Retrieval of Structured and Unstructured Data with vitrivr

Luca Rossetto
University of Basel
luca.rossetto@unibas.ch

Ralph Gasser
University of Basel
ralph.gasser@unibas.ch

Silvan Heller
University of Basel
silvan.heller@unibas.ch

Mahnaz Amiri Parian
University of Basel
University of Mons, Belgium
mahnaz.amiriparian@unibas.ch

Heiko Schuldt
University of Basel
heiko.schuldt@unibas.ch

ABSTRACT

With the increase in sensory capability of mobile devices, the data that can be generated and used in a lifelogging context gets increasingly diverse. Such data is special in the context of multimedia, not only because of its close personal relationship with its originator, but also because of its diverse multimodality and its composition from structured, semi-structured, and unstructured data. This diversity poses retrieval challenges that are unique to lifelog data but which also have implications for retrieval activity in other multimedia domains.

In this paper, we present the extensions made to the vitrivr open-source multimedia retrieval stack, in order to address some of these unique lifelogging challenges. For the participation to the 2019 Lifelog Search Challenge (LSC), we have extended vitrivr with the capability to process Boolean query expressions alongside content-based query descriptions in order to leverage the structural diversity inherent to lifelog data.

CCS CONCEPTS

• **Information systems** → *Information retrieval*; *Users and interactive retrieval*; *Multimedia and multimodal retrieval*; Information retrieval query processing;

KEYWORDS

Content-based Retrieval, Multimedia Retrieval, Lifelogging, Lifelog Search Challenge

ACM Reference Format:
Luca Rossetto, Ralph Gasser, Silvan Heller, Mahnaz Amiri Parian, and Heiko Schuldt. 2019. Retrieval of Structured and Unstructured Data with vitrivr. In *Lifelog Search Challenge'19 (LSC'19), June 10–13, 2019, Ottawa, ON, Canada.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3326460.3329160

LSC'19, June 10–13, 2019, Ottawa, ON, Canada

ACM ISBN 978-1-4503-6781-3/19/06…$15.00
https://doi.org/10.1145/3326460.3329160

1 INTRODUCTION

Lifelogging – sometimes also referred to as *Personal Big Data* – is an emerging phenomenon by which people systematically record various aspects of their everyday lives. Depending on the purpose of the logging activity, the captured data may range from images shot in a regular interval (e.g., from the perspective of a person-mounted camera) to time-series stemming from sensors such as a heart-rate monitor. But also daily blogs or diaries may qualify as some form of lifelog data. The activity of lifelogging therefore produces a huge amount of very heterogeneous data that potentially involves multiple modalities. Often, lifelogs serve to augment human memory as they can be used to rekindle personal events and experiences that lie in the past. From a data management and information retrieval perspective, lifelogging poses a series of very interesting challenges. It bears the question, how this type of heterogeneous data can be organized and stored efficiently. More importantly, however, it remains an open challenge to allow for efficient retrieval of specific items from a lifetime worth of data. The Lifelog Search Challenge (LSC) tries to tackle this challenge head-on, by organizing a setting in which competing teams and their tools try to find particular entries in a dataset in as little time as possible. The dataset comprises of one month of anonymized multimedia lifelog data involving continuous capture images, locations, biometrics and information consumption/creation activities [12].

The main challenge for information retrieval in lifelog collections lies in the heterogeneity of the data. A typical query might look something like this: "I ate a Burger in a diner in walking distance from where I work. It was a Wednesday around noon. I went jogging before. What was the name of that diner again?" If we unpack this information need, we come to realize that it contains a lot of different components. For example, the consumed burger might be visible on an image taken by a camera or a mobile phone. The image may have a time stamp that can be used to query for the time and day of the week and it may even come with location information. Furthermore, there might be some heart rate data that indicates heightened physical activity but there may also just be a diary entry mentioning the jogging that morning. All these components must then be combined in order to find events that match the initial description and ultimately infer the name of the diner from some location data. Hence, in order to satisfy such an information need, classical *Boolean retrieval* must be combined with similarity based information retrieval techniques found in text and image retrieval.

The need for combining *Boolean search* typically found in classical databases and the *vector space model* as applied in multimedia retrieval, was investigated and discussed in [8]. Based on that work,

we have built vitrivr [20], a modular multimedia information retrieval stack. vitrivr is a long-time participant to, and the winner of the 2017 and 2019 [21] installments of, the Video Browser Showdown (VBS) [2]. In the following sections, we will describe the most recent changes to vitrivr that were added in order to be able to tackle the specific challenges posed by the LSC competition. Furthermore, we will demonstrate how vitrivr's modular architecture and the combination of various modes of information retrieval makes the stack very well suited towards a wide range of different information retrieval problems, including but not limited to the settings found in LSC and VBS.

