REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE BATNA 2

Faculté des Mathématiques et Informatique

Thèse de Doctorat

Présentée par :

Benoughidene Abdelhalim

En vue de l'obtention du diplôme de **DOCTORAT** en **:**

Filière : Informatique

Option : Technologies de l'information

Approche d'apprentissage pour l'analyse des Big Data

Soutenue publiquement le : 06/11/2023 Devant le jury composé de :

Pr. Hamouma MOUMEN	Professeur	*Université de Batna 2*	Président
Pr. Faiza TITOUNA	Professeur	*Université de Batna 2*	Rapporteur
Pr. Faiza BELALA	Professeur	*Université de Constantine 2*	Examinateur
Pr. Mohamed Nadjib KOUAHLA	Professeur	*Université de Guelma*	Examinateur
Pr. Saber BENHARZALLAH	Professeur	*Université de Batna 2*	Examinateur

Année Universitaire 2022/2023

PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

Ministry of Higher Education and Scientific Research

Batna 2 – University

Faculty of Mathematics and Computer Science

Computer Science department

DISSERTATION

In view of obtaining the Doctorate degree in 3rd cycle

Field: Computer Science

Specialty: Information technology

Presented by:

Benoughidene Abdelhalim

Entitled

Learning approach for Big Data analysis

Defended: 06/11/2023: Before the board of examiners composed of:

Pr. Hamouma MOUMEN	Professor	*University of Batna 2*	Chairman
Pr. Faiza TITOUNA	Professor	*University of Batna 2*	Supervisor
Pr. Faiza BELALA	Professor	*University of Constantine 2*	Examiner
Pr. Mohamed Nadjib KOUAHLA	Professor	*University of Guelma*	Examiner
Pr. Saber BENHARZALLAH	Professor	*University of Batna 2*	Examiner

Academic year: 2022/2023

Acknowledgement

This thesis marks the end of a long journey. It would not have been possible without God's help and the support of many people. Contrary to some beliefs, research is not a solitary endeavor but rather a collaborative one that requires guidance and support from colleagues, family, and friends.

First of all, i start by thanking almighty Allah for all the blessings he bestows upon us and for his help and guidance.

I would like to express my sincere appreciation to my thesis supervisor, Professor ***Titouna Faiza****, for his invaluable scientific support and guidance throughout the development of this thesis.*

My thanks also go to Professor ***Moumen Hamouma*** *for agreeing to chair my thesis jury, and to all members of the jury, Pr.* ***Kouahla Mouhamed Nadjib*** *from the University of 8 May 1945 - Guelma, Pr.* ***Belala Faiza*** *from the University of Abdelhamid Mehri-Constantine2, and Pr.* ***Benharzallah Saber*** *from the University of Mostefa Ben Boulaïd-Batna2, for their time and effort in examining my work.*

A big thank you to all the teachers in the computer science department at Batna 2 University for their high-quality training, as well as to my fellow doctoral students.

Finally, thank you to my mother and father for all the sacrifices they have made to allow me to pursue my studies under the best possible conditions and for never ceasing to encourage me throughout my years of study. To my brothers Hayat, Amar and Ali and my dear friends Safwane, Abdellah and Adel, and all those who from afar or near have given me their encouragement.

I will conclude by thanking you with all my heart.

Abstract

The development of information technology has led to the big data revolution, with the amount of data produced increasing at a high rate. Video data is a significant component of big data, and the concept of automatically analyzing this rapidly increasing video content has become a popular research topic. In such a scenario, video automatic analysis uses a new generation of information technologies, such as artificial intelligence (machine learning), which help transform traditional video analysis to be more efficient and convenient.

Video summary (VS) is now one of the primary areas of study in video analysis. Despite the use of big data-driven models, producing accurate video summaries in an efficient and effective manner remains a challenging task. The most effective and efficient way to transform lengthy, unstructured videos into structured, condensed, understandable, and useful information is through the use of video summaries. The primary goal of summarizing a video is to break it down into shots and select key frames from each shot that best capture the essence and flow of the entire video.

The goal of this thesis is to improve the performance of video summarization systems by enhancing the quality of analytical techniques. To achieve this goal, we proposed two contributions. First, we suggested a shot boundary detection (SBD) method to adapt key shots and exploit its potential in video summaries. This is the first step in the video summarization process, and the results have a significant impact on the quality of the final summary. The main idea behind SBD is to extract features from video frames and then identify the boundaries between shots based on the differences in the features. Second, we focused on improving video summaries using unsupervised machine learning techniques (DBSCAN) and genetic algorithms (GA) to optimize DBSCAN hyperparameters. We validated the proposed methods and results obtained through extensive comparative analysis using datasets.

Keywords: *Multimedia Big Data, Machine Learning, Deep Learning, Video Analysis, Shot Boundary Detection(SBD), Video Summarization.*

Résumé

Le développement des technologies de l'information a conduit à la révolution du big data, la quantité de données produites augmentant à un rythme élevé. Une grande partie des big data se présente sous la forme d'informations vidéo, et le concept d'analyse automatique de cette croissance exponentielle du contenu vidéo est devenu un domaine de recherche populaire. Dans un tel scénario, l'analyse automatique des vidéos utilise une nouvelle génération de technologies de l'information, telles que l'intelligence artificielle (apprentissage automatique), qui aident à transformer l'analyse vidéo traditionnelle pour la rendre plus efficace et plus pratique.

Le résumé vidéo est devenu l'un des principaux sujets de recherche en matière d'analyse vidéo. Toutefois, l'élaboration efficace de résumés vidéo précis reste un problème difficile à résoudre dans le cadre des modèles axés sur les grandes données. En plus, les résumés vidéo sont la solution la meilleure et la plus efficace pour convertir des vidéos volumineuses et amorphes en informations structurées, concises, claires et significatives. La tâche principale du résumé d'une vidéo est de segmenter la vidéo originale en plans et en extraire les images clés, qui seront les plus représentatives et les plus concises de de toute la vidéo.

L'objectif de cette thèse est d'améliorer les performances des systèmes de synthèse vidéo en améliorant la qualité des techniques analytiques. Pour atteindre cet objectif, nous avons proposé deux contributions. Tout d'abord, nous avons suggéré une méthode de détection des limites de plans (SBD) pour adapter les plans clés et exploiter son potentiel dans les résumés vidéo. C'est le premier processus dans la synthèse vidéo, et sa sortie affecte significativement les processus ultérieurs. L'idée principale derrière la limite de plan vidéo est d'extraire les caractéristiques des images vidéo, puis de détecter le type de plan en fonction des différences de caractéristiques. Deuxièmement, nous nous sommes concentrés sur l'amélioration des résumés vidéo en utilisant des techniques d'apprentissage automatique non supervisé (DBSCAN) et des algorithmes génétiques (AG) pour optimiser les hyperparamètres DBSCAN. Nous avons validé les méthodes proposées et les résultats obtenus par une analyse comparative approfondie à l'aide d'ensembles de données.

Mots clés: *Big Data Multimédia, Apprentissage Automatique, Apprentissage Profond, Analyse Vidéo, Détection Des Limites Des Plans, Résumé Vidéo.*

ملخص

مع تطور تكنولوجيا المعلومات ، أصبحت كمية البيانات المنتجة هائلة وتتزايد بمعدل مرتفع ، مما أدى إلى ثورة البيانات الضخمة. يأتي جزء كبير من البيانات الضخمة في شكل معلومات فيديو ، وأصبح مفهوم التحليل التلقائي لهذا النمو الهائل في محتوى الفيديو مجال بحث شائع. في مثل هذا السيناريو ، يستخدم التحليل التلقائي للفيديو جيلًا جديدًا من تقنيات المعلومات ، مثل الذكاء الاصطناعي (التعلم الآلي). تساعد هذه التقنيات الجديدة في تحويل تحليل الفيديو التقليدي ليكون أكثر كفاءة وملاءمة.

أصبح ملخص الفيديو (VS) أحد الموضوعات البحثية الرئيسية في تحليل الفيديو. ومع ذلك ، تظل كيفية عمل ملخصات فيديو دقيقة بكفاءة وفعالية مشكلة صعبة في النماذج التي تعتمد على البيانات الضخمة. بالإضافة إلى ذلك ، تعد ملخصات الفيديو الحل الأفضل والأكثر فاعلية لتحويل مقاطع الفيديو الكبيرة غير المتبلورة إلى معلومات منظمة وموجزة وواضحة وذات مغزى. تتمثل المهمة الرئيسية لتلخيص مقطع فيديو في تقسيم الفيديو الأصلي إلى لقطات واستخراج الإطارات الرئيسية من اللقطات ، والتي ستكون الأكثر تمثيلاً وإيجازًا للفيديو بأكمله.

في هذه الرسالة ، نهتم بتحسين أداء أنظمة تلخيص الفيديو من خلال زيادة جودة التقنيات التحليلية. ولهذه الغاية ، تم اقتراح مساهمتين. أولاً ، اقترحنا طريقة اكتشاف حدود اللقطة (SBD) من أجل استغلال إمكاناتها في ملخص الفيديو من خلال تكييف اللقطات الرئيسية. إنها العملية الأولى في تلخيص الفيديو ، ويؤثر ناتجها بشكل كبير على العمليات اللاحقة. تتمثل الفكرة الرئيسية لحدود تصوير الفيديو في استخراج ميزات إطارات الفيديو ، ثم اكتشاف نوع اللقطة وفقًا للاختلافات في الميزات. ثانيًا ، ركزنا على تحسين ملخصات الفيديو باستخدام تقنيات التعلم الآلي غير الخاضعة للرقابة (DBSCAN) والخوارزميات الجينية (GA) لتحسين معلمات DBSCAN الفائقة. تم التحقق من صحة الطرق المقترحة والنتائج التي تم الحصول عليها من خلال تحليل مقارن مكثف باستخدام مجموعات البيانات.

