PraK Tool: An Interactive Search Tool Based on Video Data Services

Jakub Lokoč[1], Zuzana Vopálková[1], Michael Stroh[2], Raphael Buchmueller[3], and Udo Schlegel[3]

[1] SIRET Research Group, Department of Software Engineering
Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic
jakub.lokoc@matfyz.cuni.cz

[2] Visual Computing, Department of Computer Science
Faculty of Natural Sciences, Konstanz University, Konstanz, Germany
michael.stroh@uni-konstanz.de

[3] Data Analysis and Visualization, Department of Computer Science
Faculty of Natural Sciences, Konstanz University, Konstanz, Germany
raphael.buchmueller,u.schlegel@uni-konstanz.de

Abstract. This paper presents a tool relying on data service architecture, where technical details of all VBS datasets are completely hidden behind an abstract stateless data layer. The data services allow independent development of interactive search interfaces and refinement techniques, which is demonstrated by a smart front-end component. The component supports common search features and allows users to exploit content-based statistics for effective filtering. We believe that video data services might be a valuable addition to the open-source VBS toolkit, especially when available for the competition on a shared server with all VBS datasets, extracted features, and meta-data behind.

Keywords: interactive video retrieval, deep features, text-image retrieval

1 Introduction

The Video Browser Showdown (VBS) [14,6] is a well-established event for comparison of interactive video search systems. Every year, new findings and insights are revealed at the competition, which affects the research and implementation of new video search prototypes. For example, joint embedding approaches based on Open CLIP trained with LAION-2B [13,7] turned out to be highly competitive at the VBS competition in 2023.

For VBS 2024, we present a new system created in cooperation between Konstanz University and Charles University, Prague. The system currently relies on extracted keyframes and features from the VISIONE [1] system but can be easily extended with novel feature extraction approaches. On top of the directory with the provided meta-data, we created a stateless application component called video data services. Our ambition is to propagate the idea that teams can participate at VBS even without the need to download, process, and maintain huge volumes of data from VBS datasets (V3C [16], MVK [19]).

In the following, we review available data access options of the front-ends of selected top VBS systems. The VIREO system [11] performs ranking operations in their back-end engine. However, the front-end application still requires indexed data (videos, thumbnails) locally to start searching. The Vibro system [18] is a standalone java application where all data has to be loaded into RAM for each instance. Similarly, both VIRET [10] and CVHunter [9] are WPF .NET applications that require to load all meta-data to RAM, while thumbnail images are stored and accessed locally from disk. Only text to joint-embedding vector transformation requires an online service for CVHunter and later versions of VIRET. The SOMHunter system [8] used Node.js containing SOMHunter core library that performed computations and maintained state. The latest version of SOMHunter core used an HTTP API. However, the core library maintained its state and did not return feature vectors to the front-end. The vitrivr system [17] relies on a three-tier architecture, where the data layer is hidden behind the retrieval engine accessible by a REST API or a WebSocket. The engine allows to run queries and returns items with some meta-data but does not return features to the front-end. The VISIONE system [1] has a retrieval engine accessible via a REST API as well; however, CLIP features cannot be directly attached to ranked result sets. The VERGE system [12] uses a web service for obtaining a ranked list of image names, but no features are attached to this list. We conclude that many systems use HTTP calls for ranked result sets; however, the option to obtain a complete set of video frames and their features for a more complex front-end application is indeed not common in regularly participating VBS systems.

2 System Description

Our video search system consists of video data services providing access to subsets of VBS video datasets and a smart front-end component for refinement of the subsets. We emphasize that the services are stateless, meaning it is up to front-end components to maintain the state in more complex search scenarios over data subsets. The video data services provide an abstract interface independent of used video data sources, and thus, front-end applications can remain unchanged (except configuration or parameter tuning) in case new joint embedding models or datasets are prepared for VBS. The current version of the services uses meta-data provided by the VISIONE system [1,2,3], where selected representative keyframes are accompanied by extracted features from CLIP based models [13,7] trained on LAION-2B. However, the meta-data feature set can be easily extended by the most recent joint-embedding models released before VBS 2024. The only change in the front-end application would be a new value of an existing parameter of the video data service interface. The overall architecture of the presented video search system is depicted in Figure 1.

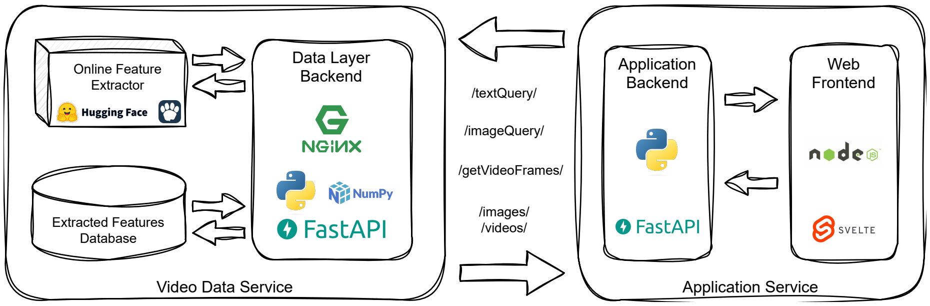


Fig. 1. Architecture of the presented video search system based on video data services. In the current version, the Application Backend provides just a web server functionality.