The remainder of this paper is structured as follows: Section 2 gives an overview of vitrivr's system architecture. Section 3 outlines the recent changes made to vitrivr in order to be able to accommodate the challenges posed by LSC. Section 4 briefly surveys related work and Section 5 concludes and describes potential future work.

2 VITRIVR

This section provides an overview of the existing functionality of the vitrivr stack while Section 3 outlines the additions made with lifelog retrieval support in mind.

2.1 General Overview / Architecture

vitrivr [20] is a content-based multimedia retrieval stack with support for several different types of media [7], including images, video, audio, and 3D models. It enables users to search in such mixed-media collections using various query modes, such as, Query-by-Sketch (QbS) with both visual and semantic representations and Query-by-Example (QbE) using external example documents from all supported media domains as well as any combination of the above. The vitrivr stack is comprised of three primary components: (i) the database that persistently stores all (meta)-information required for retrieval, (ii) the retrieval engine that handles query processing as well as feature transformations for both the offline data extraction and online retrieval stage and (iii) the user interface, which provides user facing functionality including but not limited to query formulation and results presentation.

For the database, we use ADAM$_{pro}$[9] when dealing with large volumes of content, due to its support for data distribution as well as various index structures for nearest-neighbor queries. For smaller collections, other data storage methods can be employed. Those may offer less functionality but also reduce the overall system complexity. The retrieval engine Cineast [19] takes care of query processing. Cineast offers a multitude of feature transformations provided by means of so called *feature modules*. From the built-in feature modules, a systems operator can select those that fit their particular need. Furthermore, it is very simple to implement and add new feature modules in order to support new applications. The constellation of features to be used is configurable, which enables a user to tailor a specific deployment of Cineast to the structure of the available data and the concrete use-cases. The user interface Vitrivr NG is a browser-based application implemented in Angular[1] and Typescript. It offers support for the formulation of all the various supported query modes as well as several result views that present retrieved results in slightly different ways. Query formulation is organized around the concept of *query containers* that in turn consist of one to many *query terms*. Each query term captures a particular modality (e.g., visual or text) and allows the user for formulate a query for that modality (e.g., by drawing an image or entering some text). Predicates for the different modalities in the same query container will be connected by a logical AND in the late fusion process. By configuring multiple query containers, logical OR relationships can be expressed as well. Once again, the use of the available query terms and the available result views is configurable in order to be able to easily tailor the UI to the application at hand.

All components of the vitrivr stack are released as open-source software and are available for download from their respective repositories[2]. Vitrivr has also participated to the 2016 and 2018 installments of Google Summer of Code (GSoC) and many interesting projects have been realized on top of it.

2.2 Existing Functionality for Lifelog Retrieval

Due to its successful participation to multiple instances of the VBS [18, 21], vitrivr is well equipped to handle retrieval challenges in a competitive setting. While the datasets used in the two challenges are quite different [12, 23], one of the three task types used in VBS – the *Textual Known-Item Search (KIS)* task – seems to be nearly identical to the setup found during the LSC. Therefore, we expect many of the applied and successful approaches for textual KIS tasks to be applicable to the LSC case as well.

On the data processing side, the modular architecture of Cineast allows us to specifically select feature modules that fit the data. We run the image data through a series of deep neural networks, that generate object classes and image captions, detect screen text (OCR) and even recognized actions performed by people depicted in the images. Of course, we also generate well established visual features for sketch-based or example-based image retrieval as we believe those may prove useful during the competition. The underlying database natively supports storage of more traditional data such as steps, calories, weight or blood pressure. However, we need to extend our data model in order to model the exact relationship between these data points and the individual images / events / days. More details on that will be provided in Section 3.1

Most of the data extracted from or provided with the data set can be directly queried from the existing version of Vitrivr NG. This includes but is not limited to queries for visual content through Query-by-Example (QbE) or Query-by-Sketch (QbS) and textual search for class labels, captions, song titles, or other units of information that can be represented by text. We expect semantic QbS to be particularly interesting, whenever a sufficiently detailed visual description is available in the query. However, the current iteration of the UI lacks support for the formulation of more complex Boolean queries (e.g., range queries), which is an addition to vitrivr, as outlined in Section 3.2. We expect the existing presentation of the results to be well suited for the LSC competition and do not plan any changes on that front.