كلمات مفتاحية: *البيانات الضخمة للوسائط المتعددة ، التعلم الآلي ، التعلم العميق ، تحليل الفيديو ، اكتشاف حدود اللقطة ، تلخيص الفيديو.*

Contents

List of Figures

List of Tables

Abbreviations and Acronyms

< **AE** > < AutoEncoder >

< **ART** > < Adaptive resonance theory >

< **ASLBPHD** > < Absolute Sum Local Binary Pattern Histogram Difference >

< **AUC** > < Area Under Curve >

< **CBAC** > < content-based adaptive clustering >

< **CBBH** > < Concatenated Block Based Histograms >

< **CBVIR** > < Content-Based Video Indexing and Retrieval >

< **CBVR** > < content based video retrieval >

< **CNN** > < Convolutional Neural Network >

< **CT** > < Cut Transition >

< **CUS** > < Comparison of User Summaries >

< **DBSCAN** > < density-based spatial clustering of applications with noise >

< **DENCLUE** > < DENsity-based CLUstEring algorithm >

< **DNN** > < Deep Neural Networks >

< **ECR** > < edge change ratio >

< **FC** > < Fully Connected >

< **FCM** > < Fuzzy C-Means >

< **GA** > < Genetic Algorithm >

< **GFS** > < Google File System >

< **GRU** > < Gated recurrent unit >

< **GT** > < Gradual Transition >

< **HDFS** > < Hadoop Distributed File System >

< **HOG** > < Histogram of Gradient >

< **HSV** > < Hue Saturation Value >

< **IoT** > < Internet of things >

< **IT** > < Information technology >

< **LSTM** > < Long-Short-Term-Memory >

< **MPEG** > < Moving Picture Experts Group >

< **NLP** > < natural language processing >

< **NoSQL** > < not only SQL >

< **OVP** > < Open Video Project >

< **PCA** > < Principal component analysis >

< **RAI** > < Rural Access Index >

< **RDBMS** > < relational database management system >

< **ReLU** > < Rectified Linear Lnit >

< **RNN** > < Recurrent Neural Networks >

< **ROC** > < Receiver Operating Characteristics >

< **SAE** > < stacked autoencoder >

< **SBD** > < shot boundary detection >

< **SIFT** > < scale-invariant feature transform >

< **small and medium enterprises** > < >

< **SMOTE** > < Synthetic Minority Over-sampling Technique >

< **SURF** > < method Speeded Up Robust Features >

< **SVD** > < singular value decomposition >

< **SVM** > < Support Vector Machine >

< **TCR** > < temporal collaborative representation >

< **TD** > < temporal difference >

< **TRECVID** > < Text Retrieval Conference Video >

< **TSSBD** > < two-stage method for shot boundary detection >

< **UC** > < university of california >

< **Video SUMMarization** > < VSUMM >

< **VS** > < Video summary >

< **VSBD** > < Video shot boundary detection >

< **YT dataset** > < YouTube dataset >

General Introduction

Motivations, problematic and objectives

The big data revolution in multimedia is driven by the enormous and rapidly increasing amount of multimedia data available every day, made possible by the vast development of multimedia technology and the availability of multimedia sources. The exponential growth of daily video data generation due to the fast evolution of multimedia technologies has led to voluminous and redundant data. Although video frames most vividly depict real-world scenarios, it is always challenging to choose the correct video sequence or specific section of the video from a large collection of video data. Analyzing and summarizing videos that allow users to quickly gain an understanding of the overall content of video footage have become critical and indispensable [1].

In the 3Vs (volume, variety, and velocity) of big data, videos can be considered big and varied because of their size and content. For this reason, it is important to use data analytics tools to identify the most important parts of a video and summarize them in minutes or seconds. This reduces the volume of data while preserving key information.

Video analysis is the process of analyzing video content to extract meaningful information. This can include identifying objects or people in the video, tracking their movements, and analyzing their behavior. Currently, most video analysis tools use a set of keyframes to provide a content summary of a video sequence. One of the key components of video analysis is creating a video summary. The process involves segmenting the video into shots and extracting key frames that represent the entire video [2].

The first step in video summarization is Shot Boundary Detection (SBD), which significantly affects subsequent processes. SBD extracts features from video frames and detects shot types based on feature differences. Cut Transition (CT) and Gradual Transition (GT) are the two distinct methods for detecting the boundaries of a video shot. [3].

A video summary is a short version of a longer video that captures the most important parts

of the video through the selection of the most important and pertinent semantic content within the video. There are two types of video summaries: static and dynamic. Static video summaries are created by selecting keyframes from the video and arranging them in a sequence. Dynamic video summaries are created by selecting key-frames and then adding transitions between them to create a more fluid summary [4]. In this context, several questions arise:

1. How can we analyze a big video data to extract relevant information?
2. What are the most efficient techniques for analyzing a big video data?
3. One common technique for video analysis is video summarization. How can we improve this technique to obtain a short and representative summary?

In summary, the main objective of this work is to develop an approach to video data analysis by proposing a new method of static video summarization. The latter should reduce the volume of data while preserving the key information of the videos. This static video summarization method will be based on shot boundary detection, where the latter is based on a proposed similarity model. We can condense our contributions to the research topic in the following points:

- A new method for detecting video shot boundaries has been proposed. This method uses a similarity model that extracts spatial features from the pre-trained InceptionV3 model and temporal features from LSTM. The proposed similarity model is then used to detect cut transition boundaries using a shot boundary detection algorithm.
- The proposal of a static video summarization that uses as preprocessing step a shot boundary detection method and exploits DBSCAN clustering for the extraction of keyframes.
- The clustering model is adopted by a genetic algorithm which aims to optimize the hyper-parameters of DBSCAN instead to pre-tuning them a priori by the user. For the reason that every video has its own number of key-frames.
- Validation of shot boundary detection and video summarization methods on a set of datasets.

Thesis structure

In this thesis, we examine two important categories of content-based video retrieval (CBVR) research. The first category is video shot boundary detection (SBD), which includes cuts and

gradual transitions. We provide a list of different SBD techniques. The second category is video summarizing (VS), which involves summarizing a video into a shorter version while retaining the most important information. We go over various techniques and how they are categorized in the literature. And to cover all proposals, this thesis is organized into two parts: **The first part** is the state of the art in the field. It is composed of four chapters, described as follows:

The chapter 1: is dedicated to various definitions of big data and its 3V characteristics, as well as the sources of big data and its main types. This chapter also presents the tools and technologies used to store and process big data. Additionally, big data analytics approaches were presented, including multimedia analytics such as big video analytics based on machine learning.

The chapter 2: provides an overview of the basic concepts related to machine learning, including different definitions of machine learning and how it involves the development of algorithms that can learn from data to make predictions or decisions. We have also outlined the series of steps typically followed to build a machine learning model and introduced the different categories of machine learning. Additionally, we have provided a theoretical background on deep learning and transfer learning with a detailed study of different models.

The chapter 3: provides a comprehensive survey of shot boundary detection (SBD) algorithms and discusses video definitions, transition types, and hierarchies. Each approach's benefits and drawbacks are thoroughly examined, and problems are put forth. Additionally, the chapter focuses on machine learning innovations like deep learning methods for SBD that might be employed as fresh ways in the future.

The chapter 4: gives an overview of the existing approaches for video summary, it discusses the different type of the video summary, their difficulties and the different limitations. Taking into consideration the advantages and drawbacks of several approaches.

The second part of the thesis contains our contributions. It is composed of two chapters.

The chapter 5: describes our solution for shot boundary detection (SBD) via a deep network that distinguishes input frames. The model that is used to predict shot boundaries is called the similarity model. It is learned using a Siamese recurrent architecture and it uses a pre-trained InceptionV3 model and an LSTM model to extract features. The model's final layer then fuses and categorizes these features. The proposed model focuses on cut transition detection and makes use of deep learning methods that have proven effective in tasks requiring image similarity comparison. The performance of the VSBD approach is validated on several databases and compared to other SBD methods.

The chapter 6: presents our system for static video summarization by extracting repre-

sentative key-frames based on shot boundary detection (SBD) proposed. This system relies on feature extraction with a pre-trained model (Inception-V3), then applies PCA for dimension reduction and the DBSCAN clustering algorithm to identify the best key-frames. To improve the efficiency of this system, we present a genetic algorithm that optimizes the DBSCAN hyper-parameters to extract optimal key-frames that represent the video summary. Different experiments have shown the good performance of our system. the thesis structure.

We close the thesis with a general conclusion which highlights our contributions and outlines the prospects for future research. Figure 1 gives an overview of thesis structure.

