PH-remix prototype

A non relational approach for exploring AI-generated content in audiovisual archives

Chiara Mannari[1][0000−0002−5488−4150], Davide Italo Serramazza[2][0000−0002−0227−8626], and Enrica Salvatori[3][0000−0002−2933−4341]

[1] Dipartimento di Filologia, Letteratura e Linguistica, Università di Pisa, Italy
[2] Dipartimento di Informatica, Università di Pisa, Italy
[3] Dipartimento di Civiltà e Forme del Sapere, Università di Pisa, Italy
chiara.mannari@fileli.unipi.it, davide.serramazza@fileli.unipi.it, enrica.salvatori@unipi.it

Abstract. Born in the complex and interdisciplinary scenario of digital culture, PH-Remix is a prototype of a web platform granting access and reuse of a vast amount of clips extracted from videos through AI techniques. The paper focuses both on the contribution of AI with the use of multiple machine learning algorithms specialized in the extraction of information from videos and on the possibilities derived from the use of a NoSQL database that plays a key role in the microservices architecture developed.

Keywords: Multimedia archive · Non relational database · Machine Learning · Video Remix · Public History · Film Heritage

1 Introduction

PH-Remix is a prototype platform[4] based on artificial intelligence developed in the context of a two year research project led by the Laboratory of Digital Culture of the University of Pisa, in collaboration with Festival dei Popoli (FdP), Mediateca Toscana and Fondazione Sistema Toscana. The platform enables the uploading, cataloguing, search, consultation, extraction and remix [12][2][11] of primary filmic sources. A search engine provides access to a vast index of clips extracted from the documentaries of FdP archive and a video editor allows final users to preview them and create video remixes. The platform is conceived as a tool for public history [13] and can be useful for both academics and public. It aims to help archives and other institutions devoted to video preservation to enhance the cataloging of their film heritage and promote it through the study of history and new collaborative ways to make history with the public. [3]

The FdP documentary archive is the case study for the development of the prototype. The data management is based both on traditional cataloguing techniques, assigning standard metadata to the films, and on the use of AI techniques

[4] Public History remix 2020 - 2022. http://www.labcd.unipi.it/ph-remix

to automatically extract the clips, i. e. significant video segments with different duration. For the development of the prototype a sample consisting of 400 films (about 400 hours of contents) was selected extracting more than 1 million of clips through AI processors.

The aim of the project PH-Remix is to design and develop a first prototype for exploring the possibilities that arise from the introduction of AI and remix practice in audiovisual archives. In expectation of future works with the objective to host the whole FdP archive and other collections, a flexible structure capable of supporting large amount of data has been developed. In the following chapters the technical solutions behind the project are presented focusing on the non relational approach experimented for data management and the microservices software architecture as possible alternatives to monolithic software and relational databases [7][1].

2 Platform architecture and database

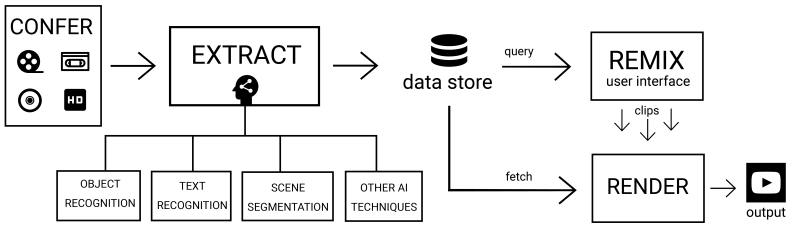

Fig. 1. Platform architecture based on microservices: the box *Other AI techniques* refers to the ease of adding other processors thanks to the architecture adopted

The architecture of the platform is shown in figure 1: it is made up of several microservice components, connected each other through rest APIs. The main components are:

- Confer: a web platform allowing to upload new films to be analysed and to insert the related metadata;
- Extract: once a video is uploaded, the extractor is in charge of its analysis: it calls different machine learning algorithms extracting *clips*, i.e. sub-segments of the whole video. They aim to detect specific features in the video portion and isolate them (more information about these algorithms are in section 3). We later refer to these algorithms as *processors*. Once the algorithms terminate their job, the Extract module collects the information about the extracted clips and stores them in the data store;
- Data store: the database of the platform described later in this section;
- Remix: a web application in which users can query *clips*, remix them and watch a preview of the built remix. It is described in section 4;

– Render: once a user is satisfied of their creation, they can request the final remix which is a video file made up of an intro, the remix itself, credits about the films from which the used clips were extracted from and an outro.