To summarize, we can state that there will be minor additions and changes to the vitrivr stack in light of the LSC competition. However, all things considered, the existing version is already very

[1] https://angular.io/

[2] https://github.com/vitrivr

well suited towards the requirements of the setup, which is surprising given, that it was not built with that goal in mind.

3 NEW FUNCTIONALITY FOR LSC

This section outlines the additions made to vitrivr specifically to support retrieval of the diverse lifelog data found during the LSC.

3.1 New media type

The visual part of the data provided in the LSC dataset [12] consists of sequences of images that were taken at regular time intervals throughout a day. Since neither the existing vitrivr support for images –which are handled individually and have no direct relationship to each other– nor the support for videos –which are treated as a sequence of video shots, all mapping to the same source video file– captures the information and relationships of the data, we have added an additional type of media document. The resulting *image sequence* media type treats a series of images as segments of one document, rather than as individual documents. This enables the representation of each day of life logging imagery as one document while keeping the individual images in their temporal order. It also enables the association of external meta information not only to an individual image but also to an entire day without needlessly duplicating the relevant information.

3.2 Boolean Queries

Cineast was designed to answer similarity queries by evaluating them across several feature modules and fusing the resulting individually ranked lists using a score-based late fusion approach. In contrast to the previously available components of a query, which are compared to the collection using some similarity measure, Boolean expressions can be directly evaluated on every element, resulting in a set of matching elements rather than a ranked list of similar ones. In order to support this distinction, some minor additions were necessary.

Boolean expressions, which can be formulated via the user interface, each have three components; the *attribute*, which describes the part of the entity that is relevant for this particular expression, a list of *values* to compare to and the *comparison operator*, (e.g., equals, in, between, less than, etc.), which is used to match the relevant attribute to the provided values. A query can contain an arbitrary amount of these expressions, which are then implicitly combined with a logical *and* within one query container. In case a query uses multiple containers, results across these containers are combined using a logical *or*. An example of how such a query might be specified can be seen in Figure 1.

The actual evaluation of the expression is handled by dedicated feature modules that transform them from the representation used by the API to something which can be evaluated by the underlying database. In contrast to other feature modules, these ones explicitly ignore the limit imposed on the size of the result set (as done for a k-nearest-neighbour query) in order to avoid generating false negatives. The result list returned by these modules is unordered and all elements have a score of one.

During the subsequent result fusion step, the results generated by a Boolean retrieval module are handled differently than the ones generated by the modules performing a similarity search. Rather than using a traditional score-based fusion, the Boolean results are applied as a filter to the fused results, removing those not matching from the final result list.

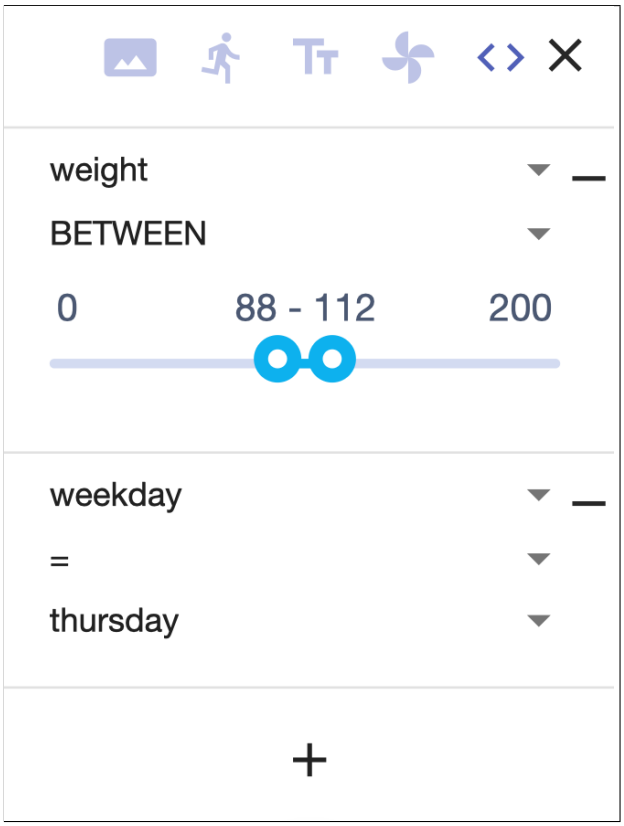

Figure 1: A Boolean query container with range and exact queries. New terms can be added to the Boolean predicate by pressing the (+)-button.

