UNIVERSITÉ DE TOURS

ÉCOLE DOCTORALE : *MIPTIS*

Laboratoire d'Informatique Fondamentale et Appliquée de Tours (UR 6300)

THÈSE présentée par :

Van Hao LE

soutenue le : **07 octobre 2024**

pour obtenir le grade de : **Docteur de l'Université de Tours**

Discipline/Spécialité : INFORMATIQUE

Performance Evaluation for Scalable Partial Video Copy Detection

THÈSE dirigée par :

M.	CARDOT Hubert	Professeur, Université de Tours, France

CO-ENCADRANT :

M.	DELALANDRE Mathieu	Maître de Conférences, Université de Tours, France

RAPPORTEURS :

M.	DUFFNER Stefan	Maître de Conférences, HDR, INSA de Lyon, France
M.	GROSSI Giuliano	Associate Professor, Université de Milan, Italie

---

JURY, présidé par M. BURIE Jean-Christophe:

M.	CARDOT Hubert	Professeur, Université de Tours, France
M.	DELALANDRE Mathieu	Maître de Conférences, Université de Tours, France
M.	DUFFNER Stefan	Maître de Conférences, HDR, INSA de Lyon, France
M.	GROSSI Giuliano	Associate Professor, Université de Milan, Italie
M.	DOMENGER Jean-Philippe	Professeur, Université de Bordeaux, France
M.	BURIE Jean-Christophe	Professeur, Université de La Rochelle, France
M.	PHAM The Anh	Associate Professor, Université de Hong Duc, Vietnam

Acknowledgments

"Gratitude is the fairest blossom which springs from the soul."
- Henry Ward Beecher

First and foremost, I would like to express my deepest gratitude to my advisor, Associate Professor. Mathieu Delalandre, and Professor. Hubert Cardot for their unwavering support, insightful advice, and constant encouragement throughout my PhD studies. I am delighted and I feel privileged to be supervised by you all. All of my achievements during my PhD studies are due to your expertise, guidance, patience, and understanding. I greatly appreciate all the open discussions we have discussed through a lager number of reading groups during this thesis. I feel so fortunate to forever remember your dedicated guidance, great efforts and remarkable enthusiasm from the bottom of my heart.

I am grateful to the reviewers for taking the time to provide valuable comments and suggestions on this manuscript. Despite my exhaustive efforts to complete this work, there are still a number of issues that need to be pointed out and corrected by you. Your comments and feedback will definitely improve this dissertation considerably.

I musk thank the administrative and technical staff at LIFAT laboratory[1], and the International Relations Office, University of Tours for their kind assistance and support during my PhD studies. Thank you very much for making all the administrative works and that part of my life in Tours easier. I would also like to acknowledge the financial support provided by L'Ecole Doctorale MIPTIS, which made this research possible. The financial support has been instrumental in allowing me to pursue and complete this research.

I would like to thank all my friends for creating a stimulating and collaborative environment. You make me more and more familiar to the new environment. To my Vietnamese friends, thank you for your constant encouragement and for helping me maintain a healthy work-life balance. To my fellow lab mates, thank you for giving me the chance to participate in our excellent PhD student network, where we can freely discuss our work and share ideas. I really hope to see all of you again on many occasions in the future.

From my personal side, I would like to thank the people who have been my anchor throughout this journey. Many thanks to my affiliation in Vietnam, Hong Duc University (HDU[2]), for making all the arrangements that allowed me to concentrate entirely on my PhD research abroad. Special thanks to Associate Professor. The Anh Pham, HDU for his invaluable support in guiding me towards finding this PhD program and shaping my

[1]LIFAT laboratory: https://lifat.univ-tours.fr, University of Tours: https://international.univ-tours.fr/
[2]Hong Duc University: https://hdu.edu.vn

career path. I would also like to extend my heartfelt gratitude to my colleagues at Faculty of Information and Communication Technology, HDU, for their support during my study abroad. Thank you for being a constant source of support and friendship. I am deeply indebted to my parents, my little family, brothers and sisters. Words cannot express how grateful I am for your endless love, sacrifices, and belief in me.

Last but not least, this thesis belongs to all of you as much as it does to me. Thank you all for being a part of this journey.

July 15th, 2024 (Tours city, France).
Van Hao Le

Résumé

La détection de segments au sein des vidéos est un problème bien connu dans le domaine de la vision par ordinateur. L'objectif est de détecter une vidéo courte au sein d'une vidéo plus longue sous contraintes de déformation. La portion de vidéo détectée est alors qualifiée de segment. La détection de segments a de nombreuses applications comme pour la recherche de vidéos en ligne, la protection de la propriété intellectuelle, la détection de publicité TV, etc. Durant les dernières années, les systèmes à base de réseaux neuronaux convolutifs se sont imposés sur ce problème. Ces systèmes extraient des caractéristiques à partir des réseaux afin de décrire les fenêtres clés des segments. Ceci soulève de nouveaux besoins en termes d'évaluation de performance des systèmes à des fins de mise à l'échelle, contrôle des déformations et d'horodatage temporel des vidéos. Cette thèse propose deux contributions clés dans le domaine de l'évaluation de performance des systèmes de détection de segments.

En première contribution, un nouveau protocole est proposé afin de générer des bases de test à l'échelle et de haute qualité, tant sur les données vidéos que celles de vérité terrain et d'horodatage. Ce protocole a été déployé afin de concevoir la base de données STVD. Cette base est la plus importante de la littérature à date avec 83 mille vidéos (ayant une durée de plus de 10 mille heures) composées de 1,6 million d'associations de segments (horodatés avec une précision à la fenêtre).

La deuxième contribution de la thèse s'intéresse aux caractéristiques extraites des réseaux neuronaux convolutifs. Ces caractéristiques sont bien adaptées aux photos mais moins aux vidéos compte tenu de la problématique d'échelle, de redondance, de floue et de faible information visuelle. Afin d'étudier les limitations, une étude de caractérisation est présentée. Sur la base du protocole d'usage de la littérature, neuf méthodes d'extraction ont été mise en oeuvre et comparées sur la base STVD. Cette comparaison s'est traduite par l'extraction de 4,4 millions de vecteurs de caractéristiques et 700 milliards d'appariements. Les résultats démontrent que la séparabilité n'est pas atteinte sur le problème de détection même si de fortes performances sont observées. Les différents réseaux présentent des performances proches, même si les architectures récentes comme ResNet50 se démarquent. De même, il y a une corrélation entre la méthode d'extraction utilisée et l'architecture du réseau utilisé. Cette étude conclue à une carence du protocole d'usage. Afin de répondre à ce problème, un nouveau protocole est proposé pour la caractérisation des fenêtres clés des segments à partir des caractéristiques réseaux. Ce protocole est basé sur un critère de qualité des fenêtres et une modélisation en série temporelle. Il permet une caractérisation fine des fenêtres clés et caractéristiques réseaux en termes de séparabilité, consistance et analyse quantitative de la qualité.

Mots clés : détection de segments · évaluation de performance · protocole · base de données · caractéristiques réseaux

Abstract

Partial Video Copy Detection (PVCD) is a well-known problem in the computer vision field. It is interested in detecting copied segments of short videos that have been transformed and embedded into longer videos. The PVCD has a wide range of real-world applications such as video retrieval, copyright protection, commercial detection and/or news verification in TV broadcasting. Over the past decade, deep learning algorithms, especially 2D Convolutional Neural Networks (CNNs), have become a key trend in designing PVCD systems. These systems extract 2D CNN features from frames for the retrieval and detection of partial video copies. This has opened new needs and challenges for the performance evaluation of PVCD systems in terms of scalability, control of degradations and temporal accuracy. The contributions of this PhD are twofold as summarized below.

We propose a new protocol to design a scalable, noise-free PVCD dataset with temporal accuracy. Several existing datasets are designed using Web-based protocols. The PVCD problem is inherent to the continuous video broadcasting, and an alternative is to process with TV-based protocols. We have defined a new TV-based protocol offering a deeper scalability, control of degradations, and temporal accuracy. It uses two different TV streams to separate candidate videos for the copy and non-copy categories in the annotation step. Within the protocol, we process with a semi-automatic and incremental groundtruthing for a fast annotation and to guarantee a domain meaning of the partial video copies. We leverage prior known metadata (i.e., TV guide/listings) and low-level frame matching for robust detection. Synthetic video degradations are applied in order to stress the systems for detection. This protocol is used to design a PVCD dataset called STVD. To the best of our knowledge, STVD is the largest public PVCD dataset containing nearly 83 thousand videos (having a total duration of 10 660 hours and containing 1 688 thousand pairs of partial video copies) and offering the temporal accuracy of frame-level.

In the second contribution, we highlight the limitations of 2D CNN features for the PVCD problem. While these 2D CNN features are effective for image processing, their performance could be degraded when faced with the unique challenges of PVCD (e.g., scalability, near-duplicate detection, motion artifacts, solid-color frames). To assess these challenges, we provide a characterization of 2D CNN features for separability/consistency. Based on our STVD dataset, we first give large-scale characterization of 9 common 2D CNNs, driven on 4.4 million feature vectors with 700 billion comparisons. From the results of the experiments, we find that the separability is not achieved on the detection problem even if strong scores are obtained. The different CNNs present competitive results. As a general trend, features extracted from recent 2D CNNs such as ResNet50 perform better. A correlation appears between the feature extraction methods and the 2D CNN architectures.

These different conclusions are consistent with the state-of-the-art in the computer vision field. The regular protocols for performance characterization are misleading for PVCD as they are bounded to the video level. To deal with this issue, we propose a new characterization protocol of key-frames with 2D CNN features. This protocol is based on a goodness criterion and a time series modelling. It provides a fine categorization of key-frames, a characterization of 2D CNN features for separability, consistency, and a quantitative analysis of the goodness of key-frames. This protocol highlights the performance limits of 2D CNN features when faced with blurred, near-constant, or almost-equivalent key-frames. In addition, a large part of key-frames cannot be classified effectively from 2D CNN features due to the scalability issues.

Keywords: partial video copy detection · performance evaluation · protocol · dataset · CNN features

Contents

List of Tables

List of Figures

List of Abbreviations

Abbreviation	**Meaning**
CPU	Central Processing Unit
CNN	Convolutional Neural Network
EPGs	Electronic Programming Guides
FPS	Frames Per Second
kps	kilobits per second
GUI	Graphic User Interface
GNN	Graph Neural Network
HSV	Hue Saturation Value
N/A	not available
No.	Number of
NTSC	National Television System Committee
NLP	Natural Language Processing
PAL	Phase Alternating Line
PVCD	Partial Video Copy Detection
SECAM	SÉquentiel Couleur À Mémoire
STVD	large-Scale TV Dataset
ZNCC	Zero mean Normalized Cross-Correlation
RGB	Red Green Blue
RNN	Recurrent Neural Network

Terminology	**Definition**
Reference video	a short video that is considered an original video.
Positive video	a long video (i.e., typically its duration is larger than reference videos) and contains at least one reference video, even if transformed.
Negative video	a video does not contain any reference videos. It serves as a distraction video.
Testing video	a long video used for matching with reference videos. It also is referred to a query video and is selected from either positive or negative videos.

Introduction

This chapter aims at providing a brief introduction to the context of this PhD which is dedicated to the research problem of Partial Video Copy Detection (PVCD). PVCD takes a part of the computer vision field, and plays a key role in many video-related applications. We first introduce in this chapter the PVCD problem with regards to its definitions, application domains, and related tasks. Solving the PVCD problem requires dealing with several areas of research in the field of computer science. From the needs of a background, we briefly introduce the research domains of the image processing, machine learning, computer vision, and performance evaluation. The last section introduces the PhD subject and organization by presenting the chapters.

Introduction to the partial video copy detection

Over the past ten years, mobile devices and TV technologies have been developed rapidly with their new versions being upgraded more regularly. In addition, there have been witnessed on advances of the Internet technology creating the environment where these devices are able to connect stably and communicate smoothly with each other. Those benefits result in growth of multimedia content as well media platforms (e.g., Youtube, Meta, TikTok) where people can share their videos and watch the others. As a result, media content that includes music videos, TV movies and the other types of video has been increasing exponentially and occupying a substantial volume of data on the Internet. For example, Youtube[3] reported that over 500+ hours of video content were uploaded to the platform every minute in 2022. Hence, the videos impose very high demands on effective retrieval and recommendation performance in large-scale datasets.

Along with the development of video applications and services, a substantial number of similar videos have been contributing to the Internet or TV services, even modified videos are not intended by their owners (e.g., re-encoding by platforms). For instance, 93% percent of near-duplicate videos can be found for some queries searched on the Web [107]. The presence of such similar videos is exposed to many challenges of video retrieval, monitoring, copyright protection, or storage. Furthermore, from an economic perspective, such phenomena may cause financial loss by reducing the number of potential customers who want to purchase the original product. It leads to the particular problem of PVCD. In next paragraphs, we first introduce the problem statement and applications of PVCD. The two last paragraphs discuss about PVCD systems and their performance evaluation.

[3]It is a video sharing platform: https://www.youtube.com

Problem statement. PVCD is a computer vision problem where copied segments of short videos may be transformed and embedded into longer videos. It is the problem of detecting similarities between videos. It has a variety of understanding and definitions on the concepts of similar videos in the literature [43, 46, 49, 72, 117]. They can be roughly divided into two categories as shown in Tab. 1.

Tab. 1 The categories of similar videos.

Category	Description
Near-duplicate / copy	refers to videos that share the same semantics and scenes with the corresponding original video, but the visual content may be modified and their temporal order varied [61, 90]. The modifications can be the results of a large variety of video transformation methods. The two terms copy or near-duplicate may be used interchangeably in the literature [15, 45, 61, 94, 107].
Partial video copy	refers to long transformed videos that contain one or more other original ones [39, 46, 99]. Similar to the previous category, many video transformations can be applied to the original video to produce copies. However, such a case presents significant semantic and visual differences, as the original video comprises only a small fraction of the long video's total duration.

Fig. 1 illustrates a French TV jingle with two concepts of near-duplicate/copy video and partial video copy. The cartoon character in the video (a) has been replaced by another in video (b), while it is still advertising the same TV jingle. Given such a characteristic, the video (b) is a near-duplicate/copy of the video (a). For the partial video copy, the near-duplicate video (b), which has been transformed from the original video (a), is embedded in the long video (c). Due to the copied segment, which is indicated by two black dashed lines, constitutes only a small fraction of the total duration of the video (c), thus, it is classified as the partial video copy case.

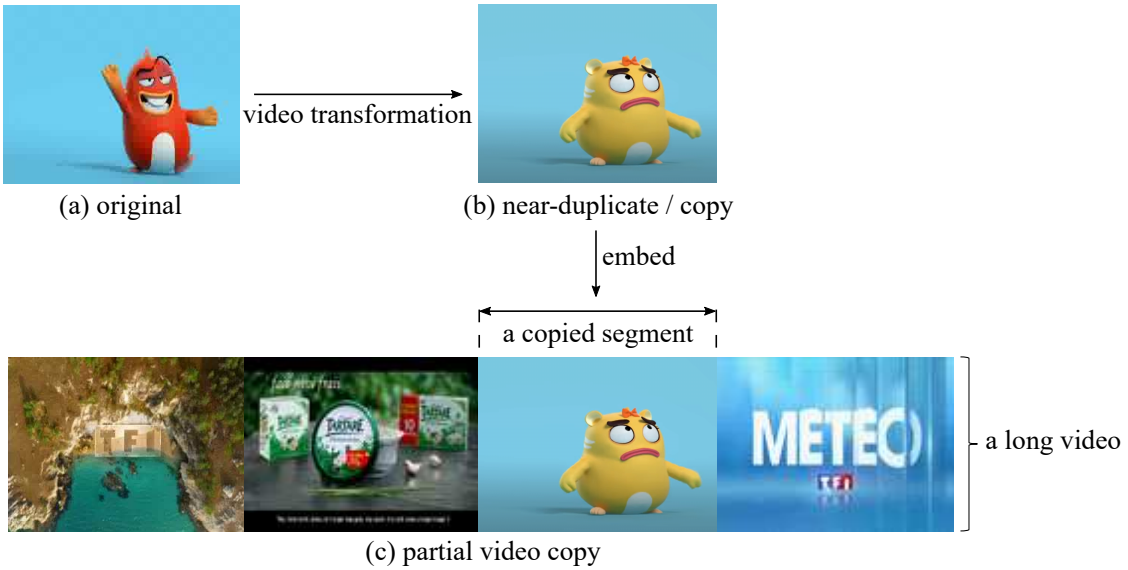


Fig. 1 Examples illustrates (a) an original video compared to:
(b) a near-duplicate/copy video, and (c) a partial video copy.

Applications. PVCD applications are designed for efficient identification, protection, search and retrieval of videos. However, distinct systems may be required for each video detection challenge. Hereafter, we summarize representative use cases of the PVCD and

discuss key challenges according to the application domains and theoretical perspectives.

Video retrieval is an important application where the PVCD techniques can be applied. Every day, we experience searching for videos by starting with a query (e.g., text, video) and ending with a list of relevant videos [43, 107]. In such systems, the quality of returned video list is a crucial aspect in order to promote the sales or retain their customers in the E-commerce and in online social entertainment services [17, 120]. In other words, the users demand the system to increase the high precision of the returned list of the videos and reduce the false-positive detection. However, this is a very challenging task for real-world systems when dealing with severely distorted videos or large-scale data. The PVCD is used to improve the diversity and reduce repetition of videos that return to the users. In addition, detecting and removing duplicate videos can mitigate the burden on storage, and thus greatly contribute on system efficiency. There are some other works which adopted the video retrieval such as detection of videos of the same incident [53] or videos of the same event [85] or mixing as a video re-localization application [23].

Copyright protection is an inherent requirement because editing and re-uploading videos via the Internet have become more convenient and low-priced, especially on the social media platforms such as YouTube, TikTok [30, 99]. As a result, video creators often find their videos uploaded on the Internet without their permission. Such cases may cause financial loss to the creators by decreasing the number of potential buyers interested in purchasing the original videos. Thus, eliminating these copied videos is an explicit need of video creators and service providers. However, these systems suffer severely from video transformation methods due to user-generated videos that may vary greatly from the original ones. These modifications are made to pass the service provider's rules and policies or even they are not intended by the users (e.g., videos are re-encoded by the platform). Taken together, these challenges increase the difficulty and the effort of detecting video violations according to the impossibility of predicting video transformation methods. For those reasons, detecting copies is more useful and urgent in order to identify the original source of a video and to protect copyright in such systems [35, 47]. With robust PVCD techniques, the system can potentially eliminate copyright violations by automatically checking uploaded videos and matching them to a set of protected videos. Also, it significantly maintains content originality as well reduces the interaction when users and moderators are exposed to harmful video content.

Video monitoring is another line of research in which the PVCD not only requires detection accuracy but also requires online processing [121]. Such a requirement can be widely applied in application domains such as commercial detection [2, 40], news verification over TV broadcasting [61, 82]. For example, a cosmetic company, named Estee Lauder, may want to monitor the broadcasting of its paid commercials by continuously detecting the placed commercial video on TVs [90]. Another scenario can be seen in many live video streaming services (e.g., YouTube, Meta) where the systems need to immediately prevent unauthorized video uploads. However, manual monitoring in such applications is a time-consuming and expensive task due to the scalability. To fulfill these needs, the PVCD techniques can be used to detect partial videos of interest. Some additional information such as the time spot, play count, and frequency are also useful in such applications. In order to deal with the time constraints, the PVCD methods require a specific approach for a trade-off between robustness and efficiency [99, 104].

Video recommendation is equally demanding from users [45, 54]. Indeed, when a user watches or likes a video, the user will also be interested in other similar videos with a high probability. It has been widely applied in many real-world applications recently. For example, in 2009, Netflix company[4] held a competition to improve their movie recommendation algorithm. Nowadays, it is very easy to see on Youtube and Meta platforms, they provide and recommend to users a list of videos that is relevant with a playing video. However, we very often notice that irrelevant videos also appear on the list of recommended videos. This is because such these systems are text-based while a large number of uploaded videos do not always include the precise meta-data in their descriptions, titles. Introducing the PVCD techniques can greatly improve accuracy in recommendations and enhance user experience in the system by recommending relevant or original video content. Especially, with the advancement of machine learning algorithms, the content-based video recommendation has attracted a lot of attention [10, 68].

PVCD systems. As was mentioned in earlier sections, the diversity of similar video categories and its applications imply that a PCVD system may be able to serve a specific task for the best. While all the tasks involve identifying content-based video correlations, it is important to justify the relation and differences between the main tasks of retrieval and detection [34, 36, 45, 47, 99]. Indeed, these two tasks share common principles and many components can be applied to both, even though their goals are not alike. The retrieval task focuses on finding and returning relevant videos from a dataset based on a given query video, the detection one focuses on identifying and locating copied videos within a dataset. Fig. 2 illustrates a general PVCD framework to demonstrate their relationship.

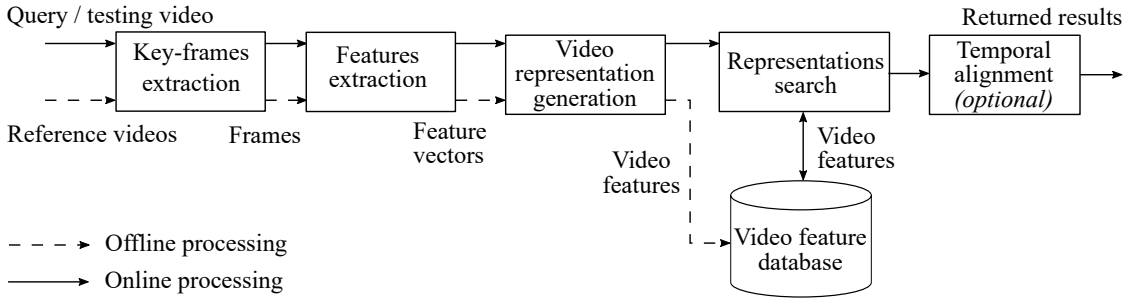


Fig. 2 A general framework of partial video copy retrieval/detection.

A general PVCD system processes videos in two phases: offline and online. In the offline phase, every video in the reference set is commonly processed within three steps. First, key-frames are extracted from the videos. These key-frames are then processed to extract features which are key and discriminative information from their visual content. Next, the set of video features is processed to generate compact and robust video representations. Finally, the obtained representations are stored into a database. Similarly, during the online phase, the system receives a query/testing video and then processes it into the key-frames selection and the features extraction. Next, the extracted features are used to generate the video representations. To detect copies, the query/testing video representations are searched in the video representation database. Then, the videos that are high similar to

[4]Netflix prize (1 million USD) attracted 5169 participating teams, http://www.netflixprize.com

the query/testing are returned to the user based on their similarity scores. The boundaries of copied segments can be given by a temporal alignment step for accurate retrieval or detection tasks. Within this framework, the detection and retrieval tasks may differ in the ways that features are extracted, the representations are obtained and processed. We briefly discuss them in the following paragraphs.

Detection aims at finding pairs of copied videos in a given video dataset [35, 47]. Thus, the outputs are labels (e.g., copied, or non-copied), and locations of detected videos within the dataset. Every reference video is matched one-by-one with a given testing video. One of the main challenges is to design features and representations that are robust to both intentional and unintentional video degradation methods. The features and representations target the robustness to the spatial noise and the geometric invariance. Additionally, timing degradations could be also investigated with temporal alignment-based detection. The detection can be made real-time where the testing video is an untrimmed/a continuous video stream [61, 62, 121]. In that case, real-time detection targets on fast and predictable features and representations for the extraction and comparison.

Retrieval refers to the task of searching videos that are relevant with a query video submitted by the user in an established reference video database [45, 107]. The output is a ranked list of videos that match with the query video. Similar to detection, retrieval needs to be robust against video degradations. However, it involves fast searching in a large-scale dataset. Thus, highly discriminative and compact features and representations are required to deal with the scalability and time efficiency. The PVCD system organizes them by effective indexing structures or clustering representations to achieve fast retrieval.

Performance evaluation of the PVCD systems. Transformations in the spatial and temporal domains, as well as scalability, are key challenges for the PVCD systems. Indeed, videos may be significantly transformed in deep ways to be searched in very large-scale datasets. That situation could lead to severe degradation of the detection and retrieval accuracy of PVCD systems. Thus, evaluating the performance of PVCD is a crucial task to measure and assess the effectiveness, efficiency, and scalability of the proposed systems. This is a specific problem of performance evaluation in the computer vision field. As with any performance evaluation problem, it requires the establishment of datasets as well as characterization metrics, but dedicated to the PVCD problem [72, 73, 90, 99].

The PVCD datasets need to propose large sets of long videos containing partial video copies as well as distraction videos. The test sets must be designed for scalability but also to address specific detection and retrieval challenges (e.g., timing degradations, impact of compression, geometric transforms of the videos). Accurate ground truth files, about the labels of video copy with their timestamping, must be provided. These datasets have to be published for users to ensure a comparison of the different algorithms and systems [45, 65]. Another key point is to define relevant protocol and characterization metrics to quantitatively and qualitatively evaluate how well a PVCD system performs [35]. These research directions are making PVCD an increasingly important task, deserving more interest and attention from the research community.

A brief introduction to research fields

The previous section has introduced the PVCD problem with regards to its definitions, application domains, and related tasks. PVCD involves addressing several areas in the computer vision science. Indeed, every PVCD system processes video frames as input data and uses machine learning algorithms to detect copies. For the needs of a background, we briefly introduce the research fields related to PVCD including the image processing, machine learning, computer vision and its performance evaluation.

Image processing. Although the first digital image appeared in 1921, the history of image processing began more recently with the development of digital computers in the 1960s. Since then, it has become increasingly important in various aspects of our lives, from entertainment to scientific research and medical diagnosis. It is a multidisciplinary field including computer science, mathematics, and engineering that is dedicated to manipulating and analyzing digital images [8]. Particularly, image processing involves the application of algorithms and techniques to transform raw visual data into meaningful information. They can be characterized into five major groups of operations as presented in Tab. 2. As technology continues to advance, and the image data grows exponentially, the image processing remains a dynamic field driving solutions to real-world challenges. Hence, many books [8, 26] have been published as well as conferences and journals (e.g., main tracks[5] ICIP, ICPR, JMIV, IP) organized for researchers, practitioners, and industry experts to exchange ideas, present novel methodologies, and discuss emerging trends.

Tab. 2 The representative grouped operations for image processing.

Group	Manipulation methods
Image acquisition	Image access, display, printing
Image compression	Lossy and Lossless image compression
Image enhancement	Brightness/contrast adjustment, frequency domain filtering, edge enhancement
Image restoration	Noise/blur removal, inverse filtering
Image analysis	Segmentation, feature extraction, object classification

At its core, the image processing aims to extract 'useful' information from the digital images. The image feature extraction plays an important role by enabling the identification and representation of key visual characteristics within digital images. These features serve as image descriptors to represent local or global features. Local features capture information about specific regions or patterns within an image (e.g., edges, corners, or key-points). In particular, it aims to identify distinctive points or patches within an image by applying techniques such as corner detection [32], scale-invariant feature transform (SIFT) [75], or Speeded Up Robust Features (SURF) [6]. By focusing on local regions, these methods enable robust and invariant representations that are resilient to changes in scale, rotation, or illumination. On the other hand, global features provide an overall representation and spatial relationships of the entire image. These features are commonly involved with the color distribution, texture, or semantic content of the image. Thus, global descriptors can be obtained by calculating color space histograms (i.e., color value distributions), texture descriptors such as the local binary patterns (LBP) [105], or spatial pyramids. Compared to

[5]The International Conference on Image Processing, International Conference on Pattern Recognition, Journal of Mathematical Image and Vision, Transactions on Image Processing

local features, global features generally offer more compact representations, thus enhancing computational efficiency. However, the selection between local and global features depends on the particular demands of the application and characteristics of image processing tasks.

Machine learning is one of the foundational parts in the field of computer science where computer programs are capable of executing tasks without explicit programming. That is, these programs can automatically operate when provided with input data instead of relying on specifically designed rules. Such algorithms require a training process, and thus it is often referred to machine learning algorithms. In particular, a computer program is said to learn with respect to a given task and a performance measure, if its performance on the task improves with the learning level [78]. Nowadays, machine learning algorithms are widely used in many real-world applications, including but not limited to image processing, virtual assistants, recommendation systems, auto-driven cars, medical diagnosis [113].

Based on types of learning, machine learning algorithms can roughly be divided first into two main categories: supervised and unsupervised. The term '*supervised*' refers to a type of learning where the algorithm is trained on a labeled dataset. That is, for each input data point, the correct output/label is provided. The algorithm is trained to map inputs to outputs based on the examples provided during training. Unlikely supervised learning, unsupervised learning commonly does not require labels. Such an algorithm is designed to discover hidden structures, or learn patterns from the unlabeled input examples.

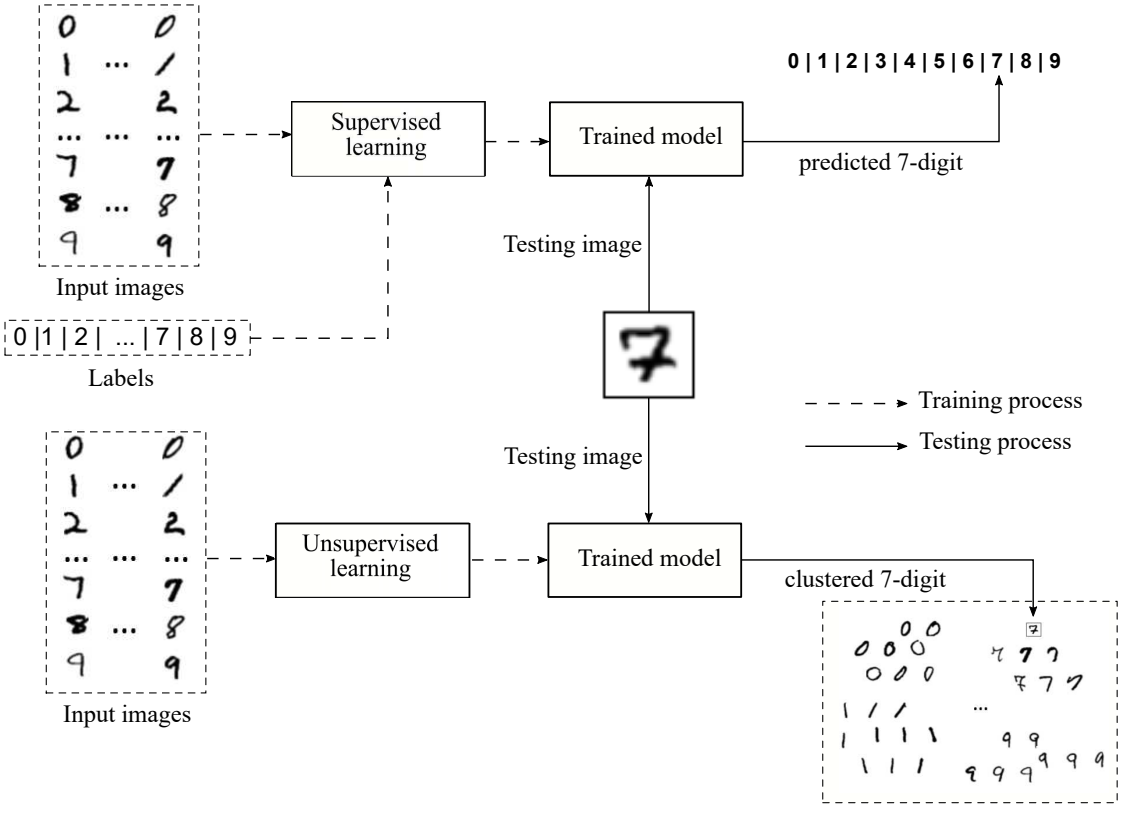


Fig. 3 Hand-written digit recognition using supervised and unsupervised learning.

Fig. 3 illustrates the supervised and unsupervised leaning through the task of hand-

written digit recognition. Considering a dataset consisting of hand-written digit images of 0 to 9, the goal of that task is to identify the digit in a testing image. To achieve this goal, the dataset is first divided into two sets for training and testing purposes. During the training process, supervised algorithms require labels for input images, while unsupervised ones do not. When it is done, the trained model is then used to predict the digit of a new testing image. As shown in Fig. 3, the supervised model categorizes the test image into one of ten classes, whereas the unsupervised model assigns the test image to the closest cluster. These tasks are known as classification and clustering, which represent supervised and unsupervised learning, respectively.

Recently, the machine learning research community has become increasingly interested in an alternative to the supervised and unsupervised learning, called reinforcement learning. The reinforcement learning is a class of algorithms designed to address specific problems involving an agent interacting with an environment that provides rewards and/or punishments. The goal of the agent is to learn a strategy for selecting actions that maximizes cumulative rewards over time. Fig. 4 describes a robot navigation system where the robot is referred to as an 'agent', and the map is considered as the 'environment'. At each step t of the training, the robot drives an action A_t based on the current environment state S_t and a reward R_t. The reward is a value that can be positive or negative resulting in a previous action A_{t-1}. After doing the action, the environment changes to a new state S_{t+1}, and a new reward value R_{t+1} is obtained. The process repeats with trial and error, the aim is to learn an optimal policy that maximizes the total accumulated reward over time. It is widely used in such applications as robotics, self-driving cars, or computer games [113].

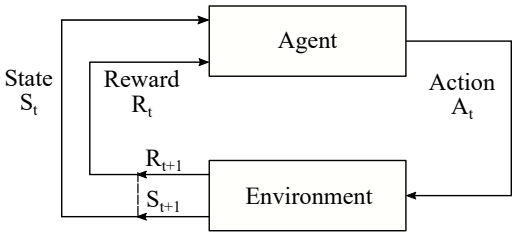


Fig. 4 A robot navigation system using reinforcement learning.

Computer vision. Together, image processing and machine learning provide the essential framework for advancing an interdisciplinary research field, namely computer vision. At the highest of its ambition, computer vision aims to replicate human vision capabilities using digital images or videos as input. In particular, from basic operations (e.g., noise reduction, edge detection, etc.) to advanced methods (e.g., feature extraction, pattern recognition, etc.), the image processing techniques serve as the foundational tool for preprocessing images or videos in computer vision applications [26]. Meanwhile, machine learning plays an essential role in enabling computer vision systems to learn from data and autonomously improve their performance over time [78]. By training models on large datasets, machine learning algorithms are able to recognize complex patterns and representations to facilitate downstream tasks such as image classification, object detection, and semantic segmentation. Interestingly, computer vision has recently attracted massive attention driven by the advancements in deep learning techniques, a subset of machine learning [113]. Indeed, a large number of studies can be found in leading computer vi-

sion conferences and journals[6] such as ECCV, ICCV, CVPR, IJCV. These conferences and journals serve as venues for researchers, industry experts to share their findings, promote collaboration, and shape the future of computer vision.

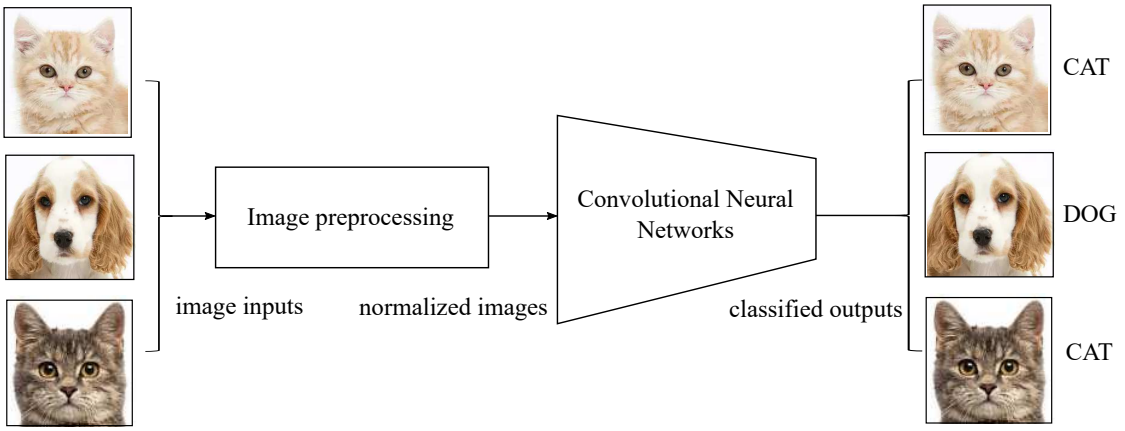


Fig. 5 An example to classify images of dogs and cats in computer vision.

Fig. 5 illustrates a typical example in computer vision where the goal is to assign a label (dog/cat) to an input image based on its visual content. To achieve this, images are usually processed with several image processing operations (e.g., resizing, color conversion) for normalization, followed by a convolutional neural network for training and prediction purposes. The network is trained on a labeled dataset consisting of thousands of images of dogs and cats. During the training phase, the network parameters are adjusted to minimize the difference between predicted and actual labels. In other words, the unique visual features of dogs and cats (e.g., eyes, nose, ear shapes, or fur patterns, etc.) can be recognized progressively. Once trained, the network can classify unseen images by computing probabilities for each class. The highest probability indicates that the image most likely contains either a dog or a cat. Nowadays, such tasks can be addressed with remarkable precision, nearly the discrimination capacities of human, by leveraging the advantages of convolutional neural networks. They motivate a shift from feature engineering (e.g., SIFT, SURF) to network engineering (i.e., CNN-based features) in computer vision.

Performance evaluation for computer vision. Clearly, the computer vision technology has the potential to solve a wide range of tasks in real-life situations [27, 113]. However, it is important to understand the performance of computer vision systems in order to ensure that the assumptions made in developing the algorithm are matched with reality. It refers to the qualitative and quantitative analyses of performance. A series of fundamental problems can be mentioned: (i) selection of synthetic/real input data in experiments related to specific domains and tasks, (ii) the generation of data that supports developers in the system configuration, (iii) the definition of groundtruth given different tasks and applications, (iv) the design of experimental sets, (v) the algorithm choice and uses of hyper-parameters, (vi) the analysis of algorithms with respect to general characteristics (e.g., complexity, resource consumption), and (vii) the definition of performance

[6]European Conference on Computer Vision, International Conference and Computer Vision, Computer Vision and Pattern Recognition, International Journal of Computer Vision

measures for each algorithm (e.g., accuracy, precision, recall, etc.). Thus, performance evaluation and characterization are key important aspects that must be comprehensively assessed when we design computer vision systems. This subject has been intensively discussed in the literature recently [52, 100].

The PhD topic and organization of the thesis

In this thesis, we propose to address the problem of partial video copy detection with a specific focus on evaluating its performance. The rest of this thesis is organized into three chapters, followed by our conclusion and perspectives. A brief introduction of these chapters is given hereafter.

- **Chapter 1**: The main goal of this chapter is to present a detailed review of state-of-the-art studies in the PVCD. The main ideas, advantages, and weaknesses of each method are first presented. Then, the key properties of each benchmark dataset used to evaluate the PVCD performance, the connection among them, and the shortcomings are carefully discussed. Next, performance evaluation of the PVCD is discussed on the datasets with standard metrics. The main results are given in terms of effectiveness, efficiency and scalability. By doing this investigation, the needs of more challenging datasets for performance evaluation on the PVCD are given as the conclusion of this chapter.

- **Chapter 2**: As was mentioned in Chapter 1, several existing datasets are designed from web videos. The PCVD problem is inherent to the continuous video broadcasting. The alternative is then to process with TV datasets offering a deeper scalability and a control of degradations for a fine performance evaluation. We propose in this chapter a TV dataset called STVD. It is designed with a protocol ensuring a scalable capture and robust groundtruthing. STVD is the largest public dataset on the task with a near $\simeq$ 83 000 videos having a total duration of 10 660 hours. Performance evaluation results of representative methods on the dataset are reported in the chapter for a baseline comparison.

- **Chapter 3**: 2D CNN are main components for the PVCD. 2D CNN-based features serve for the retrieval and matching of videos. Robustness is a key property of these features. It is a well-known problem in the computer vision field but little investigated for PVCD. The contributions of this chapter are twofold: (i) based on a public video dataset, we provide large-scale experiments with 700 billion of comparisons of 4.4 million feature vectors. We report conclusions for PVCD consistent with the state-of-the-art. (ii) the regular protocol for performance characterization is misleading for PVCD as it is bounded to the video level. A method for the characterization of key-frames with 2D CNN features is proposed. It is based on a goodness criterion and a time series modelling. It provides a fine categorization of key-frames and allows a deeper characterization of a PVCD problem with 2D CNN features.

At last, the main contributions of this thesis are summarized. The merits and limitations of this work are also reviewed. Perspectives are given for future work.