2.1 Video Data Services

The idea behind video data services is to provide access to video data, either one particular video or selected video frames with some properties (e.g., frames related to a text query). From this perspective, video data services combine a general data-sharing platform and retrieval engine available on the internet. Based on findings from previous VBS events, we identified four essential video data services for VBS search scenarios:

- *textQuery(text, k, dataset, model, addFeatures)* returning a list of the k most relevant representative video frames to a text query, cosine distance is assumed for vectors of a specified joint embedding model, temporal query option is allowed as well using > symbol in the text parameter
- *imageQuery(image, k, dataset, model, addFeatures)* returning a list of the k most relevant representative video frames to an example image query (any image can be used), cosine distance is assumed for vectors of a specified joint embedding model
- *getVideoFrames(itemID, k, dataset, model, addFeatures)* returning a list of k temporally preceding/subsequent representative frames to itemID from the same video, for $k = 0$ all representative frames are returned for the video
- *getVideo(itemID, dataset)* returning URI of a given video

The list of returned video frames is a JSON file with a sorted array of items. The items are sorted based on rank or time information for the list of video items. Each item contains a video frame thumbnail URI, item rank/time, cosine distance with respect to the query, item ID for the VBS server (for item submission), and optionally, video frame feature vector and detected classes. Since the last two fields can significantly expand the volume of each item (from several bytes to kilobytes), these fields are returned only if parameter $addFeatures = true$ is specified. Please note that a lightweight option $addFeatures = false$ leads to limited front-end functionality (e.g., no relevance feedback actions). On the other hand, both options can be combined to obtain quickly list of images and then list of corresponding features in a background process. Since the number of supported features could vary in the future, this parameter could be changed to a binary based encoding of requested features (e.g., "1010 ..." for first and third feature set).

After each video services call, a data chunk of k items is returned to a front-end that allows users to refine the result set interactively. Although primarily designed for video search applications, this architecture can provide access to various types of data for interactive search refinement. For example, the system can be used for interactive search experiments utilizing data visualizations based on Chernoff faces [4].

For all teams using video data services, a result log json file can be created and stored in the video data services back-end as long as a userID parameter is added to the interface.

2.2 Front-end Search Options and Interface

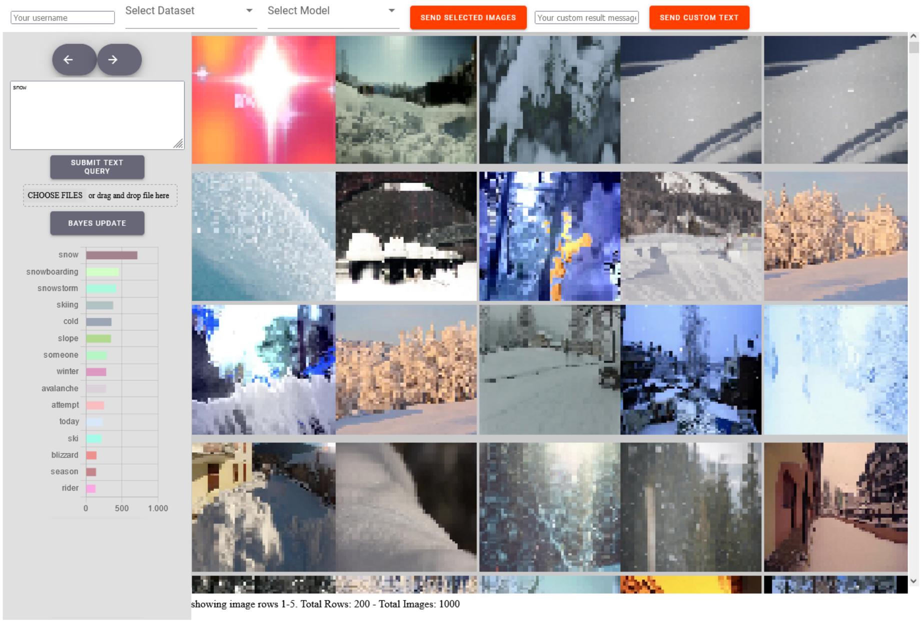
The front-end component is responsible for the refinement of subsets obtained by video data services. We first recapitulate common features of successful systems and then describe a new feature helpful in specific search scenarios.