This late filtering approach has the advantage that the individual feature modules, each evaluating their interpretation of similarity, need not be aware of additional restrictions to the results they return, which reduces the complexity per module immensely. The downside lies in the additional work performed on result elements that are later discarded. This late-filtering scheme also results in a potentially too-low number of hits, since, in the worst case, all retrieved results are discarded because of a Boolean filter while results which would have matched the filter are not retrieved in the first place due to their too-low global similarity to the query. This problem can be addressed to a certain degree by increasing the number of results returned by every individual module.

3.3 Result filters

Similar to the Boolean query component outlined in Section 3.2, vitrivr offers an additional filter stage during which Boolean expressions are evaluated. In contrast to the Boolean query component, however, filters applied during this stage only affect the already retrieved result set and are evaluated by the user interface, independently of the back-end. Consequently, no new query needs to

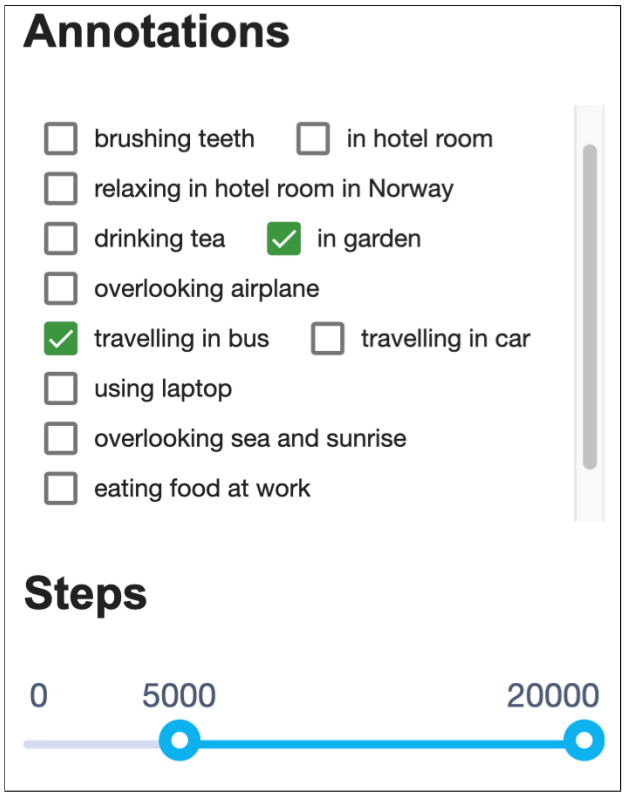

Figure 2: UI component for filtering results. The options are generated dynamically based on the available attributes.

be evaluated by the back-end and the database in order to change such a filter, which makes them very responsive.

Since the filters are applied to a known result set, the available filter options are dynamically generated based on the properties of the retrieved results. This ensures that all filter actions actually reduce the number of displayed results and keeps the filter menu from getting cluttered by unnecessary options. Generally, the filter options are being kept simple and mainly involve faceting through check boxes or filtering by selecting a value through a slider.

Figure 2 depicts an example of a filter menu. In this particular instance, all retrieved results from days with fewer than 5'000 steps will be hidden in the UI. Similarly, all results that have not been annotated with either 'travelling in bus' or 'in garden' are not shown to the user. In order to improve usability, filters with no options are ignored. This means that even though none of the check boxes in the figure are not checked by default, all results are shown until at least one check box has been selected by a user.

4 RELATED WORK

The lifelog dataset is a collection of 3 million images collected by 33 people over 3.5 years [3]. The images were recorded via a camera attached to the life logger. Furthermore, the images were enriched with metadata such as biometrics data or geo locations, hence, the dataset can be considered to be multimodal. The LSC challenge operates on a subset of the original dataset, involving 27 days stemming from a single life logger [10]. That subset has already been processed a by a standard computer vision pipeline in order to enrich the metadata by some basic concepts that could be inferred from the images.

From a data organization perspective, the LSC dataset is essentially based on images that were taken in a 45 seconds interval throughout a day, capturing the moments of a lifelogger's activities and encounters. One approach to handle such data is to treat it as a video with a very low frame rate. Based on this assumption, one could use systems mainly targeted at video storage and retrieval to handle such data. This was done by [15] and [16] and in both cases, images coming from the same day were grouped into a single video file whereas a range of images within a video were treated as individual shots. For vitrivr, we have decided to create a new media type as explained and described in Section 3.1. This allows us to attach metadata to individual images as well as the entire video depending on the semantics of that metadata and does away with the restrictions imposed by shots, which are a construct that can not be directly applied to this case.