Chapter 1

A state-of-the-art review on the performance evaluation of PVCD

Contents

1.1 Introduction

This chapter reviews state-of-the-art studies on the performance evaluation of PVCD. They are organized in three main aspects. A brief state-of-the-art of the PVCD systems is first presented in Section 1.2. Then, we discuss the state-of-the-art of the PVCD performance evaluation in more detail in Section 1.3. Finally, Section 1.4 discusses our subjective remarks on the shortcomings of current benchmarks, and highlights the needs of advanced contributions that will connect to our following chapters. For convenience, the general terminology conventions used for the PVCD are given on page 15, and Tab. 1.1 describes the main symbols and terms used in this chapter.

Tab. 1.1 Main symbols and terms used in the chapter.

Symbols	Meaning
i, j	are indexes of an array, a vector.
HI(A,B)	is a histogram intersection between two videos A and B.
CS(Q,R)	is a Cosine similarity between two vectors Q and R.
ED(Q,R)	is a Euclidean distance between two vectors Q and R.
DP(Q,R)	is a dot product between two vectors Q and R.
r_i	a reference with $i \in [1, q]$
v_i^j, p_i^k	two different positive videos ($j \neq k$) of the same reference r^i
$P(n, 2)$	2-permutation of n videos, computed by $\frac{n!}{(n-2)!}$
$C(n, 2)$	2-combination of n videos, computed by $\frac{n!}{2!(n-2)!}$
$(\mathbf{C})$	the total number of positive pairs
P, R, F_1	are Precision, Recall and F_1 metrics, respectively.
mAP, AP	mean Average Precision and Average Precision

1.2 A brief state-of-the-art of the PVCD systems

The PVCD system processes videos according to each specific application scenario (e.g., retrieval, detection, monitoring). Thus, each PVCD may consist of different components that play different roles in video processing. Fig. 2 illustrates a general framework of the PVCD system. In this section, we select representative studies to discuss each component of the system in more detail. Particularly, an overview of the PVCD systems is first presented in the subsection 1.2.1 to establish the intersection between them. Based on a general PVCD framework, we then select representative works for illustrating algorithms and methods that have been used in each component of the PVCD system. These components are interconnected as a whole and each of them is related to the previous one. Thus, they are discussed individually in the subsections 1.2.2 to 1.2.4.

1.2.1 An overview of PVCD systems

A no-training framework. The most common approach is to divide the complex PVCD system into several smaller components as illustrated in Fig. 2. A typical PVCD system consists of three major components including feature extraction, generation of video representations, and video comparison [46, 62, 107]. An optional component can be used for the more specific partial detection (e.g., boundaries of copied video segments).

Basically, given a video database, frames are first sampled from videos, and their features are then extracted. Based on these features, a video summarization can be performed to generate compact and distinctive representations in order to serve for detecting partial video copies. They can be formed as a single vector (video-level) or multiple vectors (frame-level). Thus, the different structural levels of videos (e.g., a whole video, segment, frame) can be taken into account for the retrieval or detection purposes. Today, video databases are increasing rapidly with exponential growth. To speed up the search process, these video representations are normally organized by effective indexing structures. The above steps are typically handled in an offline process. Similarly, given a query/testing video, its representations are first generated in the same manner as in the video representation component. The obtained query/testing representations are then searched in the database index to detect near-duplicate videos based on similarity measures. For example, the video in the database is considered as the copy of the query when the similarity between two video representations is greater than or equal to a predefined similarity threshold - ϵ. This process is normally done online. The offline and online processes with three components (i.e., feature extraction, generation of video representations, video comparison) are the main principles to design a typical PVCD system. In an extended scenario, the boundaries of copied video segments (i.e., the starting and ending timestamps of copied segments) in the temporal domain are needed for a more specific partial detection. This can be done with the support of some specific techniques such as graph/network mining, voting systems, or dynamic programming [29, 47, 61].

A deep learning framework. With the growth of data volumes and the development of machine learning algorithms, recent studies have proposed new approaches for designing the PVCD system [30, 47, 56]. Generally, they are motivated by the outstanding performance of deep learning algorithms, which are a subset of machine learning.

In the deep learning framework, we can roughly classify the approaches into two categories. The first approach uses a pre-trained deep learning model, which has been trained on a large image dataset in order to serve as a video feature extractor in the PVCD system. It refers to a fine-tuning technique, that allows the model to leverage previously learned features while adapting to the PVCD [27, 113]. For example, the work [114] used a pre-trained VGG-16 model, which was trained on the ImageNet dataset for an image classification task, to extract video features. Within such a framework, no training process is required and this can be adapted and used in the above framework by adapting the feature extraction component [45, 55, 104, 122].

In addition to the use of pre-trained models, recent new deep learning models can be developed to learn the similarity/dissimilarity between videos [47, 56]. The typical of these models is a Siamese neural network that involves two or more identical subnetworks (i.e., they share the same architecture and the same parameters). The core idea of the Siamese is to train a model to identify similarities or differences between data points by pulling similar pairs to be close in the feature space while pushing dissimilar pairs apart. In such a twin model, copy and non-copy video pairs are used as inputs. Particularly, two random videos from different classes are selected as a negative pair (non-copy video pair). For copy pairs, video transformations are applied to the original video to generate a positive pair (copy video pair). Each subnetwork processes one of the videos to extract meaningful features. The outputs of these subnetworks are then compared using a distance metric (e.g., the

Euclidean distance), to quantify the similarity between the videos. During training, the model weights are being updated to minimize the distance between similar video pairs and maximize the distance between dissimilar ones. When training is done, the trained Siamese network can robustly capture and measure video similarity. Within a Siamese network, the use of loss functions like contrastive loss and triplet loss is normally applied, and that deep learning technique is referred to as contrastive learning.

Another line of research involves a hybrid framework in which a pre-trained CNN model serves as a feature extractor, and a new deep learning model is developed to detect copies from matching frames through a training process [30, 39, 54]. In this framework, a video is represented with multiple feature vectors in the temporal order. These features are normally extracted from strong standard pre-trained CNN models. A video similarity matrix between two videos is then generated based on the frame-to-frame matching scores (e.g., Cosine similarity, dot product). A neural network is developed to capture or enhance these spatial and temporal structures within the similarity matrix. To detect the boundaries of the copied segments, an object detection model (e.g., YOLO model) can be adopted to estimate the locations as temporal bounding boxes.

We have presented here two representative PVCD frameworks. Among them, three main common components (i.e., feature extraction, generation of video representations, and video comparison using standard similarity measures) can be shared together. In the following sections, the state-of-the-art studies in the PVCD are discussed with a focus on the main components of the PVCD framework.

1.2.2 Features extraction

A variety of feature extraction methods have been proposed in the literature. They aim to extract informative and distinguishing characteristics from video content. Generally, these methods are mainly derived from the studies of image processing and retrieval due to videos are composed of a large number of consecutive frames. Feature types can be roughly classified into three groups including global, local, and deep learning features.

Global features refer to the statistical information of images. The color histogram (e.g., RGB, HSV histograms) is the most commonly used in the literature [11, 15, 89, 93, 94, 107]. It can be obtained by concatenating the counts of the pixels whose color values fall within the corresponding range. The color histogram is compact and computationally efficient. However, it contains no geometric, shape, or texture information and is sensitive to color changes. For example, if two videos generated from different objects or scenes have similar color patterns, they might produce similar histograms, consequently making them difficult to detect. For these reasons, color histograms are preferred to retrieve videos that are almost identical (e.g., repeated content, or duplicate videos) to the query video with minor transformations such as minor changes in brightness, or contrast.

Other works use the ordinal measure which describes the pairwise ordinal relations between blocks in terms of average gray-level values [9, 14, 87]. An ordinal image is partitioned into 3×3 blocks (resulting in 9 blocks and 2-combination $C(9, 2) = 36$ ordinal relations) and the average gray-level value in each block is computed. The ordinal measure maintains robustness in addressing the color degradation effect caused by various encoding

devices or illumination conditions. However, it does not consider the temporal structure of videos, thus, it is sensitive to the length of the video.

From the above results, global feature methods are generally effective and efficient calculations but only limited to tasks involving duplicate videos or real-time online requirements. In other words, they have significant limitations when dealing with complex scene changes or geometric transformations.

Local features detect interest points or regions from images to describe geometric and shape characteristics. They are mainly derived from advances in the image processing research community. Among various studies, a number of different detectors and descriptors are available such as the Harris corner features [49, 63], scale invariant feature transform (SIFT) [46, 47], speeded up robust feature (SURF) [14, 15], and local binary pattern (LBP) [31, 93, 94]. Compared to global features, local features capture object-level information and are invariant to scale and affine transformations. They can offer good performance, but they are computationally expensive when matching a large number of local features for copy detection (e.g., the number of local points could reach hundreds to thousands per image). To balance between speed and robustness, the principal components analysis (PCA) technique can be applied to generate a variant of SIFT, named PCA-SIFT [98].

Relying fully on either the global or local features could lead to an inadequate result in video processing. To get improved results, multiple video features are extracted, which contain a combination of global and local features of the video. Basically, the fusion feature is constituted by concatenating different types of features with weights into a unified space. For instance, a combination of the global feature using HSV histogram and the local feature using LBP descriptor is proposed in studies [31, 48, 79, 93, 94]. In addition, other works have proposed to leverage the temporal information within videos in order to generate a spatio-temporal descriptor that contains both the spatial information of frames and the temporal relations among frames. The temporal structure can be modeled by tracking key points temporally along with the video sequence [87, 115]. Generally, global-local or spatio-temporal features can offer richer video representations, although an additional computation is inevitable with such approaches.

In summary, tens of feature types have been applied for the PVCD topic. They can be modeled in a single form using either global or local features, or in multiple forms by combining global-local or spatial-temporal features. While global or local features can deal with specific video degradations, combined features are a trade-off strategy for system performance. Therefore, depending on the outcome of the PVCD system, the balance of effectiveness and efficiency should be comprehensively considered.

Deep learning features refer to features generated by deep learning algorithms which involve artificial neural networks with multiple layers in the field of machine learning. Among various algorithms, convolutional neural networks (CNNs) have shown outstanding performance compared to the previous approaches in many computer vision tasks including image classification, retrieval, and object detection [27, 113]. Motivated by this, deep learning features have been adopted and applied to the PVCD for the extraction of video features. The current popular CNN models include LeNet [66], AlexNet [59], VGG-Net [92], InceptionNet [97], and ResNet [33]. Based on these models, the image features are generally extracted from the convolutional or fully connected layers, which are two major

components of a CNN network.

As a straightforward way, considering a pre-trained CNN model which has been trained on a large dataset (e.g., ImageNet [59]), the video features can be extracted from any fully connected layers except for the last layer, which is used as a classifier. For instance, in a VGG-16 network consisting of 13 convolutional layers (1 - 13) and 3 fully connected layers (14 - 16), either the output of layer 14 or layer 15 (not 16) can be considered as the image feature. The same mechanism can be applied to different CNN models. This technique is well-known in the computer vision field, thus it has been widely investigated in the PVCD studies [38, 45, 47, 53, 55, 56, 65, 74, 104, 117]. Based on the findings of the studies, features extracted from the fully connected layers in a CNN demonstrate a competitive performance compared to studies using traditional features (e.g., SIFT, SURF features).

Another research direction is that the deep learning features can also be obtained from the activations of the convolutional layers in a CNN. For a particular object or scene image retrieval task, a higher performance could be obtained by using the features extracted from convolutional layers rather than fully connected ones [101]. Inspired by this work, a considerable amount of literature has been published on the PVCD topic [5, 30, 35, 36, 43, 53, 54, 55, 56, 88, 99, 103, 114, 119, 122]. However, these studies have a variety of ways to extract features from videos. In general, several factors should be considered experimentally such as: (i) the name of the used CNN, (ii) the version of the CNN (e.g., VGG-16/19, Inception-v1/v3), (iii) the selected convolutional layer (e.g., ResNet-50/152 layers), and (iv) the method used to extract features (e.g., average/max, layer-aggregation). Different results have been discussed and reported using this approach [5, 55, 114].

The CNN-based features exhibit a strong ability to address various types of image transformation. However, it also has great limitations in video processing due to the fact that temporal information is neglected in a CNN. To fill this gap, many deep learning algorithms are designed to learn the spatio-temporal representations for videos. Different from CNN, deep 3-dimensional convolutional networks (3D CNN) which use 3D convolution operations on three channels are used in recent studies [53, 102]. In addition, a recurrent neural network (RNN), which is a type of neural network architecture designed to handle sequential input data, is used to stack a series of frames as an input [38, 99]. Another type of neural network, namely graph neural network (GNN), can be applied to represent the relations between spatial and temporal dimensions in videos as [74, 119].

Overall, no single technique and no general feature descriptor seem to be optimal for all the PVCD systems. The feature extraction techniques are based on different features (i.e., global, local, deep learning) with different capacities against video transformations. Particularly, global features are fast but limited against complex video transformation attacks, whereas local features are more robust but computationally expensive. Deep learning algorithms can provide automatic and highly generalizable features, but they are data intensive and less interpretable (e.g., understanding which parts of a certain image/video are important).

1.2.3 Generation of video representations

After video feature extraction is done, the set of features is processed to generate compact and robust video representations for retrieval scenarios and purposes. However, different studies focus on different aspects of the video. Indeed, a video is composed of a sequence of consecutive frames as illustrated in Fig. 1.1. At its core, an image/frame is the smallest element of a video, while a segment is defined by a group of adjacent frames that are semantically correlated to the same subject or scene. A video is composed of one or more segments. Existing works can be classified into two representation categories: video-level and frame-level representations [43, 47, 74]. Video-level representation approaches summarize a whole video with a single representation vector, regardless of any individual frame information. Frame-level approaches represent a video with multiple vectors and calculate the video similarity frame-by-frame. In the following, we discuss each type of video representation in more detail.

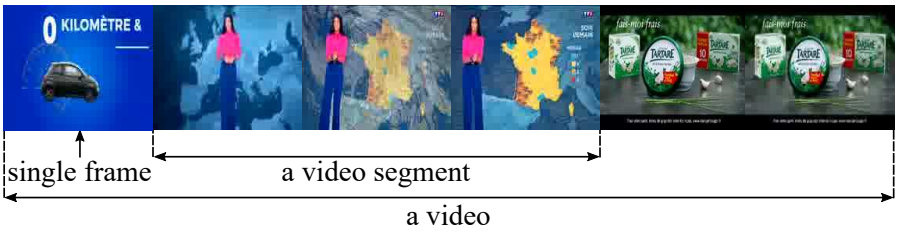


Fig. 1.1 An example of the video structure consisting of frame and segment elements.

Video-level representation approaches summarize a whole video with a single representation vector. Such approaches reduce the problem to video representation vectors in high-dimensional spaces, thus allowing them to be efficiently stored, indexed, and retrieved.

A video representation can be obtained by using aggregation schemes such as various histograms, hashing, clustering, principal component analysis, or bag-of-words. Particularly, in work [107], the authors defined a video representation as a m-dimensional vector (i.e., m is the number of bins) of an accumulative HSV histogram over all key-frames in the video. In [89], the authors proposed a video representation model called Bounded Coordinate System which extends the principal component analysis (PCA) of video features. A feature hashing framework based on a supervised method has been used in studies [48, 123]. In these works, they use multiple features and learn a group of hash functions that map the video features into the Hamming space. In works [9, 55], the bag-of-words and k-mean clustering algorithm have been applied to cluster all the video features and assign a visual word to each cluster. The final video representation is generated using the bag-of-words histograms of its features. A method for aggregating a set of local descriptors, followed by principal component analysis is proposed in the work [22].

For studies using deep learning algorithms, the video-level representation can be obtained by concatenating or averaging methods. In [45, 55, 56, 88], they extracted CNN-based features from intermediate convolution layers and concatenated all layer vectors to a single representation. A post-processing step was applied by using zero-mean and L_2-normalization. In [44], the video representation was obtained by averaging all its features extracted from the fully connected layer of the pre-trained AlexNet model.

Frame-level representations can offer a fine-grained descriptor compared to the video-level ones. A video is represented by multiple feature vectors and the video similarity is generally calculated by comparing between these individual features.

A simple approach is to stack all of the obtained features extracted from the key-frames of a video in the previous step to generate a frame-level representation. However, such an approach could result in high time complexity when matching a large number of representations. To generate compact representations, the PCA technique on the SIFT local features (i.e., PCA-SIFT) can be applied as demonstrated in works [98, 107]. Additionally, several works proposed a method that combines multiple types of features [5, 21, 31, 76, 93, 94]. The reason is that local features can capture the spatial information while global ones can carry the temporal information. In particular, the authors of the works [31, 93, 94] proposed a method to generate a frame-level video representation by combining the HSV color histogram characterizing the global feature and LBP features characterizing the local feature. Bag-of-words is another representation pattern to represent a video [14, 76]. In these works, frames containing similar visual features are clustered by the k-means clustering algorithm, and each cluster is assigned a unique visual word. All the visual words are then used to build a visual vocabulary. By doing that, a frame can be represented as a histogram of the occurrences of the visual words in that frame. Thereby, each video can be represented as a sequence of histograms.

Studies using deep learning features aim at generating spatio-temporal representations in order to represent a video. Particularly, the spatio-temporal representations are generated by using Fourier-based representations [5], transformer-based neural networks [36, 88, 99], a multi-attention mechanism in neural network architectures [103] for temporal aggregation. Recently, several works have developed neural works to learn parametric matching functions to estimate the video-to-video similarity [30, 35, 43, 53, 57, 58]. They design a video similarity network that captures spatial and temporal structures within pairwise video similarity matrices which are built on frame-to-frame similarity scores.

All of the studies reviewed here indicate that the generation of video representations attempts to reduce data to accelerate search response or improve the robustness of individual video features. Depending on the chosen scenario or purpose, different works focus on different aspects of the video (i.e., frame-level, video-level). For example, a video monitoring application requires high time efficiency while a video copyright protection needs accurate detection. Therefore, the generation of video representations should be adapted to such requirements for a specific application.

1.2.4 Video comparison

Generated video representations are referred as video embedding vectors. They are used to measure the similarity between a testing/query video and videos in the database at different levels, depending on how the video representations were generated from the previous component (e.g., frame-level, video-level). In the following, we will first review the common similarity measures that have been used to compare video representations. Several representative methods to identify the boundaries of the copied video segments for accurate detection are then discussed.

Video similarity measure involves a real-valued function that qualifies the similarity between two objects in statistics and related fields. When applied to the PCVD, these measures are used to compute the similarity between the query video and the database videos based on video embedding vectors. The studies are commonly used standard measures such as histogram intersection [87] Hamming distance [14, 36, 39, 46, 93, 94], Cosine similarity [9, 15, 34, 43, 47, 55], Euclidean distance [14, 61, 104, 107, 108], dot product [5, 22, 30, 35], learning metrics [38, 43, 54, 88, 102].

Histogram intersection $\mathrm{HI}(A, B)$ can be used to measure the similarity between two videos A and B consisting of N embedding vectors. The similarity between A and B is computed as in Eq. 1.1, where $h_{A,l}$ and $h_{B,l}$ are the values of the $l-th$ bin of the two histograms, L is the total number of bins, and N is a total number of embedding vectors. The bigger value $\mathrm{HI}(A, B)$ is more preferred.

$$\mathrm{HI}(A, B) = \frac{1}{N}\sum_{l=0}^{L-1}\min(h_{A,l}, h_{B,l}) \tag{1.1}$$

Hamming distance $\mathrm{HD}(Q, R)$ between two n-dimensional vectors Q and R is the number of positions at which the corresponding symbols are different. Mathematically, if Q and R are two feature vectors of length n. The Hamming distance $\mathrm{HD}(Q, R)$ is given as Eq. 1.2, where the binary function $\mathbb{1}(R_i, Q_i)$ is 0 if $R_i = Q_i$, and 1 otherwise. Q_i and R_i are the $i-th$ elements of vectors Q and R, respectively.

$$\mathrm{HD}(Q, R) = \sum_{i=0}^{n-1}\mathbb{1}(R_i, Q_i) \tag{1.2}$$

Cosine similarity $\mathrm{CS}(Q, R)$ primarily considers the angle between two n-dimensional vectors Q and R to determine their similarity and disregards the length of the vectors. It is calculated using the dot product of both vectors and the product of their lengths. The formula is given in Eq. 1.3, where Q_i and R_i are the $i-th$ elements of vectors Q and R, respectively. The $\mathrm{CS}(Q, R)$ value ranges from -1 (not similar) to +1 (very similar).

$$\mathrm{CS}(Q, R) = \frac{Q \cdot R}{\|Q\|\|R\|} = \frac{\sum_{i=0}^{n-1} Q_i \cdot R_i}{\sqrt{\sum_{i=0}^{n-1} Q_i^2}\sqrt{\sum_{i=0}^{n-1} R_i^2}} \tag{1.3}$$

Euclidean distance, especially in the n-dimensional space, is a measure of the straight line distance between two vectors in that space. Particularly, two vectors $Q = (Q_0, Q_1, \cdots, Q_{n-1})$ and $Q = (R_0, R_1, \cdots, R_{n-1})$ with the size of n, the $\mathrm{ED}(Q, R)$ is given in Eq. 1.4, where Q_i and R_i are the $i-th$ elements of vectors Q and R, respectively.

$$\mathrm{ED}(Q, R) = \sqrt{\sum_{i=0}^{n-1}(Q_i - R_i)^2} \tag{1.4}$$

Dot product and Cosine similarity are closely related concepts. Indeed, the dot product DP(Q,R) between two vectors Q and R can be derived from the Cosine equation 1.3 by multiplying the cosine of the angle between two vectors CS(Q,R) by the lengths of both vectors. In a special case, when both vectors are normalized with L_2 normalization, the dot product DP(Q,R) is equal to the cosine similarity CS(Q,R) (i.e., $\|Q\| = \|R\| = 1$).

$$\mathrm{DP}(Q, R) \quad = \quad \|Q\| \cdot \|R\| \cdot \mathrm{CS(Q,R)} \tag{1.5}$$

Learning metrics refer to approaches that apply deep learning algorithms to develop a video similarity network. These networks learn parametric matching functions to estimate the video-to-video similarity. At its core, the cosine similarity, euclidean distance, or dot product are mainly used to build the video-to-video similarity matrix.

Overall, there are various measures for calculating the similarity between videos. They are connected to the methods used to generate video representations. For instance, the Hamming distance can be effective to the hashing representations while Euclidean distance or Cosine similarity is more widely used in the spatio-temporal representations. When the video features are processed with L_2 normalization, the dot product can be used. At last, the video similarity neural networks have been proposed recently to estimate the similarity between videos automatically.

Temporal alignment. Besides determining copied videos, another task of the PVCD is to identify the copied segments (i.e., starting and ending timestamps) between two videos. It refers to checking the temporal consistency of matching frames for both videos to locate the copied segments [14, 38, 46, 47, 71, 98]. In particular, the temporal Hough voting method was applied to detect copied segments by searching peaks of matching scores within a fixed window size of timestamps [21, 46, 47]. An alternative is to use the temporal network [38, 46, 98, 99, 104] with a graph that considers the matched frames and their similarities as nodes and edges, respectively. The partial copy segments are then computed as a network flow optimization problem to find the longest path as the copied segment. The other works [14, 71] applied dynamic programming to extract the diagonal blocks containing the largest similarity in the frame-to-frame similarity matrix.

Recently, machine learning algorithms have been also applied to align and detect matching frames for the PVCD [5, 30, 35, 39, 43, 74]. The authors [5] used a circulant temporal network encoding to align matching videos based on the Fourier-domain representations with temporal matching kernels and trainable coefficients. The authors [74] proposed a Graph Neural Network (GNN) with self-attention and cross-attention mechanisms to detect partial copies. Another approach considers copied video segments as bounding boxes drawing from the frame-to-frame similarity matrix. Then, the object detection models (e.g., RefineDet, YOLO model) are used to detect these bounding box patterns [35, 39, 43], or to predict the probability of frame pairs lying right on the partial alignments [30].

1.3 A state-of-the-art of the PVCD performance evaluation

We have presented in the previous section a brief state-of-the-art PVCD systems. We have provided an overview of the global architecture of PVCD systems and how the de-

tection is processed at different stages from feature extraction to video comparison. This state-of-the-art highlights that PVCD systems could handle a large amount of video degradations in the spatial and as well temporal domains. However, a key problem is to identify the real performances and limits of detection for these systems. As detailed in the introduction chapter, this is related to the research field of performance evaluation in computer vision. We provide in this section a state-of-the-art of the PVCD performance evaluation. We first present in detail the PVCD datasets used in the literature in the subsection 1.3.1. Then, the common performance characterization metrics for PVCD are discussed in the subsection 1.3.2. At last, the subsection 1.3.3 reviews the main detection and timing performances of the state-of-the-art PVCD systems.

1.3.1 The PVCD datasets

Several datasets have been proposed in the literature for the PVCD performance evaluation. They are listed in Tab. 1.2 sorted by year of publication. These datasets are mainly designed to characterize the task of near-duplicate video retrieval/detection. They typically comprise two main sets of (i) references and (ii) testing videos. The reference set contains information about the representative partial copies (i.e., the original sources). The testing videos are derived from these representatives with various transformations. The goal is to identify whether two testing videos are derived from the same representative source, regardless of possible transformations. In the following paragraphs, we will present the organization of these datasets.

The reference set consists of the original sources that represent partial copies. Depending on the database design, they can be formed in text [30, 35, 46, 107] or video formats [45, 62, 81]. Both of them are statistically popular references to ensure reliability and diversity (e.g., the most popular text queries on Youtube, the videos with millions of 'likes' by Tiktok users, etc.). Another approach is to establish a large-scale reference video database by capturing very long or continuous videos (e.g., several hours) [11, 49, 61]. In that case, the references have no domain meaning (i.e., random selection) and are hidden in the long videos. We denote the 'References' column in Tab. 1.2 with a 'n/a' value for this approach. Based on these references, their corresponding transformed videos are collected from the Internet or generated from the reference videos to constitute the testing set.

Tab. 1.2 Datasets used in the literature. The 'References' column indicates the number of representative partial copies. It can be either text or video queries, depending on the database design. The 'n/a' value in the column denotes that the references of that dataset are continuous or long videos (several hours). 'Positive videos' contain at least one video segment derived from a reference. 'Negative videos' include distraction videos for the testing purpose. A pair of positive videos contains one or more copied segments, whereas a pair of copied segments contains the start and end timestamps of copies for both the positive videos. (n/a: not available, s: seconds, h: hours, m-h: man-hour).

Datasets \ Description	Year	References	Testing videos		Positive video pairs	Positive segment pairs	Copied segments	Average duration (s)	Total duration (h)	Metadata	Average resolution	Degradation methods		Annotation cost(m-h)	Annotation level	Publicly available
			Positive videos	Negative videos								real/ synthetic	Type of degradations			
TV_2007 [49]	2007	100	100	100	n/a	n/a	100	n/a	30 000	category	352×288	synthetic	capture, editing	n/a	segment	No
BBC_2007 [61]	2007	n/a	10	n/a	n/a	n/a	132	n/a	6.0	category	n/a	synthetic	capture, editing	n/a	segment	No
MUSCLE-VCD [62]	2007	101	13	5	n/a	n/a	21	3 025	100	queries description	n/a	real, synthetic	capture, editing	n/a	video, segment	Yes
CC_WEB_VIDEO [107]	2007	24	3 481	9 309	n/a	n/a	n/a	151	537	title, thumbnail, duration, view count	352×264	real	networking, editing	n/a	video	Yes
TRECVID [81]	2010	134	1 072	n/a	536	n/a	1 072	120	420	title, keywords, description	n/a	synthetic	capture, editing	n/a	video, segment	No
UQ_VIDEO [93]	2011	24	3 481	129 142	n/a	n/a	n/a	n/a	n/a	n/a	n/a	real	networking, editing	n/a	video	No
TV_2014 [11]	2014	n/a	$1{\times}10^7$	n/a	n/a	n/a	$34{\times}10^6$	n/a	380 000	duration, category, timestamp	360×288, 448×256	real, synthetic	editing	n/a	segment	No
VCDB [46]	2014	28	528	100 000	6 139	9 236	n/a	72	2 000	category	n/a	real	capture, editing	700	segment	Yes
SVD [45]	2019	1 206	n/a	526 787	10 211	n/a	n/a	17	2 705	format, duration,	240×426	real, synthetic	capture, editing	800	video	Yes
FIVR-PVCD [30]	2021	100	n/a	n/a	5 935	10 870	n/a	113	n/a	duration, category, timestamp	n/a	real	networking, editing,	n/a	segment	Yes
VCSL [35]	2022	122	9 207	n/a	167 508	281 182	n/a	364	17 416	category, keywords	n/a	real	networking, editing	20 000	segment	Yes

The testing set includes negative and positive videos. The negative videos do not contain any references, consequently serve as distraction videos. The positive video contains one or more video segments that are derived from a set of references. The testing videos may be resulted from different sources of degradation as illustrated in Fig. 1.2. These degradations cover the full video processing pipeline from capture (e.g., different sensors, perspective modification, blurring, contrast alteration, video encoding, etc.), video networking (e.g., video speeding, frame removal, compression, etc.), and geometric editing operations (e.g., logo/text adding, downscaling, flipping, black border, etc.). Depending on the designed protocol, the degradations are resulted from real/user-generated video data [47, 93, 107], or obtained with synthetic [49, 62, 81], or even mixed methods [11, 45].

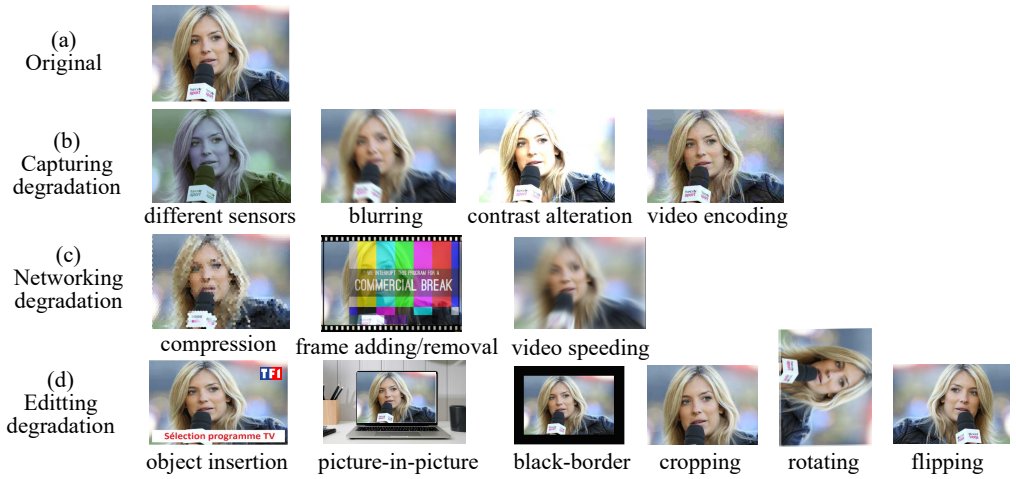


Fig. 1.2 Common video degradations: (a) original/standard version, (b) capturing degradation, (c) networking degradations, (d) editing degradations.

Along with videos, a groundtruth file identifying copied videos is provided to evaluate PVCD approaches. It can be categorized into 2 types: video-level and segment-level annotations, in which it is shown in the 'Annotation level' column of Tab. 1.2. While the video-level annotation indicates whether a video is copied or not, the segment-level annotation provides additional information on the boundaries of copied segments (i.e., the start and end timestamps). All of the existing datasets provide the segment-level with a timestamping precision of 1-second. By doing this, the total number of annotated cases is given in the 'Copied segments' columns of Tab. 1.2.

The groundtruth files are commonly generated in a manual or semi-automatic way with human support. Given a list of references, their corresponding candidate videos are first selected to detect copies. A 'positive' record in the groundtruth file is introduced by concatenating a reference video and a corresponding candidate video, if copied segments are detected and validated by users [45]. A desirable property is to distribute unbalanced numbers of positive videos between the references. First, this corresponds to the real-life use cases. Next, it stresses the systems and machine learning algorithms against the problems of scalability and accuracy for detection. To enhance this property, the metric of positive pairs given in Eq. 1.6 was introduced in [46] and has been used to characterize all the recent datasets [30, 35, 45, 46].

$$(\mathbf{C}) \quad = \quad \sum_{i=1}^{q} \frac{1}{2} P(n_i, 2) = \sum_{i=1}^{q} C(n_i, 2) \tag{1.6}$$

This metric Eq. 1.6 is applicable for counting the pairs of videos and/or segments, even if the common trend in the literature is to provide for the two cases [30, 35, 46]. Tab. 1.2 distinguishes the two, we illustrate here the Eq. 1.6 for the case of video pairs. Considering 2 arbitrary different positive videos v_i^j and v_i^k ($j \neq k$) having the same reference r_i, a pair of videos is defined as $(v_i^j, v_i^k) = (v_i^k, v_i^j)$. The total number of positive pairs for the reference r_i is the half[1] of the 2-permutation of its n positive videos $\frac{1}{2}P(n, 2)$. This is equivalent to the 2-combination of n positive videos $C(n, 2) = \frac{1}{2}P(n, 2)$. Similarly, for all the references r_i, with $i \in [1, q]$, the total number of positive pairs ($\mathbf{C}$) is the overall sum of 2-combination (see Eq. 1.6). This metric is dependent on the distribution of $[n_1, \ldots, n_q]$ values where ($\mathbf{C}$) is maximized for the exponential case and minimized for the uniform one. The number of positive pairs ($\mathbf{C}$) is shown in Tab. 1.2 for each dataset, where we detail the cases of videos and/or segments if available. The number of positive pairs of segments is still greater than the one of videos according to the dataset. In the recent dataset [35], this number has increased by nearly an order of magnitude to address the scalability issue.

The videos are collected from the Internet or TV with some metadata (e.g., title, description, video category, etc.). In addition, most of the datasets are not publicly available due to industrial or intellectual property constraints. The following paragraphs will review and discuss one-by-one main datasets that are used for the PVCD topic. In the last paragraph, we will provide a global conclusion and comparison about these datasets.

TV_2007 dataset [49] is constituted by capturing TV videos. The testing set is processed with two distinct TV sources/streams in order to separate the capture of positive and negative videos. The capture of French TV channels is selected to generate a reference video database (i.e., original source) as well as positive videos. This positive capture is organized in different sizes based on the total number of hours (e.g., 100, 10 000, 30 000 hours). Within the positive capture, 100 short videos having a 15-second long have been first extracted randomly to serve as the reference videos. Then, a combination of 5 degradation methods (i.e., min = 0 and max = 5) is applied to each reference in order to generate the positive video. It results in the total of 100 positive videos. On the other hand, a foreign TV capture is established to get the negative videos. This hypothesis is to ensure that none of the negative videos appears in the reference videos. Similar to the positive mechanism, 100 short videos of 150 seconds ($\times$10 longer than a positive video) have been extracted from the negative capture to constitute the final testing set having a total of 16 500 seconds. To detect copies, the testing videos are used to match against the original videos with a temporal precision of 2 frames. Such a protocol is more suitable for the task of online video copy detection in which the video database consists of very long streams. Another main limitation of the dataset is the lack of diversity in video degradation methods. In addition, the obtained videos have no domain meaning as they correspond to random sequences. Besides the videos, none of the metadata is provided in the dataset.

[1] due to the symmetric property $(v_i^j, v_i^k) = (v_i^k, v_i^j)$

BBC_2007 dataset [61] is obtained from the BBC[2] archive. The collected videos are used to comprise a reference video database consisting of 79 videos (about $\simeq$ 3.1 hours). For the positive videos, two scenarios of degradation methods are applied randomly on short video sequences extracted from the database. In the first scenario, each 5-second video segment is processed with one of eight predefined degradation methods. For the second scenario, each video segment that has a random duration between 1 - 10 seconds is applied with a random combination of transformations. To evaluate the system, a long testing video is constituted by inserting some positive videos within a long video captured from an external sources to ensure reliability (i.e., the external videos do not appear in the BBC, as similar to the work [49]). As a result, 10 testing videos (9 videos of 15 minutes, 1 video of 30 minutes) are fed into the system to find the best matching with the original reference videos. Due to the small number and short duration of videos, this dataset does not fulfill the needs for performance characterization of the PVCD. In addition, no negative set is provided in the dataset. Moreover, the lack of diversity in video transformations and metadata are major limitations in this dataset.

MUSCLE-VCD dataset [62] is generated to deal with both the full and partial video copy detection tasks involving different video sources such as Web video clips, TV archives, and movies. A set of 101 videos is selected to constitute a reference set. From this set, the testing set of 18 videos is processed with several predefined transformations. They are then divided into two categories including 15 and 3 videos for full and partial video copy tasks, respectively. For the full video copy, 10 positive videos are generated by applying only a single predefined transformation. The length of copies varies from 5 minutes to 1 hour. The remaining 5 videos, referred to as the negative videos, are collected separately from the reference dataset. For the partial video copy, on the other hand, each video can contain several short video segments (i.e., 21 video segments) with randomly different transformations. These segments are manually annotated in the range from 1 to 60 seconds using a professional video editing software. It results in the final positive set of 13 videos (i.e., 10 and 3 videos for the full copy and the partial copy, respectively). To evaluate the system, a copied segment/video is detected if the testing video matches the groundtruth. As a common difficulty, the limited volume of data (i.e., 100 hours) cannot provide good generalization for the PVCD methods. Applying degradation methods to the captured videos can also make it difficult to control performance characterization finely.

CC_WEB_VIDEO dataset [107] is collected from videos on social media platforms (i.e., Youtube, Google, and Yahoo). First, 24 text queries are submitted to retrieve all short relevant videos, less than 10 minutes in duration, on those websites. Then, the most popular video for each query is selected manually and served as a reference video. The testing set of 12 790 videos is manually annotated at a video-level by 3 users. It results in 3 481 and 9 309 videos for the positive and negative sets, respectively. To evaluate the system, every reference video is compared with the testing videos. Such an approach is aimed at retrieving short videos with a full copy, which does not generalize to the PVCD problem. In addition, this dataset suffers from the lack of challenges in the number of reference videos and sophisticated transformations except for the user-generated method. Moreover, the manual annotation process is very time-consuming and expensive, which

[2]British Broadcasting Corporation (BBC): http://www.bbc.co.uk

prevents further expansion of this dataset.

TRECVID dataset/task of Content-based Copy Detection [81] has been released yearly since 2001 for many tasks including video content-based copy detection, instance search, known-item search, semantic indexing, surveillance event detection, and multimedia event detection. Among the TRECVID datasets, the task of video content-based copy detection was introduced between 2008 and 2011. We selected the TRECVID 2010 for this review, as it established the regular protocol on the task. The root video set contains about 11 200 videos which are collected from the Internet and TV. They are short videos with an average duration of 2 minutes, consequently resulting in a total duration of 400 hours. The testing set is generated based on 201 original short videos (i.e., less than 3 minutes) as well as 8 predefined video transformations. It is divided equally into 3 groups including a full copy, a partial video copy, and non copy. While a full video copy is generated from a whole video segment extracted from the reference set, a short reference video is embedded in a non-reference video to generate a partial video copy. The non copy video severed as a negative video is extracted randomly from the non-reference videos. To challenge participating teams, 8 video transformations are applied to the 201 original videos to generate the final testing set consisting of 1 072 and 536 videos for the positive and negative sets, respectively. Most of the testing videos have a short duration (i.e., between 10 seconds and 3.5 minutes) and some meta-data such as title, keywords, and description are provided in this dataset. Overall, a small size and copied segments with little domain meaning are major limitations of this TRECVID dataset.

UQ_VIDEO dataset [93] is an extended version of the CC_WEB_VIDEO. Basically, it is designed by adding 119 833 videos that serve as negative/distraction videos. The videos were collected based on the 400 most popular queries from Google Zeitgeist Archives[3] over 6 years since 2004. For each submitted query, a maximum 1 000 short videos ($\leq$ 10 minutes) are downloaded to generate the negative set. This resulted in a total of 129 142 negative videos. The reference and positive sets are derived from the CC_WEB_VIDEO dataset. Such an approach may raise questions about reliability due to the negative videos are unverified. Also, it results in a very unbalanced data distribution between the positive and negative sets. Moreover, the original videos are not provided, only the features of key-frames are introduced in this dataset.