In this platform a key component is the *Data store* since each other components either store or read data from it. More in detail it is a MongoDB database, thus a NoSQL one, containing the following collections:

– Users: it stores the list of users allowed to upload films to the platform;
– Video: in this collection the metadata of the uploaded videos are stored, e.g. bit rates, aspects, audio/video codecs, etc.
– Films: it contains information about the same video present in the previous collection. In this case the information concerns the artwork as for instance title, director, nationality, production, etc.
– Segments: a collection containing information about the extracted clips from the uploaded videos. Each element contains a reference to the video which it was extracted from, the starting point and the duration of the clip, the detected information from the machine learning algorithms, etc.

There are several motivations behind the use of a NoSQL database. First of all, the information contained in these collections does not follow a predefined schema: the items in *Segments* differ because each processor stores different information in the database (more details in section 3). In addition, the items in *Films* collection may be different: although the International Federation of Film Archives (FIAF)[5] was adopted as the metadata standard for the cataloguing of FdP archive, the platform was designed to promote interoperability and support any metadata schema through custom fields and data import option available in the back-end. The policy adopted was to add as much metadata as possible in order to perform a full text search across the whole index of clips, in a way similar to search engines (more detail in section 4). Other motivations are the higher performances in retrieving documents that a NoSQL database can guarantee [9][6] and lastly, the way in which the data are stored in a NoSQL database, namely key-value pairs, i.e. the data structure required from the rest API.

3 Use of Artificial Intelligence

In this section the three *processors* developed so far will be described.

The first one, *Object recognition* relies on *RetinaNet* [10] to perform predictions on the frames making up a film. The adopted version uses as backbone *ResNet152* [4] trained using the *Open Images* dataset [8], arguably the one with the highest number and wider diversity of target classes. The developed processor uses the predictions made on consecutive frames to compose a segment lasting from the first time to the last time in which an object O is detected. In addition to the common information listed in section 2, this processor stores in

[5] https://www.fiafnet.org/images/tinyUpload/E-Resources/Commission-And-PIP-Resources/CDC-resources/20160920%20Fiaf%20Manual-WEB.pdf

the data store the information about the detected object along with the rectangle coordinates surrounding the detected object and the related confidences. This score is used for sorting the results in decreasing order: the score of a segment of this processor is the average of the single predictions score. A future development of the platform will be to have similar scores $s \in [0, 1]$ also for the other two processors in order to have results fully sorted according to this metadata.

The second processor *Text Recognition*, also analyses all the Film frames. It relies on the concatenation of two algorithms: EAST [17] which detects the frame portions in which text is present and the popular utility *Tesseract* to retrieve the texts from these areas. This processor aims to analyse just the subtitles that were impressed in the film frames thus it processes only the lower half of the frames. To build a segment starting from the single frame predictions a similarity score among the text extracted from the frame f_i and the frame f_{i+1} is used: if the Normalized *Levenshtein distance* [16] falls under the value 0.5, i.e. the two sets of characters share less than 50% of the elements, the subtitles are considered to be different. The beginning and ending times of a segment are the ones corresponding to the first and the last time a subtitle s is shown. The label in this case is the extracted subtitle.

A future development will be to take into account other information present in these segments to evolve the platform with a semantic approach. They contain results of a master thesis which aimed at analysing the extracted transcripts applying NLP algorithms on them. Some examples are keep only relevant items (eliminating the stop words), lemmatization or detecting Named Entity.

The last processor *Scene Segmentation* is based on *Trans Net v2* [14]. Differently from the previous two, this processor works directly on the video rather than on the frames composing it. It analyses the video and detects the shots within it, labeling them with their dominant colour.