As mentioned earlier, there are various types of metadata accompanying the LSC dataset. In LSC2018, only 3 out of 6 participant teams used biometric data alongside the other annotations available in the dataset [11]. LifeXplore uses this rich metadata as a means to filter and refine the search results [16] while LIFER [25] uses it simply for keyword based search. In contrast, [1] used the metadata to support a TF-IDF based ranking strategy [17]. For vitrivr, we use the metadata to facilitate exact and range Boolean queries as well as a way to drill down and refine the results post-query execution. To complement the provided metadata, [15] and [24] used deep learning based methods to recognize objects from the images of the dataset and use those concepts in keyword search. The authors of [24] even employ an image captioning method to extract action classes of the still images via Neuraltalk [14]. In addition to feature and concept extraction, [24] applied FlowNet [4] – a deep neural network to extract optical flow from two consecutive images. This modality is used to cluster the results on a visual level, but they produced inconsistent results on a temporal level.

In contrast to most conventional interactive search systems proposed for LSC 2018, [6] and [13] used a virtual reality (VR) environment to change the user interaction methods. [13] created a virtual space in which the user could access the images arranged on a map based on their geo location information. The system proposed by [5] enables the user to interact with the system through VR handles and search by concepts or filters available in the system. Although novel user interaction methods are attractive and may benefit from the larger angle of view for the user, vitrivr keeps the simplicity and user friendliness of its web-based UI, which was proven to be effective during the VBS 2019 novice session [22].

5 CONCLUSION

In this paper we presented the changes made to the vitrivr system in order to prepare for the the LSC 2019. A new media type is introduced to enable vitrivr to process and attach the metadata semantically to both individual images and temporally connected sequence of images. In addition to the existing query models in vitrivr, both the user interface and the query engine are extended

by adding means for the specification and execution of Boolean queries, which allows us to search directly in the metadata provided by the LSC dataset which can be a range of numbers or separate textual entities.

All the other functionality of vitrivr such as textual search, semantic sketching, and color sketching remains unchanged. In addition to the metadata, a series of textual labels extracted by various deep learning pipelines is used to fill the gap between existing conceptual labels in the metadata. These pipelines include screen text detection (OCR), object and action detection, and image captioning. Additional filters based on the metadata are included to refine the search results after a query is executed.

Hence, the preparation for the challenges of LSC 2019 consist only of moderate additions to the existing vitrivr stack. LSC poses an interesting addition to our existing engagement in similar activities and is one stepping stone towards our vision of a general purpose multimedia retrieval system.

REFERENCES

[1] Adrià Alsina, Xavier Giró, and Cathal Gurrin. 2018. An interactive lifelog search engine for LSC2018. In *Proceedings of the 2018 ACM Workshop on The Lifelog Search Challenge*. ACM, 30–32.

[2] Claudiu Cobârzan, Klaus Schoeffmann, Werner Bailer, Wolfgang Hürst, Adam Blažek, Jakub Lokoč, Stefanos Vrochidis, Kai Uwe Barthel, and Luca Rossetto. 2017. Interactive video search tools: a detailed analysis of the video browser showdown 2015. *Multimedia Tools and Applications* 76, 4 (2017), 5539–5571.

[3] Aiden R Doherty, Niamh Caprani, Ciarán Ó Conaire, Vaiva Kalnikaite, Cathal Gurrin, Alan F Smeaton, and Noel E OâĂŹConnor. 2011. Passively recognising human activities through lifelogging. *Computers in Human Behavior* 27, 5 (2011), 1948–1958.

[4] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. 2015. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*. 2758–2766.

[5] Aaron Duane and Cathal Gurrin. 2019. User interaction for visual lifelog retrieval in a virtual environment. In *International Conference on Multimedia Modeling*. Springer, 239–250.

[6] Aaron Duane, Cathal Gurrin, and Wolfgang Huerst. 2018. Virtual reality lifelog explorer: lifelog search challenge at ACM ICMR 2018. In *Proceedings of the 2018 ACM Workshop on The Lifelog Search Challenge*. ACM, 20–23.

[7] Ralph Gasser, Luca Rossetto, and Heiko Schuldt. 2019. Towards an All-Purpose Content-Based Multimedia Information Retrieval System. *arXiv preprint arXiv:1902.03878* (2019).