TV_2014 dataset [11] is processed with the capture of 10 TV channels over a period of 4 years. This dataset is designed to detect the repeated content of TV broadcasting. During that period of time, the dataset generation and copy detection have been processed incrementally on the full captured video database. The capture is driven at low resolution for storage issues and the videos are organized as a set 24 video files of 1 hour per day and channel (a near total 380 000 numbers 1-hour video files for 4 years). The daily database is used to search against itself and compare to the database of all past days to detect copies. With such an approach, none of the reference set is given, and the negative video material cannot be presented as a number of videos but rather as a number of hours in this dataset. The repeated content is stored in the fingerprint format within two tables including copied segments and positive videos. It results in the total of $\simeq$ 34 million copied segments that are derived from $\simeq$ 10 million videos (i.e., positive videos). Besides a real noise resulting

[3]Google Zeitgeist Archive: https://archive.google.com/

from capture, common synthetic methods are used to measure the robustness. In addition to duration and timestamp information, only a few parts of the dataset are labeled with categories. Compared to the other datasets, it offers a deeper challenge for scalability. However, this dataset is not available, and thus it is no longer used in the literature.

VCDB dataset [46] is composed of online videos from Youtube and MetaCafe websites[4]. A reference set is constituted by 28 popular text queries amongst different categories such as commercials, movies, or news. These queries are then used to retrieve the relevant videos from the two websites in order to generate the testing set. That is, no original videos (reference videos) are provided and the PVCD task aims to detect partial copies within near-duplicate/transformed videos related to the same query. In particular, an average of 20 result videos per query are downloaded to constitute a core dataset that consists of 528 videos. Due to the lack of reference videos for detection, the groundtruth has been formalized and characterized as positive pairs of videos and segments (see Eq. 1.6). In this context, one or more segments of the positive video v_i^j can appear in the positive video v_i^k, with the same reference r_i and $j \neq k$. This has required 7 well-trained annotators working for a total of around 700 man-hours. As a result, a positive set containing 6 139 pairs of videos has been annotated. From these pairs of videos, 9 236 pairs of segments were labeled manually. On the other hand, 100 000 distraction videos were collected to serve as negative videos. Due to the annotation is time-consuming and expensive, the number of labeled videos both at the video-level and the segment-level is pretty small. In addition, such an approach could cause difficulties in extending the dataset. At last, collecting directly online videos results in a black-box model of video degradation, which cannot provide a fine control for certain PVCD characterization tasks.

SVD dataset [45] is obtained from the root video dataset containing over 100 million short online videos (≤ 60 seconds) uploaded on Douyin[5]. From the root dataset, the videos are divided into three sets including the reference, labeled and negative. Particularly, the reference set was first selected from 1 206 the most liked videos in November 2018. Based on the reference set, 34 020 candidate videos were used to detect copies. Similar to the approach of [46], the groundtruth has been formalized and characterized as positive pairs (see Eq. 1.6). It can also be applied to negative pairs in which a positive video v_i^j, having the query reference r_i, is compared to a false positive video v^k, with $j \neq k$, in the query that is not linked to r_i. As a result, 37 138 video pairs (i.e., 26 927 negative and 10 211 positive) have been detected and validated by users, requiring a total of about 800 man-hours for labeling. For the negative set, 700 000 videos were chosen to label by matching methods without human annotation. Indeed, video features (i.e., BSIFT, VGG-16 features) were used to calculate the pairwise similarity between reference and the negative videos. Videos with very low similarity compared to a predetermined threshold were selected to generate the final negative set consisting of 526 787 videos. The dataset offers $\simeq$ 3 000 hours of videos, with an average length of 17 seconds per video. To evaluate the system, 4 video degradations are applied to the testing videos to match against the reference videos. The video-level annotation and most videos having a short duration are the main limitations of this dataset considering the scope of the PVCD problem.

[4]Youtube: https://youtube.com, Metacafe: http://www.douyin.com (inactive since 2021)
[5]Douyin: http://www.douyin.com

FIVR-PVCD dataset [30] is a fine-grained version of the FIVR-200K dataset [53] by providing the boundaries of copied segments. Basically, the root FIVR-200K dataset consists of 100 references and 22 960 videos. Similar to the approach of [45, 46], the groundtruth has been formalized and characterized as positive pairs of videos and segments (see Eq. 1.6). Based on the root dataset, they have selected 5 935 positive video pairs to annotate manually at a segment-level. It results in the final positive set containing 10 870 pairs of segments. As the common issue for Web video datasets, this dataset shows many limitations and open questions in the context of performance evaluation for the PVCD. On top of that, the size is pretty small, and video files are not provided for this dataset.

VCSL dataset [35] is based on videos collected from Youtube and Bilibili websites [6]. As a general trend of online Web video datasets [30, 46], the reference set is constructed based on 122 text queries covering different popular topics. They are submitted to the websites in order to retrieve 9 207 relevant videos (roughly $\simeq$ 100 videos per query) for the annotation step. Similar to the approach deployed in studies [30, 45, 46], the groundtruth has been formalized and characterized as positive pairs of videos and segments (see Eq. 1.6). Indeed, 167 508 pairs of positive videos have been annotated with 281 182 pairs of segments, requiring 30 well-trained annotators and about $\simeq$ 20 000 man-hours for groundtruthing. Overall, the annotation step of this dataset is very time-consuming and expensive, even the average video duration is pretty small (i.e., $\simeq$ 6 minutes). Besides, all of these videos are realistic copies which can be a black-box model of video degradations.

In summary, the key PVCD datasets have been reviewed deeply in this section. The protocols and the main properties of the datasets are also discussed in great detail. To name some highlights, two main sources of videos are derived from the Internet and TV broadcasts. While the TV datasets [11, 49, 61] are designed for the task of online real-time detection, the Web video datasets [30, 35, 45, 46, 107] seem to be more commonly used for the retrieval or detection tasks. The most relevant datasets are MUSCLE_VCD [62], VCDB [46], FIVR-PVCD [30], and VCSL [35], which have been specifically produced for partial video copy detection and localization. Other related datasets are likely to simulate the problem of near-duplicate video retrieval. Different key properties can be identified in these four datasets including the scalability, the source of capture for noise control, and the temporal accuracy, as summarized qualitatively in Tab. 1.3.

Tab. 1.3 Comparison of the publicly available PVCD datasets. Degradation: Realistic and Synthetic. The 'n/a' stands for not available.

Dataset	MUSCLE_VCD [62]	VCDB [46]	FIVR-PVCD [30]	VCSL [35]
References	small	small	small	small
Positive pairs	small	small	middle	large
Annotation cost	n/a	+	++	+++
Negative videos	small	middle	no	no
Overall scale	**small**	**middle**	**middle**	**large**
Degradation	R+S	R	R	R
Annotation level	segment	segment	segment	segment
Timestamps (s)	1	1	1	1

Regarding scalability, the key difficulty is to constitute the set of positive videos. In-

[6] Youtube: https://youtube.com, Bilibili: https://www.bilibili.com/

deed, the negative videos are easy to be generated and their amount must be balanced with the one of positive videos. The scalability of the positive set can be characterized by the numbers of references, videos and pairs (see Eq. 1.6). Although MUSCLE_VCD [62] is the first well-known public dataset, it offers a small scale. The two datasets VCDB [46], and FIVR-PVCD [30] present a close level of scalability, whereas the VCSL dataset [35] offers a large-scale challenge. However, the scalability in the VCSL dataset has been obtained at the price of a huge annotation cost by humans. All these datasets have been designed using Web-based protocols. Compared to the TV datasets [11, 49, 61], the Web capture generally produces realistic noise and cannot ensure a fine characterization of systems against specific degradations (see Fig. 1.2). At last, another main limitation is the temporal accuracy. Indeed, the segment-level annotation has been bounded at a precision of 1-second in all the datasets. However, the temporal accuracy is a desirable property to evaluate the PVCD performance that will be discussed in the next section. To the best of our knowledge, there is a lack of scalable, noise-free, and time-accurate PVCD datasets in the literature today.

1.3.2 Performance characterization for PVCD

To evaluate the performances of the proposed methods and systems, a characterization protocol must be established. Such a protocol involves the selection of datasets and their organization, to establish a training strategy, characterization methods, and metrics for evaluation. Tab. 1.4 presents the common characterization protocols along with their brief corresponding descriptions used for PVCD. Based on the frameworks mentioned in Section 1.2, these protocols can be roughly divided into two categories: no-training and training-based. Among the training-based, cross-validation and cross-dataset validation are two popular characterization protocols for the PVCD in the literature. Within to these protocols, we detail in the next paragraphs specific aspects related to the dataset organization, metrics and characterization methods.

Tab. 1.4 The common characterization protocols for PVCD.

Protocol	Description
no-training	is used to evaluate the performance of methods/systems that do not involve machine learning approaches. In such cases, the whole dataset is used for evaluation [11, 49, 55, 61].
cross-validation	is used in machine learning to evaluate the performance of a model on unseen data. It involves dividing the dataset into multiple subsets, using one of these subsets as a validation set, and training the model on the remaining ones (e.g., train-test split [35, 56], k-fold [43]).
cross-dataset evaluation	is used to assess how well a machine learning model trained on one dataset performs when evaluated on a different dataset [5, 30, 35, 99].

Dataset organization: The datasets described in the previous section provide testing videos, positive and negative, with their references, and groundtruth. For evaluation, they must be organized to perform the detection with PVCD systems and to characterize the performances. When a training-based protocol is used, it is common to divide the set of positive and negative videos into subsets for training, validation and testing with a cross-validation procedure Tab. 1.4. The performances of proposed methods rely not only on their ability to detect the copies but also on their ability to reject the non-copies. At a first level, testing videos contains negative segments required to reject, surrounding the

copies needed to detect. However, to stress the systems a significant amount of whole distractive videos could be included in the datasets. Indeed, in real-world applications distractive/negative videos constitute a significant part of the dataset compared to positive videos (see Tab. 1.2). This raises a hard challenge for scalability and depending the characterization protocol, it could be established to include or not the distractive/negative videos for characterization.

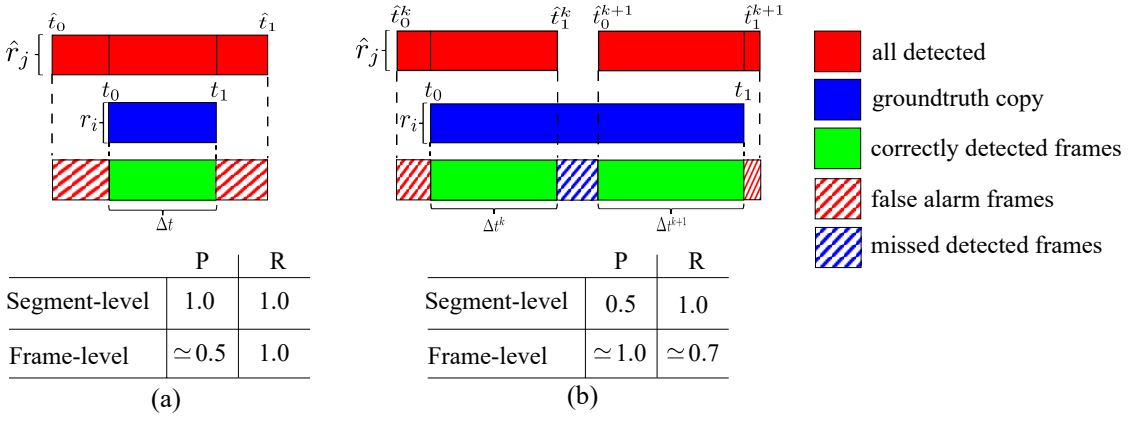
Metrics: The standard metrics of Precision (P), Recall (R), F_1, and mean average precision (mAP) are used to characterize the detection performances in most studies [72, 90]. These are baseline and popular metrics in the pattern recognition and information retrieval fields. Tab. 1.5 reminds shortly the key mathematic formulations and definitions with application to PVCD. While the P, R and F_1 scores are mainly used to characterize the detection task [5, 29, 30, 38, 43, 74, 99], the mAP metric is commonly used for the content-based video retrieval [31, 54, 56, 93, 98, 119]. Tab. 1.6 reports the main PVCD results of the representative PVCD studies, using these metrics.

Tab. 1.5 P, R, F_1, mAP, and AP metrics.

Mathematic formulations	Definition
$P = \frac{number\ of\ correctly\ detected}{all\ detected}$, $R = \frac{number\ of\ correctly\ detected}{groundtruth\ copy}$, $F_1 = 2\frac{P \cdot R}{P + R}$	The precision (P) measures the accuracy of video detection while the recall (R) measures the ability to detect all the possible video copies. Both are computed by comparing the number of correctly detected videos (or segments) to the overall detection results or the groundtruth. An alternative is to compute at the frames composing the videos/segments (i.e., frame-level). The F_1 metric is the harmonic mean of P and R. The P, R, F_1 metrics have distributions with a range $\in [0, 1]$ against a detection threshold, where 0 is the worst possible score and 1 is the best. It is common to extract the maximum F_1 with the corresponding P and R.
$mAP = \frac{1}{Q}\sum_{q=1}^{Q} AP(\mathbf{v}_q)$	The mean average precision (mAP) evaluates the accuracy of a video retrieval system across a set of query/testing videos. It varies in the range $\in [0, 1]$ with a higher score representing a more accurate result. Q is the total number of testing videos, and AP($\mathbf{v}_q$) denotes the average precision for *q-th* testing video $\mathbf{v}_q$. In other words, the mAP is the average of average precisions across all testing videos.
$AP(\mathbf{v}_q) = \frac{1}{RV_q}\sum_{k=1}^{N} P_k(q) \cdot \mathbb{1}_k$	The average precision (AP) quantifies the quality of ranked results for the single query video $\mathbf{v}_q$, where - N the total number of videos in the dataset, - $P_k(q)$ denotes the precision at cut-off k in the ranked list for $\mathbf{v}_q$, - $\mathbb{1}_k$ is a binary function which returns 1 if the *k-th* retrieved video is a relevant video, otherwise $\mathbb{1}_k = 0$, - RV_q is the total number of relevant videos for the testing video $\mathbf{v}_q$. The AP also has a value in the range from 0 (worst) to 1 (best).

Characterization methods: The P, R and F_1 metrics matter for the characterization of the detection. As a baseline, they can be computed at the video-level without considering the timing aspects. This raises miscellaneous detection cases. As an example, a false alarm segment appearing in the video, located at different time, could result in the '*correctly detected*' characterization of the video. The metrics must be adapted in order to take into account the timing aspect for PVCD. As discussed in [46], two main methods are deployed in the literature to address this issue at the segment-level and frame-level. They are discussed below and illustrated on Fig. 1.3.

The segment-level characterization computes the P, R and F_1 scores with segments. Here, the '*groundtruth copy*' of the reference r_i is defined in the groundtruth located at t_0, t_1. For each detection, a video segment, located at $\hat{t}_0$, $\hat{t}_1$, is considered '*correctly detected*' if it has the same reference $\hat{r}_j = r_i$, and there is an intersection Δt between the detected segment and the groundtruth, as illustrated in Fig. 1.3 (a). When multiple intersections could be established, the detection results in false alarms (see Fig. 1.3 (b)). In that case, one of the two segments k, $k+1$ (e.g., the one with the lowest confident score) will be rejected and considered as the false alarm. This will increase the number of '*all detected*' and subsequently decrease the precision, as presented in Tab. 1.5 and illustrated in Fig. 1.3 (b) (i.e., P = 0.5 at the segment-level). Similarly, once multiple groundtruth copies are given, the missed cases could occur for the detection. This will decrease the number of '*correctly detected*' and consequently reduce the recall.



	P	R
Segment-level	1.0	1.0
Frame-level	≃0.5	1.0

(a)

	P	R
Segment-level	0.5	1.0
Frame-level	≃1.0	≃0.7

(b)

Fig. 1.3 Illustration of the metric calculations for (a) single detection, (b) multiple detection.

The characterization at the segment-level is well adapted for the multiple detections Fig. 1.3 (b). However, it is quite tolerant of the rough localization of the detected segment, as illustrated in Fig. 1.3 (a). In that case, even if the segment is roughly detected, a perfect detection will be accounted for performance (e.g., P = R = 1.0 at the segment-level for case (a)). In order to overcome this limitation, a second method is to characterize the detection at a frame-level, as discussed next.

The frame-level characterization overcomes the limitation of the segment-level. This characterization evaluates the P, R, and F_1 scores at the frames that constitute the video segment. This is equivalent to counting the number of correctly and incorrectly detected frames between the two segments. Indeed, the number of frames composing the segments could be easily obtained from $(t_1 - t_0) \times$ FPS for the groundtruth, and from $(\hat{t}_1 - \hat{t}_0) \times$ FPS for the detection, as illustrated in Fig. 1.3 (a). All the frames in the detected and groundtruth segments are derived from the references $\hat{r}_j$ and r_i, respectively. If $\hat{r}_j = r_i$, all the overlapping frames located in the Δ_t interval are considered as '*correctly detected*'. Otherwise, they correspond to false alarms/missed cases and will increase/decrease the number of '*all detected*'/'*correctly detected*'. Given such a case, the precision and recall computed in Tab. 1.5 will consequently decrease. The characterization at the frame-level is well adapted for the rough localization Fig. 1.3 (a). However, compared to the characterization at the segment-level, it is quite tolerant of the false alarms and missed

cases, as shown in Fig. 1.3 (b). An alternative characterization metric at the frame-level, which is an extension of P, R, F_1 computations, was introduced in the work [35].

1.3.3 State-of-the-art results

In this section, we discuss the main state-of-the-art results for PVCD. Tab. 1.6 gives a global comparison with the used datasets (see the tables 1.2 and 1.3), brief descriptions of methods/systems, employed characterization protocols, dataset organization, characterization methods, and metrics. The results are organized into two subparts based on the relation of dataset characteristics to the PVCD. Different performances and results have been achieved in literature depending on considered tasks, systems, used datasets, and characterization protocols. That is, the results misalign from study to study and are not directly comparable for a fair comparison [72, 90]. Based on the synthesis of Tab. 1.6, we analyze the results here at three levels. We first provide a global discussion on used protocols, followed by analyses of the results on the least and most relevant PVCD datasets.

A global discussion: The systems use different frame features, video representations, and matching methods, which are briefly described in Tab. 1.6, and more details are given in Section 1.2. Recent systems achieve top performances mainly through deep learning methodologies, especially CNNs. Most of the systems are characterized without training or with a cross-dataset evaluation. In the last case, the motivations are even to evaluate the robustness of the methods/systems to related datasets [38, 39], or even to deal with the little scalability and suitability of datasets for training [35, 43]. Few works process with training, where the common trend is to employ a cross-validation procedure [35, 43, 56]. To relax the scalability constraint and the challenges in detection/retrieval, the most common strategy is to remove the negative/distractive videos from the datasets. The main characterization protocols in the literature focus on the segment-level while employing the P, R, F_1, and mAP metrics (see Section 1.3.2).

These divergences in the characterization protocols have a great impact on the performances. As an example, a $\simeq 10\%$ gap in the F_1 score is observed with and without the inclusion of negative/distractive videos in the VCDB dataset [47]. Similarity, the characterization at the segment-level or frame-level also has a significant impact on the performance outcomes (e.g., $\simeq 10\%$ on the F_1 score on the VCDB dataset [46]). Finally, the modification of the characterization protocol could have an incremental impact on the F_1 score (some percent on the F_1 score on the VCSL dataset [35]).

Analysis of results for less relevant PVCD datasets: Preliminary results have been reported on several less relevant PVCD datasets, as shown in the top part of Tab. 1.6. In such cases, the characterization is done at the whole-video level without considering the timing aspect that is an important outcome for the PVCD systems. Other strong limitations for the characterization are linked to the datasets that are not publicly available, have a small size, and limited relevance to PVCD (see more details in Section 1.3.1). As a result, it is difficult to draw consistent conclusions for the PVCD characterization, even though the systems appear to achieve good detection scores (F_1 or mAP scores $\geq 85\%$). Among these datasets, only SVD is publicly available and challenging (the lowest mAP $\simeq 80\%$) attracting more attention from the research community in the PVCD problem [56].

Tab. 1.6 The main state-of-the-art PVCD results. **H** and **L** stand for the hand-crafted and learning-based features, respectively. V/S/F denotes the video/segment/frame level for the corresponding performance results. The resulting text in bold indicates the best results for each dataset, and 'n/a' stands for not available.

Dataset	Method	Evaluation protocol	Dataset organization positive | negative	Feature type | Name	V/S /F	P (%)	R (%)	F_1 (%)	mAP (%)
The less relevant PVCD datasets									
TV_2007	DPSS [49]	no-training	full | yes	**H** | Harris	V	**90.00**	**80.00**	**84.70**	n/a
BBC_2007	ViCopT [61]	no-training	full | no	**H** | Harris	V	**95.00**	**82.00**	**88.02**	**86.00**
TV_2014	Fingerprints [11]	no-training	full | no	**H** | Fingerprints	V	**100.0**	**98.00**	**88.90**	n/a
CC_WEB_VIDEO	CNN-L [55]	no-training	full | yes	**L** | AlexNet, VGG, Google	V	n/a	n/a	n/a	97.60
CC_WEB_VIDEO	ViSiL-v [54]	cross-dataset evaluation	full | yes	**L** | ResNet-50	V	n/a	n/a	n/a	**99.30**
SVD	DML [56]	cross validation	partial | yes	**L** | 4096D VGG-16	V	n/a	n/a	n/a	**78.47**
The most relevant PVCD datasets									
MUSCLE-VCD	LSP-LSH [109]	no-training	full | yes	**H** | LDP	V	n/a	n/a	n/a	93.00
MUSCLE-VCD	TNP [98]	no-training	partial | yes	**H** | PCA-SIFT	V	n/a	n/a	n/a	**100**
MUSCLE-VCD	TNP [98]	no-training	partial | yes	**H** | PCA-SIFT	S	n/a	n/a	n/a	90.00
VCDB	CNN [47]	no-training	full | no	**L** | AlexNet	S	n/a	n/a	65.03	n/a
VCDB	CNN [47]	no-training	full | yes	**L** | AlexNet	S	n/a	n/a	54.78	n/a
VCDB	CNN [47]	no-training	full | no	**L** | AlexNet	F	n/a	n/a	71.01	n/a
VCDB	LAMV [5]	cross-dataset evaluation	full | no	**L** | RestNet-34 RMAC	S	n/a	n/a	68.70	n/a
VCDB	CNN-RNN [38]	cross-dataset evaluation	full | no	**L** | ResNet + SiamesLSTM	S	n/a	n/a	72.33	n/a
VCDB	Q-learning [29]	no-training	full | no	**H** | VTh + ST descriptors	S	88.29	73.55	80.25	n/a
VCDB	GANN network [74]	no-training	full | no	**L** | Inception v4-FC	S	80.00	65.00	n/a	n/a
VCDB	SPD method [43]	cross-validation	partial | no	**L** | ResNet R-MAC	S	n/a	n/a	81.86	n/a
VCDB	VSAL approach [30]	cross-dataset evaluation	full | no	**L** | Resnet-50	S	**89.71**	**84.62**	**87.09**	n/a
VCDB	Fast KNN search [99]	cross-dataset evaluation	full | no	**L** | Resnet-Transformer	S	n/a	n/a	**87.64**	n/a
FIVR-PVCD	VSAL approach [30]	cross-dataset evaluation	full | no	**L** | Resnet-50	S	**85.75**	**68.83**	**76.36**	n/a
VCSL	SPD [35]	cross-validation	partial | no	**L** | 1536-D DINO	S	**90.31**	**84.67**	**87.40**	n/a
VCSL	SPD [35]	cross-dataset evaluation	full | no	**L** | 1536-D DINO	S	90.66	81.23	85.69	n/a

Analysis of results for the most relevant PVCD datasets: The main results are presented in the bottom part of Tab. 1.6. They cover four main datasets including MUSCLE-VCD, VCDB, FIVR-PVCD, and VCSL. Among these datasets, MUSCLE-VCD seems to be outdated and is no longer challenging because near-perfect results can be obtained easily with traditional methods (i.e., mAP = 90% is achieved using the PCA-SIFT feature [98]). Another reason is the limited scale in this dataset (see tables 1.2 and 1.3). Considering the other datasets, two scalability cases must be distinguished: VCDB/FIVR-PVCD and VCSL, which are discussed in the next paragraphs.

The VCDB and FIVR-PVCD datasets present a close and low level of scalability, as detailed in tables 1.2 and 1.3. They mainly differ in their degradation levels, in which the FIVR-PVCD dataset is more challenging with abundant temporal and spatial editing of copy segments to detect. At the same evaluation protocol, this variability causes a $\simeq 10\%$ difference on the F_1 score [30]. The scalability and wrong distribution of datasets look not sufficient to support a direct training protocol with a cross-validation procedure. This is discussed in [35] and compared in [29] and [43] for the VCDB dataset. Particularly, a training-based protocol using cross-validation in [43] has a little impact on performances compared to a no-training protocol [29] ($F_1 \simeq 82\%$ vs. $F_1 \simeq 80\%$, respectively). As a result, the trend is to process with cross-dataset evaluation [30, 99] where an incremental gain of $\simeq 10\%$ on the F_1 score can be obtained.

The protocols deployed on the VCDB/FIVR-PVCD datasets lead a mis-understanding of the PVCD problem with almost saturated performances (e.g., F_1 scores $\simeq 90\%$ in [30, 99] on the VCDB dataset). This can be explained by: (i) the low level of scalability of the datasets and their limited ability for training (e.g., only $\simeq 2\%$ on the F_1 score is observed on the VCDB dataset [29, 43]), (ii) the non-inclusion of negative videos (e.g., a degradation of $\simeq 10\%$ on the F_1 score is observed on the VCDB dataset in [47]), and (iii) the use of the segment-based protocol for characterization tolerant to rough and partial localization as detailed in Section 1.3.2 (e.g., a degradation of $\simeq 10\%$ on the F_1 score is observed on the VCDB dataset in [47]). As a conclusion, the low level of scalability and timing accuracy of the two datasets cannot address advanced challenges and a fine characterization for PVCD.

VCSL is the only dataset offering a high level of scalability. As detailed in tables 1.2 and 1.3, it has more than an order of magnitude difference for the positive segment pairs compared to the VCDB/FIVR-PVCD datasets. As a result, VCSL is the only dataset where a training-based protocol can be deployed efficiently with a cross-validation procedure. This is illustrated in [35] where the procedure achieves competitive results on the VCDB dataset against a cross-dataset evaluation with a slightly gap ($\simeq 2\%$ on the F_1 score).

However, similar to the VCDB/FIVR-PVCD datasets, a phenomenon of saturated performance is observed on the state-of-the-art with VCSL (e.g., $F_1 = 87.64\%$ for VCDB [99] vs. $F_1 = 87.40\%$ for VCSL [35]). This is explained by the non-inclusion of negative videos (ii) as detailed previously. Indeed, the data augmentation of the positive set alone for training improves then the overall performance (i.e., reducing overfitting) [91]. To stress the system with scalability, the mandatory point is to include negative videos in balance with positive videos [110]. That is, VSCL dataset and the reported experiments do not meet with all the requirements for scalability. Some other strong limitations of VCSL are the non-consistency and the rough timing annotation (see tables 1.2 and 1.3).

1.4 Conclusions and perspectives

To conclude this chapter, we shall highlight several key remarks regarding the performance evaluation of the PVCD topic in terms of application domains, the existing approaches, benchmark datasets, metrics, and the results. Then this section discusses some perspectives that will connect to our contributions in the next chapters.

The PVCD has been a well-know topic in the computer vision for decades. It covers a wide range of application domains such as video retrieval, copy right protection, video monitoring, etc. Although they have differences in purpose, they share many common components within the PVCD systems (e.g., feature extraction, generation of video representations, video comparison). The existing approaches in the literature are trying to deal with two key challenges in the video transformations including both spatial and temporal domains, as well as scalability. To address the problem of the video transformations, tens of video feature types derived from the image patten recognition field have been used for the PVCD. Among these features, 2D CNN-based features which are extracted from the modern CNN models have been emerging as a potential video feature extractor over the past ten years. To deal with the scalability, the optimal techniques (e.g., PCA, hashing) have been adapted to reduce the problem of video embeddings in high-dimensional spaces.

To evaluate the performance of these PVCD systems, benchmark datasets have been proposed in the literature. After a careful investigation, many weaknesses were found and detailed among the existing datasets. Indeed, few 6/10 datasets are publicly available to be used in the literature due to the intellectual property. Among of them, only four datasets (MUSCLE-VCD, VCDB, FIVR-PVCD, and VCSL) are the most relevant to the PVCD problem. As a general trend, an increasing number of copied user-generated videos with real-world noise has been collected to constitute these datasets over time. However, none of the datasets is able to introduce a 'real' challenge for the PVCD problem in terms of scalability, capacity to recognize different specific video transformations, or reliability of groundtruth, to the best of our knowledge. Finally, it is extremely difficult to extend these datasets because of the huge annotation cost required to produce the groundtruth.

Given the benchmark datasets, they are organized to evaluate the performance of the proposed methods using standard metrics (i.e., P, R, F_1, or mAP). Despite many methods have been developed in the literature, we have found that the state-of-the-art results from different studies are not always comparable. Based on these results along with the dataset organization, and metrics, we have analyzed and classified them in more details for a better understanding in the performance characterization aspect. By doing this, we have explored many unsolved issues related to the performance evaluation and characterization for PVCD (e.g., scalability, evaluation protocols, characterization levels). One of the most important issues is the lack of large-scale experiments needed to gain a deeper understanding of the unique challenges in the recent PVCD systems, which mostly adopted deep learning models originally designed for image processing to handle video. Therefore, performance evaluation and characterization, particularly focusing on the systems using 2D CNN features, could be a promising future direction, with little effort made so far.

In this thesis, the main aim is to evaluate the performance of PVCD systems. To archive this goal, a large-scale public dataset must be proposed for the research community.

The PVCD dataset should have properties that address the limitations of the existing datasets highlighted earlier. Thus, the first our contribution is to introduce a new PVCD dataset. It involves a protocol design that provides deeper scalability, degradation control and temporal accuracy. Our protocol, dataset, and experiments are presented in Chapter 2. Following the first contribution, our methods and protocols are then proposed to evaluate the performance of PVCD systems using 2D CNN features. As was mentioned earlier, most studies in the PVCD have emphasized the use of 2D CNN features over the past decade. Through large-scale experiments and analysis, our second contribution confirms and highlights the limitations of 2D CNN features for the PVCD problem. Our methods, protocols and experiments are discussed in Chapter 3.

Chapter 2

A TV dataset for performance evaluation of PVCD methods

Contents

2.1 Introduction

In the previous chapter 1, we have presented a state-of-the-art about the PVCD problem having a particular focus on the performance evaluation side. Most of the proposed systems in the literature have been characterized as a black box using datasets composed of online videos. These datasets mix different kinds of degradations and levels making it difficult to drive a fine performance characterization. In addition, specific characterization tasks cannot be handled with the existing datasets (e.g., scalability, real-time detection).

The PVCD problem is inherent to continuous video broadcasting. An alternative is then to process with a TV dataset offering meaningful data, deeper scalability and a fine control of degradations for performance characterization. As discussed in Chapter 1 (see page 27), none of the TV datasets proposed in the literature have been made publicly available for the needs of performance evaluation. We propose in this chapter a large-Scale TV Dataset for the PVCD, called STVD. Section 2.2 describes the related work. Section 2.3 presents our protocol and pipeline for video capture and groundtruthing. Experiments to design the dataset are reported in Section 2.4 with performance evaluation results of representative methods as a baseline comparison. Section 2.5 presents a statistical comparison between the STVD and other benchmark datasets, and Section 2.6 provides conclusions and perspectives. For convenience, the list of abbreviations as well the general terminologies used for the PVCD are given on page 15, and Tab. 2.1 gives the meaning of the main symbols used in this chapter.

Tab. 2.1 Main symbols used in the chapter.

Symbols	Meaning
$\mathbf{i}, \mathbf{j}, \mathbf{k}, \mathbf{u}, \mathbf{l}$	are indexes
$\mathbf{q}, \mathbf{n}$	are sizes of an array with $\mathbf{q} < \mathbf{n}$
$\mathbf{t}, \hat{\mathbf{t}}$	the scheduled and detected timestamp for a program
$\mathbf{h}$	a hashcode generated from TV metadata
$\mathbf{C}(\mathbf{h})$,$\mathbf{m}$	a counting function returning the number of occurrences of $\mathbf{h}$, $\mathbf{m} = \mathbf{C}(\mathbf{h})$
$\mathbf{h_c}$	a hashcode with the maximum number of occurrence $\in [\mathbf{h_1}, \ldots, \mathbf{h_q}]$
$\mathbf{H} = [\mathbf{h_1}, \ldots, \mathbf{h_k}, \ldots, \mathbf{h_q}]$ $\mathbf{H}^* = [\mathbf{h}_1^*, \ldots, \mathbf{h}_\mathbf{l}^*, \ldots, \mathbf{h}_\mathbf{n}^*]$	unlabeled and labeled hashcode arrays, respectively
$\mathbf{t_1}, \ldots, \mathbf{t_i}, \ldots, \mathbf{t_n}$	a set of the ordered timestamps such as $\mathbf{t_i} < \mathbf{t_{i+1}}$
$\mathbf{t_i}, \mathbf{D_i}$	the timestamp and the duration of the reference instance i, respectively
$\mathbf{\Delta_i}$	an interval of two ordered timestamps, $\mathbf{\Delta_i} = \mathbf{t_{i+1}} - \mathbf{t_i}$
$\mathbf{W} = \mathbf{W}^- + \mathbf{W}^+$	the window model for extraction of candidate videos
$\mathbf{L} \in [\mathbf{L}_{\min}, \mathbf{L}_{\max}]$	$\mathbf{L} = \hat{\mathbf{t}} - \mathbf{t}$ is the latency, $\mathbf{L}_{\min} < \mathbf{0}$, $\mathbf{L}_{\max} > \mathbf{0}$ the min and max
$\mathbf{L}^- < 0, \mathbf{L}^+ > 0$	a negative and positive latency, respectively
$\mathbf{D} \in [\mathbf{D}_{\min}, \mathbf{D}_{\max}]$	$\mathbf{D} \in \mathbb{R}^+$ is a program duration, $\mathbf{D}_{\min}$, $\mathbf{D}_{\max}$ the min and max
$[\ldots, \mathbf{V_i}, \ldots]$, $[\ldots, \mathbf{V}_\mathbf{j}^*, \ldots]$	the sets of positive candidate videos and positive videos, respectively
$[\ldots, \mathbf{R_u}, \ldots]$, $\tilde{\mathbf{R}}$	a set of reference videos and a reference candidate video
$[\mathbf{X_1}, \ldots, \mathbf{X_m}]$	a set of matched videos, $[\mathbf{X_1}, \ldots, \mathbf{X_m}] \subseteq [\ldots, \mathbf{V_i}, \ldots]$.
ZNCC	Zero-mean Normalized Cross-Correlation
$\overline{\textbf{ZNCC}}$	A global **ZNCC** score obtained with weighted averaging
$\mathbf{T_0}, \ldots, \mathbf{T_6}$	the video degradations and transformations
$\mathbf{s_i}, \mathbf{e_i}$	the start and end times of the video segment $\mathbf{i}$
α, β	the parameters to control the degradation level
$\mathcal{S}$	a $\mathbf{T_0}$ segment starting at $\mathbf{s} = \mathbf{t} - \lvert\mathbf{L}^-\rvert$ and ending at $\mathbf{e} = \mathbf{t} + \mathbf{D} + \mathbf{L}^+$

2.2 Related work

Several datasets have been proposed in the literature for the performance evaluation of PVCD methods. However, to the best of our knowledge, there are only three TV datasets that have been proposed so far. Tab. 2.2 resumes the main properties of these datasets from Tab. 1.2. For comparison, Fig. 2.1 gives graphical representations of the protocols used to design these datasets. We will detail them in the following paragraphs.

Tab. 2.2 Comparison of the TV datasets for the PVCD performance evaluation.

Dataset	Reference videos	Testing videos	Duration (h)	Metadata	Resolution (width×height)	Degradation method	Publicly available
TV_2007 [49]	100	200	30 000	little	352×288	synthetic	no
BBC_2007 [61]	n/a	10	6.0		n/a		
TV_2014 [11]	n/a	1×10^7	380 000		360×288, 448×256		
STVD_2021 (ours)	243	83 320	10 660	rich	320×240	synthetic	yes

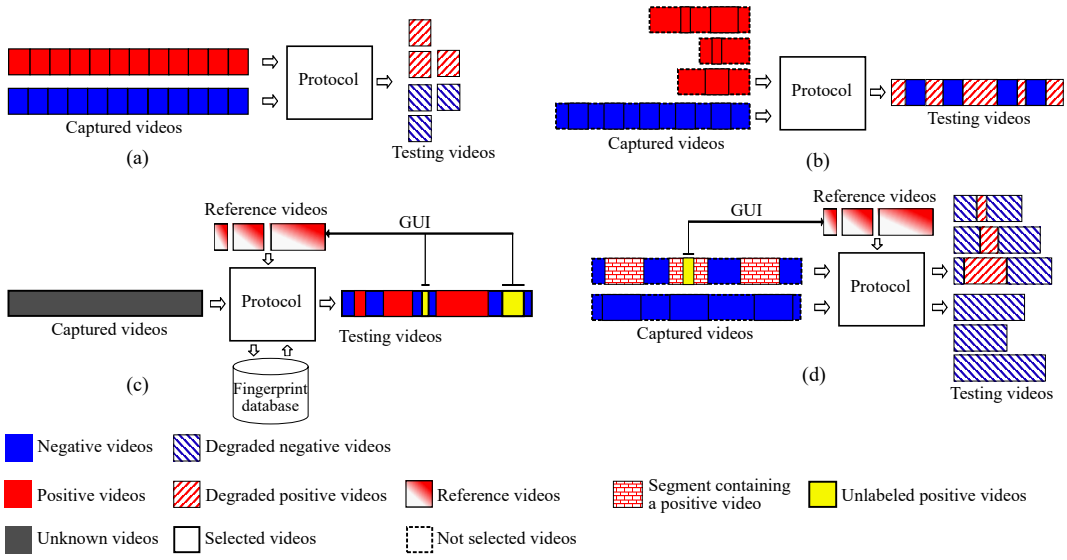


Fig. 2.1 The protocols used to design the TV datasets: (a) [49], (b) [61], (c) [11] and (d) STVD.

The authors in [49] use two TV captures to extract the positive and negative videos as shown in Fig. 2.1 (a). A long TV capture serves to extract randomly short video sequences corresponding to the positive videos. The partial copies are obtained with the application of degradation methods to the positive videos. The shortest TV capture, having little correlation with the long TV capture (e.g., a foreign channel), is employed to get the negative videos. This hypothesis makes sure that no negative videos appear in the positive videos. However, the obtained videos have no domain meaning as they correspond to successive videos having a uniform duration. In addition, none of the metadata is provided in the dataset. At last, the dataset is more designed for the task of full video copy detection as a single video instance is extracted before applying video degradations.

Another TV dataset is presented in the work [61]. The used protocol is introduced in Fig. 2.1 (b). The video database is separated into two sets for the positive and negative videos. A testing video is then constituted by merging multiple negative and positive videos. For the needs of the PVCD performance evaluation, simulated transformations are applied randomly to short segments composing the positive videos. These testing videos are used as queries within the system to find the best match with the original videos. Similar to [49] the metadata has not also been taken into account in this dataset.

The protocols used in [49, 61] illustrated in Fig. 2.1 (a)-(b) result in positive videos without domain meaning. Indeed, these videos are obtained with a uniform/random selection and aggregation. To solve this problem, a different approach is used in [11] Fig. 2.1 (c). Every day, a fingerprinting database is generated from the daily TV capture. This database is searched against itself to get candidate positive videos and then compared to an archive database. The archive database contains reference videos that are near-duplicate positive videos. To deal with the scalability, the two databases are trimmed to keep only the first occurrences of any repeated video. Any new detected positive video occurrence is labeled by a user with a graphic user interface (GUI). That is, the overall approach processes with an incremental groundtruthing. The videos that are not detected constitute the negative set. The metadata of the videos is obtained by user labeling with a GUI. The approach offers a large variability of partial copies (e.g., jingles, advertising, etc.). However, it is error-prone as there is no guarantee that false or missed detections will not occur. In addition, only a small amount of metadata can be obtained.

The protocols Fig. 2.1 (a)-(c) used in [11, 49, 61] have strong limitations. They offer little/no metadata, are not fully reliable for the groundtruthing, or dedicated to the partial video copy detection problem. In addition, the three datasets have never been made publicly available and used by external systems in the literature Tab. 2.2.