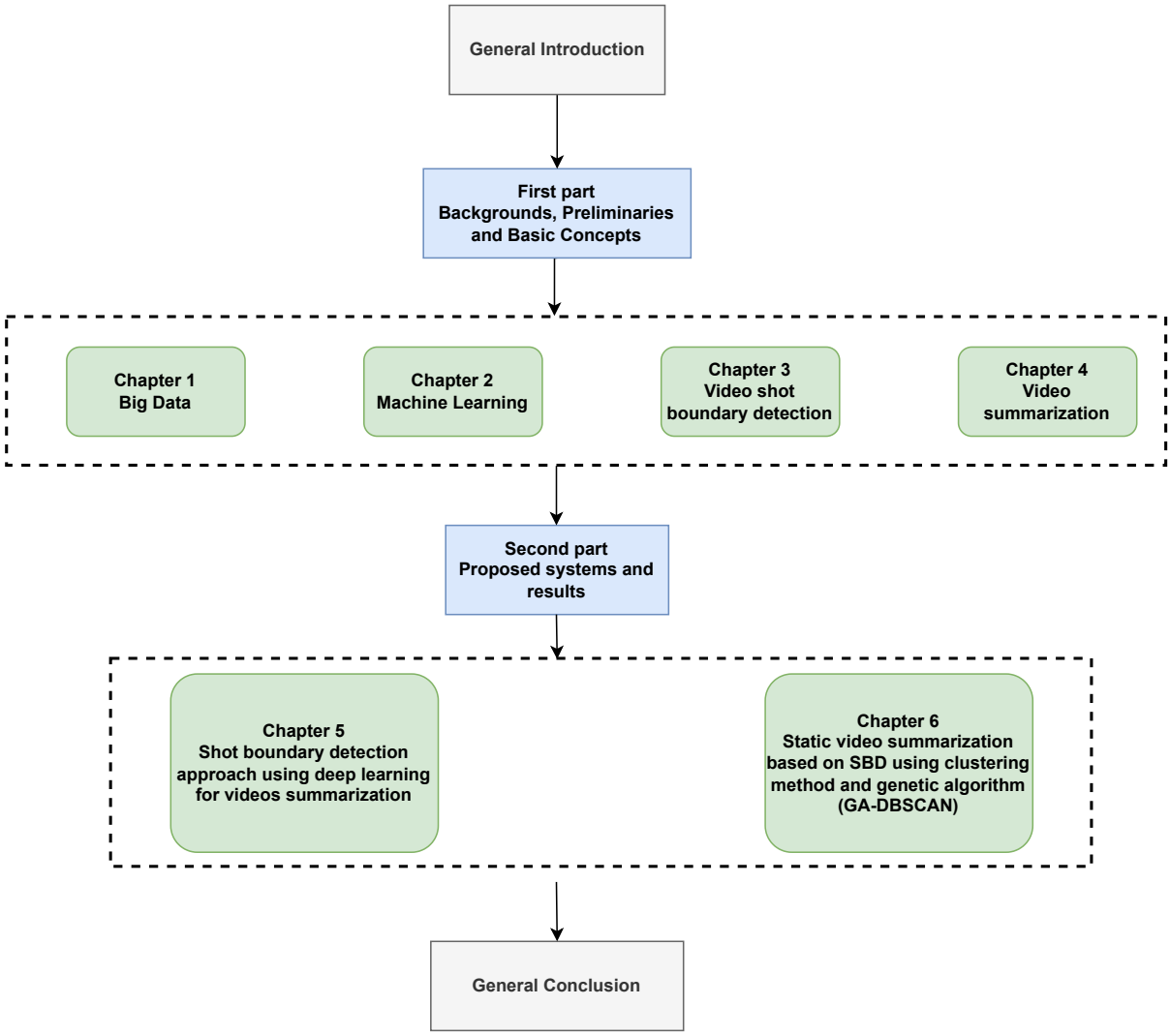


Figure 1: Thesis structure

List of publications

Journal Papers

1. Benoughidene Abdelhalim, Titouna Faiza. A novel method for video shot boundary detection using CNN-LSTM approach.In International Journal of Multimedia Information Retrieval, 2022, vol. 11, no 4, p. 653-667. https://doi.org/10.1007/s13735-022-00251-8

Conference Papers

1. Benoughidene Abdelhalim, Titouna Faiza, SERIDI, H., et al. «Shot Boundary Detection: Fundamental Concepts and Survey». In:CITSC. 2019. p. 34-40.

2. Benoughidene Abdelhalim, Titouna Faiza. «Hybrid Approach For Learning Similarity of Images». In Algerian Journal of Engineering Architecture and Urbanism Vol. 5 Nr. 2 2021 p. 319-325.

Part I

State of the Art

Chapter 1

Backgrounds of Big Data

Contents

1.1 Introduction

Big data is a term used to describe large and complex data sets that traditional data management systems cannot handle. In this chapter, we will explore the basics of big data. Section 1 will cover the various definitions of big data and its 3Vs characteristics, as well as the sources of big data such as social media and the Internet of Things (IoT). We will also discuss the different types of big data. Section 2 will focus on the tools and technologies used to store and process big data, including the Hadoop ecosystem. In section 3, we will delve into big data analytics. Finally, we will examine the relationship between big data and machine learning.

1.2 Definitions of Big Data

"Big Data" is a phrase used to refer to a lot of data. The literature contains a number of definitions of big data from various angles. For instance, in [5] big data is defined as vast amounts of data with a high velocity or variety that need to be captured, stored, and analyzed using novel technologies and methods. It is also utilized to support and optimize processes, offer information and discovery, and enhance decision-making. According to [6], "Big data" is defined as data whose volume, velocity, or variety is too great to be processed using traditional methods, necessitating a scalable architecture for effective data manipulation, storage, and analysis. On the other hand, according to [7], the phrase "Big Data" refers to the storing and processing of large and/or complex data sets using a variety of techniques, such as NoSQL, MapReduce, and machine learning. Other researchers, like [8], define big data as the enormous volume of data collected from a wide variety of sources that is too large, raw, or unstructured to be analyzed by traditional database techniques.

1.2.1 Sources of Big Data

Modern technology allows us to collect large quantities of diverse data from sources such as sensors and videos. Big data comes from various sources, including operational and business data from enterprises, logistical and sensory data from the Internet of Things, location and human interaction data from the online world, and scientific research data [9]. Figure 1.1 [10] shows some sources of big data.

- **Media as a Big Data source:** The development of social networks has led to millions or even billions of daily audio and visual uploads. Examples include videos on YouTube, music recordings on SoundCloud, and photos on Instagram. The volume of this media continues to grow rapidly[1];

- **Cloud computing as a Big Data source:** By storing their data on the cloud, businesses have moved beyond traditional data sources. They can store both structured and unstructured data and quickly retrieve real-time data. Cloud computing is scalable, adaptable, cost-effective, and versatile. Big data can be delivered and stored via networks and computers on public or private clouds[2];

- **Web as a Big Data source:** The public web constitutes big data that is widespread

[1] https://subscription.packtpub.com/book/big-data-and-business-intelligence/9781783554393/1/ch01lvl1sec13/sources-of-big-data

and easily accessible. Individuals and businesses can access data on the Internet or Web, and websites like Wikipedia provide quick and free access to information. The size of the Web makes it useful for startups and SMEs in many different ways[2];

- **Internet of things (IoT) as a Big Data source:** The Internet of Things (IoT) is a significant source of big data generated from sensors connected to electronic devices. In smart cities based on IoT, big data can come from devices such as computers and smartphones, traffic and transportation systems, medical care, public departments, and household appliances [9];

- **Databases as a Big Data source:** Companies are combining traditional and modern databases to achieve relevant megadata with little cost and IT infrastructure, creating a hybrid data architecture. Insights extracted from these datasets can boost corporate earnings. Popular databases use data sources such as MS Access, DB2, Oracle, SQL, and Amazon Simple[2].

Figure 1.1: Big data sources

1.2.2 Types of Big Data

Big data comes from many sources and formats and can be categorized in many ways. One fundamental difference is between structured and unstructured data. Unstructured data includes text files, geographical data, multimedia files, and financial data, which are expected to make up 80-90% of future data growth according to [11]. Big Data can be broadly classified into three categories.

[2] https://www.allerin.com/blog/top-5-sources-of-big-data

- **Structured data:** It is data that is stored in a format that specifies its structure. It is often easy to store, process, analyze and query structured data with conventional analysis tools and software. Examples of typical data files include spreadsheets, transaction data and relational database management systems (RDBMS). Microsoft Excel is an example of a great tool for working with organized data. Knowing the appropriate combinations of formulas and functions will make it simple to handle millions of rows with numbers and titles[12]. According to experts, between 5 and 10 percent of all data worldwide is structured [13]. Figure 1.2, displays a few illustrations of structured data.

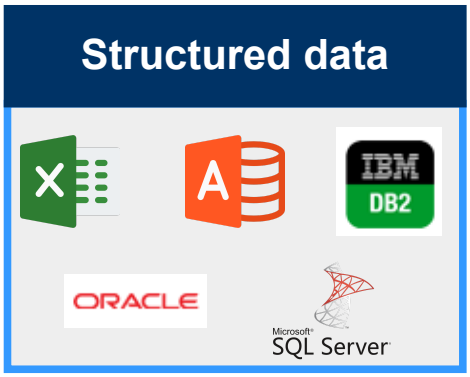


Figure 1.2: Structured data

- **Semi-structured data:** Semi-structured data is data that does not fit neatly into a database or data table. It may have tags or other distinguishing marks that apply hierarchical hierarchies and segregate items, making it self-descriptive. According to [13], semi-structured data may be managed and converted into structured data using XML and JSON. Some instances of semi-structured data are shown in Figure 1.3.

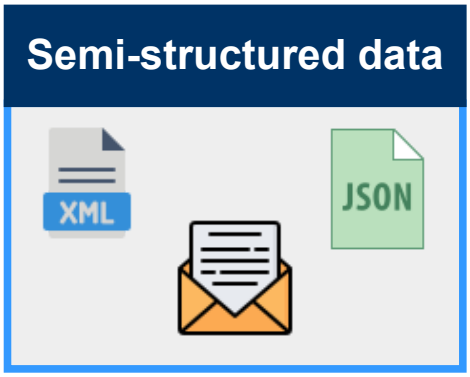


Figure 1.3: Semi-structured data

- **Unstructured data:** Unstructured data is difficult to classify and is not naturally structured, making it the opposite of structured data. It accounts for 90% of all data in the

world according to [13]. Unstructured data cannot be analyzed using conventional tools and services. Figure 1.4 shows some examples of unstructured data.