The labels that are assigned at each segment from the different processors are used in the remix platform to search for segments, fitting the users needs to build their own remix. In order to allow users to perform queries in different languages, the segments generated from the first and third processors contain in the labels also the wikidata entry [15] related to the detected object/colour.

4 Remix Platform

The platform architecture and the extraction processes described above have been designed with the aim to support the access of final users to the remix service: the front-end application for video remix.

Represented in figure 2, the remix tool is a single page application with a rich user interface divided into three areas:

- an area at the top right corner of the screen for searching and browsing clips;
- a video editor at the bottom to remix clips and perform basic video editing operations through drag and drop. This area includes also a button for launching the server side task to export remixes in the final mp4 format;
- a video player at the top left for the local preview of single clips and remixes.

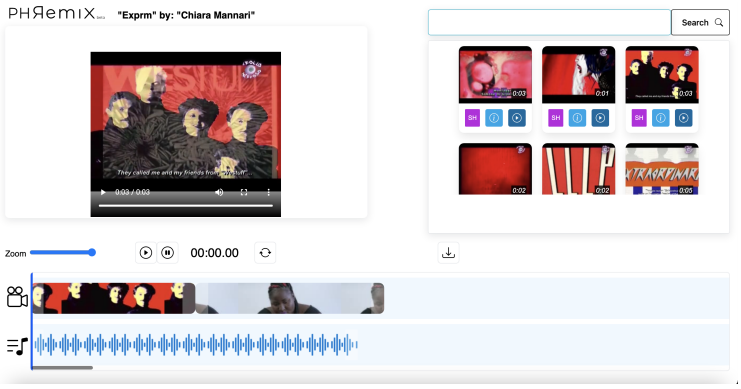

Fig. 2. Remix application

The core of the remix application is the search engine of clips that is still in an early version. Through the interaction with a simple text field it is possible to build queries at different level of complexity, then send the request to the server through the search API (as illustrated in figure 1). The query is processed by a server side script in NodeJS which performs searches in the MongoDB database.

In simple searches the algorithm queries directly in the collection of clips performing a proximity search (i.e. the result may be the exact string, a substring or a similar string according to MongoDB text search). Clips to be returned are subsequently ordered following criteria similar to search engines and IR models: first are provided results more similar to the value searched, then are returned the more different. The score associated to the clips extracted by the object detection processor (see section 3) is particularly relevant for ordering such clips.

On the other hand, in complex searches users can use filters to narrow the search to particular films with specific values. This kind of search is developed both at client and at server side. At client side, a query-text component provides a list of suggestions that appear when the user types in a prefix. Prefixes used in the prototype refer to common entities that are of interest to the users: people, places, film titles and countries of production. Results originate from an index generated from the extraction of single or multiple field values in the collection Films of the data store. As shown in figure 3, by typing "@" followed by at least three characters, the prefix for people is activated and a list of names which aggregates values from multiple fields of the collection Films appears. These fields store person entities, e.g. film director, producers, technicians, staff members but also anthroponyms, i.e. people or characters mentioned in the film's plot. When the server receives a request for a filtered search, the script performs a two-level search in the MongoDB database. First, the films matching the specified criteria are selected, then the query is limited only on these films. As described in section 2, the reference that each segment has to the film which it was extracted from is essential both for this kind of search and for the Render microservice that generates credits for the films used in the current remix.

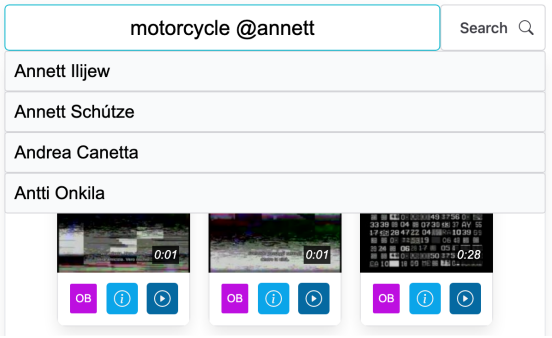

Fig. 3. Advanced search with filter for people active

The search could be further improved developing an extended query language and a more complex IR system to facilitate users in their searches. The structure of the collections in the database supports extensions such as language independent searches based on wikidata, entity based searches or a system to manage licenses for copyrighted materials. Eventually, the use of a NoSQL database to manage archival data stimulates additional evaluations on the use of different approaches to explore catalogues. The full-text search based on entities and autocompletion proper of IR systems is a user friendly solution spread over the internet thanks to search engines and social networks[5] and future tests could contribute to the evaluation of this approach as an alternative solution to the traditional grid interfaces provided by catalogues based on relational databases.