We propose in this chapter a new dataset, called STVD. Fig. 2.1 (d) describes our protocol used to design the dataset. First, similar to the protocols [49, 61] Fig. 2.1 (a)-(b) we use two separated TV streams to ensure no false-positive detections during the groundtruthing. Secondly, as [11] our strategy processes with an incremental groundtruthing to guarantee a domain meaning for the partial video copies. However, we take advantage of a priori known metadata and low-level frame matching for robust detection. This ensures no false-positive videos in the groundtruth. For the needs of performance evaluation, synthetic video degradations are applied in order to stress the methods for detection. Our protocol is presented thoroughly in the next section 2.3.

2.3 STVD: A large Scale TV Dataset

2.3.1 System overview

STVD has been built based on a protocol with 3 major components from (**C1**) to (**C3**) Fig. 2.2. We process first with a video capture (**C1**) to extract positive/negative candidate videos. This component processes with TV metadata and long captured videos. We have used two separate streams from the TV capture to ensure no false-positive videos as similar to [49, 61]. The candidate videos are then processed within the component (**C2**)

to produce a reference, positive and negative video sets. In the overall pipeline, the latency and video duration parameters (i.e., **L**, **D**) are selected first with expert setting to control the capture **(C1)**. They are refined next with the component **(C2)** to generate the final video degradation **(C3)**. For clarification, Tab. 2.3 extends some key definitions from the list given in page 15 to demonstrate video data handled within the pipeline.

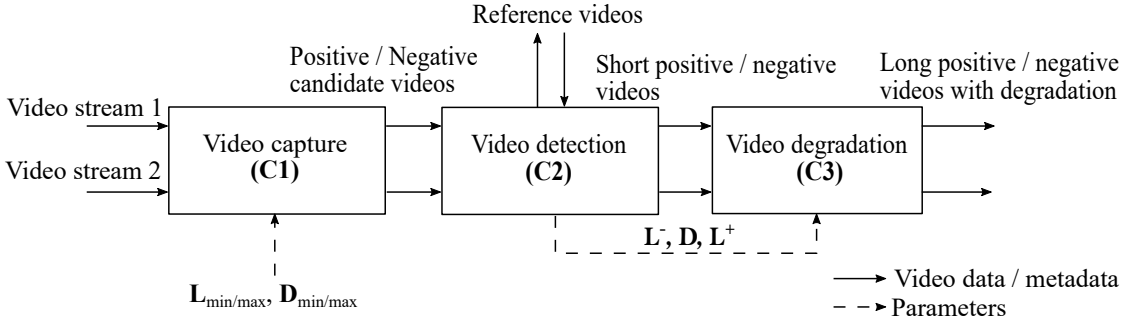


Fig. 2.2 The pipeline for constructing the STVD dataset.

Tab. 2.3 Definition of the video data terms used in this chapter.

Terminology	Length	Definition
A positive candidate video	**W**	is extracted from the stream 1. The candidates are selected according to hashcodes with their occurrences. A part of them will serve to constitute the final set of positive videos.
A negative candidate video		is extracted from stream 2. The candidates are selected by subtracting the idle segments that may contain possible reference videos. They will serve as instances to get the final set of negative videos.
A reference video	**D**	is extracted from a positive candidate video by using a GUI. It has a short duration, and it is linked to a set of positive candidate videos.
A short positive video		is the copied segment obtained from matching between a reference video and a positive candidate video. It is a near-duplicate of the reference video.
A short negative video		is randomly extracted from a negative candidate video based on the distribution of **D** values obtained from the video detection (**C2**).
A long positive video	$\lvert\mathbf{L}^-\rvert + \mathbf{D} + \mathbf{L}^+$	is extended from a short positive video by introducing left/right segments and applying video degradations. It contains at least one near-duplicate of a reference video.
A long negative video		is extended from a short negative video by introducing left/right segments and applying video degradations.

2.3.2 Video capture (C1)

The component **(C1)** extracts positive/negative candidate videos. It processes with the video streams and metadata collected from the Electronic Programming Guides (EPGs) to get the candidate videos as illustrated in Fig. 2.3. This is handled into two subcomponents **(C1.1)** and **(C1.2)**. Then, the long captured videos and metadata serve for the detection of positive and negative candidate videos handled into two sub pipelines. We obtain the positive candidate videos from a selection process using metadata and the application of a windows model. It is processed with the subcomponents **(C1.3)** and **(C1.4)**. The

negative candidate videos are selected by using an adapted selection process demonstrated in the subcomponent **(C1.5)** while setting the windows model into the idle mode. The subcomponents are presented and discussed in the following paragraphs.

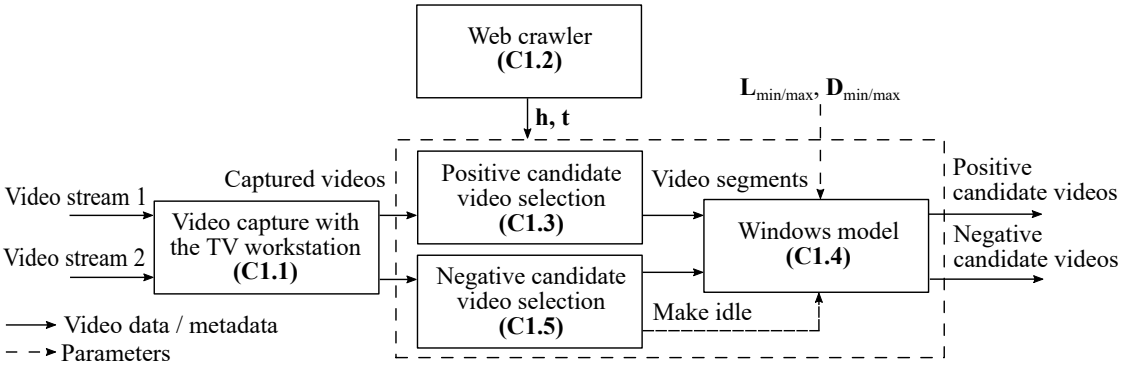


Fig. 2.3 A block diagram of the component **(C1)**.

Video capture with the TV workstation (C1.1)

A large number of TV videos need to be captured to constitute the dataset. This requires a dedicated platform for the scalable capture of TV programs. We have used the TV workstation depicted in Appx. A.1. Within this workstation, we have used the DELL Precision 5820 computer dedicated to the video capture. This computer can record up to ×8 TV channels simultaneously. It embeds ×4 dual-channel Avermedia CL332-HN cards for the hardware-supported video capture. Moreover, the computer is equipped with high-capacity hard disks (e.g., 38 terabytes) for a long capture. The capture setting is detailed in Tab. 2.4 and the next paragraphs.

Tab. 2.4 Protocol for capture.

Channels per month	Daily file					
	Compression	Resolution	kbps	Aspect ratio	Length	FPS
8	MPEG-4	320 × 240	560	4 : 3	20 h	30

We have set the workstation to capture French public channels. They are derived from digital TV signals broadcast over the air via terrestrial transmission. The digital TV signals are compressed with the MPEG-4/H264 AVC standard[1] (Advanced Video Coding developed by Motion Picture Experts Group). For scalability, the capture has been driven with a round-robin policy on a set of non-duplicate channels with one-month interval (e.g., channels #1-8 month 1, #9-16 month 2, etc.). For storage optimization, we have set a resolution 320×240 (width×height) for the capture with a parameter of 560 kbps (kilobits per second) for compression. The resolution was selected similarly to the dataset [11] Tab. 2.2. It constitutes the best trade-off between the memory cost and distortion as highlighted in our experiments in Appx. A.2. However, this resolution 320×240 shifts the video aspect ratio to 4:3 whereas the French public channels are delivered in 16:9.

[1]TDF's report: https://www.tdf.fr/sites/default/files/TDF-LIVRE-BLANC-TNT_0.pdf

This is due to the hardware capture constraints as discussed in Appx. A.2. While using this capture, the video noise is mainly bounded to the compression artifacts and contrast deviation as highlighted in Section 2.4.

We have also bounded the capture to 20 hours a day per channel[2] as few TV programs are scheduled during the night. Audio tracks were removed as we studied the PVCD as a computer vision task, focusing on the visual video content. The TV is delivered at a 25 FPS rate within the Phase Alternating Line (PAL) standard[3]. However, for normalization with the National Television System Committee (NTSC) standard and web videos, we set the capture at 30 FPS.

Web crawler (C1.2)

The capture with the workstation in component **(C1)** results in videos having a long duration (e.g., 20 hours). Detecting partial video copies within such long videos is error-prone due to the scalability problem [11]. However, a large number of TV programs are delivered with a priori schedule in the EPGs. These EPGs provide the schedule of the main public TV programs (e.g., news, movie, live show, etc.) and contain a large amount of repeated content (e.g., jingle, daily show, commercial, etc.). They can be then used for the selection of positive/negative candidate videos. In addition, they also provide ready-to-use metadata about the programs (e.g., title, category, timestamp, description, etc.).

The EPGs are publicly available on websites or services[4]. We have developed a web crawler to gather the TV programs from the website[5]. The crawling is launched every day to guarantee up-to-date data. From the gathered data, we have collected for every program the scheduled timestamp **t** and generated a corresponding hashcode **h**. The hashcode **h** is first derived from a text key resulting by concatenating between a channel name and a program title as illustrated in Tab. 2.5. For uniqueness, this key text is applied to the SHA-3 function (having a 512-bit encoding) to generate the final hashcode **h**. As detailed in Fig. 2.3, the obtained metadata **t**, **h** are used as input data of subcomponents **(C1.3)**, **(C1.4)** and **(C1.5)**. They serve for the selection of the positive/negative candidate videos. We will present these subcomponents in the next paragraphs.

Tab. 2.5 Processing of metadata and hashing method
(Ch) Channel, (LD) Levenshtein distance.

Ch name	Program title	Text key	Hashcode (h)	Normalized title	LD
France 2	Journal 08h00	france2journal08h00	04c7 ... 4d41	a = journal08h00	LD(a,a) = 0
France 2	Journal 20h00	france2journal20h00	8b57 ... b572	b = journal20h00	LD(a,b) = 2
France 2	C'est bon à savoir	france2cestbonasavoir	3da2 ... e81e	c = cestbonasavoir	LD(a,c) = 12

[2]From 6AM day_i to 2AM day_{i+1}

[3]PAL and NTSC are two of three major analog color television standards, the other one is SECAM. In France, TNT (Télévision Numérique Terrestre), which translates to Digital Terrestrial Television (DTT), typically follows the same standards as PAL for frame rates.

[4]https://www.telerama.fr, https://www.france.tv, https://xmltv.ch

[5]https://www.telerama.fr

Selection of positive candidate videos (C1.3)

The selection of positive candidate videos must be handled from the captured video with the help of metadata (i.e., the hashcode **h** with its timestamp **t**). Although every program could be a candidate, we are mainly focusing on programs with periodic broadcasting (e.g., weather forecasts, movie series, etc.). Indeed, these programs are good candidates for the PVCD as they appear with a large amount of occurrences. To limit the user interaction and for reliability, we have developed a specific approach to extract these periodic programs Fig. 2.4. We will detail the process in the next paragraphs.

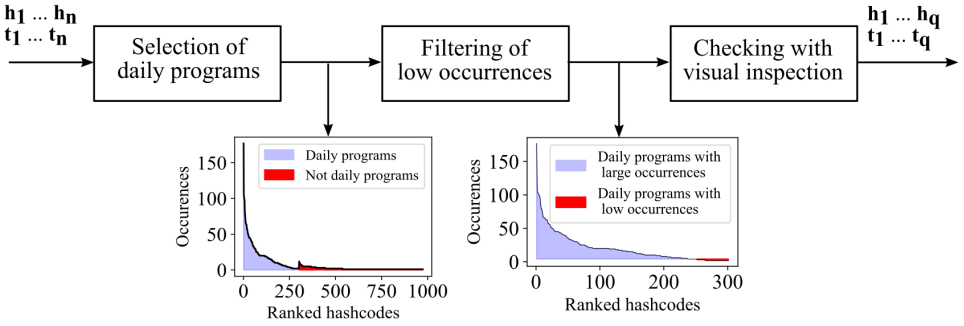


Fig. 2.4 The pipeline for extracting periodic TV programs.

As the first step in our process, we extract all periodic TV programs. A program is considered as periodic when a day interval $\mathbf{\Delta} = 1$ is appearing between the ordered timestamps $\mathbf{t_1}, ..., \mathbf{t_i}, ...\mathbf{t_n}$ with $\forall \mathbf{i}\, \mathbf{\Delta_i} = \mathbf{round}(\frac{\mathbf{t_{i+1} - t_i}}{\mathbf{Day}}) \leq 1.5$, where the **round** is an operator, and the **Day** is a constant for the daily duration. However, some TV schedules may be canceled (e.g., due to live news) or appear at certain times during the week (e.g., only the weekend or from Monday to Wednesday). Thus we apply a threshold to the rate given as $\frac{1}{\mathbf{n-1}} \sum_{\mathbf{i=1}}^{\mathbf{n-1}} \Delta_\mathbf{i}$. The periodic programs are detected for a minimum rate (e.g., 0.75) of timestamp differences $\mathbf{\Delta_i} = 1$. Next to this, we apply the final threshold to filter out periodic programs having a very low level of occurrences as illustrated in Fig. 2.4. This corresponds to programs ending/starting at the beginning/end of the month interval used for the capture in Component **(C1.1)**.

After the periodic program selection, we have applied a grouping with visual inspection during the final step. Indeed, the titles of programs could present variations in the EPGs due to the writing styles and naming conventions. Such errors could appear in the groundtruth and we have to apply then a checking process for grouping. A NPL text preprocessing[6] was first applied for the text key normalization before hashing, as shown in Tab. 2.5. Then, we have applied heuristics to recommend tentative programs for grouping with visual inspection. The Levenshtein distance was applied to compare the normalized program titles. We selected this distance as it is commonly used for string matching [3]. The closest titles were checked with visual inspection (e.g., based on description of programs, image and video content, etc.). As an example given in Tab. 2.5, the two TV programs "Journal 08h00" and "Journal 20h00" are grouped after the visual inspection because they are the same program (i.e., the program is scheduled twice everyday).

[6] e.g., lowercase/uppercase normalization, removal of spaces and special characters, etc.

The windows model (C1.4)

Although the timestamp $\mathbf{t}$ is delivered by metadata, no information is given about the exact location and duration of the repeated content. In addition, the TV broadcasting suffers from latency. To solve these problems, we have triggered the capture to get the jingles appearing at the beginning of programs with a window model $\mathbf{W} = \mathbf{W}^- + \mathbf{W}^+$ as illustrated in Fig. 2.5. The size parameter $\mathbf{W}^-$ guaranties the minimum latency with the TV broadcasting $\mathbf{W}^- \geq |\mathbf{L}_{\text{min}}|$. The parameter $\mathbf{W}^+$ is set with the maximum latency and reference video duration $\mathbf{W}^+ \geq \mathbf{D}_{\text{max}} + \mathbf{L}_{\text{max}}$. It is common to have $\mathbf{W}^+ \gg \mathbf{W}^-$. The detection of positive candidate video is then achieved within the interval $[\mathbf{t} - \mathbf{W}^-, \mathbf{t} + \mathbf{W}^+]$. While using the window $\mathbf{W}$, our protocol can guarantee that every jingle will appear in the selected video segment, even with a large delay or if they have already started. As illustrated in Fig. 2.3, the $\mathbf{D}_{\text{min / max}}$, $\mathbf{L}_{\text{min / max}}$ parameters have been selected with large values and expert setting for the needs of video capture. Then, they are refined through the video detection **(C2)** and the video degradation **(C3)**.

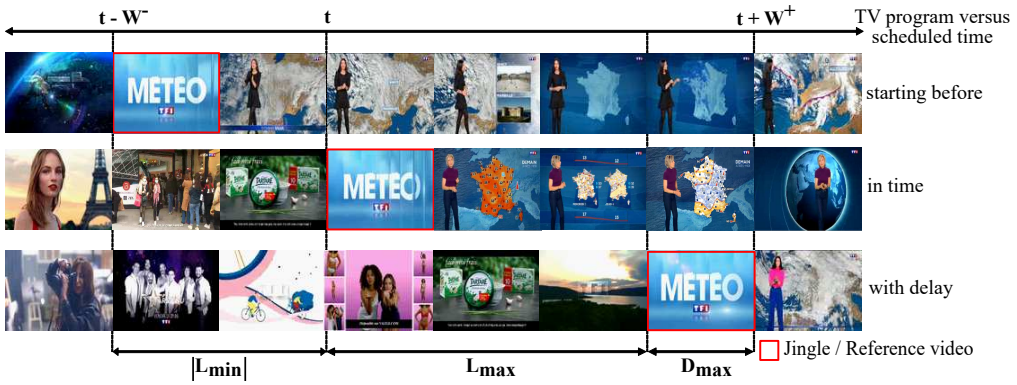


Fig. 2.5 The window model $\mathbf{W}$ processes for the candidate videos.

Applying the large window size $\mathbf{W}$ to the captured videos could cause overlapping cases for close positive candidate videos as illustrated in Fig. 2.6. Considering two consecutive programs $\mathbf{i}$ and $\mathbf{j}$, having ordered timestamps $\mathbf{t_i} < \mathbf{t_j}$, it is common to have $\mathbf{t_j} - \mathbf{t_i} \ll \mathbf{W}^+$. We have applied a merging step to solve this problem as demonstrated in Fig. 2.6 (a). We merge the two programs $\mathbf{i}$ and $\mathbf{j}$ if $\mathbf{t_i} + \mathbf{W}^+ > \mathbf{t_j} - \mathbf{W}^-$. That is, such a video segment may contain several positive candidate videos.

Selection of negative candidate videos (C1.5)

The component **(C1)** captures both the positive and negative candidate videos, as shown in Fig. 2.2. The negative candidate videos are not supposed to appear in the positive set. For reliability, similar to [49, 61], we have used a separated stream to get the negative candidate videos as shown in Fig. 2.1 (a), (b), and (d). For better robustness, we only have selected video segments, apart from jingles. For every program, we made idle for the selection of all the segments in negative candidate videos where a jingle could appear in the range $[\mathbf{t} - \mathbf{W}^-, \mathbf{t} + \mathbf{W}^+]$. Similar to the positive case Fig. 2.6 (a), overlapping segments were grouped together, as illustrated in Fig. 2.6 (b). We split then the remaining

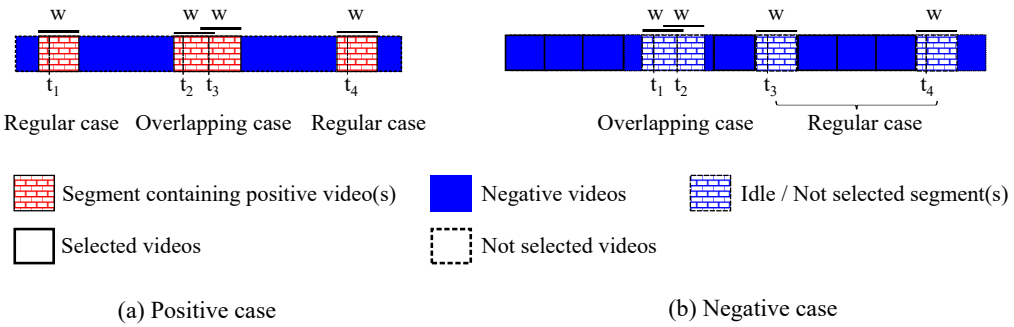


Fig. 2.6 An example of the merging step.

valid segments into successive intervals having a duration $\mathbf{W}$. Within each interval, the negative candidate video is timestamped at $\mathbf{t} = \mathbf{W}^{-}$.

2.3.3 Video detection (C2)

The positive candidate videos extracted from the previous component **(C1)** must be processed to detect the partial copies and to constitute the set of positive videos. Along with the positive video set, it is necessary to add both the set of reference videos and the set of negative videos for the PVCD problem. The annotation step could be time-consuming and costly due to the large number of candidate videos that require labeling. An alternative is to develop a fully automatic detection method. However, this needs a matching method able to deal with the scalability of the database, variety of video content, level of noise, and required accuracy for groundtruthing. Such a task is difficult to handle considering that no training database could be constituted for the needs.

To address this issue, we propose a pipeline supported by user interaction for video detection as illustrated in Fig. 2.7. Within this pipeline, we apply a semi-automatic groundtruthing with a loop-based methodology. The full pipeline is designed in order to guarantee robust detection and to minimize the user interaction cost. It consists of three subcomponents, from **(C2.1)** to **(C2.3)**. We first extract reference videos by using expert settings, as presented in subcomponent **(C2.1)**. These reference videos are then matched with the positive candidate videos to detect the real copies, as discussed in subcomponent **(C2.2)**. The subcomponents **(C2.1)** and **(C2.2)** are driven within a loop-based methodology to support both goals including video selection and validation. At last, the parameters serve in the final step to generate the negative videos as demonstrated in subcomponent **(C2.3)**. The following sections will present the subcomponents and the loop-based methodology in more details.

The extraction of the reference videos (C2.1)

As was mentioned in Component **(C1.4)** and illustrated in Fig. 2.5, the positive candidate videos contain TV jingles (i.e., short videos used in television to announce shows, series, or movies). The extraction of reference videos aims to cut the video segments corresponding to the jingles from positive candidate videos. It is a standard task that can

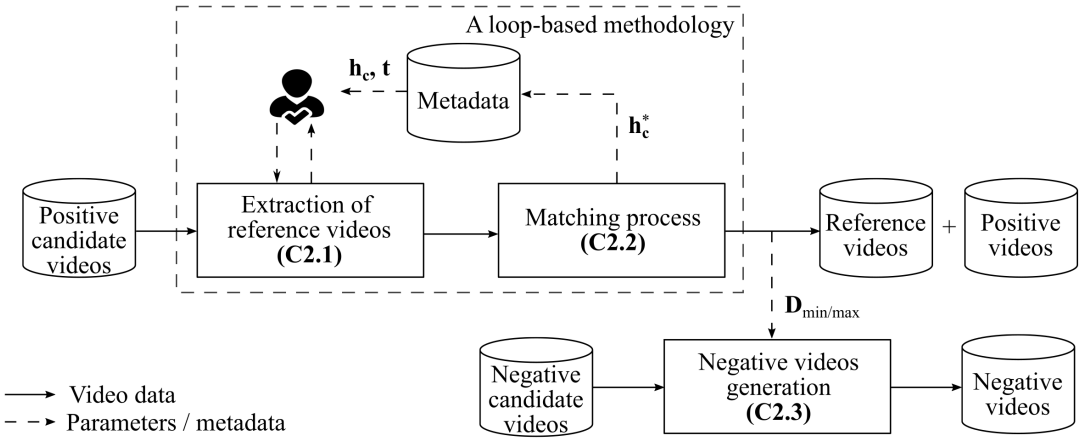


Fig. 2.7 Our pipeline for video detection.

be done by a user handling a GUI video editor. The user needs only general knowledge about the TV programs (e.g., channels that broadcast, content of popular shows, series, etc.).

However, this process could require a huge amount of user interaction considering the large number of positive candidate videos. In addition, the reference videos must be extracted precisely (i.e., temporal boundaries of the reference video). This is a desirable property for the PVCD datasets (see Chapter 1). It is a sophisticated and time-consuming task even for an experienced annotator. To achieve this aim, we have used the specific GUI video editor[7] supporting a frame-to-frame display mode for an accurate extraction. To minimize the user interaction cost, we have used the metadata to automatically recommend and locate the right video content for processing, as illustrated in Fig. 2.7.

We first have marked the list of hashcodes as unlabeled $\mathbf{H} = [\mathbf{h_1}, \ldots, \mathbf{h_q}]$. We then selected the unlabeled hashcode $\mathbf{h_c}$ with the highest number of occurrences. This guarantees to maximize the number of partial copies for each extraction of a new reference video. The scheduled timestamp information $\mathbf{t}$ in the metadata is used to locate the video segment for extraction. In addition, the positive candidate videos have the maximum size of $\mathbf{W}$, fixed by Component **(C1.4)**, allowing the user to bound the range of segments to investigate. To validate the correct video content for extraction, the user double-checks the reference video from positive candidate videos by visual inspection. When the hashcode $\mathbf{h_c}$ has been processed, the next item in the list $\mathbf{H}$ is selected to proceed until it is completed.

After extraction, the reference video is matched against all positive candidate videos having the same hashcode. This guarantees to bound the search space for detection accuracy and time processing optimization. It is important to note that the final set of reference videos requires a validation step to ensure there are no near-duplicates among the reference videos. To mitigate that issue in this section, we will address it after discussing the matching process **(C2.2)** in the following section.

[7]The free video editor software Avidemux: http://avidemux.sourceforge.net/

The matching process (C2.2)

After the extraction, any reference video needs to be matched against the positive candidate videos, having the same hashcode, to produce the groundtruth. To do this, a brute-force strategy in the literature is the full frame matching [39, 104]. Here, all the successive frames of a reference video are matched against the successive frames of the positive candidate videos. This process could be error-prone and it has a quadratic time complexity requiring a large amount of comparisons. Thus, robust and time-efficient features with metrics must be proposed to support the matching between frames.

Many features in the literature have been investigated to determine the similarity between frames (e.g., BRIEF [118], SIFT [123], CNN features [114]). They can handle major degradations and geometric transformations but require a huge amount of time processing for extraction and matching. In addition, they are little suitable for the detection problem as the captured videos in Component **(C1)** are "noise-free" with a low-level distortion error and contrast deviation (our experimental results are reported in Section 2.4). We therefore have considered as a direct matching method the Zero-mean Normalized Cross-Correlation (**ZNCC**) metric. This metric fits well with the detection problem as it is robust to noise and contrast-invariant [29, 64].

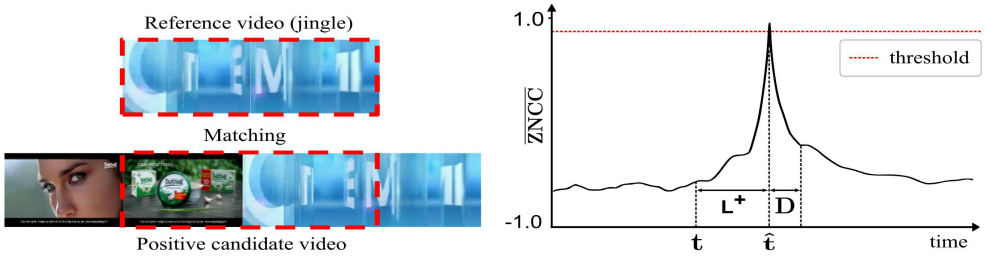


Fig. 2.8 Demonstration of the video matching with **ZNCC** metric.

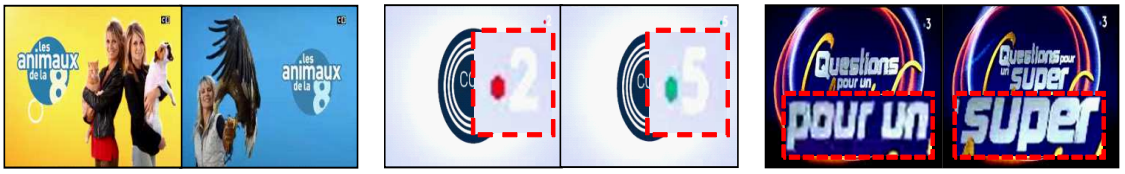
Our approach is illustrated in Fig. 2.8, and the detailed implementation is presented in Appx. A.3. A global $\overline{\mathbf{ZNCC}}$ score was computed with an averaging from the individual frame matching and then compared to a threshold for the detection. To ensure high accuracy, the matching has been done at a high resolution (i.e., 64×48 pixels). The threshold was fixed based on the matching results obtained from comparisons between the reference videos and a subset of negative candidate videos. This subset of negative candidate videos has been selected randomly. For a better result, we have set a weighting method for timing accuracy in frame-level annotation (e.g., 2-3 frames precision). This accuracy is mainly dependent on the weighting methods used with the averaged $\overline{\mathbf{ZNCC}}$. As discussed in Appx. A.3, mean averaging appears as the simple but the best method. The maximum score gives the timestamp for detection $\hat{\mathbf{t}}$. The difference with the scheduled timestamp is the latency $\mathbf{L} = \hat{\mathbf{t}} - \mathbf{t}$.

When the matching is done, a validation step is required from the user to compare the results at the metadata and video levels. The validation at the metadata level ensures that the number of positive videos reaches its maximum capability. The one at the video level avoids the near-duplicate videos among the reference videos. This requires several additional video matching steps. Hence, we have driven a loop-based method to support

the user for video selection which is presented in the next section.

The loop-based methodology

As discussed in the two previous sections and illustrated in Fig. 2.7, the processing of reference and positive videos is performed within a loop-based methodology. This methodology aims to minimize the user interaction cost when driving the two subcomponents **(C2.1)** and **(C2.2)**. The unlabeled hashcode $\mathbf{h_c}$, having the highest number of occurrences, is selected for inspection and extraction. It is labeled into a hashcode $\mathbf{h_c^*}$ when a jingle is detected in the positive candidate videos and validated by the user. These steps are repeated until a sufficient number of hashcodes have been labeled with extraction/detection of reference and positive videos.



(a) (b) (c)

Fig. 2.9 References (a) with a different visual content (b) (c) that are near-duplicate.

However, a key constraint is the variety in visual content of reference videos Fig. 2.9. This requires to correct the hashcodes. The case (a) is related to reference videos having different visual content where the hashcode needs to be split. The cases (b) and (c) correspond to near-duplicated reference videos that require to merge two hashcodes together. Hence, an overall procedure must be fixed to select/correct the hashcodes while minimizing the user interaction cost. The procedure is given below in a pseudo-code using the symbols introduced in Tab. 2.1.

* **Initialization.** $\mathbf{H} = [\mathbf{h_1}, \ldots, \mathbf{h_k}, \ldots, \mathbf{h_q}]$ and $\mathbf{H}^* = [\mathbf{h_1^*}, \ldots, \mathbf{h_l^*}, \ldots, \mathbf{h_n^*}]$ are unlabeled and labeled hashcode arrays, respectively, at the initialization $\mathbf{H}^* = \emptyset$. The sets of positive candidate videos and positive videos are $[\ldots, \mathbf{V_i}, \ldots]$ and $[\ldots, \mathbf{V_j^*}, \ldots]$, respectively. $[\ldots, \mathbf{R_u}, \ldots]$ is a set of reference videos, $\tilde{\mathbf{R}}$ is a candidate for this set. A set of matched videos is $[\mathbf{X_1}, \ldots, \mathbf{X_m}] \subseteq [\ldots, \mathbf{V_i}, \ldots]$.

Step 1. $\mathbf{h_c} := \max_{\forall \mathbf{k} \in [\mathbf{1}, \mathbf{q}]} (\mathbf{C}(\mathbf{h_k}))$ with a counting function $\mathbf{C}$ returning the number of occurrences of a hashcode $\mathbf{h_k}$, the max function affecting a hashcode variable $\mathbf{h_c}$. Using a GUI, the user double-checks visual content from positive candidate videos $\mathbf{V_i}$ having the hashcode $\mathbf{h_c}$, and extracts a reference video $\tilde{\mathbf{R}}$, go to Step 2.

Step 2: match $\tilde{\mathbf{R}}$ with the set of positive candidate videos $[\ldots, \mathbf{V_i}, \ldots]$ to get the linked videos $[\mathbf{X_1}, \ldots, \mathbf{X_m}] \subseteq [\ldots, \mathbf{V_i}, \ldots]$, then check,

- if $\mathbf{m} = \mathbf{C}(\mathbf{h_c})$, label $\mathbf{h_c}$ as $\mathbf{h_c^*}$, move $\mathbf{h_c^*}$ from $\mathbf{H}$ to $\mathbf{H}^*$, go to Step 3.

- else if $\mathbf{m} \neq \mathbf{C}(\mathbf{h_c})$, create[8] a new hashcode $\mathbf{h_{q+1}}$, set the occurrences of the hashcode $\mathbf{h_{q+1}}$ with the value $\mathbf{C}(\mathbf{h_c})-\mathbf{m}$, label this hashcode and set $\mathbf{h_c^*} = \mathbf{h_{q+1}^*}$, push $\mathbf{h_c^*}$ to $\mathbf{H}^*$, go to Step 3.

Step 3. match $\tilde{\mathbf{R}}$ with the set of reference videos $[\dots, \mathbf{R_u}, \dots]$, then check,

- if there is no near duplicate, add $\tilde{\mathbf{R}}$ to the set $[\dots, \mathbf{R_u}, \dots]$, update all the video $[\mathbf{X_1}, \dots, \mathbf{X_m}]$ to $[\dots, \mathbf{V_j^*}, \dots]$ with $\mathbf{h_c^*}$, go to step 1 or stop.
- else if $\exists \mathbf{R_l} \approx \tilde{\mathbf{R}}$, get $\mathbf{h_l^*}$ the hashcode of the $\mathbf{R_l}$ the reference video, update $[\mathbf{X_1}, \dots, \mathbf{X_m}]$ to $[\dots, \mathbf{V_j^*}, \dots]$ with $\mathbf{h_l^*}$, delete $\tilde{\mathbf{R}}$, clear $\mathbf{h_c^*}$, go to Step 1 or stop.

Within the proposed protocol, the minimization of the user interaction cost is achieved with three key mechanisms: (i) a time-efficient implementation for matching that is suitable for user interaction as depicted in Appx. A.3, (ii) the max operator to get the hashcode $\mathbf{h_c}$ at Step 1, and (iii) the unlabelling of the hashcode $\mathbf{h_c}$ after creating the new hashcode $\mathbf{h_{q+1}}$ at Step 2. Indeed, the first extraction of the reference video having the lowest number of occurrences (e.g., $\mathbf{m} = \mathbf{m_1}+\mathbf{m_2}$ with $\mathbf{m_1} = \mathbf{C}(\mathbf{h_{q+1}})$ and $\mathbf{m_1} \ll \mathbf{m_2}$) involves to re-process the hashcode $\mathbf{h_c}$ to recover the $\mathbf{m_2}$ positive candidate videos.

The negative videos generation (C2.3)

In order to stress the system, negative videos must be provided. However, it would make little sense to generate these videos by either a fixed or fully random duration. To avoid this issue and be close to a realistic scenario, we have adopted the video distribution extracted from the subcomponent **(C2.2)** Fig. 2.7 . This distribution is computed from the durations of reference videos $\mathbf{D} \in [\mathbf{D}_{\min}, \mathbf{D}_{\max}]$ obtained from the last loop. As discussed in our component **(C1.4)**, within any interval $\mathbf{W}$ a negative candidate video is timestamped at $\mathbf{t} = \mathbf{W}^-$. The final selection of a negative video in **(C2.3)** is obtained by randomly applying a uniform duration of $\mathbf{D}$ and then extracting a video segment located at $[\mathbf{t}, \mathbf{t}+\mathbf{D}]$.

2.3.4 Video degradation (C3)

The positive/negative videos obtained from the components (**C1**) and (**C2**) correspond to true-life captures with real noisy conditions. For needs of performance evaluation, a common strategy is to apply additional synthetic methods to degrade the videos [11, 45, 49, 62, 81]. By performing transformations, a fine performance characterization can be handled, and more challenging datasets can be designed to stress the methods for detection.

Similar to the works [11, 45, 49, 62, 81], we have selected a set of representative methods detailed in Tab. 2.6 labeled, $\mathbf{T_0}$ to $\mathbf{T_6}$. These are applied to both the positive and negative videos. For the performance evaluation of PVCD methods, we use first a transformation $\mathbf{T_0}$ to get long videos embedding the positive videos. Then, the methods enter in two categories (see Fig. 2.10): pixel attacks $\mathbf{T_{1-2}}$ (b) and global transformations $\mathbf{T_{3-5}}$ (c). A final transformation $\mathbf{T_6}$ is used for video speeding. The $\mathbf{T_0}$ is a specific transformation in our approach designed with our latency measure $\mathbf{L}$ (see Fig. 2.8). $\mathbf{T_{1-6}}$ are baseline image processing that performs on $\mathbf{T_0}$. We detail these transformations as follows.

[8] e.g., with a text labeled from -00 to -99 adds to the text key (see Tab. 2.5).

Tab. 2.6 Degradation methods for video transformation.

Label	Method	Parameters
$\mathbf{T_0}$	video cut	uses the latency distribution to cut segments before/after the video and having a duration $\|\mathbf{L}^-\|, \mathbf{L}^+$, respectively.
$\mathbf{T_1}$	downscaling	applies a random downscaling $\alpha \in [0.1, 0.9]$ to get frames from 24×32 up to 216×288 for a robust matching with time optimization [96].
$\mathbf{T_2}$	compression	processes with a parameter $\frac{1}{\beta}$ with $\beta \in [1, 80]$ applied to the recommended kbps $\in \{140, 280, 420\}$ for capture [4] such as $\frac{1}{\beta} \times$ kbps.
$\mathbf{T_3}$	flipping	applies randomly (yes/no) a flipping transformation to the video.
$\mathbf{T_4}$	rotating	applies a random vertical/horizontal rotation $\in \{0, \frac{\Pi}{2}, \Pi, \frac{3}{2}\Pi\}$.
$\mathbf{T_5}$	black border & stretching	selects an aspect ratio $\frac{w}{h} \in \{0.46, 0.56, 0.63, 0.75, 1.33, 1.6, 1.78, 2.17\}$ to introduce left/right borders ($\frac{w}{h} < 1$) or to stretch the image ($\frac{w}{h} > 1$).
$\mathbf{T_6}$	video speeding	speeds down the videos at a FPS $\in [15, 25]$ using a network throughput predictor [67].

(a) (b) (c)

Fig. 2.10 Degradations (a) reference (b) pixel attack (c) global transformations.

For the needs of the PVCD, short negative/positive videos must be embedded into the longest segments $\mathcal{S}$. The issue here is how to fix the duration of the negative video segments to include before and after the copy of positive videos. To address this problem, we have characterized the video latency appearing in the TV broadcasting and modeled it to generate our partial video copies. More specifically, $\mathbf{T_0}$ extracts additional left/right video segments within the window of size $\mathbf{W}$ (see Fig. 2.11 (a)). The duration of $\mathcal{S}$ must be fixed, we have set $\mathbf{T_0}$ with the latency distribution obtained with the component (**C2**). As shown in our experimental Section 2.4, the latency values obtained with (**C2**) respect the testing conditions of a normal distribution. We have modeled then the latency as a Gaussian distribution. This latency model was used for every short negative/positive video $\mathbf{i}$. Considering a positive video detected at $\hat{\mathbf{t}}$ from (**C2.2**), or a short negative video timestamped at $\mathbf{t} = \mathbf{W}^-$ from (**C2.3**), $\mathcal{S}_\mathbf{i}$ is obtained by cutting a long video segment at $\mathbf{s_i} = \hat{\mathbf{t}}_\mathbf{i} - |\mathbf{L}^-|$ and $\mathbf{e_i} = \hat{\mathbf{t}}_\mathbf{i} + \mathbf{D_i} + \mathbf{L}^+$ (and with $\mathbf{t_i}$, respectively) with $\mathbf{L}^-, \mathbf{L}^+$ random negative/positive latency values.

A segment $\mathcal{S}$ could be extracted for any short negative video. Indeed, a selection at $\mathbf{t} = \mathbf{W}^-$ within a window of size $\mathbf{W}$ by (**C2.3**) cannot result in an overlapping case while using the latency in $\mathbf{T_0}$. However, such overlapping cases could appear within the short positive videos while introducing values $\mathbf{L}^-, \mathbf{L}^+$ as illustrated in Fig. 2.11 (b). Therefore, we have used a merging step in order to solve this problem. Considering two videos detected at $\hat{\mathbf{t}}_\mathbf{i}, \hat{\mathbf{t}}_\mathbf{j}$ where $\hat{\mathbf{t}}_\mathbf{i} < \hat{\mathbf{t}}_\mathbf{j}$ and $\mathbf{e_i} > \mathbf{s_j}$, a mean segment $\mathcal{S}_\mathbf{u}$ must be computed and preserved if $\forall \mathcal{S}_\mathbf{i} \in \mathcal{S}_\mathbf{u}$, $\mathbf{s_u} < \hat{\mathbf{t}}_\mathbf{i}$ and $\hat{\mathbf{t}}_\mathbf{j} + \mathbf{D_j} < \mathbf{e_u}$ as demonstrated in Fig. 2.11 (b). That is, a long positive video could contain several reference videos for testing.