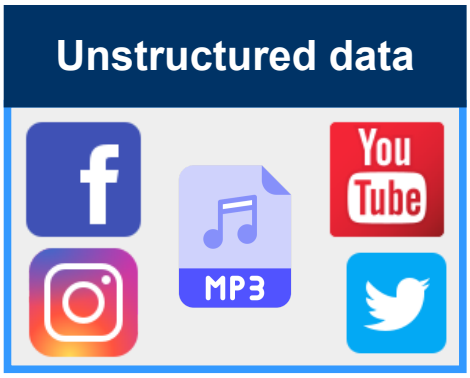


Figure 1.4: Unstructured data

1.2.3 Characteristics of Big Data

The amount of data created each day has grown dramatically since the invention of the Internet. The velocity and variety of data have also increased significantly. In 2001, Gartner suggested a three-dimensional view of the challenges and opportunities driven by data growth, known as the three V's: Volume, Variety, and Velocity according to [9]. Figure 1.5 illustrates these defining characteristics of Big Data.

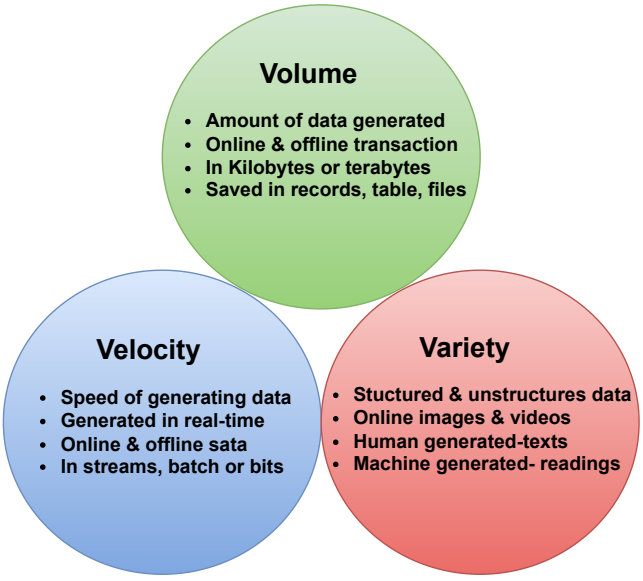


Figure 1.5: The 3V's of Big Data

- **Volume:** Volume refers to the quantity of data produced by websites, portals, and online programs, especially social networks that provide streaming data. In 2000, 800,000 petabytes of data were stored globally according to [14]. By 2020, this is predicted to

reach 30-40 zettabytes. For example, Facebook generates over 500 terabytes of data daily according to [15].

- **Variety:** Variety in Big Data refers to the range of structured and unstructured data generated by humans or machines, including documents, contracts, sensor data, social media, medical records, and emails. However, much of this data is unstructured or has a complex structure that is difficult to represent in structured or semi-structured databases according to [16].

- **Velocity:** Velocity in Big Data refers to the speed of data generation and processing. It also refers to the increasing speed of data generation, processing, and use according to [17].

1.3 Big Data Technologies

Big Data issues such as volume and velocity can be mitigated using appropriate technologies for parallel and distributed computing. These technologies provide cost-effective computing power and exceptional efficiency in processing large amounts of data in real-time [18]. This is achieved through a successful abstraction layer based on an acceptable programming paradigm [19]. The goal is to improve performance, facilitate innovation in business model products and services, and provide decision support. In the following section, we briefly discuss the common data processing platforms.

1.3.1 Hadoop and MapReduce

Advancements in computing technology have made it possible to manage immense volumes of data without the need for supercomputers or expensive equipment. Popular big data technologies and tools such as Hadoop, MapReduce, and Spark have revolutionized data management due to their ability to quickly and cheaply analyze massive amounts of data [20]. The following section discusses Hadoop and MapReduce in more detail

- **High-availability distributed object-oriented platform (Hadoop):** Hadoop is a software framework developed by Apache that is based on Google's Big Table and MapReduce technologies. It was created by Doug Cutting and named after his son's toy elephant.[21]. A Hadoop framework typically consists of two parts: the Hadoop distributed file system

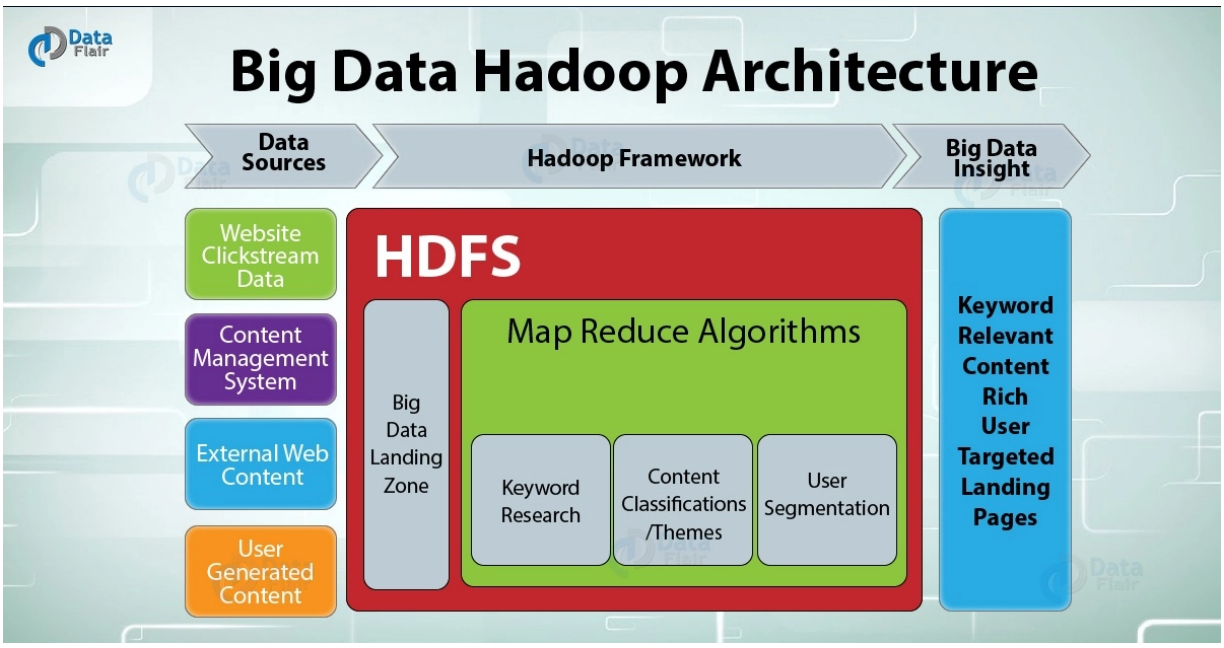


Figure 1.6: Hadoop architecture

and the MapReduce programming model as shown in Figure 1.6 [22]. The Hadoop distributed file system is a computer network environment designed to effectively manage Big Data applications, while the MapReduce programming model provides libraries and a graphical user interface for effective communication with the Hadoop distributed file system [23]. These are explained in more detail in the following subsections.

1. **Hadoop Distributed File System (HDFS):** The Hadoop Distributed File System (HDFS) is a distributed, scalable, and portable file system inspired by the Google File System (GFS). It allows access to data dispersed across multiple devices and can run on inexpensive machines with high fault tolerance. HDFS is primarily used for persistent data storage that offers quick access to application data [24]. HDFS nodes are divided into two categories: data nodes and name nodes. The name node acts as a regulator between the client and the data node, directing the client to the specific data node that contains the requested data. Data is stored in blocks of files that are duplicated among various data nodes [25].

2. **MapReduce Programming Model:** MapReduce is a programming model and implementation for processing and generating large datasets across multiple computers in a Hadoop cluster. Each server in the cluster has a collection of low-cost internal disk drives. MapReduce improves performance by distributing tasks to the servers where the data is stored. The cluster nodes are used to schedule data processing. The underlying runtime system automatically parallelizes the computation over large

clusters of computers, handles machine failures, and schedules inter-machine communication to optimize network and disk usage once the user specifies the calculation in terms of a map and reduce function [26, 27].

1.3.2 Spark

Spark is a fast computing cluster developed with the help of nearly 250 developers from 50 companies at UC Berkeley's AMPLab. It was originally developed as a research project for the Berkeley Lab RAD (later renamed AMPLab) in 2009. After using Hadoop MapReduce in the lab, researchers found it to be ineffective for interactive and iterative data processing tasks. As a result, Spark was created with the goal of being fast for interactive queries and iterative algorithms, incorporating features such as efficient fault recovery and in-memory storage support [28].

Apache Spark is an open-source processing framework that builds upon the MapReduce model to effectively support various types of computations, including iterative processing, interactive queries, and stream processing. Spark is designed for speed, ease of use, and the ability to handle large and diverse data sets such as text and graph data [29]. The architecture of Spark is shown in the following figure 1.7 [30].