5 Conclusions

The microservice architecture facilitates the evolution of the platform through the improvement of existing modules and the development of new ones. The FdP archive is the case study for the development of the prototype but in a future more archives could be added to the platform. As described in the previous sections, the flexibility of the NoSQL database and the APIs developed allow to easily add new contents coming from different archives. The support of different schemas promotes interoperability between archives and the platform.

The availability of a tool capable to perform searches within large quantities of data provided both by machine learning algorithms and by human cataloguing leads the way to new approaches for the exploration of audiovisual cultural heritage. The prototype developed is now an essential tool for directly evaluating the possibilities offered by the information extraction from videos by AI.

Automatic video analysis and segmentation is particularly challenging nowadays with the huge amount of data produced and spread through the internet. For this reason, PH-Remix platform, including searchable index of clips and editor for video remix aims to be an innovative tool to be experimented in different scenarios: research, dissemination of cultural heritage, users engagement.

References

1. Andreas Meier, M.K.: SQL & NoSQL Databases. Springer Vieweg Wiesbaden (2019)
2. Gallagher, O.: Reclaiming critical remix: the role of sampling in transformative works. Routledge (2018)
3. Grasso, G., Mannari, C., Serramazza, D.: Intelligenza artificiale e archivi audiovisivi: potenzialità e sfide del progetto ph-remix. In: AIUCD 2022 - Proceedings. pp. 141–144 (2022)
4. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. arxiv 2015. arXiv preprint arXiv:1512.03385 (2015)
5. Hu, S.: Efficient text autocompletion for online services. In: Takeda, K., Ide, I., Muhandiki, V. (eds.) Frontiers of Digital Transformation: Applications of the Real-World Data Circulation Paradigm, pp. 171–185. Springer Singapore, Singapore (2021)
6. Jose, B., Abraham, S.: Performance analysis of nosql and relational databases with mongodb and mysql. Materials today: PROCEEDINGS **24**, 2036–2043 (2020)
7. Kalske, M., Mäkitalo, N., Mikkonen, T.: Challenges when moving from monolith to microservice architecture. In: Garrigós, I., Wimmer, M. (eds.) Current Trends in Web Engineering. pp. 32–47. Springer International Publishing, Cham (2018)
8. Kuznetsova, A., Rom, H., Alldrin, N., Uijlings, J., Krasin, I., Pont-Tuset, J., Kamali, S., Popov, S., Malloci, M., Kolesnikov, A., et al.: The open images dataset v4. International Journal of Computer Vision **128**(7), 1956–1981 (2020)
9. Li, Y., Manoharan, S.: A performance comparison of sql and nosql databases. In: 2013 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM). pp. 15–19. IEEE (2013)
10. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. pp. 2980–2988 (2017)
11. Navas, E.: Remix theory. The aesthetics of sampling. Springer (2012)
12. Navas, E., Gallagher, O., burrough, x.: The Routledge Handbook of Remix Studies and Digital Humanities. Routledge (2021)
13. Salvatori, E.: Digital (public) history: the new road of an ancient discipline. RiMe Rivista dellIstituto di Storia dellEuropa Mediterranea **1**(1), 57–94 (2017)
14. Souček, T., Lokoč, J.: Transnet v2: An effective deep network architecture for fast shot transition detection. arXiv preprint arXiv:2008.04838 (2020)
15. Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Communications of the ACM **57**(10), 78–85 (2014)
16. Yujian, L., Bo, L.: A normalized levenshtein distance metric. IEEE transactions on pattern analysis and machine intelligence **29**(6), 1091–1095 (2007)
17. Zhou, X., Yao, C., Wen, H., Wang, Y., Zhou, S., He, W., Liang, J.: East: an efficient and accurate scene text detector. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 5551–5560 (2017)