We apply next on $\mathbf{T_0}$ a set of baseline image processing $\mathbf{T_{1-6}}$ for degradation. In particular, $\mathbf{T_1}, \mathbf{T_2}$ are used to get videos at low-level resolutions with compression. Two

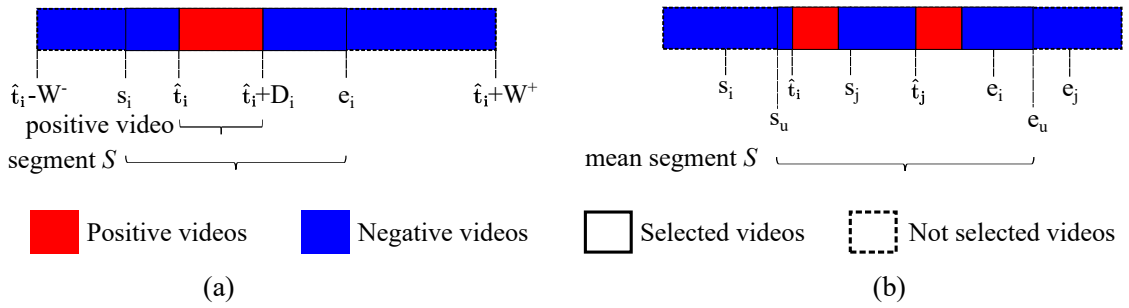


Fig. 2.11 (a) a segment $\mathcal{S}$ (b) overlapping case.

correlated parameters α, β control the level of video degradation. Further details can be found in Appx. A.4. $\mathbf{T_3}$ and $\mathbf{T_4}$ apply realistic geometric transformations for video rendering as the flipping and the horizontal/vertical rotations. The aspect ratio parameters in $\mathbf{T_5}$ have been fixed with the standard screen resolutions[9] to introduce or rescale the videos. For the video speeding $\mathbf{T_6}$, the videos have been controlled with the predefined FPS values of $\in [15, 25]$ as similar to [45]. However, in order to fit with real-world scenarios, the FPS parameters have been generated from the popular autoregressive integrated moving average model which was used in the prediction of network throughput [67].

We have combined the degradations $\mathbf{T_0}$ - $\mathbf{T_6}$ to generate the test sets A to F as detailed in Tab. 2.7. The test set A gives a root capture while applying only $\mathbf{T_0}$. It is given for the needs of tuning a performance evaluation task. The test sets B and C apply a pixel attack with $\mathbf{T_1}$ and $\mathbf{T_2}$ at two levels of degradation with the control of parameters α, β. The predefined resolution and compression parameters used in set A ($\alpha = \beta = 1$) have been fixed to ensure a negligible degradation with a storage optimization strategy. The test set B has a low-level of distortion and scalability that constitutes a "hello world" benchmark. The test set C presents a hard pixel attack. The test set D is related to the global transformations with $\mathbf{T_3}$ to $\mathbf{T_5}$, whereas the test set E applies $\mathbf{T_6}$ for video speeding. At last, the test set F combines the sets C, D, and E.

Tab. 2.7 Test sets.

Test set	$\mathbf{T_0}$	$\mathbf{T_{1-2}}$	$\alpha \in$	$\beta \in$	$\mathbf{T_{3-5}}$	$\mathbf{T_6}$	Description
Set A	√		1	1			Root capture for further tuning
Set B	√	√	$[0.25, 0.9[$	$[1, 40[$			“Hello world" test set
Set C	√	√	$[0.1, 0.25]$	$[40, 80]$			Pixel attack with scalability
Set D	√	√	0.6	20	√		Global transformations with scalability
Set E	√	√	0.6	20		√	Video speeding with scalability
Set F	√	√	$[0.1, 0.25]$	$[40, 80]$	√	√	Combination of sets C, D and E

2.4 Experiments and results

We report in this section our experiments and results. While applying our pipeline shown in Fig. 2.2, our experiments have been conducted to generate the final dataset

[9]For desktop, tablet, and phone https://gs.statcounter.com

detailed in Tab. 2.8. Sections 2.4.1 to 2.4.3 report experiments on our different components **(C1)** to **(C3)**. Performance evaluation results of representative methods on the dataset are given in Section 2.4.4. A statistical comparison between the STVD and the main recent publicly available datasets is presented in Section 2.5. At last, conclusions and perspectives are discussed in Section 2.6.

Tab. 2.8 The STVD dataset.

	(C1)				**(C2)**		**(C3)**		
	Channel	Duration	Captured videos	Candidate videos	Videos	Duration	Test set	Videos	Duration
Positive set	8	4 800 h	240	5 737	3 780	6 h	6	19 280	2 515 h
Negative set	16	9 600 h	480	16454	12 165	21 h	6	64 040	8 145 h

2.4.1 Video capture (C1)

We have first driven a capture campaign by our TV workstation for three consecutive months. After capturing 24 public channels, we obtained a root dataset composed of 14 400 hours of TV programs corresponding to 720 daily captured video files and having a volume size of 3.46 TB. The root dataset is divided into two subsets for the positive and negative videos, respectively. We have processed these videos with three components from **(C1)** to **(C3)** in order to produce the STVD dataset as shown in Tab. 2.8. Considering thousands of hours of recorded video, we have selected a one-month capture from 8 TV channels, resulting in 240 video files as the positive set. The remaining 16 channels, containing 480 video files, have been used to generate the negative set.

Along with these captured videos, 720 metadata files were collected daily. We have processed the metadata to extract the list of unique hashcodes $\mathbf{h_1}, \ldots, \mathbf{h_q}$ and their occurrences, as depicted in Fig. 2.12 (a). We have obtained nearly one thousand hashcodes for a total number of 7 310 videos. As discussed in Section 2.3.2, to limit the user interaction and ensure reliability, we have considered only the daily programs for groundtruthing. Thus, we have analyzed the hashcodes to classify TV programs according to their frequency (i.e., daily vs. not daily) as shown in Tab. 2.9. It can be highlighted in Fig. 2.12 (a) that the main part of the videos ($\simeq$ 6 000 video occurrences occupying 83% in the total of video occurrences Tab. 2.9) is related to a small number of hashcodes ($\simeq$ 300 daily programs). In addition, to limit user interaction we have filtered out daily programs with very low occurrence levels by applying a threshold of 4 as illustrated in Fig. 2.12 (b).

Tab. 2.9 Program categories processed with the metadata.

	Daily	Not daily	Total
Percentage of hashcodes (%)	31	69	100
Percentage of video occurrences (%)	83	17	100

Regarding the error-prone cases caused by the naming convention of the TV program titles, we have computed a distribution of Levenshtein Distance (LD) distances as shown in Fig. 2.13 (a). For feasibility, a suspicious subset was selected from this distribution using a threshold (i.e., 8) to analyze the consistency of those programs. For instance, Fig.

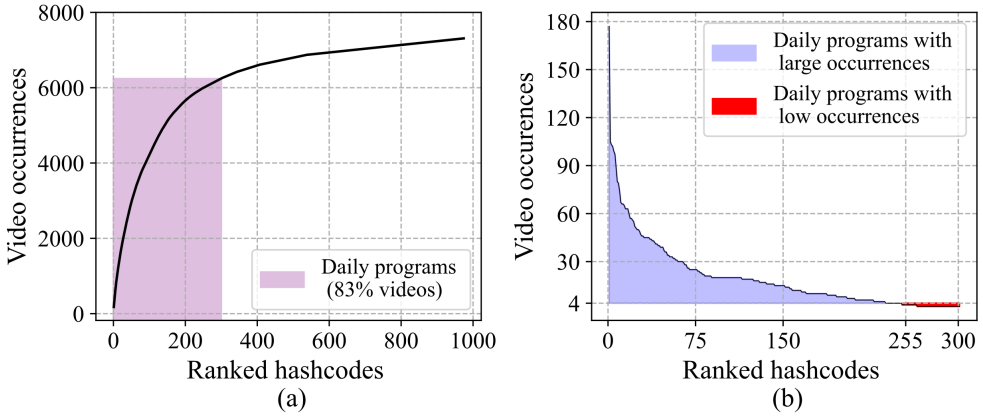
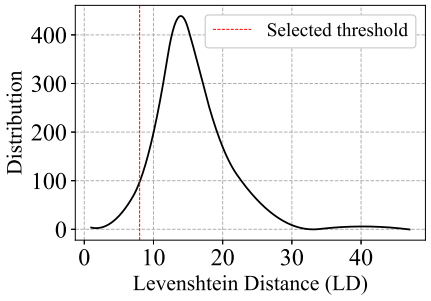


Fig. 2.12 Hashcodes and video occurrences:
(a) cumulated distribution for all the programs, (b) distribution of daily programs.

2.13 (b) shows some programs that require a further visual inspection step. From the row (3) to (4) in Tab. 2.10, we can see that only a small portion of the programs need to be grouped by the visual inspection step (i.e., $< 1\%$).



(a)

Programs were removed		Programs were kept		LD
Normalized title	No. videos	Normalized title	No. videos	
le1245	5	le1945	19	1
swat	9	swat2017	15	4
teleshoppingsamedi	5	teleshopping	24	6
balancetonpostlafter	4	balancetonpost	8	6
28minutessamedi	5	28minutes	20	6

(b)

Fig. 2.13 (a) The LD distribution, (b) removal of the closest titles from the LD distribution.

In summary, we have obtained a set of the positive candidate videos as shown in Tab. 2.10. It is highlighted that we have filtered out a main part of hashcodes ($\simeq 70\%$), that could require a huge amount of user interaction for labeling, while preserving a large amount of positive candidate videos ($\simeq 86\%$) as given on the row (2) in Tab. 2.10. For higher reliability and lower user interaction cost, a small part of the hashcodes and positive candidate videos (i.e., $\simeq 7\%$, and 8%, respectively) have been removed if they present in the metadata with low occurrences or inconsistent names. As a result, the final set of 5 737 positive candidate videos corresponding to 248 reference videos were generated as shown in the last row in Tab. 2.10. It indicates that Component (**C1**) results in the lowest values of hashcodes (25.5%) while it covers a large number of positive candidate videos (78%).

For negative candidates, on the other hand, we have processed the metadata to make idle/elimination for selection of all the segments where reference videos could appear as presented in Section 2.3.2. As a result, we have obtained 16 454 valid/not idle intervals for

Tab. 2.10 The number of hashcodes and positive candidate videos processed with our pipeline.

	Hashcodes		Positive candidate videos	
	No.	%	No.	%
All (1)	974	100	7 310	100
Daily programs (2)	301	31.0	6 260	86
Thresholding for daily programs (3)	255	26.0	5 737	78
Visual inspection (4)	248	25.5	5 737	78

the selection of the negative candidates as shown in Tab. 2.8.

2.4.2 Video detection (C2)

For this experiment, we have applied a loop-based method for the selection of reference videos as mentioned in Section 2.3.3. First, we have cut a total of 248 samples using the GUI, which will be severed as the reference videos Tab. 2.10 (i.e., row (4)). At the first iteration, a threshold was then fixed based on the maximal interclass matching score obtained from comparing between the reference videos and a subset of negative videos to ensure none of the false positive case with high probability. That is, a positive video is detected if its matched score is greater than the maximal interclass matching score (e.g., $\overline{\mathbf{ZNCC}} > 0.8$) as illustrated in Fig. 2.14 (a). When the matching is done, these videos were corrected by a validation step with the user support to produce the final set of 243 reference videos. They were used for the second loop to detect the 3 780 positive videos having a total duration of 6 hours as shown in Tab. 2.8. That is, the partial copies a have short duration $\mathbf{D} \in [1, 25]$ seconds. Besides the positive set, the negative set contains 12 165 videos having 21 hours in total duration.

From the results in Fig. 2.14 (a), strong evidence of the **ZNCC** metric was found in distinguishing between the interclass and intraclass $\overline{\mathbf{ZNCC}}$ distributions $\in [0.79, 0.90]$. Furthermore, we conducted the experiments to characterize the compression and contrast noises within the $\overline{\mathbf{ZNCC}}$ distribution by using the standard metrics Mean Square Error (MSE) [41] and Contrast Noise Ratio (CNR) [50]. As shown in Fig. 2.14 (b), very low levels of noise were obtained with the MSE < 20 and CNR < 0.01 for most of the matching cases. Together these results firmly demonstrate that the **ZNCC** metric fits very well with our detection problem.

As a common issue of TV broadcasting (see Fig. 2.5), we have observed a list of latency values compared with the timestamp given by the metadata. These values are meaningful to be used in order to introduce before/after copied segments for the PVCD problem. Thus, we have characterized the values for this purpose. In particular, we have observed an approximately Gaussian latency distribution as shown in Fig. 2.15 (a). We have then used the duration with $\mathbf{D} \in [1, 25]$ seconds and the latency distribution with $\mathbf{L} \in [-590, 820]$ seconds at $\pm\mathbf{3}\sigma$ shown in Fig. 2.15 (b) to compute the $\mathbf{D}_{\min\,/\,\max}$, $\mathbf{L}_{\min\,/\,\max}$ parameters values that will be used in Components (**C2**) and (**C3**).

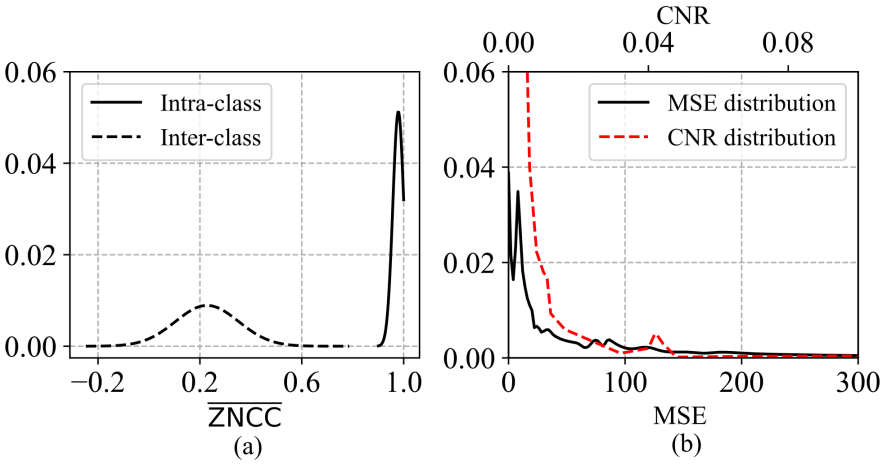


Fig. 2.14 Distribution of (a) $\overline{\mathbf{ZNCC}}$, (b) MSE and CRN.

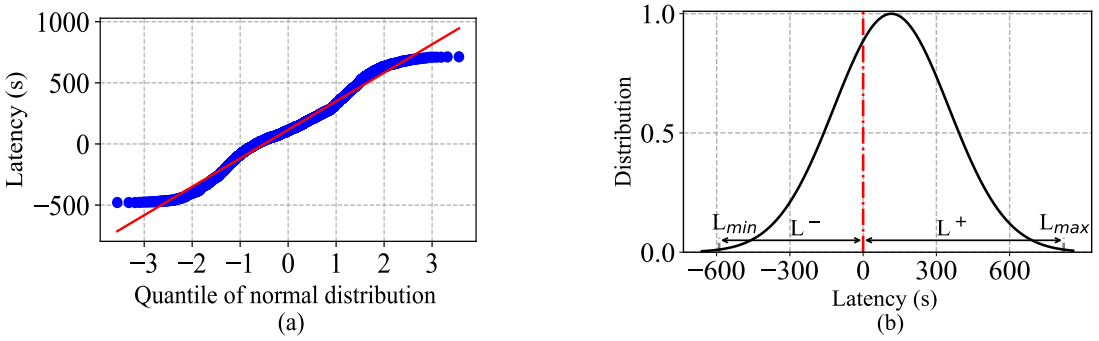


Fig. 2.15 (a) quantile of normal distribution (b) latency.

2.4.3 Video degradation (C3)

We have then applied the component (**C3**) to get the 6 test sets as discussed in Section 2.3.4. For the test sets A, C, D, E, and F, we have obtained $5 \times 3\,780 = 18\,900$ and $5 \times 12\,165 = 60\,825$ positive/negative videos, respectively. The test set B has been generated in balance for low scalability with a total number of $2 \times 3\,780 = 7\,560$ videos. However, we have observed a $\simeq 15\%$ of covering cases with the positive videos (Fig. 2.11 (b)). Therefore, we have obtained the final dataset with a total amount of $\simeq 83$ thousand videos composed of the 19 280 positive and 64 040 negative videos (Tab. 2.8). Considering the latency distribution (Fig. 2.15 (b)), the application of $\mathbf{T_0}$ has resulted in an average duration of 7.5 minutes for the videos. The total duration of the dataset is 10 660 hours with 2 515 hours and 8 145 hours for the positive/negative videos, respectively (Tab. 2.8). Each test set C to F contains $\simeq 1\,960$ hours of testing videos for scalability.

2.4.4 Performance evaluation

In this section, we present performance evaluation results on the STVD dataset of representative PVCD methods [114, 118, 123] Tab. 2.11. These methods are processed in two steps for key-frame extraction and matching. The key-frame extraction selects candidate frames for matching based on sampling methods [114, 118] or temporal features [123]. The matching processes with features (CNN [114], BRIEF [118] and SIFT [123]) and optimization components for the time processing requirement.

Tab. 2.11 Representative PVCD methods for the performance evaluation.

Method	Key-frame extraction	Frame matching
[114]	Frame clustering with an adaptive color histogram	Deep features with the VGG16 network. CNN features are aggregated for optimization with mean pooling.
[118]	FPS sampling and a Coset Group Method (CGM).	Matching with the BRIEF features, Binary Temporal Alignment (BTA) is used for optimization.
[123]	Temporal-concentration SIFT (TCSIFT) features.	Multi-assignment K-NN searching of TCSIFT features with LSH.

We have applied a protocol for a fair comparison. We have normalized the key-frame extraction step within all the methods. The SIFT and BRIEF features do not supporting the global transformations. We have bounded the evaluation to the test sets B and C only. We have characterized the methods in a learning-free/pre-trained mode. Only the SIFT and BRIEF features of testing frames have been stored for comparison. The CNN features have been obtained from a VGG16 network pre-trained on the ILSVRC dataset[10]. We have also removed the optimization components for a strongest accuracy. The F_1 score has been used as it is common to characterize the PVCD as discussed in Section 1.3.2.

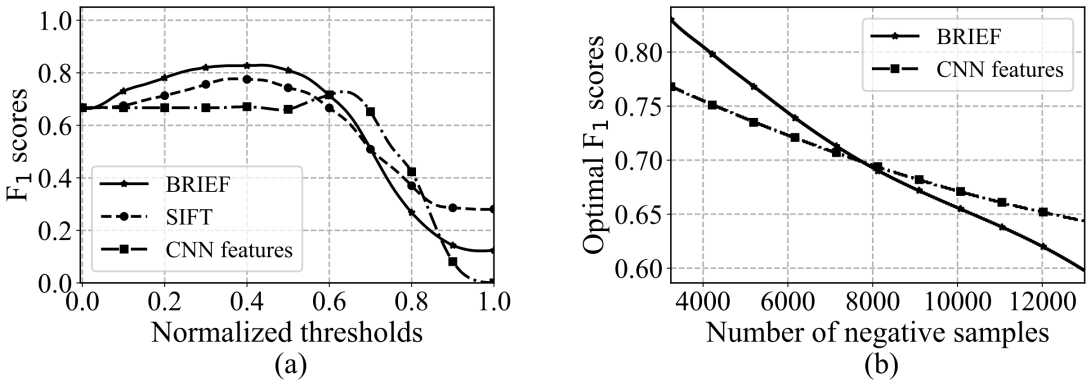


Fig. 2.16 F_1 scores on the test set C:
(a) comparison of methods [123, 118, 114] (b) performance with scalability for [118].

We have evaluated first the method [118] on the test set B. We have obtained a score $F_1 = 0.98$ highlighting the "hello world" ability. Further experiments have been investigated on the test set C (Fig. 2.16). We have constituted first a subset with balanced 3 000 + 3 000 for positive and negative videos, respectively. Fig. 2.16 (a) gives the F_1 scores

[10]ImageNet Large Scale Visual Recognition Challenge https://image-net.org/

against the normalized thresholds for all the methods. We have obtained optimum scores $F_1 \in [0.73, 0.83]$ with a top $F_1 = 0.83$ for the method [118]. A gap $\simeq 0.15$ appears for [118] between the test sets B and C due to the pixel attack. Fig. 2.16 (b) reports the results of the top methods [118, 114] on the full test set C while increasing the negative videos up to 12 000. We have observed a gap $\simeq 0.25$ for the F_1 score due to the scalability with better robustness for the CNN features [114].

2.5 Statistical comparison

As shown in Tab. 2.2, none of the existing TV datasets is publicly available, thus they are no longer used by the research community. We have selected three publicly available recent datasets from Tab. 1.2 for a statistical comparison. Tab. 2.12 presents a comparison between STVD and three major datasets in the literature. Compared with the existing datasets, several key strengths of the STVD dataset have emerged to contribute to this growing topic of PVCD: (i) it is captured from TV and is almost noise-free allowing a fine control of degradations with synthetic methods, (ii) it is the largest dataset with ten thousand hours of video, 243 references, and 1 688 thousand positive pairs, (iii) it offers a balanced distribution between the negative and positive videos, and (iv) it is delivered with an accurate timestamping for video alignment.

Tab. 2.12 Comparison between STVD and main existing datasets.

Datasets	VCDB	SVD	STVD	VCSL
Paper	[47]	[45]	[65]	[35]
Year	2014	2019	2021	2022
Source of capture	Web	Web	TV	Web
Degradation	real	synthetic	synthetic	real
References	28	1 206	243	122
Positive videos	528	n/a	19 280	9 207
Positive pairs	9 K	10 211	1 688 K	281 K
Negative videos	100 000	526 787	64 040	n/a
Duration (h)	2 030 h	2 705 h	10 660 h	17 416 h
Timestamps (s)	1 s	n/a	$\frac{1}{30}$ s	1 s
Annotation cost (m-h)	700	800	105	20 000

(h): hours, (s): seconds, (m-h): man-hours, and n/a: not available

2.6 Conclusions and perspectives

This chapter has presented a new protocol to design a dataset for the performance evaluation of the PVCD methods. This protocol was designed to process multiple TV channel videos. Three major advantages of our protocol can be exploited. First, given the dedicated TV workstation, which can support a stable multi-channel capture in real-time, our protocol can be applied to produce the videos with a fine control of noise. This is a critical point to apply the robust **ZNCC** metric to deal with such noisy features compared to others. Second, a window size **W** was proposed to address a common latency issue as well as to avoid a full-search strategy while processing TV videos compared to state-of-the-art approaches in the literature. At last, our protocol can be simply adapted to design

such scalable datasets for several problems, for instance, the parallel machine scheduling[11] or the fact-checking [84].

Another important outcome is that our protocol has been used to generate a new dataset, named STVD, for performance evaluation of the PVCD problem. Indeed, STVD was designed with the protocol ensuring a scalable capture and robust groundtruthing. To the best of our understanding, STVD is currently the largest public dataset on the task. It covers nearly 83 thousand videos for a total duration of 10 660 hours. As a baseline comparison, performance evaluation results of representative methods on the dataset were reported to show room for improvement.

For further improvements to this work, more competitive systems should be tested. However, considering the complexity and variety of these systems, such a task is difficult to handle at the individual level. A sensible solution is to promote the STVD dataset within the research community by exchanging and announcing it through mailing lists. Additionally, more experiments and results can be provided if the STVD will be developed as a PVCD contest taking part in such key computer vision conferences[12]. Hence, we will endeavor that the STVD dataset remains available for the coming years to promote research in the PVCD domain. A terms of service agreement is provided to establish the rules and guidelines for the use of the STVD datasets[13].

Our dataset is the first part of this dissertation focusing on performance evaluation of the PVCD methods. In the next part, we attempt to bring a new contribution related to recent trends of deep learning algorithms for the PVCD problem. To achieve this goal, state-of-the-art deep learning approaches must be considered and discussed thoroughly along with many further experiments. First, we plan to investigate some recent models embedding spatial-temporal deformation to make the systems robust against the spatial degradation as well the video latency. Second, an alternative is to process in two steps with a deep learning method and a temporal network for timing performance. At last, considering the possibility of fine control for the degradation with the STVD and its scalability, our dataset could help to design some new deep learning models. We will present this aspect in the next chapter.

[11] Parallel machine scheduling dataset: https://dataset-stvd.univ-tours.fr/
[12] ICIAP, ECCV, CVPR competitions
[13] Terms of Service: https://dataset-stvd.univ-tours.fr/cgu_en.pdf

Chapter 3

Performance evaluation of 2D CNN for the PVCD

Contents

3.1 Introduction

In Chapter 1, we have presented the state-of-the-art studies that particularly focus on the performance evaluation aspect of the PVCD systems. We highlight many limitations of the existing datasets in terms of scalability, the accuracy of the groundtruth, and ability to address fine performance characterization tasks. To fill these gaps, we propose a new protocol for large-scale video capture with high stability and semi-automatic groundtruthing. The protocol was first described in Chapter 2. Then, using this protocol, we have designed a new PVCD dataset, called STVD, appearing as one of the top PVCD datasets in the literature. We show that STVD is significantly larger in scale and diversity than the existing PVCD datasets. To the best of our knowledge, there is only our dataset that can provide a frame-level accuracy in the groundtruth file. Hence, it allows driving a fine performance characterization task for different PVCD problems.

Tab. 3.1 Main symbols and mathematical notations used in the chapter.

Symbols	Meaning
K, M, B, F, f	thousand 10^3, million 10^6, billion 10^9, float and frame/feature vector
x, y, z	scalar values
m, n or m_i, n_j	sizes of sets/vectors with $i, j = 1, 2, \ldots$
$X = [x_1, \ldots, x_n], Y$	X is the feature vector of positive frame ($x_1, \ldots, x_n$ the elements), Y is negative
$\tilde{X}, X^*$	$\tilde{X} \simeq X$ is the near duplicate of X, $X^* \neq X$ has a different reference
$\{X_1, \ldots, X_n\}$	set of feature vectors
$\|\|X\|\|$	L_2-norm of X with $\|\|X\|\| = \sqrt{\sum_{\forall i} x_i^2}$
$X \cdot Y$	dot product between X and Y with $X \cdot Y = \sum_{\forall i} x_i y_i$
$SC(X, Y)$	Cosine similarity $SC(X, Y) = X \cdot Y \in [-1, 1]$ with $\|\|X\|\| = \|\|Y\|\| = 1$
$F_1 = 2\frac{P \cdot R}{P+R}$	F_1 score with P the precision and R the recall
$\phi(X)$	goodness criterion characterizing the separability with X when $\phi(X) \geq 0$
t, $[z_1, \ldots, z_{m+1}]$	observation at t with $[z_1, \ldots, z_{m+1}]$ the $\phi(X)$, $\phi(\tilde{X}_1)$, $\phi(\tilde{X}_m)$ criteria
$z_{\min}, \overline{z}, z_{\max}, \sigma, \tau$	baseline statistics of $[z_1, \ldots, z_{m+1}]$, τ the rate of positive values $z_k > 0$
α, β	thresholds for categorization of frames
$\overline{\mathcal{Z}}$	mean of indices with $\tau = 0$ and $\sigma \leq \alpha$ for a reference to fix the threshold $\beta = \overline{\mathcal{Z}}$
$E_{RR} = 1 - \frac{TP+TN}{Total}$	error rate with TP, TN, FP, FN the true/false negative/positive, $Total = TP + TN + FP + FN$ is the total of classification

In this chapter, we propose to go further and deeper into the performance evaluation of the PVCD while addressing the characterization problem of systems and methods. As was discussed in Chapter 1, deep learning algorithms have emerged as an active research trend in designing PVCD systems and addressing many other computer vision tasks over the past decade. Despite extensive research has been carried out on the PVCD, few studies have quantitatively analyzed the impact of deep learning algorithms. Hence, the primary aim of this chapter is to provide the characterization of PVCD performance, with a particular focus on the systems using deep learning algorithms. An investigation of other PVCD methods may be beyond this PhD scope. Section 3.2 systematically reviews the state-of-the-art studies related to the PVCD systems using deep learning algorithms. In Section 3.3, we present our method and protocol for characterizing the performance of PVCD systems using 2D CNN features. Experiments and results are discussed in Section 3.4, and conclusions along with perspectives are presented in Section 3.5. For convenience, the general terminology conventions used for the PVCD are given on page 15, and Tab. 3.1

gives the main symbols and mathematical notations used in this chapter.

3.2 Related work

As was mentioned in Chapter 1, deep learning algorithms have emerged as an active research trend for designing PVCD systems over the past decade. Among them, 2D CNNs have been widely used in many studies as shown in Tab. 3.2. The table extends the PVCD framework illustrated in Fig. 2, with a focus on the studies using 2D CNNs. Within each component, various techniques have been proposed to improve the overall PVCD performance. Basically, frames are first sampled from videos, and fed into a 2D CNN to extract features. These features are then used to generate the video representations. Finally, a pair of the representations between two videos are matched to determine copies based on a similarity measure. In the following sections, we discuss briefly these components.

Tab. 3.2 Main components of a typical PVCD system using 2D CNN features.

Key-frame extraction	• Sampling method: [5, 30, 34, 35, 36, 38, 39, 45, 47, 53, 54, 55, 56, 65, 74, 88, 99, 103, 104, 117, 119, 122] • Adaptive methods [12, 43, 114]
2D CNN	• VGGNet [45, 53, 55, 65, 99, 104, 114, 117, 119] • ResNet [5, 30, 35, 36, 38, 43, 53, 54, 88, 99] • InceptionNet [53, 55, 56, 74, 103] • AlexNet[47, 55, 56, 117, 122] • Other CNNs [34, 36, 39, 43, 88]
Feature extraction	• Fully connected layers [38, 45, 47, 53, 55, 56, 65, 74, 104, 117] • Convolutional layers [5, 30, 35, 36, 43, 53, 54, 55, 56, 88, 99, 103, 114, 119, 122] • Other methods [43, 119] • Low-dimensional [30, 34, 36, 39, 45, 104, 114]
Video representation	• Frame-level [5, 30, 34, 35, 36, 38, 39, 43, 47, 54, 65, 74, 88, 99, 117] • Video-level [36, 45, 55, 56, 88, 103, 104, 119, 122]
Video comparison	• Cosine similarity [5, 30, 34, 35, 43, 47, 55, 53, 54, 65, 88, 99, 103, 119] • Euclidean distance: [45, 53, 104, 114, 117, 122] • Hamming distance [36, 39, 45, 88] • Chamfer similarity [35, 88] • Learning metrics [38, 39]

3.2.1 Key-frame extraction

A typical video consists of a sequence of frames. In contrast to an individual image, a frame within a video contains not only spatial information but also structural and temporal information across frames. As was illustrated in Fig. 1.1, a video can be organized in a hierarchical structure (i.e., frame, segment, video). Among frames, a key-frame can be selected to comprehensively represent the content of certain consecutive frames. Hence, key-frame extraction methods choose these representative frames to summarize the video while removing redundant frames. In most studies, extracting a series of key-frames to represent the video is a common operation as shown in Tab. 3.2. Indeed, it is done by sampling at the fixed FPS (e.g., 1, 2). The alternative is to fix a constant number of key-frames per video [38, 103]. Additionally, the works [36, 122] fixed the number of segments per video and then applied a time interval to each segment in order to extract key-frames.

While sampling reduces the amount of video data to process, it may still result in repeated or unrepresentative samples. For instance, a video with non-uniform content distribution, where certain segments contain richer information than others, could result in inaccurate representations by sampling methods. To generate a better summarization of the video, the authors [114] proposed a key-frame extraction method based on the k-

mean clustering algorithm. In this work, the specific steps are proposed to compute the number of clusters, followed by the k-means algorithm. The frame closest to the cluster center would be selected as a key-frame. The other work by Jiang et al. [43] proposed a method that formulates the key-frame extraction as a multi-frame classification problem. In particular, after a uniform video sampling, all sampled frames are resized and organized in a tiled pattern. This pattern is then fed into a lightweight pre-trained 2D CNN model to classify it into two classes (i.e., 1: key-frame, 0: non-keyframe). A recent work [12] used an open-source tool to detect the visual content changes, and selected the mid-position frame in each video segment as a key-frame.

Once the key-frames selection is done, these key-frames are then used to extract representative and distinguishing features. It is closely related to image processing techniques in which standard 2D CNNs can be used as a feature extractor. However, there is a variety of pre-trained 2D CNN models as well as the feature extraction methods that have been used in the literature as shown in Tab. 3.2. In the following paragraphs, we review common approaches for video feature extraction with a focus on the studies using 2D CNN models in terms of architectural and methodological aspects.

3.2.2 2D CNN

Over the past decade, many state-of-the-art applications in the computer vision, including image classification, object detection, and segmentation have been developed [27, 113]. They are driven primarily by the outstanding performance of convolutional neural networks in deep learning, which is a subset of the machine learning discipline. A CNN is a type of deep learning model specifically designed for processing structured grid data, such as images. It consists of several types of layers, each of which plays distinct roles in a feature extraction process, also known as feature learning or representation learning techniques. Within each CNN, a cross-correlation operation (or a convolution operation in the sense of CNN) is a key computation that produces a feature map, capturing a variety of features such as edges, textures, and more complex patterns. To generalize to new data, it requires a training process using back-propagation and gradient descent methods on a large number of data samples (e.g., millions of images). Due to their capability to automatically and hierarchically learn spatial hierarchies from input images, 2D CNNs have attracted significant effort, leading to a shift from feature engineering to network engineering.

Among the PVCD studies, the selected 2D CNNs are derived from the deep learning models for the image classification task. They process images as arrays of pixel values. That is, image data is represented as square matrices (e.g., common image sizes $\in [224; 299]$) in the RGB color space as shown in Tab. 3.3. It is then fed into a CNN which is composed of the convolutional layers and followed by the fully connected layers. In the convolutional layers, data values are formed as two-dimensional grids of numbers while they are reshaped into one-dimensional real vectors within the fully connected layers. The last fully connected layer is known as the output layer which has the same number of output nodes as the number of classes needed for the classification purpose. Such standard principles are still adopted and widely applied to design a CNN today.

Main 2D CNNs used for the PVCD studies include AlextNet, VGGNet, InceptionNet, and ResNet as shown in Tab. 3.2. Each of these CNNs may have different versions (e.g.,

VGGNet-11/13/16/19, ResNet-18/34/50/101/152). Tab. 3.3 presents the most popular models used for PVCD in order of time to convey a sense of the CNN history. In particular, AlexNet [59], introduced by Alex Krizhevsky et al. in 2012, achieved a breakthrough performance in the ImageNet competition[1]. This achievement is possible thanks to a deep architecture (i.e., 8 layers) and the use of Rectified Linear Units (ReLUs) for non-linearity, dropout to mitigate overfitting, and data augmentation techniques to improve generalization. VGGNet [92], developed by the Visual Geometry Group at the University of Oxford in 2014. The VGGNet consists of 16 to 19 layers with very small (3×3) convolution filters, which are applied consistently across the network. InceptionNet [97], also known as GoogLeNet published in 2015, introduced a novel architecture called the Inception module. This module simultaneously applies convolutional filters of different sizes (1×1, 3×3, and 5×5) within the same layer, allowing the network to capture features at various scales. In 2016, ResNet, or Residual Network [33] addressed the problem of vanishing gradients in deep networks by introducing residual learning through skip connections. The ResNet architectures can be extremely deep, with various versions such as ResNet-50, ResNet-101, and ResNet-152 having 50, 101, and 152 layers, respectively.

Tab. 3.3 Main CNNs have been used for PVCD. Accuracy indicates the top-1 accuracy of those models on the ImageNet dataset for the image classification task. The CPU inference time was computed by feeding one image for each run and the final result was obtained by the mean value of 1 000 runs. N/A stands for not available.

Model	Year	Selected version	Input size	Hyper-params	Acc (%)	CPU inference time Intel (ms)	AMD (ms)	ARM (ms)
AlexNet [59]	2012	AlexNet	227×227×3	60 965 224	57.2	N/A	N/A	N/A
VGGNet [92]	2014	VGG-16	224×224×3	138 357 548	71.5	18.21	57.63	162.49
InceptionNet [97]	2015	version 3	299×299×3	27 182 195	77.9	10.67	26.37	84.00
ResNet [33]	2016	ResNet-50	224×224×3	25 610 156	74.9	5.73	18.79	45.73

As shown in Tab. 3.3, these models provide a wide range of complexities, hyper-parameter configurations, and processing times. AlexNet, one of the earlier 2D CNNs, has a relatively simple architecture with $\simeq$ 60 million parameters. VGGNet, particularly the VGG-16 version, increases complexity with up to 138 million parameters (i.e., the parameters of VGGNet rely heavily on the fully connected layers occupying $\simeq$ 80% of the total parameters.). InceptionNet introduces inception modules that allow for wider networks with ×5 fewer parameters and more ×2 efficient computational cost compared to VGGNet. ResNet, with $\simeq$ 25 million parameters for ResNet-50, achieves a balance between accuracy and computational efficiency compared to the Inception-v3 model. Overall, as these architectures progress from AlexNet to ResNet, there is a general trend towards more efficient use of parameters and architecture designs that could support achieving more accuracy and optimized processing times. However, their performance can vary across architectures and hyper-parameter choices. Hence, selecting a CNN for PVCD could be a critical decision in the context of performance.

In addition to the popular 2D CNN models reviewed here, several other 2D CNNs could be used for the PVCD studies, as presented in Tab. 3.2. Xception [13], which is an improvement of Inception-v3, was used in a previous work [39]. Recent 2D CNNs (e.g.,

[1]ImageNet challenge: https://www.image-net.org/challenges/LSVRC/

Transformer, Attention networks) were customized to combine with standard CNNs to enhance the quality of video representations, as the proposed works [34, 36, 43]. Another line of research is to develop a twin CNN model, also known as a Siamese network that consists of two identical CNN networks, to learn the similarity between videos by using contrastive learning metrics [54, 56, 90].

3.2.3 Video feature extraction

In the PVCD framework, video feature extraction is a crucial step that can directly influence the final detection result. It involves a process to transform raw video data into a set of representative characteristics which are used for subsequent algorithms (e.g., retrieval, detection, classification). When video key-frames are selected, image processing techniques are generally employed to extract their features. In the early studies, tens of feature types can be applied such as the color histogram, local, or texture features, as were mentioned in Section 1.2.2. However, in the existing PVCD studies using 2D CNN-based features, video features are mainly extracted from popular pre-trained CNN models to leverage the knowledge (i.e., learned representations) gained during the training. Indeed, they are computed through the forward propagation of an image over a pre-trained CNN model. Given a pre-trained CNN model, the features can be obtained from the Fully Connected (FC) layers or the convolutional ones. They are summarized in Tab. 3.2.

As a straightforward method, video features can be extracted from any fully connected layers within a CNN, except for the last fully connected layer which serves as the final output (e.g., classification probabilities, predicted values). While convolutional layers focus on detecting local patterns, FC layers are in charge of capturing global patterns and high-level abstractions. The convolutional layers first identify local patterns by applying many filter and pooling operations to the input image to produce feature maps that are formed into high-dimensional matrices. These feature maps are then flattened into a one-dimensional vector that can be considered as a global feature vector of the input image. Next, this vector is then passed through one or more intermediate FC layers, which reduces the data size while retaining significant information. Each node in the FC layer is connected to every node in the previous layer, consequently, it allows the CNN to combine the learned features in various ways and capture relationships among them. As a result, the global feature often provides a more compact and informative representation of the input image.

To illustrate this approach, we selected the VGG-16 network, which consists of 13 convolutional layers followed by 3 fully connected layers. The architecture of the VGG-16 network is presented in Tab. 3.4. Once the input data reaches the fully connected layers (i.e., after layer #13), the FC layer takes the feature maps and flattens them into a vector of 25 088 dimensions. This global vector then continues to pass through two FC layers (i.e., layers #14, #15) to generate a compact vector of 4 096 dimensions. Since the VGG-16 is trained on the ImageNet dataset [19], the final fully connected layer has 1 000 output nodes corresponding to 1 000 classes. According to this architecture, the output of the FC layer #14 or #15 (not #16) can be extracted to serve as the input image features. The size of the image representation corresponds to 4 096 dimensions.

In addition to the feature extraction from FC layers, the convolutional ones can be considered a good candidate as introduced in the work [101], 2016. The authors of the study

Tab. 3.4 VGG-16 architecture.

