thessaloniki, Greece, January 8-11, 2019, Proceedings, Part I*. 349–360. https://doi.org/10.1007/978-3-030-05710-7_29

[33] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision* 115, 3 (01 Dec 2015), 211–252.

[34] Klaus Schoeffmann, Bernd Münzer, Andreas Leibetseder, Jürgen Primus, and Sabrina Kletz. 2019. Autopiloting Feature Maps: The Deep Interactive Video Exploration (diveXplore) System at VBS2019. In *MultiMedia Modeling - 25th International Conference, MMM 2019, Thessaloniki, Greece, January 8-11, 2019, Proceedings, Part II*. 585–590. https://doi.org/10.1007/978-3-030-05716-9_50

[35] Tomáš Souček. 2018. Known-item search in image datasets using automatically detected keywords, bachelor thesis.

[36] Tomáš Souček, Jaroslav Moravec, and Jakub Lokoč. 2019. TransNet: A deep network for fast detection of common shot transitions. *CoRR* abs/1906.03363 (2019). arXiv:1906.03363 http://arxiv.org/abs/1906.03363

[37] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*. 1–9.

[38] Bart Thomee and Michael S. Lew. 2012. Interactive search in image retrieval: a survey. *International Journal of Multimedia Information Retrieval* 1, 2 (01 Jul 2012), 71–86. https://doi.org/10.1007/s13735-012-0014-4

[39] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2017. Show and Tell: Lessons Learned from the 2015 MSCOCO Image Captioning Challenge. *IEEE Trans. Pattern Anal. Mach. Intell.* 39, 4 (2017), 652–663.

[40] Jin Yuan, Huanbo Luan, Dejun Hou, Han Zhang, Yan-Tao Zheng, Zheng-Jun Zha, and Tat-Seng Chua. 2012. Video Browser Showdown by NUS. In *Advances in Multimedia Modeling*, Klaus Schoeffmann, Bernard Merialdo, Alexander G. Hauptmann, Chong-Wah Ngo, Yiannis Andreopoulos, and Christian Breiteneder (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 642–645.

[41] X. Zhou, C. Yao, H. Wen, Y. Wang, S. Zhou, W. He, and J. Liang. 2017. EAST: An Efficient and Accurate Scene Text Detector. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2642–2651. https://doi.org/10.1109/CVPR.2017.283

[42] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. 2017. Learning Transferable Architectures for Scalable Image Recognition. *CoRR* abs/1707.07012 (2017). arXiv:1707.07012 http://arxiv.org/abs/1707.07012