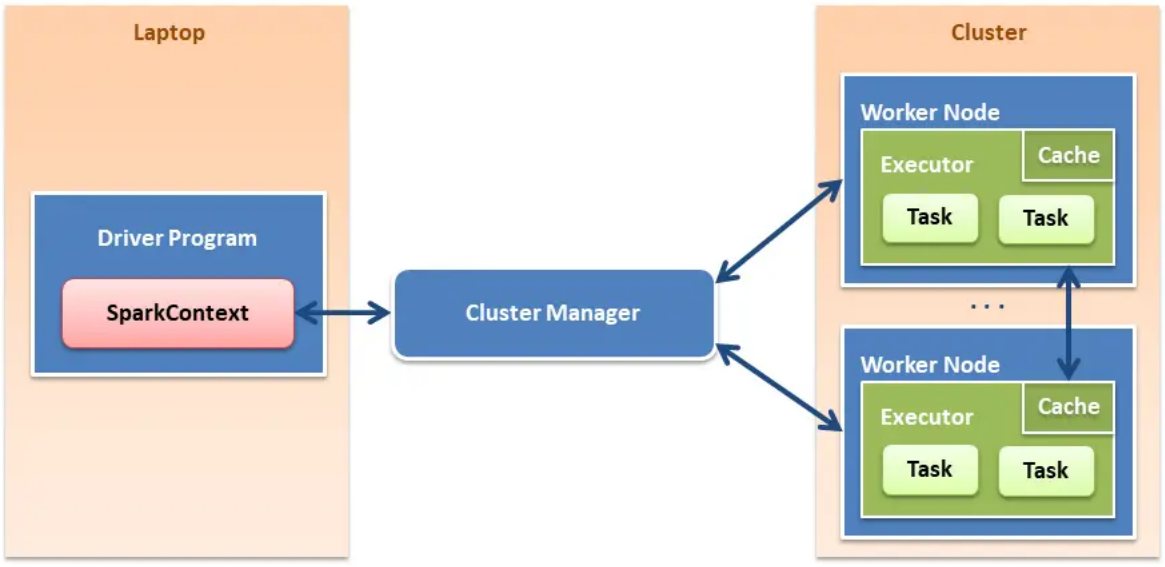


Figure 1.7: Spark architecture

1.4 Big Data Analytics

Big data analytics has become a scientific and statistical tool to extract the information needed to gain knowledge. Traditional techniques are unable to handle this type of data. Big data analytics aims to collect, store, and analyze data to support evidence-based decision-making in the real world. According to the figure 1.8 [31], it can be broadly divided into three types: descriptive analysis, predictive analysis, and prescriptive analysis [32].

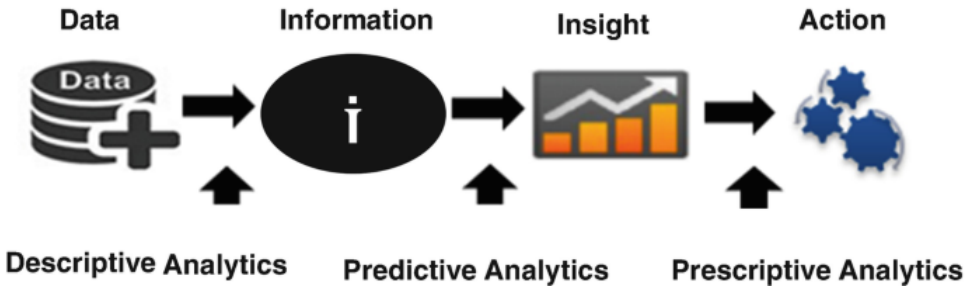


Figure 1.8: Types of analytics techniques

- **Descriptive Analytics:** Provides details on what occurred. In this method, fresh insights are created based on historical data utilizing statistical descriptions like statistic summary, correlations, or sampling and clustering techniques like K-means [33].
- **Predictive Analytics:** Predicts the future probabilities and trends using diverse feature modeling methods and predictive algorithms such as Decision Tree or Linear SVM (Support Vector Machine). It provides information on what is likely happen in the future and what actions can be taken [33].
- **Prescriptive Analytics:** Prescriptive analysis involves suggesting various actions and includes experimental design and optimization techniques. Experimental design is used to explain why a phenomenon occurs by conducting experiments in which independent variables are manipulated and extraneous variables are controlled. This allows conclusions to be drawn about the actions that the decision-maker should take [33].

1.4.1 Big Data Life Cycle

Big Data follows the life-cycle shown in the figure below 1.9 from data generation to data visualization through data preparation, data storage, and data analytics. The life-cycle of Big Data as shown in figure 1.9 typically involves several stages, including data generation, data preparation, data storage, data analysis, data visualization. These stages are often described as a cycle because the insights gained from one data project can inform the next.

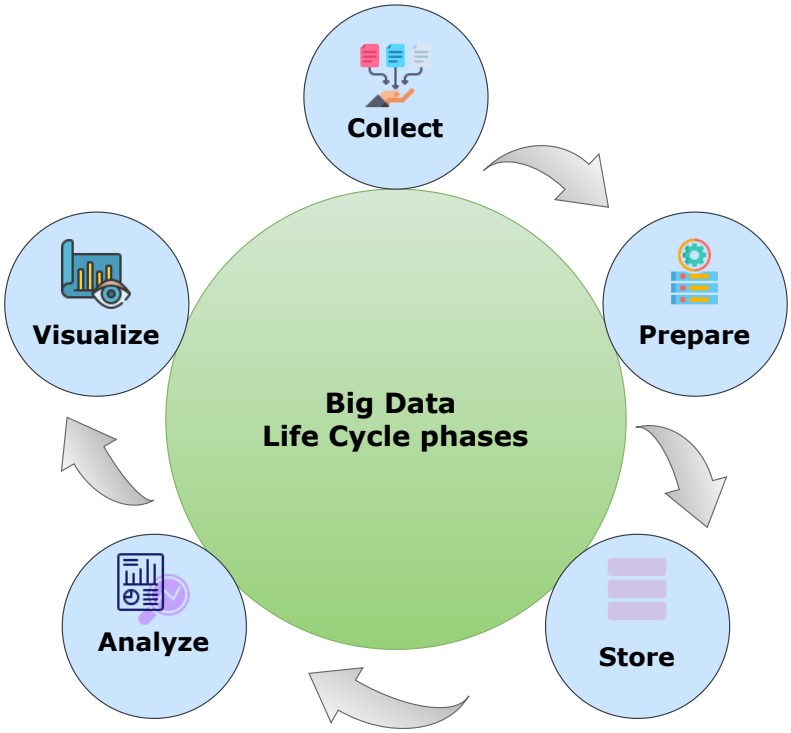


Figure 1.9: Big Data life cycle

- **Data Generation:** It is the most crucial part of the big data cycle, and the internet provides an abundance of data sources such as cameras, videos, and click streams [34].
- **Data Preparation:** It is the process of cleaning, transforming, and structuring data so that it can be analyzed effectively. The goal of data preparation is to create a dataset that is accurate, consistent, and complete. [35].
- **Data Storage:** Data storage is the process of saving data in a database using memory. It is an essential part of computing. Without data storage, computers would not be able to save any information [33].
- **Data Analytics:** Data analysis is the scientific and statistical tool for analyzing raw data to renovate the information needed for knowledge acquisition using qualitative and quantitative techniques such as regression, classification, and clustering [36].
- **Data Visualization:** Data visualization is a technique that allows users to interact with the results of data analysis while learning more about the raw data that has been gathered from diverse sources. It places the significance of the data in a visual perspective[37].

1.4.2 Big Data in Multimedia Analytics

Multimedia big data, which mainly consists of images, audio, and video, is growing rapidly due to the increasing use of cameras, cell phones, and social media. Sites such as Yahoo, Flickr, iCloud, YouTube, and social networks like Facebook, Instagram, and Twitter are valuable sources of big multimedia data [38]. The widespread use of multimedia data and the ease of access to multimedia sources have led to a big data revolution in multimedia management systems. Vast amounts of multimedia data are produced and analyzed to extract practical knowledge and semantic understanding. However, due to the heterogeneous nature of multimedia data and its richer information content compared to structured or textual data, information extraction faces significant challenges [39]. Research in multimedia big data analysis spans many disciplines and includes topics such as multimedia summarization, annotation, indexing and retrieval, suggestion, and event detection.

Video Big Data Analysis

In the era of multimedia big data, video data is a major form of unstructured data and an important area for big data analytics. Storing and retrieving video data presents new challenges, and despite the availability of applications and services for editing and processing video data before storage, efficiency in content-based video retrieval has not yet been achieved due to the large size of video data and the complexity of processing and analysis methods. Various techniques have been proposed for effective retrieval of video content, such as video summarization [40]. Video summarization involves interpreting the most important or representative sequence of video content and can be static or dynamic. Static video summarization methods use key frames to represent a video, while dynamic methods use a series of video frames and other measures to make the final summarization look more natural [9]. Multimedia summarization typically involves two procedures: structural analysis and feature extraction. Structural analysis segments a video into semantic structural elements such as lens boundaries, key frames, and scenes. Feature extraction involves further mining the features of key frames, objects, texts, and movements based on the results of structural analysis [41].

1.5 Big Data And Machine Learning

Traditional analytical methods are not sufficient to fully extract the value of big data. Machine learning has revolutionized the way we handle big data by using algorithms to analyze it and

unlock its potential. This technology allows us to develop statistical models to evaluate data sets and draw conclusions or make predictions based on identified patterns, without human intervention [3].

Machine learning and big data are complementary concepts that can produce remarkable results when used together. Effective management of big data's various characteristics can enhance the performance and accuracy of machine learning models. By providing analytics teams with large amounts of relevant and high-quality data, big data management techniques can improve the success of machine learning model construction [4].

Many companies have already realized the benefits of using machine learning to enhance their big data analytics. For instance, Netflix uses machine learning algorithms to understand each user's viewing preferences and make better recommendations, keeping them engaged with the platform. Google also employs machine learning to provide a more personalized experience for its users, including predictive text in emails and optimized directions on Google Maps [4].

Big data is essential for the existence of artificial intelligence and machine learning. It provides the necessary information for AI to understand and learn from human behavior. Big data enables faster learning and automation of data analysis. As a machine learning system processes more data, it learns more and becomes more accurate [3].