Assuming a ranked list of video frames obtained by a primary ranking model, a list of common search options [15,6] that our tool supports comprise:

- Two types of result set visualization techniques – grid of ranked images and per-line visualization of ranked items with their temporal video context
- Text query reformulation interface as frequent query reformulation is indeed a popular search option
- Example image query interface, including images from external sources (e.g., as was used by VIRET [10])
- Presentation filters allowing to show only given $k \in \{1, 2, ..., 9\}$ most relevant items from each video (e.g., as was used in CVHunter [9])
- Bayesian relevance feedback reformulation [5] (e.g., used by SOMHunter [8]), where the score is updated only for items available for the front-end, joint-embedding features are necessary

In addition to these popular features, we plan to add a novel feature relying on content-based statistics of the available subsets on front-end. Specifically, we assume that each data item has a list of detected classes (e.g., using zero-shot CLIP classification) or that the classes correspond to centroids obtained by an unsupervised clustering technique (e.g., k-means). With these classes and corresponding class confidence scores, it is possible to add interactive search features like informative statistic charts or label-selection-based filters/rankers. Mainly, label-based filters/rankers represent a novel approach at VBS (to the best of our knowledge) that combines automatically detected content-based statistics (e.g., classes that can equally divide the current set into two subsets) and user knowledge of the searched target, which affects filtering/re-ranking decisions.

The current interface of our tool is illustrated on Figure 2.



snow

showing image rows 1-5. Total Rows: 200 - Total Images: 1000

Fig. 2. The Front-end of the presented video search system based on video data services. In the top panel, users can select dataset, feature representation model and submit custom text for question answering tasks. The left panel allows users to run text queries or use external image to initialize the search. Bayes update button is available for selected images as well. Bellow the button, statistics of detected class labels are presented for the current result set. Users can click on each class label to set it as filter.

3 Conclusions

In the paper, we present an example of a VBS system relying on stateless video data services and a smart front-end, allowing interactive refinement of actual result sets. The prototype supports common search features popular at VBS and investigates a novel approach based on detected classes in video frames. For future VBS events, it would be a convenient option to run such video data services for (new) VBS teams that would like to focus more on front-ends with result-set refinement and HCI techniques. The data processing burden would be significantly lower, while the available feature set could be sufficiently competitive. Especially if the joint-embedding features are up-to-date. At the same time, result logs could be created automatically for the teams.

Acknowledgment

This work has been supported by Czech Science Foundation (GAČR) project 22-21696S. We would like to thank the authors of the discussed VBS systems for clarification of architecture details.

References

1. Amato, G., Bolettieri, P., Carrara, F., Falchi, F., Gennaro, C., Messina, N., Vadicamo, L., Vairo, C.: Visione at video browser showdown 2023. In: International Conference on Multimedia Modeling. pp. 615–621. Springer (2023)
2. Amato, G., Bolettieri, P., Carrara, F., Falchi, F., Gennaro, C., Messina, N., Vadicamo, L., Vairo, C.: VISIONE Feature Repository for VBS: Multi-Modal Features and Detected Objects from MVK Dataset (Sep 2023). https://doi.org/10.5281/zenodo.8355037, `https://doi.org/10.5281/zenodo.8355037`
3. Amato, G., Bolettieri, P., Carrara, F., Falchi, F., Gennaro, C., Messina, N., Vadicamo, L., Vairo, C.: VISIONE Feature Repository for VBS: Multi-Modal Features and Detected Objects from V3C1+V3C2 Dataset (Jul 2023). https://doi.org/10.5281/zenodo.8188570, `https://doi.org/10.5281/zenodo.8188570`
4. Chernoff, H.: The use of faces to represent points in k-dimensional space graphically. Journal of the American Statistical Association **68**(342), 361–368 (1973). https://doi.org/10.1080/01621459.1973.10482434
5. Cox, I.J., Miller, M.L., Minka, T.P., Papathomas, T.V., Yianilos, P.N.: The bayesian image retrieval system, pichunter: theory, implementation, and psychophysical experiments. IEEE transactions on image processing **9**(1), 20–37 (2000)
6. Heller, S., Gsteiger, V., Bailer, W., Gurrin, C., Jónsson, B.., Lokoc, J., Leibetseder, A., Mejzlík, F., Peska, L., Rossetto, L., Schall, K., Schoeffmann, K., Schuldt, H., Spiess, F., Tran, L., Vadicamo, L., Veselý, P., Vrochidis, S., Wu, J.: Interactive video retrieval evaluation at a distance: comparing sixteen interactive video search systems in a remote setting at the 10th video browser showdown. Int. J. Multim. Inf. Retr. **11**(1), 1–18 (2022). https://doi.org/10.1007/s13735-021-00225-2, `https://doi.org/10.1007/s13735-021-00225-2`