[8] Ivan Giangreco. 2018. *Database support for large-scale multimedia retrieval.* Ph.D. Dissertation. University of Basel.

[9] Ivan Giangreco and Heiko Schuldt. 2016. ADAM$_{pro}$: Database Support for Big Multimedia Retrieval. *Datenbank-Spektrum* 16, 1 (2016), 17–26. https://doi.org/10.1007/s13222-015-0209-y

[10] Cathal Gurrin, Hideo Joho, Frank Hopfgartner, Liting Zhou, Rashmi Gupta, Rami Albatal, Dang Nguyen, and Duc Tien. 2017. Overview of NTCIR-13 Lifelog-2 task. NTCIR.

[11] Cathal Gurrin, Klaus Schoeffmann, Hideo Joho, Andreas Leibetseder, Liting Zhou, Aaron Duane, Duc-Tien Dang-Nguyen, Michael Riegler, Luca Piras, Minh-Triet Tran, et al. 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.

[12] Cathal Gurrin, Klaus Schoeffmann, Hideo Joho, Bernd Munzer, Rami Albatal, Frank Hopfgartner, Liting Zhou, and Duc-Tien Dang-Nguyen. 2019. A test collection for interactive lifelog retrieval. In *International Conference on Multimedia Modeling*. Springer, 312–324.

[13] Wolfgang Hürst, Kevin Ouwehand, Marijn Mengerink, Aaron Duane, and Cathal Gurrin. 2018. Geospatial access to lifelogging photos in virtual reality. In *Proceedings of the 2018 ACM Workshop on The Lifelog Search Challenge*. ACM, 33–37.

[14] Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3128–3137.

[15] 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*. ACM, 15–19.

[16] Bernd Münzer, Andreas Leibetseder, Sabrina Kletz, Manfred Jürgen Primus, and Klaus Schoeffmann. 2018. lifeXplore at the lifelog search challenge 2018. In *Proceedings of the 2018 ACM Workshop on The Lifelog Search Challenge*. ACM, 3–8.

[17] Stephen Robertson. 2004. Understanding inverse document frequency: on theoretical arguments for IDF. *Journal of documentation* 60, 5 (2004), 503–520.

[18] Luca Rossetto, Ivan Giangreco, Ralph Gasser, and Heiko Schuldt. 2018. Competitive video retrieval with vitrivr. In *International Conference on Multimedia Modeling*. Springer, 403–406.

[19] Luca Rossetto, Ivan Giangreco, and Heiko Schuldt. 2014. Cineast: a Multi-feature Sketch-based Video Retrieval Engine. In *2014 IEEE International Symposium on Multimedia*. IEEE, Taichung, Taiwan, 18–23.

[20] Luca Rossetto, Ivan Giangreco, Claudiu Tanase, and Heiko Schuldt. 2016. vitrivr: A flexible retrieval stack supporting multiple query modes for searching in multimedia collections. In *Proceedings of the 2016 ACM on Multimedia Conference*. ACM, 1183–1186.

[21] Luca Rossetto, Mahnaz Amiri Parian, Ralph Gasser, Ivan Giangreco, Silvan Heller, and Heiko Schuldt. 2019. Deep Learning-Based Concept Detection in vitrivr. In *International Conference on Multimedia Modeling*. Springer, 616–621.

[22] Luca Rossetto, Mahnaz Amiri Parian, Ralph Gasser, Ivan Giangreco, Silvan Heller, and Heiko Schuldt. 2019. Deep Learning-based Concept Detection in vitrivr at the Video Browser Showdown 2019-Final Notes. *arXiv preprint arXiv:1902.10647* (2019).

[23] Luca Rossetto, Heiko Schuldt, George Awad, and Asad A Butt. 2019. V3C–A Research Video Collection. In *International Conference on Multimedia Modeling*. Springer, 349–360.

[24] Thanh-Dat Truong, Tung Dinh-Duy, Vinh-Tiep Nguyen, and Minh-Triet Tran. 2018. Lifelogging retrieval based on semantic concepts fusion. In *Proceedings of the 2018 ACM Workshop on The Lifelog Search Challenge*. ACM, 24–29.

[25] Liting Zhou, Zaher Hinbarji, Duc-Tien Dang-Nguyen, and Cathal Gurrin. 2018. Lifer: an interactive lifelog retrieval system. In *Proceedings of the 2018 ACM Workshop on The Lifelog Search Challenge*. ACM, 9–14.