#	Input	Output	Layer
1	224 × 224 × 3	224 × 224 × 3	Conv + ReLU
2	224 × 224 × 64	224 × 224 × 64	Conv + ReLU
	224 × 224 × 64	112 × 112 × 64	maxpool
3	112 × 112 × 64	112 × 112 × 128	Conv + ReLU
4	112 × 112 × 128	112 × 112 × 128	Conv + ReLU
	112 × 112 × 128	56 × 56 × 128	maxpool
5	56 × 56 × 128	56 × 56 × 256	Conv + ReLU
6	56 × 56 × 256	56 × 56 × 256	Conv + ReLU
7	56 × 56 × 256	56 × 56 × 256	Conv + ReLU
	56 × 56 × 256	28 × 28 × 256	maxpool
8	28 × 28 × 256	28 × 28 × 512	Conv + ReLU
9	28 × 28 × 512	28 × 28 × 512	Conv + ReLU
10	28 × 28 × 512	28 × 28 × 512	Conv + ReLU
	28 × 28 × 512	14 × 14 × 512	maxpool
11	14 × 14 × 512	14 × 14 × 512	Conv + ReLU
12	14 × 14 × 512	14 × 14 × 512	Conv + ReLU
13	14 × 14 × 512	14 × 14 × 512	Conv + ReLU
	14 × 14 × 512	7 × 7 × 512	maxpool
	7 × 7 × 512	1 × 1 × 25 088	flatten
14	1 × 1 × 25 088	1 × 1 × 4 096	FC + ReLU
15	1 × 1 × 4 096	1 × 1 × 4 096	FC + ReLU
16	1 × 1 × 4 096	1 × 1 × 1 000	FC + ReLU

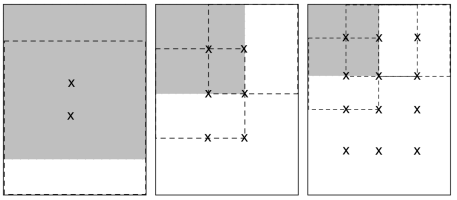


Fig. 3.1 Sample regions extracted at 3 different scales (i.e., 1, 2, 3) reproduced from [101]. While the top-left region of each scale is gray colored region, its neighboring regions towards each direction are dashed borders. The centers of all regions are depicted with a cross.

proposed two methods to apply the max-pooling operation to the activations of convolution layers in order to generate the image representation of the input image. We summarize here the main principles of these methods. Motivated by the work, these methods can be used for the different CNN methods or architectures, as presented in Tab. 3.2.

Regarding the max activation of convolution (MAC) method, the image features can be easily obtained from maximum activation values computed on the convolution feature maps. Particularly, the input image is first fed into a CNN to produce the convolutional feature maps. Given a convolutional layer, a global max-pooling operation is then applied to return a single maximum value from each feature map. At last, the final feature vector is composed of these maximum values from all feature maps. That is, the dimension of the MAC feature vector is equal to the number of feature maps in the chosen layer. For instance, in our demonstration using the VGG-16 network (see Tab. 3.4), the MAC features extracted from convolutional layer #13 form a vector of 512 dimensions. This dimension results from 512 global max-pooling operations performed across 512 feature maps, each of size 14×14. Noticeably, this method can also be applied to different pre-trained CNN models as well as different convolutional layers.

Regional maximum activation of convolutions (R-MAC) is the extension of MAC method that considers multiple regions of the feature map. Similar to the MAC, the input image is fed into a CNN to produce the convolutional feature maps. However, these feature maps here are divided into several overlapping regions at multiple scales, as illustrated in Fig. 3.1 (e.g., scale 1, 2, 3). For each region, a max-pooling operation is applied across the feature map. Next, these maximum values from all regions are aggregated by summing them to produce a single value for the feature map. At last, the final R-MAC feature vector is obtained after applying the L_2 normalization on the aggregated features. Thus, this approach also keeps the dimension of the image feature equal to the number of feature

maps in a chosen convolutional layer. For example, the VGG-16 network results in the R-MAC feature vector of 512 dimensions at the convolutional layer #13. Several variant approaches based on the R-MAC features have also proposed in the literature [43, 119]. The authors [43] introduced the L_3-iMAC feature, which is a variant R-MAC feature with some post-processing steps, to enhance the quality of the feature for retrieval. Another work [119] extracted video features directly from the feature maps with different regions and scales (i.e., 14 regions at 3 scales) to generate one feature graph of a given key-frame.

The features obtained from both of the above approaches generally produce high dimensional vectors (e.g., the FC features of VGG-16 have 4 096-D shown in Tab. 3.4). To convert them into low dimensional spaces, it can be done with pooling methods [104, 114], the principal component analysis [34, 104], hashing methods [36, 39], or bag-of-words [55].

3.2.4 Generation of video representations

After the key-frame features of a video are extracted, they are used to generate the video representations with the purposes of reducing data to speed up the response time or improving the robustness of these individual key-frame features. However, different works focus on different aspects of the video or the retrieval purposes. Existing PVCD studies can be classified into two categories, namely video-level, and frame-level as presented in Tab. 3.2. The frame-level approaches model the video in a spatio-temporal manner based on a sequence of feature vectors. In contrast to this, the video-level methods represent the whole video as a single feature vector. Fig. 3.2 illustrates a pipeline to process the video from the key-frame extraction to the generation of video representations. We summarize the representative studies on the generation of video representations for PVCD as follows.

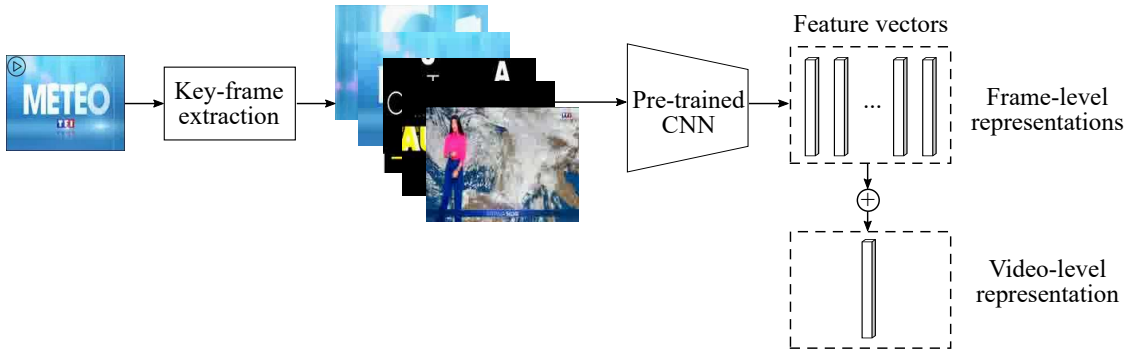


Fig. 3.2 An example illustrating the generation of video representations.

Frame-level methods represent a video with multiple feature vectors. The goal is to generate fine-grained video representations capturing the spatio-temporal information in the video. As a straightforward way, the video representations are obtained by stacking a sequence of feature vectors extracted from the key-frames of the video [38, 39]. Several methods have also been proposed to improve the quality of the video representations, which are relied mainly on the performance of pre-trained CNN models. Particularly, the authors [5] proposed the temporal layer parameterized in the Fourier domain for temporal aggregation to combine with the R-MAC features extracted from the pre-trained ResNet-

34. The authors [36, 88] used a Transformer-based neural network to aggregate the key-frame features in the temporal order. The multi-attention mechanisms (i.e., self-attention, cross-attention) used in the work [34] to enhance the quality of the key-frame features.

Video-level methods extract a single global representation for an input video. Due to a video containing a large number of the key-frame feature vectors with high dimensions, these methods focus on reducing the problem of high-dimensional spaces by using aggregation schemes. They are summarized in Tab. 3.2. Particularly, the authors [36] processed a set of key-frame features with an average pooling operation, followed by the L_2 normalization. Similarly, the authors [56, 88] computed video-level representations by averaging and normalizing (i.e., zero-mean and L_2 normalization) of all key-frame features. A graph-based representation for a video is proposed by the authors [119]. In [55, 122], the authors generated video-level representations integrating the bag-of-words histograms of all key-frames. Such approaches are less effective to detect partial copies.

3.2.5 Video comparison

PVCD systems compare videos based on their representations using standard similarity measures as illustrated in Tab. 3.2. The first approach is based on the computation of the pairwise similarity between the corresponding video representations [35, 65, 99, 114]. That is, the testing video representations are used to search in the global video representation database. To achieve fast search, the database normally is organized with effective data structures such as an indexing structure [55, 114, 122], the KD-tree [104], graph-based [74, 117]. Hence, such an approach aims at addressing the detection on a large scale.

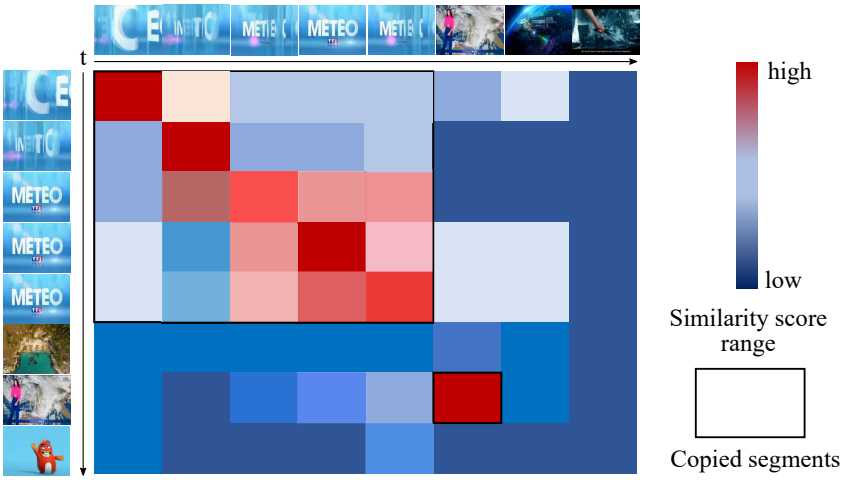


Fig. 3.3 An example of a frame-to-frame similarity matrix.

The matching can be made global with a frame-to-frame similarity matrix [30, 34, 35, 36, 43, 99, 117]. Fig. 3.3 shows a visual example of a frame-to-frame similarity matrix. This matrix is determined by computing the similarity of the representation sequences between the testing and the reference videos. Depending on the types of video representation, various similarity measures can be used such as Euclidean distance, dot product, or as summarized in Tab. 3.2. The approach to detect copied videos is to find the proportion

of similar cells in the matrix. Such an approach would be time-consuming considering the large number of frames needed to calculate similarity. Recent studies train video similarity networks using matching functions (e.g., contrastive loss, triplet loss) to estimate the video-to-video similarity [36, 38, 47, 54, 56]. They developed a neural network that captures fine-grained spatial and temporal structures within a similarity matrix. The goal is to learn a matching function that assigns higher similarities (smaller distances) to positive pairs compared to negative ones.

In all the cases, a temporal alignment of videos could be applied to locate copied segments [5, 34, 39, 47]. The first approach is to check the temporal consistency between matching frames using a temporal network, or the temporal Hough voting [5, 47]. Indeed, while the temporal network uses a network flow optimization strategy, the temporal Hough voting aligns matched frames based on accumulated votes over time and across matching frames. The authors [34, 39] adopted the YOLO network, which is an object detection neural network, to detect the boundaries of copied segments as temporal bounding boxes.

Comparison of 2D CNN features is a well-known topic in image classification and retrieval [24]. However, PVCD raises specific problems such as the highest level of scalability and noise due to the video data. To the best of our knowledge, comparisons of 2D CNN for PVCD have been addressed in [53, 55, 56, 99, 103]. However, the characterization has been done for global matching methods [55, 99] or video descriptors [56, 103] embedding the 2D CNN features. In addition, datasets with a small scale (e.g., SVD [45]) [53, 55, 56], unbalanced (e.g., VCDB [47]) [103] or private [99] have been used. The fine characterization of 2D CNN features for PVCD has never been investigated.

3.3 Protocol and performance evaluation

The PVCD systems process videos using 2D CNN features. These features serve for the retrieval and matching of videos. A key property of any PVCD system is the robustness of 2D CNN features. This is little investigated in the literature, and we will address this problem. Our contributions of this chapter are twofold **(i)**, **(ii)** as shown in Fig. 3.4.

First **(i)**, a protocol is developed to investigate the effect of the 2D CNN-based features. As shown in Tab. 3.2, a variety of 2D CNN models and methods have been proposed in the literature. These models and methods generate various types of 2D CNN features that can be used in the PCVD systems. Thus, the performance characterization of 2D CNN features for the PVCD systems is first discussed. Second **(ii)**, the sampling key-frame technique, which has been a dominant approach in the literature, could extract frames that are not suitable for 2D CNN due to the unique challenges of PVCD (e.g., scalability, near-duplicate detection, motion artifacts, solid-color frames). To investigate this, we propose a method for the characterization of key-frames. Our method is based on a goodness criterion and a time series model. By doing so, it provides a fine categorization of key-frames, and allows for a deeper characterization of the PVCD problem using 2D CNN features.

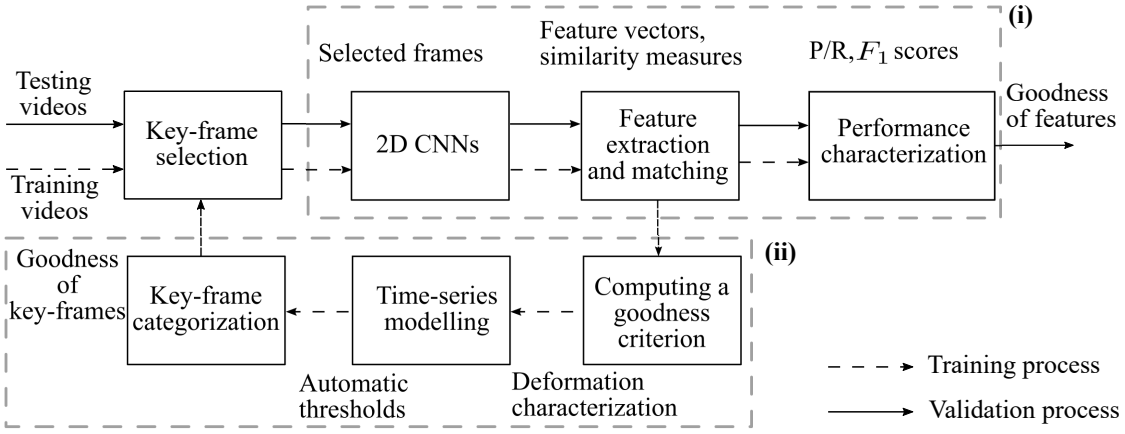


Fig. 3.4 Our protocol for performance evaluation and key-frame characterization.

3.3.1 Characterization of 2D CNN features

A PVCD dataset is divided into 2 sets: training and validation. The 2D CNN features in the validation set are used to match with the features in the training test to detect copies. The goal is to evaluate the goodness of 2D CNN features by investigating different models and methods. Fig. 3.4 **(i)** illustrates our protocol to characterize 2D CNN features.

For key-frame extraction, we have sampled with a static FPS for the negative videos and copied segments within the positive videos. Static FPS is a common method for key-frame selection as presented in Tab. 3.2.

In terms of 2D CNN we have selected VGG-16, ResNet50-v1, and Inception-v3 for characterization. These networks are typical for PVCD studies as shown in Tab. 3.2. The frames of videos were formed in matrices with sizes from 224×224 up to 299×299 (i.e., VGG-16, ResNet50-v1 require the input image size of 224×224, whereas 299×299 is needed for Inception-v3) encoded into the RGB color space. ResNet50-v1 and Inception-v3 are compact networks with $\simeq$ 30 M parameters whereas VGG-16 goes up to $\simeq$ 140 M as shown in Tab. 3.3. The three common methods the Last FC, MAC, and R-MAC have been used for feature extraction. These features were then applied with a normalization step to generate a unit L_2-norm. At last, they resulted in 9 databases of feature vectors (the vector dimensions range of 512-F, 2 048-F and 4 096-F).

For matching, we have compared the feature vectors with the cosine similarity $SC(X, Y)$ (with X and Y two vectors). It is a common measure for matching of CNN features as presented in Tab. 3.2, and is time-efficient and robust [24]. With a unit L_2-norm, it can be achieved with a single dot product $X \cdot Y$. Considering m and n the size of the training and validation sets, the brute-force comparison has a computational complexity $O(mn)$.

We have applied the characterization protocol similar to works [35, 65, 99] to evaluate the individual performance of 2D CNN features. All the extracted frames from the copied segments have been labelled with their corresponding references in the ground truth. The negative frames have no label. The performance evaluation has been computed with the P, R, and F_1 scores. That is, the maximum cosine similarity will matter, and at least one detected frame is required to detect the video.

3.3.2 Characterization of key-frames with 2D CNN features

The selection of 2D CNN features has a performance impact. Another aspect is the selection of key-frames, which can contribute interpretively to the use of 2D CNN features. Indeed, images illustrated in Fig. 3.5 with a high level of noise (a), near-constant (b), or almost duplicate (c) could be difficult to detect. This is a typical problem in image classification and retrieval [24]. We will investigate this aspect here for PVCD by providing a characterization of key-frames with 2D CNN features. The goal is to evaluate the performance accuracy of 2D CNN features when faced with a great deal of variability in the key-frames as well as such types of noises for PVCD.

(a) (b) (c) (d) (e)

Fig. 3.5 Examples of key-frames
(a) blurred NC (b) near constant W (c) almost-duplicate W
(d) foreground/background FS (e) symmetrical layout FS.

To achieve this goal, the videos in the training set are selected for processing and analysis with the 2D CNN features. By doing so, it supports the selection of appropriate key-frames suitable for 2D CNN models, thus facilitating the detection of copied videos during the validation process. In other words, when new unseen videos reach the validation step, key-frames will be extracted from specified positions to reduce matching errors. The positions are determined based on the goodness of the key-frames. Fig. 3.4 **(ii)** presents our protocol to characterize the key-frames. It involves a goodness criterion and a time series model. We will describe it in detail in the following paragraphs.

For the needs of characterization, we propose the goodness criterion of Eq. (3.1). The $SC(X,Y) \in [0,1]^2$ is defined for two any X, Y feature vectors obtained from a pre-trained CNN using a ReLU activation function, followed by L_2 normalization. This criterion maximizes the intra and interclass similarity. X is the 2D CNN feature of a positive frame, and $\{\tilde{X}_1,\ldots,\tilde{X}_m\}$ its corresponding near-duplicate. $\{Y_1,\ldots,Y_{n_1}\}$ is the set of negative 2D CNN features, and $\{X_1^*,\ldots,X_{n_2}^*\}$ the positive ones obtained from the other references. $SC_{\min}$ and $SC_{\max}$ are operators to get the minimum and maximum SC between the template X and feature sets. That is, $\phi(X) \in [-1,1]$ and $\phi(X) > 0$ guarantee a separability (i.e., no possibility for X to be classified as a false negative (X matched with a negative frame, or assigned to another video reference).

[2]Due to ReLU produces a non-negative vector, and it is scaled to unit magnitude by L_2 normalization.

$$\phi(X) = SC_{\min}(X, \{\tilde{X}_1, \ldots, \tilde{X}_m\}) - SC_{\max}(X, \{Y_1, \ldots, Y_{n_1}\}, \{X_1^*, \ldots, X_{n_2}^*\}) \quad (3.1)$$

Any frame X with its near-duplicates $\{\tilde{X}_1, \ldots, \tilde{X}_m\}$ are aligned with a timestamp t having a frame precision level. The overall set of frames can be modeled with time series as illustrated in Fig. 3.6. In these series, the $z_1, \ldots, z_{m+1}$ values at t give the $\phi(X)$, $\phi(\tilde{X}_1)$, $\ldots$, $\phi(\tilde{X}_m)$ criteria. These values can be characterized with baseline statistics ($z_{\min}$, $\overline{z}$, $z_{\max}$, σ), and a rate τ accounting the amount of positive criteria. The frames can be easily categorized as detailed in Tab. 3.5, and illustrated in Fig. 3.6. This categorization is based on the statistics with the rate compared to thresholds α, β. The large variability in scores can be detected as σ outliers compared to α. This constitutes the set of not-consistent frames. The no separable frames are obtained when $z_{\max} < 0$ and then $\tau = 0$. Some worst frames can be filtered out such as $z_{\max} < \beta$. The partially and fully separable frames are obtained for $\tau \in]0, 1[$ and $\tau = 1$, respectively.

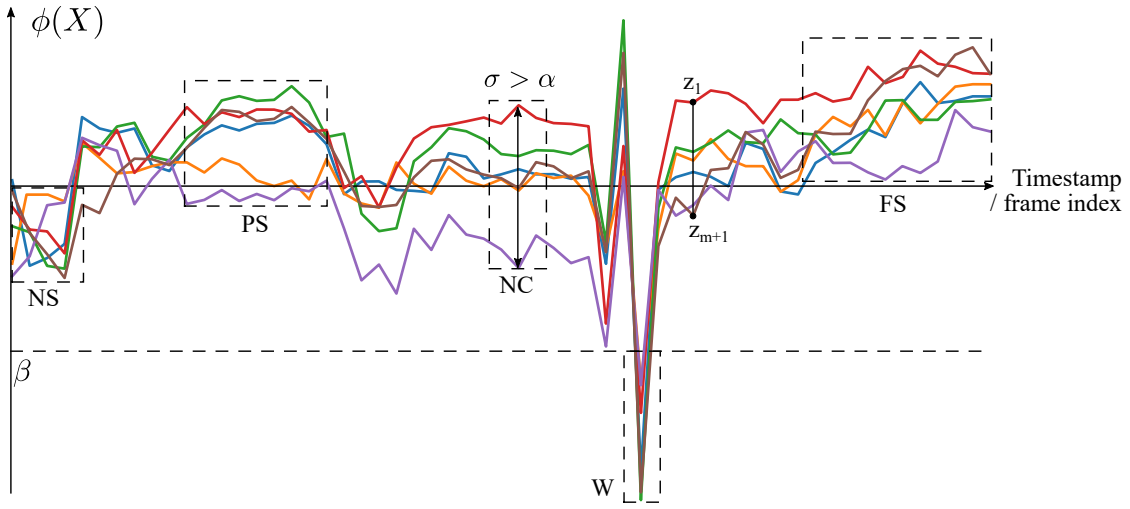


Fig. 3.6 Key-frame modelling with a time series.

Tab. 3.5 Categorization of frames.

Category	σ	$z_{\min}$	$z_{\max}$	τ
Not Consistent (NC)	$> \alpha$	$\in [-1, 1]$		$\in [0, 1]$
Worst (W)	$\leq \alpha$	$\in [-1, \beta[$		$= 0$
Not Separable (NS)		$\in [\beta, 0[$		
Partially Separable (PS)		< 0	≥ 0	$\in]0, 1[$
Fully Separable (FS)		≥ 0		$= 1$

3.4 Experiments and results

3.4.1 Video dataset

Before driving a performance characterization, a dataset must be selected. Several main PVCD datasets have been proposed in the literature. Tab. 3.6 gives a comparison.

We have selected the STVD dataset [65]. This dataset has several key properties: it is captured from TV and is almost noise-free allowing a fine control of degradations with synthetic methods, it is the largest dataset of the literature with ten thousand hours of video, 243 references and 1 688 thousand positive pairs (i.e., a positive pair (v_i, v_j) is a combination of two partial video copies v_i and v_j. The term *copied segment pairs* could be used instead [47, 35]), it offers a balance distribution between the negative and positive videos, and it is delivered with an accurate groundtruth for video alignment.

Tab. 3.6 Datasets for PVCD performance evaluation.
The h, s and n/a stand for in hours, in seconds and not available.

Datasets	VCDB	SVD	STVD	VCSL
Paper	[46]	[45]	[65]	[35]
Year	2016	2019	2021	2022
Source of capture	Web	Web	TV	Web
Degradation	real	synthetic	synthetic	real
Duration (h)	2 030 h	2 705 h	10 660 h	17 416 h
References	28	1 206	243	122
Positive videos	528	n/a	19 280	9 207
Positive pairs	9 K	10 K	1 688 K	281 K
Negative videos	100 000	526 787	64 040	n/a
Timestamps (s)	1 s	n/a	$\frac{1}{30}$ s	1 s

The STVD dataset is constituted of six test sets A to F having different sources of degradation (e.g., pixel attack, video speeding). We have selected the test set D related to scalability and global transformations as illustrated in Fig 3.7. It includes 3 213 and 12 165 positive and negative videos having a total and a mean duration of 1 960 hours and 7.5 minutes, respectively. The video resolution is fixed at 144×192 pixels (i.e., height$\times$width), and compressed at 28 kbps with the FPS of 30. Global transformations have been applied and combined including flipping, rotation, and inclusion of black borders. This test set fits well with the 2D CNN features that are translation, scale, and rotation invariant.

Fig. 3.7 Global transformations in the test set D of the STVD dataset.

As our objective is performance characterization, we have randomly split the videos into two sets for training and validation with rates of 60% and 40%, respectively. They are detailed in Tab. 3.7. We have used the 12 165 negative videos without any modification. We have re-generated and added 656 videos to the 3 213 positive videos. Indeed, the 243 references have occurrences from 1 up to 167. To fit with the splitting process, a minimum of 10 occurrences per reference is needed. We have then used the timestamps information to extract the copied segments. As detailed in [65], these segments have a duration $\in [1; 25]$ seconds, and one or several copied segments could appear in a positive video. We have

obtained 4 436 copied segments having a total duration of 7.5 hours.

Tab. 3.7 Dataset for performance characterization.

Videos	Number	Duration (h)	FPS	60% training	40% validation	Total
Negative videos	12 165	1 545 h	$\simeq 0.08$	259 050 f	172 700 f	431 750 f
Positive videos	3 869	415 h				
Copied segments	4 436	7.5 h	1	16 200 f	10 800 f	27 000 f
						458 750 **f** $\simeq$ 7 GiB of 4 096-F features

3.4.2 Comparison of 2D CNN features

A FPS = 1 has been applied to the copied segments. We have obtained a total of 27 000 feature vectors which are split into the training and validation sets. For the negative videos, we have fixed a FPS $\simeq 0.08$ for a trade-off between the scalability and memory constraint. All the experiments have been carried out on a *Nvidia* GPU RTX 2070 having a 8 GiB of memory. We have allocated 7 GiB to the features (1 GiB was used to store the program, frameworks, and for the results.) for a total amount of 458 750 vectors considering the maximum size of 4 096-F. We have extracted 431 750 feature vectors from the negative videos dispatched into the training and validation sets as shown in Tab. 3.7.

Based on the dataset and our protocol, we compare here the accuracy of 2D CNN features. It requires $\simeq$ 50.5 billion matchings for each feature database, consequently resulting in a total of $\simeq$ 455 billions. To facilitate the matching, this can be achieved in 7 to 8 hours of with a time efficient implementation in which the dataset is fully loaded in the memory of the RTX 2070 GPU, and the matching is computed with a fast vector multiplication using all the 2 560 cores of the GPU.

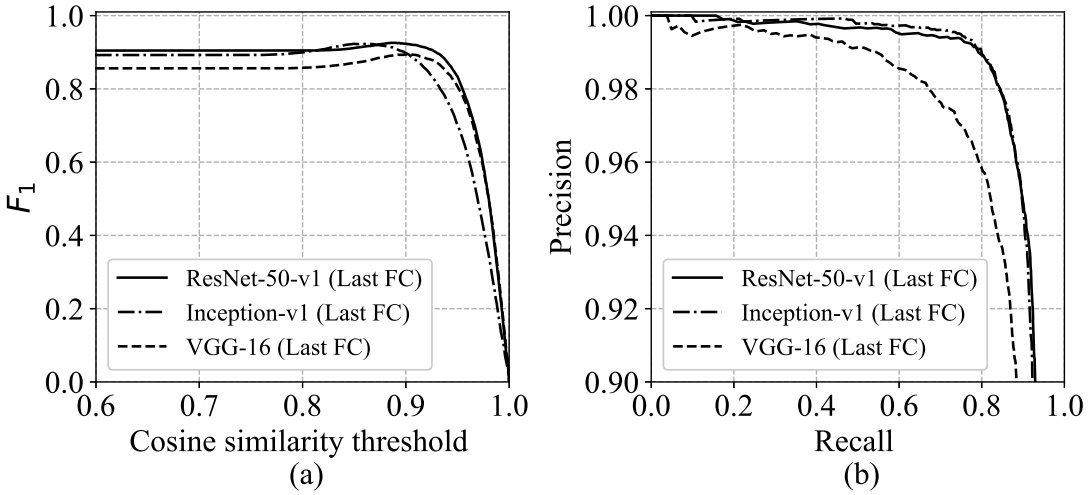


Fig. 3.8 Comparison of 2D CNN with the Last FC (a) F1, (b) P/R

Fig. 3.8 (a) gives the F_1 scores over a threshold based on the cosine similarity. The most common feature extraction method (Last FC) of different 2D CNN models is selected

for comparisons. From these results, we have found that the separability for the detection is not achieved even if strong scores are obtained. A maximum of $F_1 \simeq 0.93$ is performed with the ResNet50-v1 network. The different networks present competitive results with a maximum gap of $F_1 \simeq 0.03$. These results are consistent with previous comparisons of 2D CNN in the state-of-the-art [33]. For further analysis, Fig. 3.8 (b) provides the P/R plot. All the 2D CNNs maintain a strong precision at a high level of recall.

For a comparison of the feature extraction methods, Tab. 3.8 gives the top F_1 scores of 9 feature databases resulting from the different 2D CNN models with the Last FC, MAC, and R-MAC. For VGG-16, MAC and R-MAC outperform the Last FC method with a slight gap of $F_1 \simeq 0.03$. These methods provide a performance degradation for ResNet50-v1 and Inception-v1 up to a gap of $F_1 \simeq 0.18$. This can be mainly explained by the larger sizes of convolution layers in the VGG-16 network compared to ResNet50-v1 and Inception-v1. This leads to more accurate localizations with the MAC and R-MAC features. An equivalent conclusion is also reported in [16].

Tab. 3.8 Comparison of feature extraction methods with the top F_1 scores

	Last FC	MAC	R-MAC
ResNet50-v1	**0.926**	0.828	0.823
Inception-v1	**0.923**	0.738	0.782
VGG-16	0.894	**0.922**	0.918

Overall, the selection of the 2D CNN features has a performance impact. Our experiments report a maximum score gap of $F_1 \simeq 0.2$, where the use of the ResNet50-v1 with the Last FC features compared to the use of the Inception-v1 with the MAC features.

3.4.3 Comparison of the key-frame categories

For the experiments, we have extended the number of positive frames from 16 200 to 486 000 by sampling at the full FPS = 30. We have used the VGG-16 with the MAC feature extraction method for a trade-off between a strong detection score $F_1 \simeq 0.92$ Tab. 3.8, and the memory constraint (i.e., MAC features of the VGG-16 result in vectors of 512-F). With m and n the numbers of positive and negative frames, the Eq. (3.1) has a computational complexity $O(m\left(\frac{m+1}{2}\right) + mn)$, where $SC(X, X^*) = SC(X^*, X)$, and the comparison number of m features is $m\left(\frac{m+1}{2}\right)$. This requires $\simeq 244$ billion matchings.

Tab. 3.9 reports the results of categorization on the training set Tab. 3.6. From these results, we have applied $\alpha = 0.05$ and $\beta \in [-0.4, 0]$ thresholds obtained with automatic methods which will be detailed below.

Tab. 3.9 Categorization results of the training set at full FPS= 30.

Total indices	NC	W	NS	PS	FS
50 844	6 966	4 169	33 049	4 881	1 780
100 %	13.7 %	8.2 %	65 %	9.6 %	3.5 %
	21.9 %			**78.1 %**	

A total of 50 844 timestamps/indices have been obtained from the STVD dataset as presented in Tab. 3.6. About $\simeq 22\%$ of frames have been categorized as not consistent

NC and worst W. Within the remaining $\simeq 78\%$, only $\simeq 13\%$ fit with the partial PS or full separability FS. That is, only a very small number of "good" key-frames appear in the several videos corresponding to the categories PS and FS. Nearly 87% of key-frames are hard to detect from their 2D CNN features due to they are not consistent or little discriminative.

The categorization is derived from applying $\alpha = 0.05$ and $\beta \in [-0.4, 0]$ thresholds. We have fixed them with automatic methods illustrated in Fig. 3.9. Fig. 3.9 (a) plots the cumulative distribution of σ over the 50 844 indices. The threshold $\alpha \simeq 0.05$ can be easily obtained with an automatic elbow detection. For clarification, the cumulative rate of indices with $\tau = 0$ (over all the indices $\tau \in [0, 1]$) is given for $\sigma > \alpha$. $\ll 1\%$ of indices have a $\tau \neq 0$. The threshold β has been fixed to detect outliers for indices with $\tau = 0$ and $\sigma \leq \alpha$ reference per reference. Fig. 3.9 illustrates the method. For each reference, a mean $\overline{\mathcal{Z}}$ of indices is computed. This mean serves to fix the threshold $\beta = \overline{\mathcal{Z}}$. The indices with $z_{\max} < \overline{\mathcal{Z}}$ are categorized as worst frames W. Considering the 243 references Tab. 3.6, we have obtained a range $\beta \in [-0.4, 0]$.

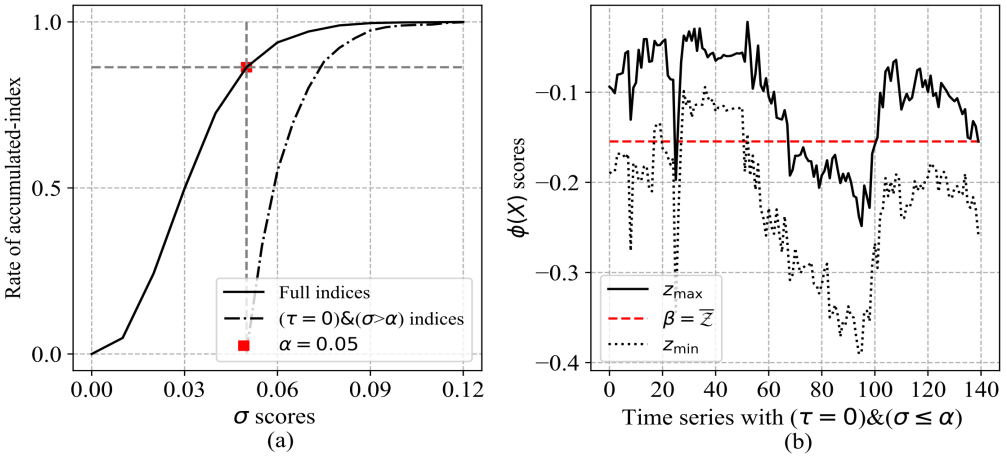


Fig. 3.9 (a) distribution of σ (for α), (b) times series with $\tau = 0$ and $\sigma \leq \alpha$ (for β).

Fig. 3.5 provides several visual examples of key-frames for the different categories. Fig. 3.5 (d, e) gives key-frames labeled FS containing distinguished shapes (e.g., background/foreground text). They are easy to detect with 2D CNN features [116]. However, they are difficult to catch from videos as they constitute only $\simeq 3\%$ of the total amount of key-frames as shown in the last column of Tab. 3.9. Fig. 3.5 (b, c) gives key-frames having a label W with a near-constant (e.g., low amount of visual content, solid-color), or an altered visual content (e.g. inclusion of logos). Even if they constitute a small part of key-frame $\simeq 8\%$, which is shown in the third column of Tab. 3.9, they must be carefully avoided for PVCD. Fig. 3.5 (a) shows a key-frame with a high level of blurring labeled NC. Such key-frames have 2D CNN features with large variability and little discriminant in their visual content. They are hard to detect. At last, $\simeq 65\%$ of key-frames are categorized as NS. The 2D CNN features of these key-frames cannot be detected effectively.

Based on the categorization as shown in Tab. 3.9, and some examples illustrated in Fig.

3.5, while some key-frame positions (e.g., frame indices classified as FS, PS) remain the performance regardless of degradation methods, the other positions should be avoided for key-frame selection (e.g., frame indices classified as NC, W). To demonstrate the goodness of these positions, we have evaluated the accuracy of 2D CNN features for the classification of key-frames. In other words, for each video in the validation set, the key-frames have been selected based on their corresponding positions (i.e., with the same frame index) in the training set. This requires refining the groundtruth from the video-level to the frame-level. Given a key-frame category, for each reference video Ref of $n+1$ frame indices, one position i-th is selected in the training set (i.e., Ref_i with $i \in [0, n]$ is the i-th position/index of the reference video Ref). Similarly, this position i-th has been used to extract the corresponding frame i-th of the video in the validation set within the same reference. To detect copies, the key-frames in the validation set are used to match with the key-frames in the training set. We have selected three key-frame categories to generate the three datasets as detailed in Tab. 3.10.

Tab. 3.10 Classification of key-frames with categorization.

Category	References	Positive frames		Negative frames		E_{RR}
		Training	Validation	Training	Validation	
FS	27	168 f	112 f			0.16 ‰
NC	142	1 746 f	1 164 f	259 050 f	172 700 f	1.31 ‰
W	170	1 602 f	1 068 f			1.90 ‰
	339	5 860 **f**		431 750 **f**		

From the positive frames, we have obtained $\simeq 340$ frame references and $\simeq 5\,900$ feature vectors (i.e., $\simeq 1\%$ of the overall positive frames). The distribution of negative frames is similar to our previous experiments Tab. 3.7 for a fair comparison. Considering the unbalanced distributions between the negative/positive frames and references, the error rate E_{RR} has been used for characterization.

These error rates shown in Tab. 3.10 confirm our previous findings and provide additional evidence that suggests the key-frames should be extracted carefully to further improve the performance of PVCD systems. Specifically, the FS category has the lowest error rate E_{RR} at 0.16‰ indicating a high level of accuracy in the classification of these key-frames, whereas the W category has the highest error rate E_{RR} at 1.90‰ highlighting the greater challenge in accurately classifying key-frames. Overall, nearly an order of magnitude appears between the error rates of the categories FS and NC/W, respectively.

3.5 Conclusions and perspectives

This chapter gives a performance characterization of 9 common 2D CNN features used for PVCD. Based on a large-scale video dataset, the experiments have been driven on 4.4 M feature vectors with 700 B comparisons. The separability is not achieved on the detection problem even if strong scores are obtained with a maximum of $F_1 \simeq 0.93$. The different networks present competitive results with a maximum gap of $F_1 \simeq 0.03$. As a general trend, features extracted from recent 2D CNN such as ResNet50 perform better. A correlation appears between the feature extraction methods and the 2D CNN architectures (e.g., VGG-16 with the MAC and R-MAC features). These different conclusions are consistent with

the state-of-the-art in the field of computer vision.

From 2D CNN features modelled as time series, a method for categorization of key-frames is proposed. This method allows a deeper characterization of a PVCD problem with 2D CNN features. It provides (**i**) a fine categorization of key-frames (**ii**) a characterization of 2D CNN features for separability and consistency (**iii**) a quantitative analysis of the goodness of key-frames. It highlights the performance limits of 2D CNN features when facing blurred, near-constant, or almost-equivalent key-frames. In addition, a large part of key-frames ($\simeq 87\%$) cannot be classified efficiently from 2D CNN features. These limitations will be explored in our future works by investigating the robust key-frame selection and learning of 2D CNN features to further improve the PVCD performance.

Conclusions and perspectives

In this chapter, we first summarize the main contributions of this thesis for the performance evaluation of scalable partial video copy detection. Particularly, it involves a stable protocol to design a large-scale PVCD dataset, and methods as well as protocols to evaluate the performance of the PVCD systems using 2D CNN features. The favorable characteristics and shortcomings of these two contributions are carefully discussed. Possible lines of future research for each of these contributions are then given.

Beginning with the chapter 1, we have provided reviews on the state-of-the-art studies for the performance evaluation of the PVCD problem. The key components of a PVCD system are discussed. The existing benchmark datasets used for PVCD are investigated in detail. Based on these benchmarks, the performance evaluation of the PVCD is then studied. The advantages and limitations of each dataset are highlighted. By taking these reviews together, it motivates us to drive our contribution to the performance evaluation aspect of the PVCD problem.

Based on the discussion of the existing benchmark datasets, our first contribution introduces a design protocol, dataset, test set challenges, and baseline comparisons for the performance evaluation of PVCD systems. They are presented in the chapter 2. The protocol comes with a stable design for video capturing of multiple TV channels. It allows us to generate the largest PVCD dataset compared to the existing ones, to the best of our knowledge. Our dataset provides diverse test sets that can support the characterization of different challenges for the PVCD problem. The performance of representative methods has been evaluated through experiments in which our dataset reported baseline comparisons to show room for improvement. For short, our protocol and dataset have the following key features.

- Stable protocol: Web and TV videos are two major sources of data for many computer vision tasks. Gathering a large number of videos from the Internet could require a great deal of human intervention, and annotating them is a time-consuming and costly task due to realistic noise. By using a dedicated TV workstation, which can support a stable multi-channel capture in real-time, our protocol is designed to produce videos with a little human intervention and a semi-automatic annotation process.