7. Ilharco, G., Wortsman, M., Wightman, R., Gordon, C., Carlini, N., Taori, R., Dave, A., Shankar, V., Namkoong, H., Miller, J., Hajishirzi, H., Farhadi, A., Schmidt, L.: Openclip (Jul 2021). https://doi.org/10.5281/zenodo.5143773, `https://doi.org/10.5281/zenodo.5143773`, if you use this software, please cite it as below.
8. Kratochvíl, M., Mejzlík, F., Veselý, P., Souček, T., Lokoč, J.: Somhunter: lightweight video search system with som-guided relevance feedback. In: Proceedings of the 28th ACM International Conference on Multimedia. MM '20, ACM (2020), in press
9. Lokoč, J., Mejzlík, F., Souček, T., Dokoupil, P., Peška, L.: Video search with context-aware ranker and relevance feedback. In: ór Jónsson, B., Gurrin, C., Tran, M.T., Dang-Nguyen, D.T., Hu, A.M.C., Huynh Thi Thanh, B., Huet, B. (eds.) MultiMedia Modeling. pp. 505–510. Springer International Publishing, Cham (2022)
10. Lokoč, J., Kovalčík, G., Souček, T., Moravec, J., Čech, P.: A framework for effective known-item search in video. In: In Proceedings of the 27th ACM International Conference on Multimedia (MM'19), October 21-25, 2019, Nice, France. pp. 1–9 (2019), `https://doi.org/10.1145/3343031.3351046`
11. Ma, Z., Wu, J., Loo, W., Ngo, C.W.: Reinforcement learning enhanced pichunter for interactive search. In: Conference on Multimedia Modeling (2023)
12. Pantelidis, N., Andreadis, S., Pegia, M., Moumtzidou, A., Galanopoulos, D., Apostolidis, K., Touska, D., Gkountakos, K., Gialampoukidis, I., Vrochidis, S., Mezaris, V., Kompatsiaris, I.: VERGE in vbs 2023. In: International Conference on Multimedia Modeling. pp. 658—-664. Springer (2023), `https://doi.org/10.1007/978-3-031-27077-2_55`
13. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision. CoRR **abs/2103.00020** (2021), `https://arxiv.org/abs/2103.00020`
14. Rossetto, L., Gasser, R., Lokoč, J., Bailer, W., Schoeffmann, K., Muenzer, B., Souček, T., Nguyen, P.A., Bolettieri, P., Leibetseder, A., Vrochidis, S.: Interactive video retrieval in the age of deep learning - detailed evaluation of vbs 2019. IEEE Transactions on Multimedia (2020)
15. Rossetto, L., Gasser, R., Lokoč, J., Bailer, W., Schoeffmann, K., Muenzer, B., Souček, T., Nguyen, P.A., Bolettieri, P., Leibetseder, A., et al.: Interactive video retrieval in the age of deep learning–detailed evaluation of VBS 2019. IEEE Transactions on Multimedia **23**, 243–256 (2020), `https://doi.org/10.1109/TMM.2020.2980944`
16. Rossetto, L., Schuldt, H., Awad, G., Butt, A.A.: V3C - A research video collection. In: MultiMedia Modeling - 25th International Conference, MMM 2019, Thessaloniki, Greece, January 8-11, 2019, Proceedings, Part I. pp. 349–360 (2019). https://doi.org/10.1007/978-3-030-05710-7_29
17. Sauter, L., Gasser, R., Heller, S., Rossetto, L., Saladin, C., Spiess, F., Schuldt, H.: Exploring effective interactive text-based video search in vitrivr. In: Dang-Nguyen, D., Gurrin, C., Larson, M.A., Smeaton, A.F., Rudinac, S., Dao, M., Trattner, C., Chen, P. (eds.) MultiMedia Modeling - 29th International Conference, MMM 2023, Bergen, Norway, January 9-12, 2023, Proceedings, Part I. Lecture Notes in Computer Science, vol. 13833, pp. 646–651. Springer (2023). https://doi.org/10.1007/978-3-031-27077-2_53, `https://doi.org/10.1007/978-3-031-27077-2_53`

18. Schall, K., Hezel, N., Jung, K., Barthel, K.U.: Vibro: Video browsing with semantic and visual image embeddings. In: Dang-Nguyen, D.T., Gurrin, C., Larson, M., Smeaton, A.F., Rudinac, S., Dao, M.S., Trattner, C., Chen, P. (eds.) MultiMedia Modeling. pp. 665–670. Springer International Publishing, Cham (2023)
19. Truong, Q.T., Vu, T.A., Ha, T.S., Lokoč, J., Tim, Y.H.W., Joneja, A., Yeung, S.K.: Marine video kit: A new marine video dataset for content-based analysis and retrieval. In: MultiMedia Modeling - 29th International Conference, MMM 2023, Bergen, Norway, January 9-12, 2023. Lecture Notes in Computer Science, Springer (2023)