1.6 Conclusion

In this chapter, we have presented the various definitions of big data and its 3Vs characteristics, as well as the sources of big data and its main types; structured, semi-structured, or unstructured, and can come in various formats such as text, images, and videos. Then we presented the tools and technologies used to store and process big data, including the Hadoop ecosystem. In addition, big data analytics approaches were presented, including multimedia analytics like big video analytics. We have also presented one of the important related task to big data which is the machine learning, in particular, the big data analytics based on machine learning.

In conclusion, big data is a rapidly growing field that has the potential to transform industries and shape our world. By understanding the definition of big data, its sources, types, and characteristics, we can better harness its power and unlock its potential. As we continue to generate and collect massive amounts of data, it will be increasingly important to develop new methods and technologies for processing and analyzing this information.

[3] https://www.lebigdata.fr/machine-learning-et-big-data

[4] https://www.techtarget.com/searchbusinessanalytics/tip/Big-data-vs-machine-learning-How-they-differ-and-relate

Chapter 2

Basic Concepts of Machine Learning

Contents

2.1 Introduction

Machine learning is a subset of artificial intelligence that involves the development of algorithms that can learn from and make predictions or decisions based on data. These algorithms improve their performance as the amount of data they are exposed to increases. In this chapter we will discuss basic concepts related to machine learning. We start with different definitions given to machine learning, and then introduce a series of steps typically followed to build a machine learning model. Next, machine learning categories such as supervised learning, unsupervised learning, and reinforcement learning will be detailed. In the second part of this chapter, we will provide a theoretical background on deep learning and transfer learning with a detailed study

of different models, for example: AutoEncoder (AE), Convolutional Neural Network (CNN) and Recurrent Neural Networks (RNN).

2.2 Concepts of machine learning

Recent advancements in computer technology have enabled us to store and process large amounts of data. As a result, machine learning has grown substantially in recent years [42]. The goal of machine learning is to give computers human-like intelligence so that they can make predictions based on vast amounts of data, which would be difficult for humans to do [43].

2.2.1 Definition of machine learning

Machine learning is a method of data analysis that automates the creation of analytical models based on previous experiences without any external human assistance [44]. In a general definition, machine learning is the branch of computer science that uses past experience to learn and use its knowledge to make future decisions [45].

More specifically, machine learning is a subfield of artificial intelligence that allows computers to learn without being explicitly programmed. Machine learning algorithms are trained on data to identify patterns and make predictions. The goal of machine learning is to identify a function f:X→Y that maps the input X into the output Y. The type of function f depends on the type of learning algorithm used [46].

According to the sorts of data sets utilized as training, machine learning algorithms can be divided primarily into three categories.(See Section 2.1.3 for more information.)

The workings of machine learning models can be understood by the example of identifying an image of a car or not. To identify this, the ML model takes images of either cars or not cars as input, extracts the different features of the images such as shape, height, etc., applies the classification algorithm, and predicts the output. Look at the below figure 2.1 [47]:

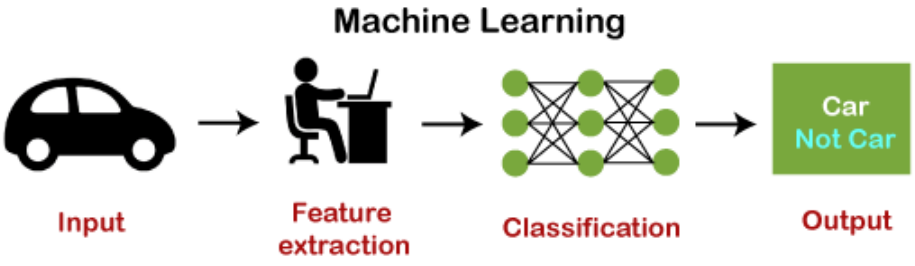


Figure 2.1: Machine learning model

2.2.2 Machine learning process

Machine learning is a data-driven process that involves collecting data, preprocessing it, building a model, and generating predictions and insights. The development, validation, and implementation of machine learning models require a series of steps, which are explained below.

1. **Data pre-processing:** Machine learning can work on structured and unstructured data (voice, images, and text). The result of this step is usually a representation of the data we will use for learning (labeled, meaning supervised learning, or unlabeled, meaning unsupervised learning). Data preprocessing is the first and most important step in creating a machine learning model. It involves cleaning, formatting, and transforming the raw data into a format that can be used by a machine learning algorithm. The specific steps involved in data preprocessing vary depending on the source and format of the data, but some common steps include:

   - **Data cleaning:** It is identifying erroneous, incomplete, irrelevant, or inaccurate data elements and then changing, amending, or deleting the row data. It can be done on photos, tables, databases, or other data types [48].
   - **Missing value:** Machine learning, which is the mainstay of data analysis and information extracting, often faces the problem of missing values. Missing values can occur for a variety of reasons, such as system failure during data collection or human error during data preprocessing. This can slow down the analytic process and lead to faulty conclusions [49].
   - **Data normalization:** This phase is typically used to adjust for variations in the data recording process. It involves scaling the data to a specific range, such as between 0 and 1 or between -1 and +1. This scaling technique is helpful when the data set does not contain any outliers. The pixel normalization method is a common technique used in computer vision to speed up model learning. In this method, an image is normalized by dividing each of its pixel values by the highest value that a pixel can accept[50].
   - **Imbalanced data problem:** It is a common problem in machine learning that makes it difficult to perform data analytics in many real-world applications. Imbalanced data refers to a classification problem where the classes are not equally represented. There are two main approaches to addressing imbalanced data sets: data augmentation and re-sampling [51]. Data augmentation is a technique that artificially increases the size of the minority class by creating new data points from existing data points. Re-

sampling is a technique that balances the size of the classes by either removing data points from the majority class (under-sampling) or duplicating data points from the minority class (over-sampling) as shown in figure 2.2 [52].

Certain algorithms can automatically create synthetic samples. Synthetic Minority Over-sampling Technique (SMOTE) is the most popular of these techniques. SMOTE is an oversampling technique, which means it generates synthetic samples from the minority class instead of duplicating them. The algorithm chooses two or more instances that are similar to each other (using a distance metric) and perturbs each instance's attributes, changing one attribute at a time according to the differences between the chosen instances [53].

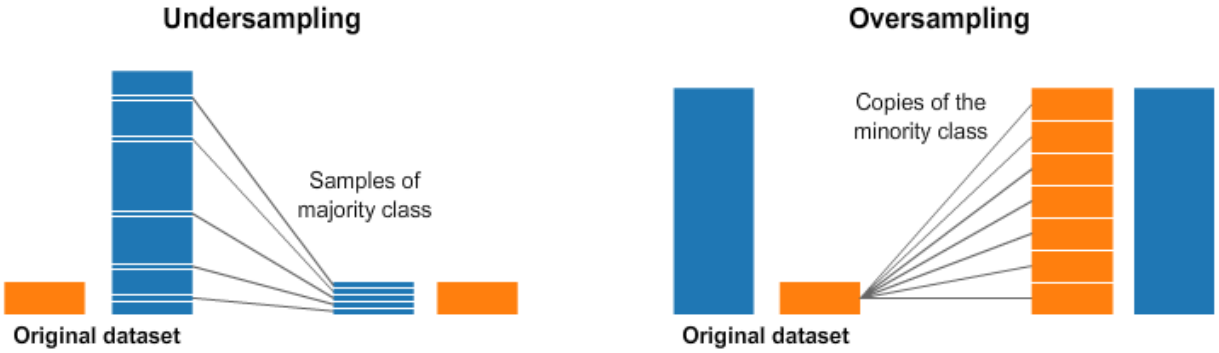


Figure 2.2: Re-sampling imbalanced datasets

2. **Feature selection:**Feature selection is the process of identifying the most important features for a data mining model. This is important because not all features are equally important, and including irrelevant or redundant features can actually hurt the performance of the model. Furthermore, adding more and more features to a model increases the overall complexity of the model. Feature selection is an important step in the data mining process, and it can significantly improve the performance of the model.

   Feature selection is a dimensionality reduction technique that seeks to select the most relevant features from a dataset. This is done by identifying and removing duplicate, irrelevant, or noisy features [54]. In the literature, a variety of feature selection techniques have been presented to obtain the pertinent features or feature subsets they need to accomplish their classification and clustering goals. For feature selection, there are three basic methods:

   - **Filter methods:** In machine learning, filter techniques are a way of selecting features that uses the properties of the data, rather than the learning algorithm, to choose fea-

tures. These methods are computationally efficient because they use cross-validation performance, rather than univariate statistics, to quantify the inherent qualities of the features. They are also faster and more computationally efficient than wrapper methods. This makes them a good choice for working with high-dimensional data. However, filter methods can omit features that are not individually significant but can be useful when combined with other features. Some common filter techniques include information gain, the chi-square test, the Fisher's score, and the correlation coefficient. [54].

- **Wrapper methods:** are a type of feature selection technique that uses a learning algorithm to evaluate the quality of feature subsets. This approach is more computationally expensive than filter methods so they should only be used when necessary, but it can lead to better performance. The main difference between wrapper methods and filter methods is that wrapper methods consider the impact of the feature subset on the performance of the learning algorithm. Filter methods, on the other hand, only consider the intrinsic properties of the features. In most cases, wrapper methods outperform filter methods in terms of prediction accuracy. However, filter methods are much faster and can be used when computational resources are limited[54].
- **Embedded methods:** are a type of feature selection technique that combines the advantages of filter and wrapper methods. They are computationally efficient, like filter methods, and they can take feature interactions into account, like wrapper methods. The main difference between embedded and wrapper methods is that embedded methods do not require an explicit evaluation of all possible feature subsets. Instead, they incorporate feature selection into the learning algorithm itself. This can lead to better performance than filter methods, without the high computational cost of wrapper methods. There are many different embedded methods, but they all follow the same basic approach. They first train a learning algorithm on the full set of features. Then, they use the learning algorithm to identify the features that are most important for predicting the target variable. Finally, they use these features to train a new learning algorithm [54].