- Reliable protocol: Our protocol can provide noise-free captured videos in order to have the fine control of degradation levels. The **ZNCC** metric is applied to demonstrate the robustness of our protocol configuration under such noisy conditions. From captured videos, a window size **W** is proposed to address a common latency issue

and avoid a full-search strategy while video processing for the problem of PVCD. Standard methods for generating experimental video data are also provided in detail to ensure consistency and reproducibility.

- Adaptable protocol: Our protocol can be simply adapted to design scalable datasets for several problems such as the parallel machine scheduling[3]or the fact-checking [84].
- Large-scale dataset: Using our protocol, we have generated a PVCD dataset, namely STVD. To the best of our knowledge, STVD is currently the largest public dataset on the task. It covers a near 83 thousand videos (having a total duration of 10 660 hours and containing 1 688 thousand pairs of partial video copies). More interestingly, our dataset not only provides an accurate frame-level groundtruth but also requires less annotation cost compared to the existing datasets.
- Usefulness dataset: Our dataset is organized into different test sets that support addressing various challenges at a high scale (e.g., pixel attacks, geometric transformations, timing degradation) for the PVCD problem. In addition, we provide a root test set where video augmentation methods can be easily applied to fine-tune the data to meet new requirements without collecting new data. Data augmentation plays an important role in improving the robustness and generalization of models for machine learning algorithms.
- Thoughtful design: STVD is made publicly available on the sub-domain of University of Tours[3]. A terms of service agreement[4] is provided to establish clean legal terms and guidelines (in contrast with some datasets that only provide access links). Hence, we will endeavor that STVD dataset remains available for the coming years to promote research in PVCD domain.

In addition to these positive features, we are also aware of several shortcomings in our protocol and dataset. First, as this protocol considers the TV jingles as candidates for video segment copies, their duration could be very short compared to the entire video (e.g., for hard cases, a 5-minute video has a 1-second copied segment). For that reason, the performance could be degraded when copied frames are missed during key-frame selection. Second, video transformation is one of the challenges for PVCD, it would be worthwhile to measure the performance between the degradation levels in our dataset with the existing ones such as VCDB, VCSL, and SVD. Therefore, to promote the performance evaluation of our dataset as well as the PVCD problem, several potential directions of research could be considered as follows.

- New challenging test sets: Despite our dataset consisting of 6 test sets, new challenging ones could be developed for different PVCD applications. For example, TV advertisements, breaking news with a high broadcast frequency could be good candidates for the PVCD, apart from TV jingles. Additionally, a higher video resolution (e.g., HD, Full HD) is needed to detect copies with a high near-duplicate level. Capturing data that combines video and audio can assist in better understanding PVCD. Such data can be used for the vision-language models such as BERT or GPT.

[3]https://dataset-stvd.univ-tours.fr/

[4]Terms of Service: https://dataset-stvd.univ-tours.fr/cgu_en.pdf

- PVCD promotes: The PVCD plays an important role in many video-related applications. Thus, our dataset can be considered as a reference for evaluating more advanced algorithms and models. To this end, several potential solutions can be driven to promote the benefits of PVCD. For example, a PVCD contest taking part in such key computer vision conferences[5] could be organized. An alternative is to exchange and announce our dataset by using different the Internet-based tools for the research community in the computer vision field (e.g., mailing lists, online platforms).

Regarding the second contribution of this thesis, we evaluate the performance of the PVCD systems using 2D CNN features that have emerged over the past decade. They are discussed in the chapter 3. State-of-the-art studies are first reviewed with a focus on the PVCD systems using deep learning algorithms. The main components, common methods, and techniques for key-frame extraction, selection of 2D CNN features, generation of video representations, and video comparison are then carefully studied. Based on the state-of-the-art studies, a new contribution has been made for characterizing the performance of the PVCD systems using 2D CNN features. It comes with our methods and protocols. The proposed approach has been evaluated through large-scale experiments involving 700 billion comparisons of 4.4 million feature vectors. Two main insights derived from our contributions are given below for a better understanding of the use of 2D CNN features in the PVCD systems.

- While various strong 2D CNN models and methods have been used for the feature extraction of images, the PVCD introduces unique challenges (i.e., scale, temporal degradation, near-duplicate detection, motion artifacts, solid-color frames). Indeed, through large-scale experiments, we have reported performance characterization of 9 common 2D CNN features used for PVCD. We find that the separability is not achieved on the detection problem even if strong scores are obtained with a maximum of $F_1 \simeq 0.93$. The different models present competitive results with a maximum gap of $F_1 \simeq 0.03$. As a general trend, features extracted from recent 2D CNN such as ResNet50 perform better. A correlation appears between the feature extraction methods and models (e.g., VGG-16 with the MAC and R-MAC features).

- The selection of key-frames should be carefully considered due to the lack of robustness in 2D CNN features for video processing. Based on a public PVCD dataset with an accurate groundtruth, we have processed videos with a time series model to characterize key-frames. Based on the experiment results, five key-frame categories are quantitatively determined through the analysis of their goodness. Among these categories, the majority of key-frames ($\simeq 87\%$) cannot be classified efficiently from 2D CNN features. Especially, the most challenging cases are videos with blurred frames, or near-constant, almost-equivalent visual content. Indeed, they introduce more errors during the matching step compared to 'good' key-frames.

Although the obtained results are interesting, we realize that there are many different research directions for this work. Several plans for future work are given below.

[5]ICIAP, ECCV, CVPR competitions

- Efficient computation features. The study of 2D CNN is increasingly attracting interest from the research community. It would be interesting to evaluate the computation cost as well as the resource constraint of the 2D CNN features so that the PVCD system is able to handle a large amount of video data.
- Robust spatio-temporal video representations. As a common trend, video-level representations make little sense for the PVCD problem. Hence, the frame-level representations, especially the spatio-temporal ones, would be considered to study robust video representations for the PVCD problem.
- Key-frame selection. It is agreed that the performance of the PVCD is also dependent on the selection of key-frames. Therefore, studying the selection of robust key-frames would further improve the PVCD performance.

Finally, we expected that the two contributions in this thesis will draw much interest and attention from the research community.

List of publications and datasets

International conferences with peer review and proceedings

- V.H. Le, M. Delalandre and H. Cardot, *"Performance Characterization of 2D CNN Features for Partial Video Copy Detection"*, Conference on Computer Analysis of Images and Patterns (CAIP), pp. 205-215, Limassol, Cyprus, 2023, (https://hal.science/hal-04231596v1/document).
- F. Rayar, M. Delalandre and V.H. Le, *"A large-scale TV video and metadata database for French political content analysis and fact-checking"*, Conference on Content-Based Multimedia Indexing (CBMI), pp. 181-185, Graz, Austria, 2022, (https://hal.science/hal-03747122v1/document).
- V.H. Le, M. Delalandre and D. Conte, *"A large-Scale TV Dataset for partial video copy detection"*, International Conference on Image Analysis and Processing (ICIAP), Lecture Notes in Computer Science (LNCS), vol 13233, pp. 388-399, Lecce, Italy, 2022, (https://hal.science/hal-03638514v1/document).
- V.H. Le, M. Delalandre and D. Conte, *"Real-time detection of partial video copy on TV workstation"*, Conference on Content-Based Multimedia Indexing (CBMI), pp. 1-4, Lille, France, 2021, (https://hal.science/hal-03552671v1/document).

National conference with peer review and proceedings

- V.H. Le, M. Delalandre and D. Conte, Une large base de données pour la détection de segments de vidéos TV, Journées Francophones des Jeunes Chercheurs en Vision par Ordinateur (ORASIS), Saint-Ferréol, France, 2021, (https://hal.science/hal-03339724/document).

Our datasets

- Partial Video Copy Detection: https://dataset-stvd.univ-tours.fr/pvcd
- Fact-checking: https://dataset-stvd.univ-tours.fr/fc
- Parallel Machine Scheduling: https://dataset-stvd.univ-tours.fr/pms

Appendix

Appendix A

This appendix discusses different aspects related to the STVD dataset presented in Chapter 2. The architecture of TV workstation is first introduced in Section A.1. Then, Section A.2 discusses the problem of real-time video capture at low resolution while using the TV workstation. Next, Section A.3 demonstrates time-efficient video matching using ZNCC with GPU support. At last, our protocol for video degradation with compression is discussed in Section A.4. For convenience, Tab. A.1 gives the main symbols and terms used in the appendix.

Tab. A.1 Main symbols and terms used in the appendix.

Symbols	Meaning
$\mathbf{I}$, $\mathbf{I}^*$	the reference and test images
$\mathbf{M}$, $\mathbf{N}$	size of an image having the width $\mathbf{M}$ and height $\mathbf{N}$
$\mathbf{I}(\mathbf{x})$, $\mathbf{I}^*(\mathbf{x})$	discrete 1D functions taking values at $\mathbf{x}$ in an image
$\sigma_{\mathbf{I}}$, $\sigma_{\mathbf{I}^*}$	standard deviations for $\mathbf{I}$, $\mathbf{I}^*$
$\overline{\mathbf{I}}$, $\overline{\mathbf{I}}^*$	means of $\mathbf{I}$, $\mathbf{I}^*$
$\hat{\mathbf{I}}(\mathbf{x})$, $\hat{\mathbf{I}}^*(\mathbf{x})$	zero-mean images of $\mathbf{I}$, $\mathbf{I}^*$
$\mathbf{m}, \mathbf{n}$	sizes with $\mathbf{m} \ll \mathbf{n}$
$\mathbf{k}, \mathbf{l}$	indexes with $\mathbf{k} \in [\mathbf{0}, \mathbf{m}]$, $\mathbf{l} \in [\mathbf{0}, \mathbf{n}]$
$\mathbf{X} = \{\mathbf{I_0}, \ldots, \mathbf{I_m}\}$	a reference video composed of frames/images
$\mathbf{Y} = \{\mathbf{I_0^*}, \ldots, \mathbf{I_n^*}\}$	a testing video composed of frames/images
$\tilde{\mathbf{X}} \simeq \mathbf{X}$	a near duplicate video with $\tilde{\mathbf{X}} \in \mathbf{Y}$
$\mathbf{S}$	a scoring matrix of size $(\mathbf{m}+\mathbf{1}, \mathbf{n}+\mathbf{1})$
$\mathbf{s_{kl}}$	element of $\mathbf{S}$ with $\mathbf{s_{kl}} = \mathbf{ZNCC}(\mathbf{I_k}, \mathbf{I_l^*})$
$\lambda_\mathbf{0}, \ldots, \lambda_\mathbf{m}$	weighting parameters with $\sum_{\mathbf{k=0}}^{\mathbf{m}} \lambda_\mathbf{k} = \mathbf{1}$
$\mathbf{w_l}$	diagonal weighted averaging $\mathbf{w_l} = \sum_{\forall \mathbf{k}} \mathbf{s_{k,l+k}} \cdot \lambda_\mathbf{k}$
$\mathbf{W} = [\mathbf{w_0}, \ldots, \mathbf{w_{n-m}}]$	vector of weighted averaging
$\mathbf{S'}$	a scoring matrix of size $(\mathbf{m}+\mathbf{1}, \mathbf{m}+\mathbf{1})$
$\bar{\mathbf{s}}_\mathbf{kl}$	$\in [\mathbf{0}, \mathbf{1}]$ is the normalized value of $\mathbf{s_{kl}}$
$\hat{\bar{\mathbf{s}}}_\mathbf{k}$	$= \min_{\forall \mathbf{l}} \bar{\mathbf{s}}_\mathbf{kl}$ is a min value in the column $\mathbf{k}$ in $\mathbf{S'}$
$\sigma_{\mathbf{I},\mathbf{I}^*}$	the covariance of pixel values between $\mathbf{I}$ and $\mathbf{I}^*$
c_1, c_2	constants to stabilize the division with weak denominator

A.1 The TV workstation

We introduce in this section the TV workstation used for video capture to design the STVD dataset. The TV workstation is a project developed by the LIFAT Laboratory[1].

[1]LIFAT laboratory, Tours city, France: https://lifat.univ-tours.fr/

It is a parallel architecture able to capture and process TV video content in real-time. The workstation aims to promote research activities at the LIFAT Laboratory on the topics of real-time video processing, natural language processing, machine learning and artificial intelligence. Fig. A.1 presents the workstation and Tab.A.2 details the hardware specification. We will discuss in this appendix different aspects related to real-time and multi-channel video capture and processing.

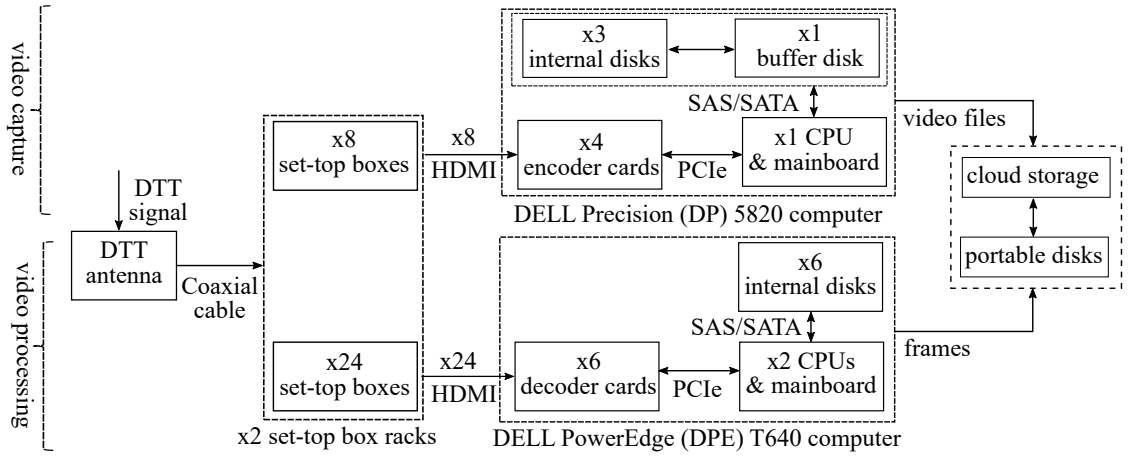


Fig. A.1 The architecture of the TV workstation.

Tab. A.2 Hardware specification of the TV workstation.

Type	Specification	Qty	Description
Computer	DELL Precision 5820	1	Intel Xeon W-2295, 64 GB RAM
	DELL PowerEdge T640	1	$\times$2 Intel Xeon Gold 5218R CPU 2.10, 32 GB RAM
Encoding card	Avermedia CL332-HN	4	2-Channel Full HD HW H.264
Decoding card	Avermedia CE314-HN	6	4-Channel Full HD
Set-top box	Astrell DVB-T 011128	32	Adapter DVB-T HDMI
Data storage	Buffer disk (2 TB)	1	Master Boot Record (MBR) partition format
	Internal disks (108 TB)	9	$\times$12 TB GUID Partition Table (GPT) format
	Portable disks (105 TB)	5	$\in [12, 20]$ TB connected by USB cables
	Cloud (5 TB)	1	connected to the Internet

The TV streams are delivered over different networks including the Internet Protocol TeleVision (IPTV), Satellite Television (SaT) and Digital Terrestrial Television (DTT) signals. Although the IPTV signal can offer new multimedia services, they suffer from a big latency compared to the DTT and SaT signals [25]. In addition, IPTV also requires a huge bandwidth for broadcasting. These are crucial points to design a system that has a capacity to handle multi-channel in real-time. To solve these problems, the capture within the workstation is driven by the DTT signal.

The DTT is supported with different standards (e.g., ATSC, ISDB-T, DTMB) at the worldwide level[2]. As the workstation is hosted in France, the DVB-T[3] standard matters.

[2]see "Digital terrestrial television" https://en.wikipedia.org/.

[3]Digital Video Broadcasting - Terrestrial (DVB-T) is a standard for the broadcast transmission of DTT. DVB-T means broadcasting a package of TV channels called multiplex whereas analog broadcasting means one transmitter operating on some frequency and transmitting only one TV channel. This standard applies to the whole Europe and other countries such as Russia, Australia, India, etc.

The DTT analog signal must be decoded into digital video streams and delivered to the computers through an interface (e.g., VGA, HDMI). For this reason, the workstation is equipped with ×32 Astrell set-top boxes that cover almost the digital terrestrial channels transmitted for free to use. These set-top boxes are dispatched into two racks (×8 and ×24 items) connected to two computers for the video capture and processing tasks, respectively. Within the racks, each of the set-top boxes demodulates the DTT signal and delivers a video stream of a TV channel to a computer at the full HD resolution. The connections with the computers are established by HDMI interfaces using the MPEG format.

The real-time video capture and processing have different purposes. Thus, they require each a dedicated computer to handle the task. As shown in Fig. A.1, the workstation comprises two dedicated computers that are the DELL Precision 5820 and DELL PowerEdge T640 for the video capture and processing, respectively. For convenience, we use the short names of DP and DPE computers.

Both of the DP and DPE computers are designed to enhance multitasking abilities. They contain a large number of PCI slots to plug capture and processing cards. These cards process the video streams at the hardware level for encoding and decoding, control of the FPS, downscaling, color-space conversion and formatting, transfer to the main memory and disk. That is, the video encoding and decoding are handled in real-time at the hardware level without any load of CPU resources. Several cards are available on the market[4], the workstation embeds the Avermedia CL332-HN and CE314-HN cards [4] for the capture and processing, respectively. These cards not only offer the multiple channels encoding / decoding with hardware support but also provide a software development kit allowing to develop image processing and computer vision methods.

The video capture is processed with the DP computer having the specification shown in Tab. A.2. More specifically, the DP computer embeds four Avermedia CL332-HN cards. Each card can process up to two video streams simultaneously. These Avermedia CL332-HN cards are connected to ×8 set-top boxes through HDMI interfaces. As a result, the DP computer is capable of capturing ×8 TV channels at the same time. Apart the Avermedia cards, the DP computer is equipped with a strong processor Intel Xeon W-2295 3.0 GHz, a main memory of 64 GB, and additional disks with high capacity (i.e., 38 TB) for video storage. The CPU supports the Avermedia CL332-HN cards for the audio capture (not performed at the hardware level within the cards). The driver for the CL332-HN cards is compatible with a Master Boot Record (MBR) partition format that limits the storage capacity to 2 TB. To support long captures, a data pipeline is designed in the DP computer where the video / audio files are first stored in a buffer disk 2TB/MBR and then swapped to high capacity disks of 12 TB formatted in a GUID Partition Table (GPT). The swapping is triggered by operating system services during idle mode (e.g., 02:00 AM to 06:00 AM).

On the other hand, real-time processing applications are processed with the DPE computer having the specification shown in Tab.A.2. The TV streams are processed with Avermedia CE314-HN cards for frame decoding. Each card can process up to 4 video streams simultaneously. Similar to the DP computer, these Avermedia CE314-HN cards are connected to set-top boxes through HDMI interfaces. The computer embeds ×6 CE314-HN cards resulting in an overall processing capacity of ×24 TV channels. Regarding a hard

[4]e.g., Avermedia, Blackmagic Design and Magewell.

real-time constraint, it is recommended to process by CPU instead of GPU, which can suffer from latency [25]. Hence, to support the processing of 24 video streams in real-time, the DPE computer is equipped of × 2 processors Intel Xeon Gold 5218R CPU 2.10 GHz. They can execute up to 80 threads simultaneously to drive 3 processing applications (with an architecture of one thread per channel). The DPE computer processes in the spatial/frame domain requiring a high capacity of storage and bandwidth. For this purpose, the computer is set with 6 high capacity disks handled through two SAS buses 3.0. As a result, the DPE computer supports a storage capacity of 72 TB and an I/O rate of $\simeq$ 600 MB/s.

In summary, the TV workstation built-up according to the above architecture is able to process in real-time up to ×8 and ×24 TV channels for the video capture and processing, respectively. This has been done by taking advantage of two computers capable of integrating multiple channels capture cards with hardware support and processing with the DTT signal that guarantees a minimum latency. In addition, a set of external disks, having a near capacity of 105 TB, is used to handle the data produced by the workstation (the DP and DPE computers). At last, a high speed performance cloud storage solution ($\simeq$ 16 MB/s for downloading) has been opened[5] to make publicly available the databases produced with the workstation.

A.2 Scalable video capture at low resolution

The DP computer presented in Section A.1 captures video and audio data. However, driving a capture over a long period of time with multiple channels raises memory constraints. Hence, the video content has to be compressed and downscaled for time-optimization and memory cost issues as in many computer vision applications [83, 86, 95]. We will discuss here main aspects related to this problem including the video capture on the workstation and memory cost, the software and hardware interpolation. At last, setting recommendations are provided for the video capture on the workstation at low resolution.

• **Video capture and memory cost:** the capture on the DP computer could be performed continuously (e.g., 20 or 24 hours a day) on multiple channels (e.g., 4 or 8). It is asynchronous and delivers separately audio/video files due to the hardware constraint (i.e., the CPU supports the Avermedia CL332-HN cards for the audio capture). The amount of data required to store for common settings is given in Tab. A.3.

Tab. A.3 The video bitrate settings and memory cost for video storage needed to capture a channel during a full day (24 hours) on the DP computer.

Resolution	Mpixels	Average bit rates (Mbps)			Audio	video + audio (GB)		
		DTT	Selected	Avermedia	(Kbps)	1 day	15 days	30 days
Low (320x240)	0.07	0.19	0.56	0.56	128	(5.6 + 1.3)	103.2	206.4
SD (720x576)	0.40	1.00	1.60	3.00	160	(16.1 + 1.6)	265.5	531.0
HD (1280x720)	0.88	2.22	3.00	7.00	256	(30.1 + 2.6)	490.5	981.0

The videos could be captured at either low or high levels of resolution, where the CL332-HN cards support up to the HD one. Given a level of resolution, a key issue for storage optimization is the setting of bitrate parameters. Based on the DTT signal throughput (i.e.,

[5]STVD cloud storage: https://dataset-stvd.univ-tours.fr/

French DTT encodes full HD resolution at 5 Mbps.) and the Avermedia recommendations [4], we have set the capture with different bitrate parameters as detailed in Tab. A.3. These parameters result in a trade-off for the capture in which we selected a high quality for the low-level resolutions and an average quality for the high-level resolutions. Along with video bitrate, the audio compression rate could contribute to a total sum of the memory cost. The French DTT applies an Advanced Audio Coding (AAC) within a range of $\in [128-256]$ Kbps. Using these settings, a one-month capture of $\times 8$ channels at high quality using the workstation requires a near $\simeq$ 8 TB of storage (where $\simeq 85\%$ is used to store the video data showed on the last row in Tab. A.3).

• **Software interpolation:** as discussed in the previous section. A key issue for the storage optimization is to downscale the videos. For noise minimization, the interpolation method used for downscaling must be selected carefully [51, 77, 111]. A first strategy is to perform this interpolation off-line using CPU. However, such an approach becomes challenging when facing to scalability. To clarify these aspects, similar to the work reported on the web-page[6] we have investigated five standard interpolation methods providing by the OpenCV[7] library. The response time results are given in Tab. A.4. These results are derived from the distribution of response times obtained from different threads.

Tab. A.4 The response time with standard interpolation methods using the OpenCV 4.5 library performing on a laptop DELL Precision 7550 equipped with an Intel(R) Core(TM) i9-10885H CPU, 2.40 GHz using Windows 10 OS.

Method	Response time* (ms)		
	1 stream	8 streams	12 streams
AREA	5.98	717.42	1 188.19
LANCZOS4	4.98	718.64	1 161.89
LINEAR	0.99	702.72	1 102.97
CUBIC	0.99	672.21	1 071.92
NEAREST	0.99	663.78	1 058.34

Note*: The response time was computed at an upper bound on its distribution after removing outliers. The outliers are involved with the local thrashing/hard context switch in the system, and thus they should be ignored for the response time measurement.

The results of Tab. A.4 highlight very short RT $\in [1-5]$ ms while processing a single video stream. However, compared with the single threading, large RT $\in [660-1\,190]$ ms can be observed while processing multiple streams (e.g., 8 or 12) at a time with multi-threading. This is mainly explained by the hardware support used by OpenCV[8] with the processor instruction such as the Streaming SIMD Extensions (SSE) or Advanced Vector Extensions (AVX)[9]. It raises a critical section problem with a multi-threading processing. In addition, other systematic factors can be a burden for processing, e.g., the disk access, the time processing required by the video decoding and encoding in main memory [42]. As a result, a near 28 hours could be needed to downscale the $\times 8$ 24-hour videos produced by the workstation. This makes the CPU/software interpolation little suitable and applicable for processing long and multiple video streams [60, 112].

[6]Comparison of OpenCV Interpolation methods by Anthony Tanbakuchi https://gist.github.com/

[7]OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. https://opencv.org/

[8]Extremely slow bilinear interpolation (compared to OpenCV) https://stackoverflow.com/

[9]SSE and AVX are instruction sets to allow parallel processing by multiple cores in a single CPU.

• **Hardware interpolation** is then needed to downscale long and multiple video streams. Within the workstation, Avermedia CL332-HN cards offer video captures with the hardware support. However, no information is provided by Avermedia about the used hardware interpolation method. To clarify this aspect, we have processed with a reverse engineering approach. The videos have been captured at the HD resolution and then downscaled with hardware and software interpolations. We selected the Mean Squared Error (MSE) metric Eq. A.1 to compare the videos obtained with the hardware interpolation to the videos obtained with the software ones. Tab. A.5 shows the minimum noise level for the linear method compared to the other hardware interpolations. That is, the hardware interpolation embedded in the Avermedia cards looks like a linear method.

Tab. A.5 MSE results between a low resolution video (i.e., 80×60) captured by Avermedia and the corresponding videos which have been captured at the HD resolution and downscaled then with the standard interpolation methods provided by the OpenCV library.

	LINEAR	CUBIC	LANCZO4	AREA	NEAREST
MSE	**227.55**	232.85	236.78	244.39	436.60

• **Video capture at low resolution:** as discussed previously, processing with low resolution videos could be acceptable and reasonable for computer vision tasks. In addition, the downscaling process could be performed in real-time with hardware support. However, selecting an appropriate low resolution is a critical point due to the effects of noise. To evaluate the impact of noise at different resolutions, we have captured a long video at the HD resolution (the maximum resolution that CL332-HN card can support) and compared this video to a set of low resolution videos obtained with hardware interpolation at different settings. Then, we have computed the MSE between the HD video and the low resolution ones. For a fair comparison, the low resolution videos have been post-processed by upscaling to the HD resolution with software interpolation.

It can easily see in Fig. A.2 that the MSE remains low for up to the SD resolution and even for some lower resolutions. With the hardware interpolation, a low level of noise could be still maintained up to a size of 75 kilopixels per frame. Indeed, only a noise gap of $\simeq 5\%$ is observed with the minimum MSE score. Let's us note that this gap could be compensated while shifting the capture to a size of 330 kilopixels per frame but resulting in a $\simeq \times 5.5$ memory cost overhead for storage. A size of 75 kilopixels per frame could be achieved with several aspect ratio such as 16:9 (366×206) and 4:3 (320×240). We have selected the aspect ratio 4:3 with a frame size of 320 × 240 pixels to fit with the hardware constraint for the capture. This constraint can be explained that hardware processing looks like AVX instructions where the image with and height should be multiple of 8 or 16 pixels (processed at a time with large size registers).

Note on the Mean Squared Error: the MSE metric is given in Eq. A.1. It is used to measure[10] image quality. It provides the mean of quadratic differences between the pixel intensities of two $\mathbf{I}$ and $\mathbf{I}^*$ having a size (height and width) $\mathbf{M}$, $\mathbf{N}$, respectively. For simplification, Eq. A.1 gives the vectorial form of the MSE computation.

[10]Other metrics (PSNR, SNR, or SSIM) are dedicated to the characterization of severe distortions [18, 106].

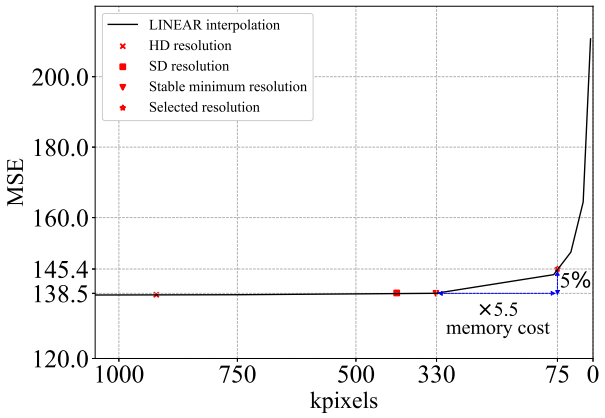


Fig. A.2 The MSE vs. frame sizes in kilopixels when downscaling with the hardware interpolation.

$$\mathbf{MSE}(\mathbf{I}, \mathbf{I}^*) = \frac{\mathbf{1}}{\mathbf{MN}} \sum_{\forall \mathbf{x}} [\mathbf{I}(\mathbf{x}) - \mathbf{I}^*(\mathbf{x})]^{\mathbf{2}} \tag{A.1}$$

A.3 A time and memory efficient implementation of ZNCC

We present in this section different aspects related to the video matching using the **ZNCC** metric. An introduction to the **ZNCC** is given first. Global matching and weighted averaging methods are then presented. At last, a time and memory efficient implementation of the matching on GPU is discussed.

• **ZNCC metric** is given below in Eq.A.2. For simplification, we use here the one dimensional notation. $\mathbf{I}(\mathbf{x})$ is a discrete function taking values in an image. $\mathbf{ZNCC}(\mathbf{I}, \mathbf{I}^*)$ compares the image $\mathbf{I}$ to an image $\mathbf{I}^*$. $\sigma_{\mathbf{I}}$ and $\sigma_{\mathbf{I}^*}$ are the standard deviations of the two images and $\bar{\mathbf{I}}$, $\bar{\mathbf{I}}^*$ are the image means. The zero-mean image of $\mathbf{I}(\mathbf{x})$, and $\mathbf{I}^*(\mathbf{x})$ are computed as $\hat{\mathbf{I}}(\mathbf{x}) = \mathbf{I}(\mathbf{x}) - \bar{\mathbf{I}}$, and $\hat{\mathbf{I}}^*(\mathbf{x}) = \mathbf{I}^*(\mathbf{x}) - \bar{\mathbf{I}}^*$, respectively. The **ZNCC** score is in range $\in [-1, 1]$ where 1 is the perfect correlation.

$$\mathbf{ZNCC}(\mathbf{I}, \mathbf{I}^*) = \frac{\sum_{\forall \mathbf{x}} (\mathbf{I}(\mathbf{x}) - \bar{\mathbf{I}})(\mathbf{I}^*(\mathbf{x}) - \bar{\mathbf{I}}^*)}{\sigma_{\mathbf{I}} \sigma_{\mathbf{I}^*}} = \frac{\sum_{\forall \mathbf{x}} \hat{\mathbf{I}}(\mathbf{x}) \hat{\mathbf{I}}^*(\mathbf{x})}{\sigma_{\mathbf{I}} \sigma_{\mathbf{I}^*}} \tag{A.2}$$

• **ZNCC similarity matrix:** given $\mathbf{X}$ a reference video and $\mathbf{Y}$ a testing video. They have sizes of $\mathbf{m+1}$, $\mathbf{n+1}$ and $\mathbf{m \ll n}$, respectively. To compare $\mathbf{X}$ and $\mathbf{Y}$ we have conducted a frame-to-frame similarity matrix based on **ZNCC** scores. In particular, two sets of ordered frames for $\mathbf{X}$ and $\mathbf{Y}$ have been extracted as $\mathbf{X} = \{\mathbf{I_0}, \ldots, \mathbf{I_k}, \ldots, \mathbf{I_m}\}$, and $\mathbf{Y} = \{\mathbf{I_0^*}, \ldots, \mathbf{I_l^*}, \ldots, \mathbf{I_n^*}\}$ with $\mathbf{k} \in [\mathbf{0}, \mathbf{m}]$, $\mathbf{l} \in [\mathbf{0}, \mathbf{n}]$. A partial copied segment of $\mathbf{X}$ can be detected if a near-duplicate $\widetilde{\mathbf{X}}$ of $\mathbf{X}$ appears in $\mathbf{Y}$ such as $\widetilde{\mathbf{X}} \simeq \mathbf{X}$ and $\widetilde{\mathbf{X}} \in \mathbf{Y}$. Thus, we have computed the matrix $\mathbf{S}$ (Eq. A.3) of global matching having a size $(\mathbf{m+1}, \mathbf{n+1})$ and

where the element $\mathbf{s_{kl}} = \mathbf{ZNCC}(\mathbf{I_k}, \mathbf{I_l^*})$. Fig. A.3 illustrates the global matrix $\mathbf{S}$ between two videos $\mathbf{X}$ and $\mathbf{Y}$ based on $\mathbf{ZNCC}$ scores in the form of heat-map visualization.

$$\mathbf{S} = \begin{bmatrix} \mathbf{s_{00}} & \cdots & \mathbf{s_{0n}} \\ \vdots & \mathbf{s_{kl}} & \vdots \\ \mathbf{s_{m0}} & \cdots & \mathbf{s_{mn}} \end{bmatrix} \tag{A.3}$$

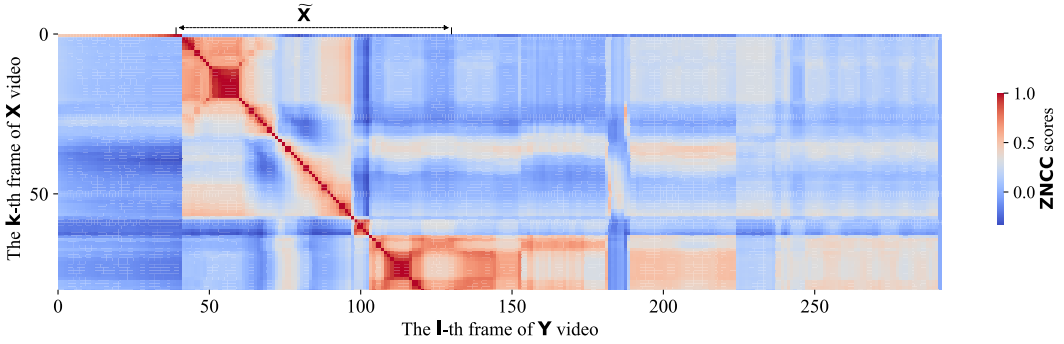


Fig. A.3 An example of the ZNCC similarity matrix.

• **Weighted averaging:** from the global matrix $\mathbf{S}$, partial copies $\widetilde{\mathbf{X}}$ can be obtained by finding the diagonal block with the largest similarity score. However, the final score and the timing accuracy could be affected by weighting methods. To investigate this aspect, we have extracted $\mathbf{W} = [\mathbf{w_0}, \ldots, \mathbf{w_l}, \ldots, \mathbf{w_{n-m}}]$ the vector of weighted averaging obtained from the diagonal elements of $\mathbf{S}$. Each element $\mathbf{w_l}$ is computed as the equation below.

$$\mathbf{w_l} = \sum_{\mathbf{k=0}}^{\mathbf{m}} \mathbf{s_{k,l+k}} \cdot \lambda_\mathbf{k} \tag{A.4}$$

In Eq. A.4, $\lambda_\mathbf{k}$ are the weighting parameters such as $\sum_{\forall \mathbf{k}} \lambda_\mathbf{k} = \mathbf{1}$. The first approach is applying with mean averaging where $\lambda_\mathbf{k} = \frac{\mathbf{1}}{\mathbf{m+1}} \forall \mathbf{k}$ is a constant. Fig A.4 plots an example of $\mathbf{W}$ values obtained from the mean averaging.

An alternative is to set the $\lambda_\mathbf{k}$ parameters from the self-matching of $\mathbf{X}$ features such as the $\mathbf{NCC\_}diff$ [28], the SIFT [46] or the entropy-based features [37]. Indeed the maximum matching $\mathbf{ZNCC}(\mathbf{I_k}, \mathbf{I_l})$ of a frame $\mathbf{I_k} \in \mathbf{X}$ to other frames $\mathbf{I_l} \in \mathbf{X}$, with $\mathbf{l} \neq \mathbf{k}$, could serve to minimize the weighting parameter. This characterizes how the frame $\mathbf{I_k}$ has near duplicates in the segment $\mathbf{X}$ and is less discriminant. For a fair comparison and a normalized score between the above weighting methods, a square matrix $\mathbf{S'}$ having of size $(\mathbf{m+1}, \mathbf{m+1})$ have been written from the $\mathbf{S}$ matrix as illustrated in Eq. A.5. That is, in $\mathbf{S'}$, $\bar{\mathbf{s}}_\mathbf{kl} \in [\mathbf{0}, \mathbf{1}]$ is the normalized value of $\mathbf{s_{kl}}$ to be adapted from methods to methods (e.g., for the $\mathbf{NCC\_}diff$, $\bar{\mathbf{s}}_\mathbf{kl} = \frac{\mathbf{1-s_{kl}}}{\mathbf{2}}$ is the normalized and inverse value of the $\mathbf{ZNCC}$ score $\mathbf{s_{kl}}$ in Eq. A.2 and A.3).

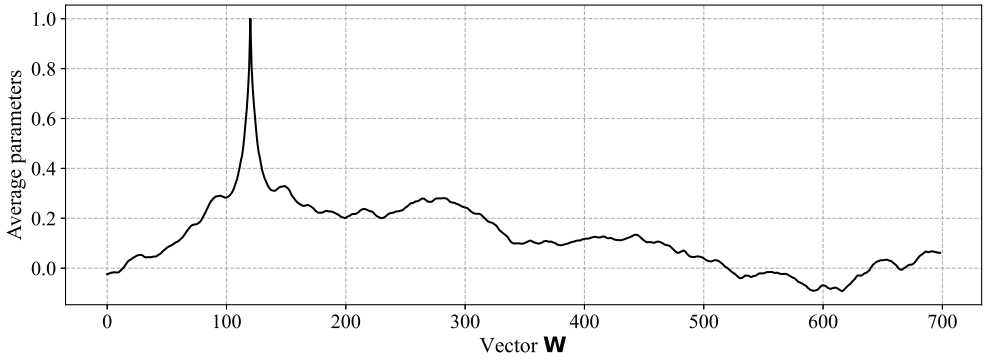


Fig. A.4 The vector $\mathbf{W}$ with mean averaging.

$$\mathbf{S}' = \begin{bmatrix} \_ & \cdots & \bar{\mathbf{s}}_{\mathbf{0m}} \\ \vdots & \_ & \vdots \\ \bar{\mathbf{s}}_{\mathbf{m0}} & \cdots & \_ \end{bmatrix} \tag{A.5}$$

A min operator could be applied to the columns of $\mathbf{S}'$ matrix to get $[\hat{\bar{\mathbf{s}}}_{\mathbf{0}}, \ldots, \hat{\bar{\mathbf{s}}}_{\mathbf{k}}, \ldots, \hat{\bar{\mathbf{s}}}_{\mathbf{m}}]$ where $\hat{\bar{\mathbf{s}}}_{\mathbf{k}} = \min_{\forall \mathbf{l}} \bar{\mathbf{s}}_{\mathbf{kl}}$. Each term $\hat{\bar{\mathbf{s}}}_{\mathbf{k}}$ characterizes a maximum matching between two frames $\mathbf{k}$, $\mathbf{l}$ with $\forall \mathbf{k} \neq \mathbf{l}$. They could be normalized in the final step to get the $[\lambda_{\mathbf{0}}, \ldots, \lambda_{\mathbf{k}}, \ldots \lambda_{\mathbf{m}}]$.

To identify the accurate weighting method, we have conducted experiments on 150 pairs of reference videos $\mathbf{X}$ and testing videos $\mathbf{Y}$. Fig. A.5 shows that the exact matching can be observed by the mean averaging method compared to the other methods. Hence, this mean averaging method has been used as a weighting standard.

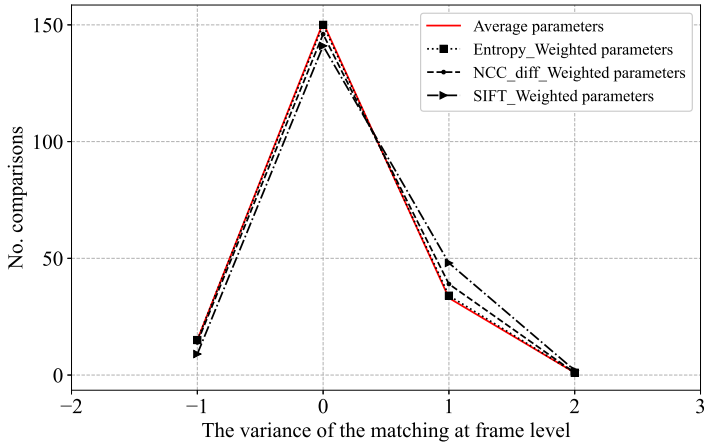


Fig. A.5 The comparison of various weighting methods.

• **Efficient matching on GPU:** computing the global matching $\mathbf{S}$ matrix could require a huge time-processing and memory consumption. Given $p = ||\mathbf{I}|| = ||\mathbf{I}^*||$ the frame size, and $q = (\mathbf{m}+\mathbf{1})(\mathbf{n}+\mathbf{1})$ the total number of frames needed for matching, it requires a

$O(pq)$ computational cost and a rough $O(q + p(m + 1) + p(n + 1))$ memory requirements. However, each element of the $\mathbf{S}$ can be computed independently. Thus, this process can be accelerated by taking advantages of the parallel programing on GPU. We have developed a full pipeline for the global matrix $\mathbf{S}$ computation described as below.

Step 1. to apply a gray-level conversion/downscaling to the frame $\mathbf{I_0}, \ldots, \mathbf{I_m}$ of $\mathbf{X}$. To speed up the **ZNCC** product, compute the zero-mean images $\mathbf{I}(\mathbf{x}) - \bar{\mathbf{I}}$. The final **ZNCC** product needs a floating-point value for pixel coding. This would require some MiB for storage purpose. For instance, 64×48 pixels per frame, 4 bytes (32-bit floating-point) per pixel, 30 FPS, a duration $\mathbf{D} = 3$ seconds will require 1.05 MiB.

Step 2. $\forall \mathbf{k}$, compute and store the $(\sigma_{\mathbf{I_0}}, \ldots, \sigma_{\mathbf{I_m}})$ parameters. These are scalars and require a little memory for storage aspect. (i.e., < 1 KiB). Repeat steps 3 and 4.

Step 3. apply steps 1, 2 to the video $\mathbf{Y}$. For instance, 64×48 pixels per frame, 30 FPS, a duration $\mathbf{D} = 1,200$ seconds will require 422 MiB in main memory with a floating-point format and a near 140 KiB for $(\sigma_{\mathbf{I_0^*}}, \ldots, \sigma_{\mathbf{I_n^*}})$.

Step 4. the global matrix $\mathbf{S}$ would require a memory allocation at a $O(q)$ for a float coding. For each matching $\mathbf{s_{kl}} = \mathbf{ZNCC}(\mathbf{I_k}, \mathbf{I_l^*})$, the result could be obtained with a matrix multiplication and summation at time complexity of $O(2p)$. Therefore, the $\mathbf{S}$ matrix has a memory requirement of $O(q + p(m + 1) + p(n + 1))$. For instance, it could require 436 MiB to compute 64×48 pixels per frame, 4 bytes per pixel, 30 FPS and two videos $\mathbf{X}$, and $\mathbf{Y}$ with $\mathbf{D} = 3$ and 1 200 seconds, respectively. The final divide operation is done while applying the $\sigma_{\mathbf{I_k}} \times \sigma_{\mathbf{I_l^*}}$ parameters. The matrix addition, multiplication and reduction could implemented efficiently on GPU [69, 80].

Tab. A.6 presents a performance comparison of the CPU and GPU implementations. As detailed, the processing time on GPU improves $\simeq$ two orders of magnitude compared to the CPU. To take advantage of this benefit, we have selected this GPU implementation for our experiments presented in Chapter 2.

Tab. A.6 Comparison of the average processing time between CPU and GPU for **ZNCC** matching.

	CPU (14 threads)	GPU
Time processing (s)	390	5

Note: we selected the reference and testing videos having FPS = 30 with duration of 3 and 1 200 seconds, respectively. The frame size was fixed at a 64×48 pixel in gray-scale. The experiments were performed on the laptop DELL Precision 7550 equipped a CPU Intel(R) Core(TM) i9-10885H supporting up to 18 logical threads, a GPU Quadro-T2000 4 GiB memory, and main memory of 32 GiB. The number of threads was fixed at 14 for computation, while keeping the remaining threads for the safety of the operating system. The processing time was computed as the average value of the results after 1 000 run-times.

A.4 Video degradation with compression and downscaling

In this section, we present our protocol for video degradation with compression. Video compression is a key process for the reduction of the memory cost and network bandwidth.

However, it results in a degradation of videos where the compression method, parameters and resolution are key factors for controlling the noise level. This is a well-known problem in the computer vision field addressed in several studies [1, 70].

In our work, we have considered the MPEG (Moving Picture Experts Group) method for compression. The MPEG is a popular method to compress videos, and it is widely used for broadcasting digital TV signals and online web videos. It enters in the category of lossy compression methods using inexact approximations (i.e., where the compression is irreversible). The MPEG uses various strategies to achieve a high-quality compression taking into account the temporal and spatial redundancies [70]. It relies on two encoding standards H.264/MPEG-4 AVC (Advanced Video Coding) and H.265/MPEG-H HEVC (High-Efficiency Video Coding), where the H.265 is the last recent one delivering a significantly better visual content at the same level of compression. In our protocol, we have used the H.264/MPEG-4 AVC as it is consistent with the compression norm of French DTT signal[11]. This compression is supported too by hardware encoding in the TV workstation with the 312-HN Avermedia cards (see Appx. A.1). Thus, it constitutes then a good candidate method for our degradation protocol.

Similar to the T4 - video transformation in the popular benchmark TRECVID [81], we have controlled the encoding parameter kbps (kilobit per second) and video resolution for degradation with the H.264/MPEG-4 AVC compression. Two parameters α, β have been defined for the needs as detailed in Tab. A.7. We have evaluated the impact of these parameters on the degradation of a candidate video (encoded at the 320×240 resolution / 30 FPS / 560 kbps with a duration of 20 minutes). Fig. A.6 provides an analysis of the impact of parameters and their correlation on the video degradation.

Tab. A.7 Parameters used to control the video degradations with compression methods.

Parameter	Description	Degradation level	
		average	hard
α	is a downscaling parameter to control the video resolution at different scales $\alpha \in [0.1, 0.9]$. It results in frame sizes from 32×24 up to 288×216.	$]0.25, 0.9]$	$[0.1, 0.25]$
β	is a bit-rate parameter to control the video compression at different scales $\beta \in [1, 80]$. It is applied according to the recommended kbps $\in \{140, 280, 420\}$ for capture [4] such as $\frac{1}{\beta} \times$ kbps.	$]1, 40]$	$[40, 80]$

As a first step, we have characterized the impact of video downscaling with the α parameter. The video at the 320×240 resolution has been downscaled based on the α parameter, then upscaled and compared to the root video. To evaluate the degradation, the MSE metric, that is a term the PSNR (Peak Signal-to-Noise Ratio) metric, has been used to characterize the noise (see the end note). Indeed, it is a popular characterization metric of the compression artifacts [7]. As given in Fig. A.6 (a), this parameter α could be characterized with a null tangent at $\alpha \simeq 0.9$, an elbow at $\alpha \simeq 0.25$ and an infinite tangent at alpha $\simeq 0.1$. Then, we have fixed two ranges with downscaling at $\alpha \in]0.25, 0.9]$ and $\in [0.1, 0.25]$ for average and hard degradations, as detailed in Tab. A.7.

In the second step, we have analyzed the variation of β parameter to control the compression quality Fig. A.6 (b). This β parameter has been correlated to the two specific values $\alpha_1 = 0.1$ and $\alpha_2 = 0.25$. Similar to the above experiments, the videos obtained at

[11]TDF's report: https://www.tdf.fr/sites/default/files/TDF-LIVRE-BLANC-TNT_0.pdf

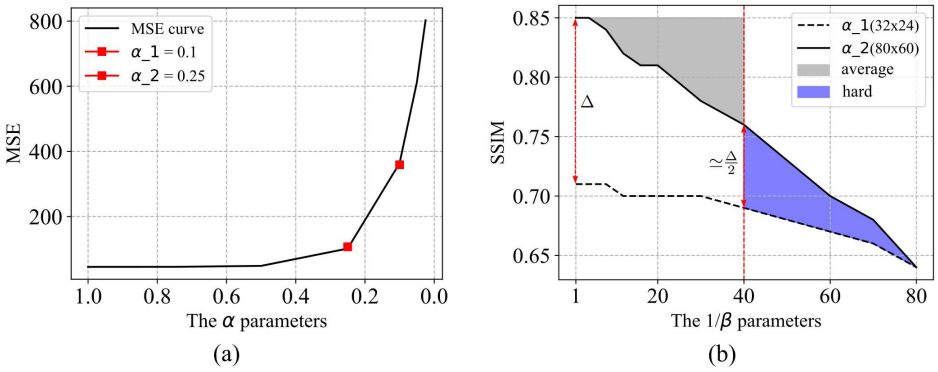


Fig. A.6 An analysis for selection of α, β parameters using the MSE and SSIM metrics.

the lowest resolutions and qualities have been upscaled and compared to the root video (at the 320×240 resolution). The Structural Similarity Index Measure (SSIM) metric (see the end note) has been used for a finer characterization. This metric is more sensitive than the MSE and then suitable for such these cases [20]. As shown in Fig. A.6 (b), the β parameter has the strongest impact at a high-level of resolution with $\alpha_2 = 0.25$. A near half reduction $\simeq \frac{\Delta}{2}$ of the gap between two SSIM scores could be obtained at $\beta \simeq 40$, whereas the SSIM scores become equally with α_1 and α_2 for $\beta \simeq 80$.

Based on these analyses, we have fixed two correlated ranges for α, β as detailed Tab. A.7. These ranges result in two levels of degradation illustrated in Fig. A.6 (b).

Note on MSE / SIMM: For simplification, we use the vectorial form of image data to define the MSE and SSIM metrics presented in Eq. A.6 and Eq. A.7, respectively.

The MSE metric is given in Eq. A.6. It is used to measure image quality. It provides the mean of quadratic differences between the pixel intensities of two images $\mathbf{I}$ and $\mathbf{I}^*$ having a size (height and width) $\mathbf{M}$, $\mathbf{N}$, respectively.

$$\mathbf{MSE}(\mathbf{I}, \mathbf{I}^*) = \frac{\mathbf{1}}{\mathbf{MN}} \sum_{\forall \mathbf{x}} [\mathbf{I}(\mathbf{x}) - \mathbf{I}^*(\mathbf{x})]^{\mathbf{2}} \tag{A.6}$$

The SSIM metric is given in Eq. A.7. It quantifies the similarity in terms of luminance, contrast, and structure between the two images $\mathbf{I}$ and $\mathbf{I}^*$. The parameters $c_1 = (k_1 \times L)^2$ and $c_2 = (k_2 \times L)^2$ are constants to stabilize the division, where L is the dynamic range of pixel values (e.g., 255 for 8-bit/gray images) and $k_1 = 0.01$, $k_2 = 0.03$ by default.

$$\mathbf{SSIM}(\mathbf{I}, \mathbf{I}^*) = \frac{(2 \cdot \overline{\mathbf{I}} \cdot \overline{\mathbf{I}^*} + c_1)(2 \cdot \sigma_{\mathbf{I},\mathbf{I}^*} + c_2)}{(\overline{\mathbf{I}}^2 + \overline{\mathbf{I}^*}^2 + c_1)(\sigma_{\mathbf{I}}^2 + \sigma_{\mathbf{I}^*}^2 + c_2)} \tag{A.7}$$

Bibliography

[1] P. Akyazi and T. Ebrahimi. Comparison of compression efficiency between hevc/h. 265, vp9 and av1 based on subjective quality assessments. In *International Conference on Quality of Multimedia Experience (QoMEX)*, pages 1–6, 2018.

[2] T. Arsan, E.E. Bulut, B. Eren, A. Uzgor, and S. Yolcu. A novel iptv framework for automatic tv commercials detection, labeling, recognition and replacement. *Multimedia Tools and Applications (Multimed. Tools Appl)*, pages 1–19, 2022.

[3] K.M.M. Aung and A.N. Htwe. Comparison of levenshtein distance algorithm and needleman-wunsch distance algorithm for string matching. *National Journal of Parallel and Soft Computing*, 2019.

[4] AVerMedia. Avermedia capture card software developement kit. Technical Report 4.2, AVerMedia Technologies, Inc. www.avermedia.com, 2015.

[5] L. Baraldi, M. Douze, R. Cucchiara, and H. Jégou. Lamv: Learning to align and match videos with kernelized temporal layers. In *IEEE conference on computer vision and pattern recognition (CVPR)*, pages 7804–7813, 2018.

[6] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *European Conference on Computer Vision (ECCV)*, pages 404–417, 2006.

[7] A. Bhat, S. Kannangara, Y. Zhao, and I. Richardson. A full reference quality metric for compressed video based on mean squared error and video content. *IEEE transactions on circuits and systems for video technology (TCSVT)*, 22(2):165–173, 2011.

[8] A.C Bovik. *Handbook of image and video processing*. Academic press, 2010.

[9] Y. Cai, L. Yang, W. Ping, F. Wang, T. Mei, X.S. Hua, and S. Li. Million-scale near-duplicate video retrieval system. In *ACM international conference on Multimedia (MM)*, pages 837–838, 2011.

[10] D. Chand and H. Ogul. Content-based search in lecture video: A systematic literature review. In *Conference on Information and Computer Technologies (ICICT)*, pages 169–176, 2020.

[11] J.H. Chenot and G. Daigneault. A large-scale audio and video fingerprints-generated database of TV repeated contents. In *International Workshop on Content-Based Multimedia Indexing (CBMI)*, pages 1–6, 2014.

[12] T.H. Chiang, Y.C. Tseng, and Y.C. Tseng. A multi-embedding neural model for incident video retrieval. *Pattern Recognition*, 130:108807, 2022.

[13] F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *IEEE conference on computer vision and pattern recognition (CVPR)*, pages 1251–1258, 2017.

[14] C.L. Chou, H.T. Chen, and S.Y. Lee. Pattern-based near-duplicate video retrieval and localization on web-scale videos. *IEEE Transactions on Multimedia*, 17(3):382–395, 2015.

[15] W.T. Chu, P.C. Chuang, and J.Y. Yu. Video copy detection based on bag of trajectory and two-level approximate sequence. In *IPPR Conference on Computer Vision, Graphics, and Image Processing Conference (CVGIP)*, 2010.

[16] A. Cools, M.A. Belarbi, and S.A. Mahmoudi. A comparative study of reduction methods applied on a convolutional neural network. *Electronics*, 11:1422, 2022.

[17] P. Covington, J. Adams, and E. Sargin. Deep neural networks for youtube recommendations. In *ACM conference on recommender systems (RecSys)*, pages 191–198, 2016.

[18] R. Dass and N. Yadav. Image quality assessment parameters for despeckling filters. *Procedia Computer Science*, 167:2382–2392, 2020.

[19] J. Deng, W. Dong, R. Socher, L.J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Conference on computer vision and pattern recognition (CVPR)*, pages 248–255, 2009.

[20] R. Dosselmann and X.D. Yang. A comprehensive assessment of the structural similarity index. *Signal, Image and Video Processing*, 5:81–91, 2011.

[21] M. Douze, H. Jégou, and C. Schmid. An image-based approach to video copy detection with spatio-temporal post-filtering. *IEEE Transactions on Multimedia*, 12(4):257–266, 2010.

[22] M. Douze, H. Jégou, C. Schmid, and P. Pérez. Compact video description for copy detection with precise temporal alignment. In *European Conference on Computer Vision (ECCV)*, pages 522–535, 2010.

[23] Y. Feng, L. Ma, W. Liu, T. Zhang, and J. Luo. Video re-localization. In *European Conference on Computer Vision (ECCV)*, pages 51–66, 2018.

[24] S. Gkelios, A. Sophokleous, S. Plakias, Y. Boutalis, and S.A. Chatzichristofis. Deep convolutional features for image retrieval. *Expert Systems With Applications*, 177(114940), 2021.

[25] V. Golyanik, M. Nasri, and D. Stricker. Towards scheduling hard real-time image processing tasks on a single gpu. In *International Conference on Image Processing (ICIP)*, pages 4382–4386, 2017.

[26] R.C Gonzalez and R.E Woods. *Digital Image Processing (4th Edition).* Pearson, 2018.

[27] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning.* MIT press, 2016.

[28] Z.J. Guzman-Zavaleta and C. Feregrino-Uribe. Towards a video passive content fingerprinting method for partial-copy detection robust against non-simulated attacks. *PLOS ONE*, 11(11):1–19, 2016.

[29] Z.J. Guzman-Zavaleta and C.F. Uribe. Partial-copy detection of non-simulated videos using learning at decision level. *Multimedia Tools and Applications (Multimed. Tools Appl)*, 78(2):2427–2446, 2019.

[30] Z. Han, X. He, M. Tang, and Y. Lv. Video similarity and alignment learning on partial video copy detection. In *ACM International Conference on Multimedia (MM)*, pages 4165–4173, 2021.

[31] Y. Hao, R. Mu, T.and Hong, M. Wang, N. An, and J.Y. Goulermas. Stochastic multiview hashing for large-scale near-duplicate video retrieval. *IEEE Transactions on Multimedia (MM)*, 19(1):1–14, 2016.

[32] C. Harris and M. Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, pages 10–5244, 1988.

[33] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Conference on computer vision and pattern recognition (CVPR)*, pages 770–778, 2016.

[34] S. He, Y. He, M. Lu, C. Jiang, X. Yang, F. Qian, X. Zhang, L. Yang, and J. Zhang. Transvcl: Attention-enhanced video copy localization network with flexible supervision. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2023.

[35] S. He, X. Yang, C. Jiang, G. Liang, W. Zhang, T. Pan, Q. Wang, F. Xu, C. Li, J. Liu, et al. A large-scale comprehensive dataset and copy-overlap aware evaluation protocol for segment-level video copy detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21086–21095, 2022.

[36] X. He, Y. Pan, M. Tang, Y. Lv, and Y. Peng. Learn from unlabeled videos for near-duplicate video retrieval. In *ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 1002–1011, 2022.

[37] Y. Hou, X. Wang, S. Liu, and Y. Zhang. Video copy detection based on uniform local binary pattern. *DEStech Transactions on Computer Science and Engineering*, 2017.

[38] Y. Hu and X. Lu. Learning spatial-temporal features for video copy detection by the combination of cnn and rnn. *Journal of Visual Communication and Image Representation (J. Vis. Commun. Image Represent)*, 55:21–29, 2018.

[39] Y. Hu, Z. Mu, and X. Ai. STRNN: End-to-end deep learning framework for video partial copy detection. *Journal of Physics: Conference Series (JPCS)*, 1237(2):022112, 2019.

[40] Muhammad Faraz Hyder, Maria Andleeb Siddiqui, Muhammad Mukarram, et al. Tv ad detection using the base64 encoding technique. *Engineering, Technology & Applied Science Research (ETASR)*, 11(5):7605–7609, 2021.

[41] O. Ieremeiev, V. Lukin, K. Okarma, and K. Egiazarian. Full-reference quality metric based on neural network to assess the visual quality of remote sensing images. *Remote Sensing*, 12(15):2349, 2020.

[42] Y. V Ivanov and C.J. Bleakley. Real-time h. 264 video encoding in software with fast mode decision and dynamic complexity control. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 6(1):1–21, 2010.

[43] C. Jiang, K. Huang, S. He, X. Yang, W. Zhang, X. Zhang, Y. Cheng, L. Yang, Q. Wang, and F. Xu. Learning segment similarity and alignment in large-scale content based video retrieval. In *ACM International Conference on Multimedia (MM)*, pages 1618–1626, 2021.

[44] J. Jiang, Yu. Tong, H. Lu, B. Cui, K. Lei, and L. Yu. Gvos: a general system for near-duplicate video-related applications on storm. *ACM Transactions on Information Systems (TOIS)*, 36(1):1–36, 2017.

[45] Q.Y. Jiang, Y. He, G. Li, J. Lin, L. Li, and W.J. Li. SVD: A Large-Scale Short Video Dataset for Near-Duplicate Video Retrieval. In *International Conference on Computer Vision (ICCV)*, pages 5281–5289, 2019.

[46] Y.G. Jiang, Y. Jiang, and J. Wang. Vcdb: a large-scale database for partial copy detection in videos. In *European conference on computer vision (ECCV)*, pages 357–371. Springer, 2014.

[47] Y.G. Jiang and J. Wang. Partial copy detection in videos: A benchmark and an evaluation of popular methods. *IEEE Transactions on Big Data*, 2(1):32–42, 2016.

[48] W. Jing, X. Nie, C. Cui, X. Xi, G. Yang, and Y. Yin. Global-view hashing: harnessing global relations in near-duplicate video retrieval. *World wide web*, 22(2):771–789, 2019.

[49] A. Joly, O. Buisson, and C. Frélicot. Content-based copy retrieval using distortion-based probabilistic similarity search. *Transactions on Multimedia*, 9(2):293–306, 2007.

[50] M. Kanmani and V. Narsimhan. An image contrast enhancement algorithm for grayscale images using particle swarm optimization. *Multimedia Tools and Applications (Multimed Tools Appl)*, 77:23371–23387, 2018.

[51] D. Khaledyan, A. Amirany, K. Jafari, M.H. Moaiyeri, A.Z. Khuzani, and N. Mashhadi. Low-cost implementation of bilinear and bicubic image interpolation for real-time image super-resolution. In *2020 IEEE Global Humanitarian Technology Conference (GHTC)*, pages 1–5, 2020.

[52] R. Klette, H.S. Stiehl, M.A. Viergever, and K.L. Vincken. *Performance characterization in computer vision*. Springer, 2000.

[53] G. Kordopatis-Zilos, S. Papadopoulos, I. Patras, and I. Kompatsiaris. Fivr: Fine-grained incident video retrieval. *IEEE Transactions on Multimedia*, 21(10):2638–2652, 2019.

[54] G. Kordopatis-Zilos, S. Papadopoulos, I. Patras, and I. Kompatsiaris. ViSiL: Fine-grained spatio-temporal video similarity learning. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6350–6359, 2019.

[55] G. Kordopatis-Zilos, S. Papadopoulos, I. Patras, and Y. Kompatsiaris. Near-duplicate video retrieval by aggregating intermediate cnn layers. In *International conference on multimedia modeling (MMM)*, pages 251–263. Springer, 2017.

[56] G. Kordopatis-Zilos, S. Papadopoulos, I. Patras, and Y. Kompatsiaris. Near-duplicate video retrieval with deep metric learning. In *International Conference on Computer Vision Workshops (ICCV)*, pages 347–356, 2017.

[57] G. Kordopatis-Zilos, G. Tolias, C. Tzelepis, I. Kompatsiaris, I. Patras, and S. Papadopoulos. Self-supervised video similarity learning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4756–4766, 2023.

[58] G. Kordopatis-Zilos, C. Tzelepis, S. Papadopoulos, I. Kompatsiaris, and I. Patras. Dns: Distill-and-select for efficient and accurate video indexing and retrieval. *International Journal of Computer Vision*, 130(10):2385–2407, 2022.

[59] A. Krizhevsky, I. Sutskever, and G.E. Hinton. Imagenet classification with deep convolutional neural networks. In *Conference and Workshop on Neural Information Processing Systems (NeurIPS)*, volume 25, 2012.

[60] Y. Kui-Ying, S. Fa-Long, Z. Sheng-Hua, and Z. Changchun. Par model sar image interpolation algorithm on gpu with cuda. *IETE Technical Review*, 31(4):297–306, 2014.

[61] J. Law-To, L. Chen, A. Joly, I. Laptev, O. Buisson, V. Gouet-Brunet, N. Boujemaa, and F. Stentiford. Video copy detection: a comparative study. In *ACM International Conference on Image and Video Retrieval (CIVR)*, pages 371–378, 2007.

[62] J. Law-To, A. Joly, and N. Boujemaa. Muscle-vcd-2007: a live benchmark for video copy detection, 2007. http://www-rocq.inria.fr/imedia/civr-bench/.

[63] J. Law-To, A. Joly, L. Joyeux, N. Boujemaa, O. Buisson, and V. Gouet-Brunet. Video and image copy detection demo. In *International conference on Image and Video Retrieval (CIVR)*, pages 97–100, 2007.

[64] V.H. Le, M. Delalandre, and D. Conte. Real-time detection of partial video copy on tv workstation. In *International Conference on Content-Based Multimedia Indexing (CBMI)*, pages 1–4. IEEE, 2021.

[65] V.H. Le, M. Delalandre, and D. Conte. A large-scale tv dataset for partial video copy detection. In *International Conference Image Analysis and Processing (ICIAP)*, pages 388–399, 2022.

[66] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[67] D. Lee, M. Lee, D.and Choi, and J. Lee. Prediction of network throughput using arima. In *International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, pages 1–5, 2020.

[68] J. Lee, N. Kothari, and P. Natsev. Content-based related video recommendations. *Advances in Neural Information Processing Systems (NIPS) Demonstration Track*, 2, 2016.

[69] F. Li, Y. Ye, Z. Tian, and X. Zhang. Cpu versus gpu: which can perform matrix computation faster–performance comparison for basic linear algebra subprograms. *Neural Computing and Applications*, 31(8):4353–4365, 2019.

[70] Z.N. Li, M.S. Drew, and J. Liu. Modern video coding standards: H. 264, h. 265, and h. 266. *Fundamentals of Multimedia*, pages 423–478, 2021.

[71] H. Liu, Q. Zhao, H. Wang, P. Lv, and Y. Chen. An image-based near-duplicate video retrieval and localization using improved edit distance. *Multimedia Tools and Applications (Multimed. Tools Appl)*, 76(22):24435–24456, 2017.

[72] J. Liu, Z. Huang, H. Cai, H.T. Shen, C.W. Ngo, and W. Wang. Near-duplicate video retrieval: Current research and future trends. *ACM Computing Surveys (CSUR)*, 45(4):1–23, 2013.

[73] M. Liu, L.M. Po, Y.A. Ur Rehman, X. Xu, Y. Li, and L. Feng. Video copy detection by conducting fast searching of inverted files. *Multimedia Tools and Applications (Multimed. Tools Appl)*, 78(8):10601–10624, 2019.

[74] X. Liu, X. Feng, and P. Pan. Gann: A graph alignment neural network for video partial copy detection. In *Intl Conference on Big Data Security on Cloud (BigDataSecurity), Intl Conference on High Performance and Smart Computing,(HPSC) and Intl Conference on Intelligent Data and Security (IDS)*, pages 191–196. IEEE, 2021.

[75] D.G. Lowe. Object recognition from local scale-invariant features. In *International conference on computer vision (ICCV)*, volume 2, pages 1150–1157, 1999.

[76] A.K. Mallick and S. Maheshkar. Near-duplicate video retrieval based on spatiotemporal pattern tree. In *International Conference on Computer Vision & Image Processing (CVIP)*, pages 173–186, 2018.

[77] Y. Marin, J. Miteran, J. Dubois, B. Heyrman, and D. Ginhac. An fpga-based design for real-time super-resolution reconstruction. *Journal of Real-Time Image Processing (JRTIP)*, 17(6):1769–1785, 2020.

[78] T.M. Mitchell. *Machine Learning*. McGraw–Hill, Inc., 1997.

[79] X. Nie, X. Li, J. Sun, and Y. Yin. Ufvh: unified feature video hashing for near-duplicate video retrieval. In *Workshop on Visual Analysis in Smart and Connected Communities (VSCC)*, pages 17–24, 2017.

[80] C NVIDIA. Cuda c++ programming guide. *NVIDIA, Aug*, 2020.

[81] P. Over, G. Awad, J. Fiscus, B. Antonishek, M. Michel, W. Kraaij, A. Smeaton, and G. Quénot. Trecvid 2010 - an overview of the goals, tasks, data, evaluation mechanisms and metrics. In *TRECVID 2010 workshop participants notebook papers, Gaithersburg, MD, USA*. NIST, USA, 2010.

[82] G. Özbulak, F. Kahraman, and S. Baykut. Robust video copy detection in large-scale tv streams using local features and cfar based threshold. In *International Conference on Digital Signal Processing (DSP)*, pages 124–128, 2016.

[83] A. Petreto, T. Romera, F. Lemaitre, M. Bouyer, B. Gaillard, P. Menard, Q. Meunier, and L. Lacassagne. Real-time embedded video denoiser prototype. In *International Symposium-Optronics in Defense and Security (Optro)*, 2020.

[84] F. Rayar, M. Delalandre, and V.H. Le. A large-scale tv video and metadata database for french political content analysis and fact-checking. In *International Conference on Content-Based Multimedia Indexing (CBMI)*. IEEE, 2022.

[85] J. Revaud, M. Douze, C. Schmid, and H. Jégou. Event retrieval in large video collections with circulant temporal encoding. In *Conference on computer vision and pattern recognition (CVPR)*, pages 2459–2466, 2013.

[86] S. Saponara, A. Elhanashi, and A. Gagliardi. Real-time video fire/smoke detection based on cnn in antifire surveillance systems. *Journal of Real-Time Image Processing (JRTIP)*, 18(3):889–900, 2021.

[87] L. Shang, L. Yang, F. Wang, K.P. Chan, and X.S. Hua. Real-time large scale near-duplicate web video retrieval. In *ACM international conference on Multimedia (MM)*, pages 531–540, 2010.

[88] J. Shao, X. Wen, B. Zhao, and X. Xue. Temporal context aggregation for video retrieval with contrastive learning. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 3268–3278, 2021.

[89] H.T. Shen, X. Zhou, Z. Huang, J. Shao, and X. Zhou. Uqlips: a real-time near-duplicate video clip detection system. In *International conference on Very large data bases (VLDB)*, pages 1374–1377, 2007.

[90] L. Shen, R. Hong, and Y. Hao. Advance on large scale near-duplicate video retrieval. *Frontiers of Computer Science (Front. Comput. Sci.)*, 14(5):1–24, 2020.

[91] C. Shorten and T.M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

[92] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *Open-access arXiv:1409.1556*, 2014.

[93] J. Song, Y. Yang, Z. Huang, H.T. Shen, and R. Hong. Multiple feature hashing for real-time large scale near-duplicate video retrieval. In *ACM international conference on Multimedia (MM)*, pages 423–432, 2011.

[94] J. Song, Y. Yang, Z. Huang, H.T. Shen, and J. Luo. Effective multiple feature hashing for large-scale near-duplicate video retrieval. *IEEE Transactions on Multimedia*, 15(8):1997–2008, 2013.

[95] S. Song, Y. Li, Q. Huang, and G. Li. A new real-time detection and tracking method in videos for small target traffic signs. *Applied Sciences*, 11(7):3061, 2021.

[96] J. Su, D.V. Vargas, and K. Sakurai. One pixel attack for fooling deep neural networks. *Transactions on Evolutionary Computation (TEVC)*, 23(5):828–841, 2019.

[97] C. Szegedy, L. Wei, J. Yangqing, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015.

[98] H.K. Tan, C.W. Ngo, R. Hong, and T.S. Chua. Scalable detection of partial near-duplicate videos by visual-temporal consistency. In *ACM international conference on Multimedia (MM)*, pages 145–154, 2009.

[99] W. Tan, H. Guo, and R. Liu. A fast partial video copy detection using knn and global feature database. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2191–2199, 2022.

[100] N.A. Thacker, A.F. Clark, J.L. Barron, J.R. Beveridge, P. Courtney, W.R. Crum, V. Ramesh, and C. Clark. Performance characterization in computer vision: A guide to best practices. *Computer vision and image understanding (CVIU)*, 109(3):305–334, 2008.

[101] G. Tolias, R. Sicre, and H. Jégou. Particular object retrieval with integral max-pooling of cnn activations. In *International Conference on Learning Representations (ICLR)*, pages 1–12, 2016.

[102] Du. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *IEEE international conference on computer vision (ICCV)*, pages 4489–4497, 2015.

[103] K.H. Wang, C.C. Cheng, Y.L. Chen, Y. Song, and S.H. Lai. Attention-based deep metric learning for near-duplicate video retrieval. In *International Conference on Pattern Recognition (ICPR)*, pages 5360–5367, 2021.

[104] L. Wang, Y. Bao, H. Li, X. Fan, and Z. Luo. Compact cnn based video representation for efficient video copy detection. In *International conference on multimedia modeling (MMM)*, pages 576–587, 2017.

[105] L. Wang and D.C. He. Texture classification using texture spectrum. *Pattern recognition*, 23(8):905–910, 1990.

[106] Z. Wang, J. Chen, and S.C.H. Hoi. Deep learning for image super-resolution: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 43(10):3365–3387, 2021.

[107] X. Wu, A.G. Hauptmann, and C.W. Ngo. Practical elimination of near-duplicates from web video search. In *ACM international conference on Multimedia (MM)*, pages 218–227, 2007.

[108] X. Wu, C.W. Ngo, A.G. Hauptmann, and H.K. Tan. Real-time near-duplicate elimination for web video search with content and context. *IEEE Transactions on Multimedia*, 11(2):196–207, 2009.

[109] X. Yang, Q. Zhu, and K.T. Cheng. Near-duplicate detection for images and videos. In *ACM workshop on Large-scale multimedia retrieval and mining (LS-MMRM)*, pages 73–80, 2009.

[110] Z. Yang, M. Ding, T. Huang, Y. Cen, J. Song, B. Xu, Y. Dong, and J. Tang. Does negative sampling matter? a review with insights into its theory and applications. *Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 46(8):5692–5711, 2024.

[111] Y. Yoo, J. Shin, and Joonki P. Real-time continuous-scale image interpolation with directional smoothing. *Transactions on Smart Processing and Computing*, 3(3):128–134, 2014.

[112] O. Zachariadis, A. Teatini, N. Satpute, J. Gómez-Luna, O. Mutlu, O.J. Elle, and J. Olivares. Accelerating b-spline interpolation on gpus: Application to medical image registration. *Computer Methods and Programs in Biomedicine (CMPB)*, 193:105431, 2020.

[113] A. Zhang, Z.C. Lipton, M. Li, and A.J. Smola. *Dive into deep learning*. Cambridge University Press, 2023.

[114] C. Zhang, B. Hu, Y. Suo, Z. Zou, and Y. Ji. Large-scale video retrieval via deep local convolutional features. *Advances in Multimedia*, 2020:1687–5680, 2020.

[115] J.R. Zhang, J.Y. Ren, F. Chang, T.L. Wood, and J.R. Kender. Fast near-duplicate video retrieval via motion time series matching. In *IEEE International conference on multimedia and expo (ICME)*, pages 842–847. IEEE, 2012.

[116] X. Zhang and J. Gao. Measuring feature importance of convolutional neural networks. *IEEE Access*, 8:196062–196074, 2020.

[117] X. Zhang, Y. Xie, X. Luan, J. He, L. Zhang, and L. Wu. Video copy detection based on deep cnn features and graph-based sequence matching. *Wireless Personal Communications*, 103(1):401–416, 2018.

[118] Y. Zhang and X. Zhang. Effective Real-Scenario video copy detection. In *International Conference on Pattern Recognition (ICPR)*, pages 3951–3956. IEEE, 2016.

[119] G. Zhao, B. Zhang, M. Zhang, Y. Li, J. Liu, and J.R Wen. Star-gnn: Spatial-temporal video representation for content-based retrieval. In *International Conference on Multimedia and Expo (ICME)*, pages 01–06. IEEE, 2022.

[120] X. Zhao, C. Gu, H. Zhang, X. Yang, X. Liu, J. Tang, and H. Liu. Dear: Deep reinforcement learning for online advertising impression in recommender systems. In *Conference on Artificial Intelligence (AAAI)*, volume 35, pages 750–758, 2021.

[121] X. Zhou and L. Chen. Monitoring near duplicates over video streams. In *ACM international conference on multimedia*, pages 521–530, 2010.

[122] Z. Zhou, J. Chen, C.N. Yang, and X. Sun. Video Copy Detection Using Spatio-Temporal CNN Features. *IEEE Access*, 7:100658–100665, 2019.

[123] Y. Zhu, X. Huang, Q. Huang, and Q. Tian. Large-scale video copy retrieval with temporal-concentration sift. *Neurocomputing*, 187:83–91, 2016.

Van Hao LE

Evaluation de performance pour la détection à l'échelle de segments de vidéo

Résumé :
La détection de segments au sein des vidéos est un problème bien connu dans le domaine de la vision par ordinateur. L'objectif est de détecter une vidéo courte au sein d'une vidéo plus longue sous contraintes de déformation. La portion de vidéo détectée est alors qualifiée de segment. La détection de segments a de nombreuses applications comme pour la recherche de vidéos en ligne, la protection de la propriété intellectuelle, la détection de publicité TV, etc. Durant les dernières années, les systèmes à base de réseaux neuronaux convolutifs se sont imposés sur ce problème. Ces systèmes extraient des caractéristiques à partir des réseaux afin de décrire les fenêtres clés des segments. Ceci soulève de nouveaux besoins en termes d'évaluation de performance des systèmes à des fins de mise à l'échelle, contrôle des déformations et d'horodatage temporel des vidéos. Cette thèse propose deux contributions clés dans le domaine de l'évaluation de performance des systèmes de détection de segments. En première contribution, un nouveau protocole est proposé afin de générer des bases de test à l'échelle et de haute qualité, tant sur les données vidéos que celles de vérité terrain et d'horodatage. Ce protocole a été déployé afin de concevoir la base de données STVD. Cette base est la plus importante de la littérature à date avec 83 mille vidéos (ayant une durée de plus de 10 mille heures) composées de 1,6 million d'associations de segments (horodatés avec une précision à la fenêtre). La deuxième contribution de la thèse s'intéresse aux caractéristiques extraites des réseaux neuronaux convolutifs. Ces caractéristiques sont bien adaptées aux photos mais moins aux vidéos compte tenu de la problématique d'échelle, de redondance, de floue et de faible information visuelle. Afin d'étudier les limitations, une étude de caractérisation est présentée. Les résultats démontrent que la séparabilité n'est pas atteinte sur le problème de détection même si de fortes performances sont observées. Les différents réseaux présentent des performances proches, même si les architectures récentes comme ResNet50 se démarquent. Cette étude conclut à une carence du protocole d'usage. Afin de répondre à ce problème, un nouveau protocole est proposé pour la caractérisation des fenêtres clés des segments à partir des caractéristiques réseaux. Il permet une caractérisation fine des fenêtres clés et caractéristiques réseaux en termes de séparabilité, consistance et analyse quantitative de la qualité.
Mots clés : détection de segments · évaluation de performance · protocole · base de données · caractéristiques réseaux

Performance Evaluation for Scalable Partial Video Copy Detection

Abstract :
Partial Video Copy Detection (PVCD) is a well-known problem in the computer vision field. It is interested in detecting copied segments of short videos that have been transformed and embedded into longer videos. The PVCD has a wide range of real-world applications such as video retrieval, copyright protection, commercial detection and/or news verification in TV broadcasting. Over the past decade, deep learning algorithms, especially 2D Convolutional Neural Networks (CNNs), have become a key trend in designing PVCD systems. These systems extract 2D CNN features from frames for the retrieval and detection of partial video copies. This has opened new needs and challenges for the performance evaluation of PVCD systems in terms of scalability, control of degradations, and temporal accuracy. This PhD provides two main contributions dedicated to the performance evaluation of the PVCD systems. We propose a new protocol to design a scalable, noise-free PVCD dataset with temporal accuracy. This protocol is used to design a PVCD dataset called STVD. To the best of our knowledge, STVD is the largest public PVCD dataset containing nearly 83 thousands videos (having a total duration of 10 660 hours and containing 1 688 thousand pairs of partial video copies) and offering the temporal accuracy of frame-level. In the second contribution, we highlight the limitations of 2D CNN features for the PVCD problem. While these 2D CNN features are effective for image processing, their performance could be degraded when faced with the unique challenges of PVCD (e.g., scalability, near-duplicate detection, motion artifacts, solid-color frames). To assess these challenges, we provide a characterization of 2D CNN features for separability/consistency. Based on our STVD dataset, we give large-scale characterization of 9 common 2D CNNs, driven on 4.4 million feature vectors with 700 billion comparisons. From the results of the experiments, we find that the separability is not achieved on the detection problem even if strong scores are obtained. The different CNNs present competitive results. As a general trend, features extracted from recent 2D CNNs such as ResNet50 perform better. A correlation appears between the feature extraction methods and the 2D CNN architectures. Furthermore, the regular protocols for performance characterization are misleading for PVCD as they are bounded to the video level. To deal with this issue, we propose a new characterization protocol of key-frames with 2D CNN features. This protocol is based on a goodness criterion and a time series modelling. It provides a fine categorization of key-frames, a characterization of 2D CNN features for separability, consistency, and a quantitative analysis of the goodness of key-frames.
Keywords : partial video copy detection · performance evaluation · protocol · database · cnn features.