3. **Choosing a model:** When selecting a machine learning model, there are many factors to consider, including the type of problem, the amount of data available, the desired accuracy, and the available resources. The best type of model for a particular problem will depend on the specific requirements of the problem. For example, if the desired output is known,

then a supervised learning model is a good choice. If the desired output is not known, then an unsupervised learning model is a good choice. If the problem involves making decisions, then a reinforcement learning model is a good choice. Once a type of model has been chosen, the next step is to select a specific model from the available options. There are many different models available for each type of machine learning problem. The best model for a particular problem will depend on the specific characteristics of the problem, such as the amount of data available, the desired accuracy, and the available resources. The process of selecting a machine learning model can be challenging. However, it is important to carefully consider all of the factors involved in order to select the best model for the problem at hand [55]. Three major categories are available for exploring the machine learning model options, as shown in Figure 2.3 [56].

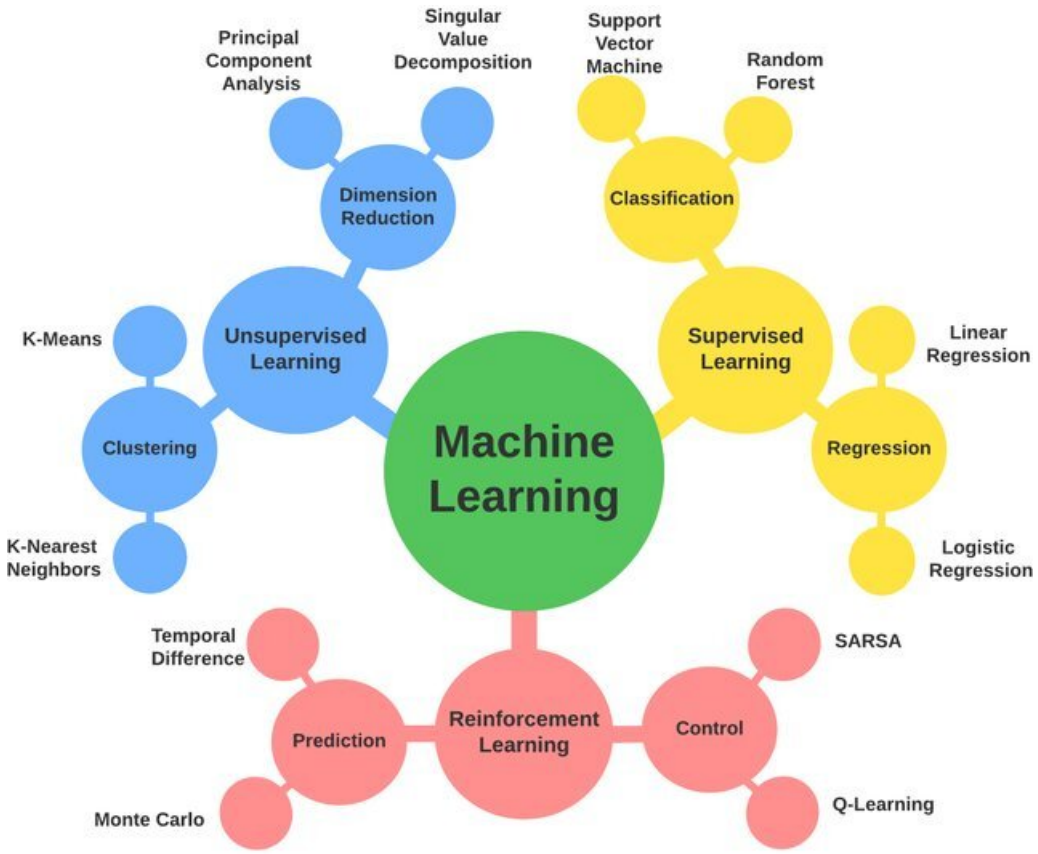


Figure 2.3: A categorization of major machine learning techniques with relevant examples

4. **Model evaluation:** Model evaluation is the process of using evaluation measures to understand the effectiveness of a machine learning model. It is important to evaluate a model to ensure that it is performing well and to identify any areas where it can be improved. There are many different evaluation measures that can be used, depending on the type of model and the task that it is being used for. For classification models, common evaluation

measures include accuracy, precision, recall, and F1 score. For regression models, common evaluation measures include mean squared error (MSE), root mean squared error (RMSE), and mean absolute error (MAE). The Figure 2.4 [57] shows a variety of evaluation measures that can be used for different machine learning tasks. [1].

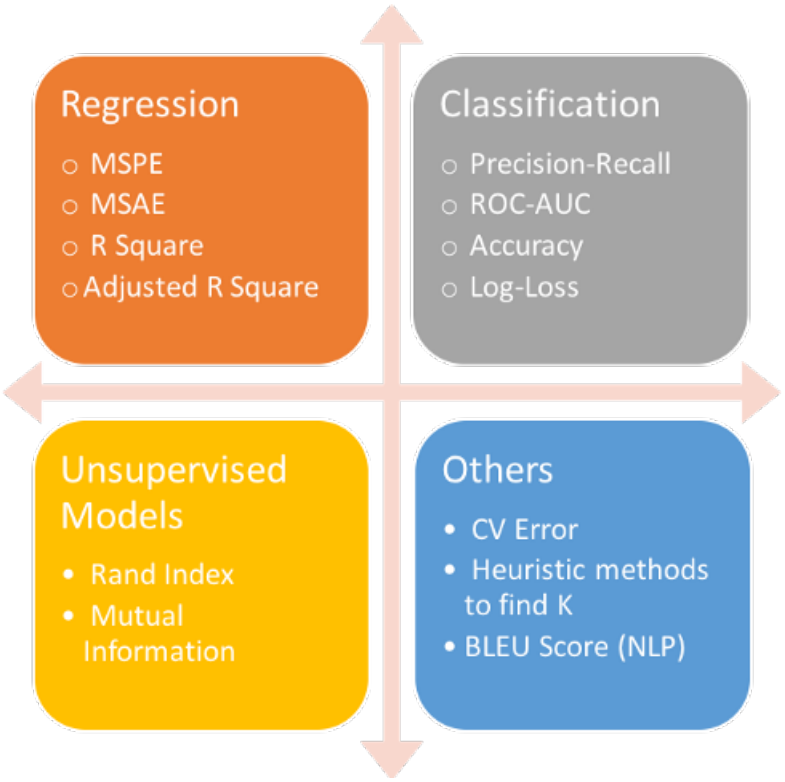


Figure 2.4: Evaluation metrics

(a) **Accuracy:** It is a metric that summarizes the model's performance across all classes. It is useful when all classes are equally important. It is calculated as the proportion of correctly predicted events to the total number of predicted events [2].

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.1}$$

(b) **Precision:** It is calculated as the ratio between the number of Positive samples correctly classified to the total number of samples classified as Positive (either correctly or incorrectly) 2.

$$Precision = \frac{TP}{TP + FP} \tag{2.2}$$

(c) **Recall:** It is a metric that measures how well a model can identify positive samples. It is calculated as the proportion of positive samples that have been accurately identified

[1] https://www.dominodatalab.com/data-science-dictionary/model-evaluation
[2] https://blog.paperspace.com/deep-learning-metrics-precision-recall-accuracy/

as positive to all positive samples. A higher recall indicates that the model is better at identifying positive samples 2.

$$Recall = \frac{TP}{TP + FN} \tag{2.3}$$

(d) **F1 score:** It is a measure of a model's accuracy that combines both recall and precision. It ranges from 0 to 1, with 1 being the highest possible score 2.

$$F1score = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \tag{2.4}$$

Where: T= True ; N= Negative; F= False; P= Positive.

(e) **Receiver Operating Characteristics (ROC) curve:** It is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied.

(f) **Area Under Curve (AUC):** It is a measure of a binary classifier's overall performance. It is calculated by finding the area under the receiver operating characteristic (ROC) curve. The AUC provides insight into the classifier's consistency and stability.

2.2.3 Machine learning categories

There are three main types of machine learning algorithms: supervised learning, unsupervised learning, and reinforcement learning. An overview of these three main types is provided in the following paragraphs. Figure 2.5 [58] illustrates the categories of machine learning.

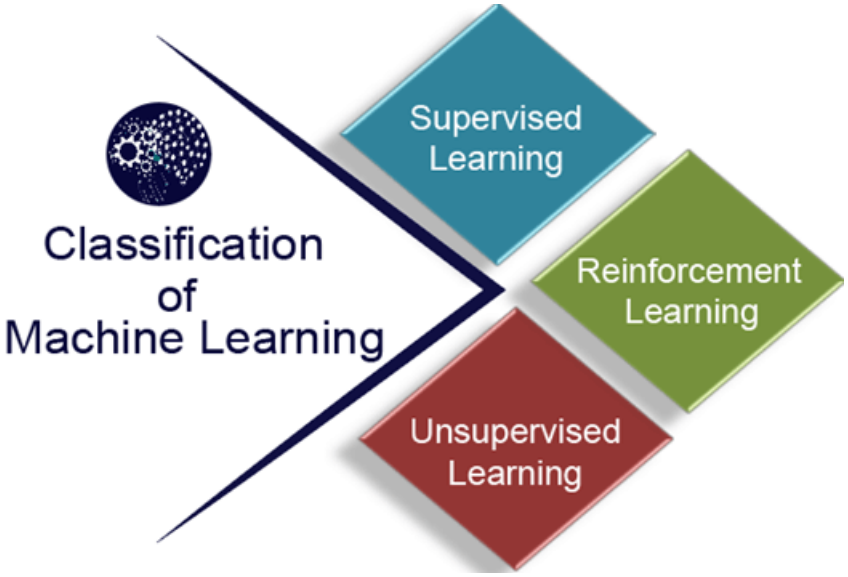


Figure 2.5: Classification of machine learning

1. **Supervised learning:** is a type of machine learning that uses labeled data to train a model. Labeled data is data that has been categorized into different classes. For example, a dataset of images of cats and dogs would be labeled data, with each image being labeled as either a cat or a dog. Supervised learning algorithms are trained on labeled data to learn the relationship between the input features and the output labels. Once the algorithm is trained, it can be used to predict the output labels for new input data [43]. The following diagram 2.6 [58] illustrates how supervised learning works:

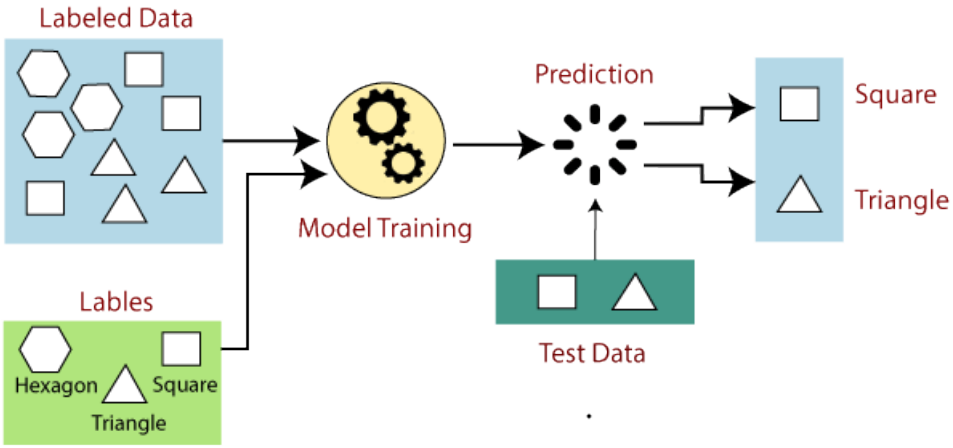


Figure 2.6: Diagram of supervised learning

Supervised learning can be further divided into two types of problems:

- **Classification:** Classification methods are used to predict the category of an output variable. The output variable is categorical if it can take on one of a limited number of values, such as Yes, No, Male, Female, True, or False. The categories of the output variable are also known as targets, labels, or classes (See Figure 2.7 [58]).

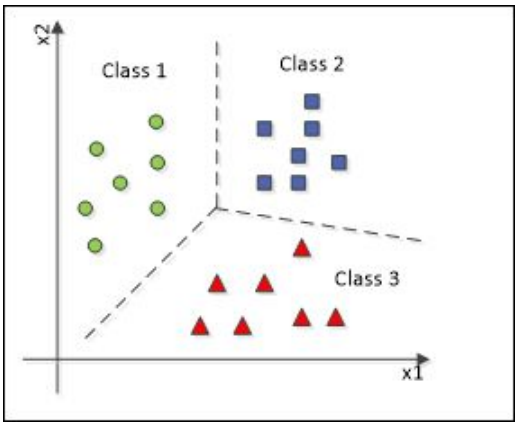


Figure 2.7: Classification problem

- **Regression:** Regression methods are used to predict a continuous value, such as a price, weight, or height. The input variables are used to predict the output variable. The input and output variables are correlated if they have a linear relationship. For example, the price of a house might be correlated with the size of the house. Regression methods are used in a wide variety of industries, including finance, healthcare, and retail. (See Figure 2.8 [58])

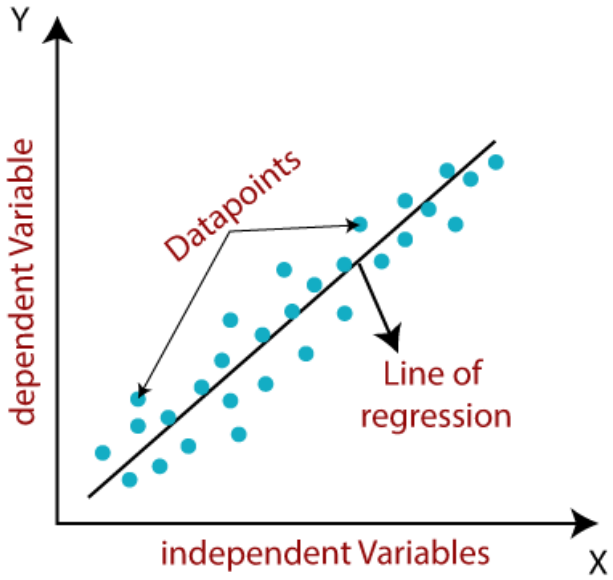


Figure 2.8: Linear regression model

There are many examples of supervised learning algorithm, including:

- Linear regression;
– Naive Bayes;
– Random forest;
– Support vector machines;
– Decision Tree;
– K-Nearest Neighbours.

2. **Unsupervised learning:** Unsupervised learning systems use unlabeled data to train their algorithms. Unlike supervised learning, which requires labeled data, unsupervised learning systems learn from unlabeled data by finding patterns and relationships between the data points. This type of learning is similar to how humans learn new information, by observing and making inferences from the world around them [59]. One of the most common tasks in unsupervised learning is clustering. Clustering algorithms divide a dataset into groups, or clusters, of similar data points. The distance function is often used to measure the

similarity between data points [43].

Diagram 2.9 [58] below explains how unsupervised learning functions.

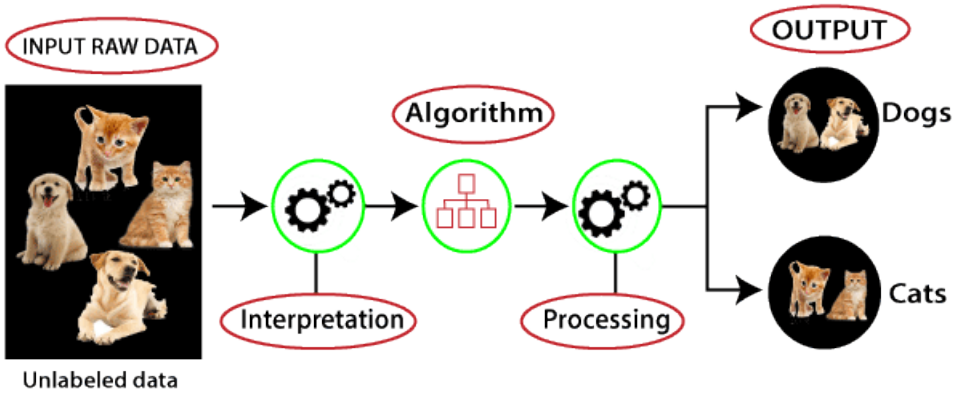


Figure 2.9: Unsupervised learning diagram

Unsupervised learning issues can be further grouped into clustering and association problems:

- **Clustering:** is a technique for grouping items into clusters based on their similarities. Items that are more similar to each other are grouped together, while items that are less similar are grouped separately. Clustering is accomplished by identifying patterns in the data, such as shape, size, color, behavior, etc., and then classifying the data according to the presence or absence of these patterns. (See Figure 2.10 [58]).

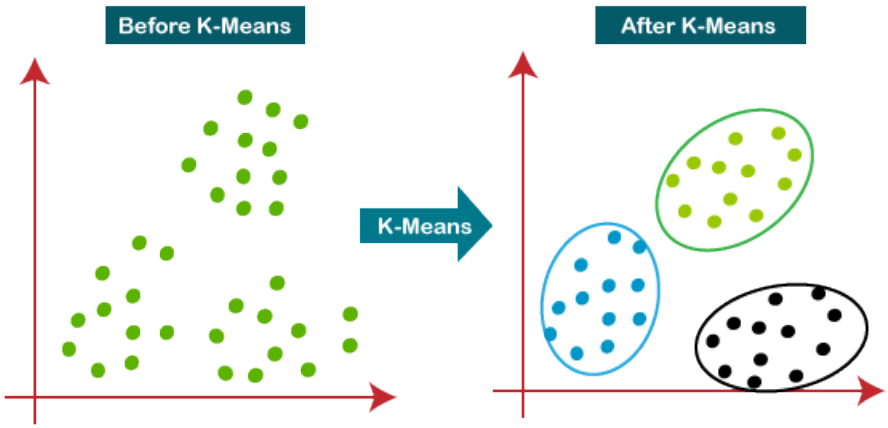


Figure 2.10: K-Means clustering algorithm

- **Association:** is an unsupervised machine learning technique that can be used to find relationships between variables in a large dataset. It does this by looking for patterns in the data, such as which items are often bought together. This information can then

be used to make more informed decisions about marketing and product placement [60]. For example, an association rule might reveal that people who buy bread often also buy butter or jam. This information could be used to place bread, butter, and jam near each other in a grocery store, or to target customers who buy bread with ads for butter and jam.

There are many examples of unsupervised learning algorithm, including:
– Apriori algorithm for association rule learning problems;
- k-means for clustering problems;
- Singular Value Decomposition (SVD) and Principle component analysis (PCA) for dimensionality reduction.

3. **Reinforcement learning:** is a type of machine learning that allows systems to learn from their own experiences. Unlike supervised learning, where systems are trained on a fixed dataset, reinforcement learning systems learn by trial and error. They are given feedback on their actions, and they use this feedback to improve their performance over time [61]. Reinforcement learning is often used for prediction and control in problem domains where time and event sequences matter, feedback may be delayed, and actions have consequences [55].
Diagram 2.11 [62] below explains how reinforcement learning functions.

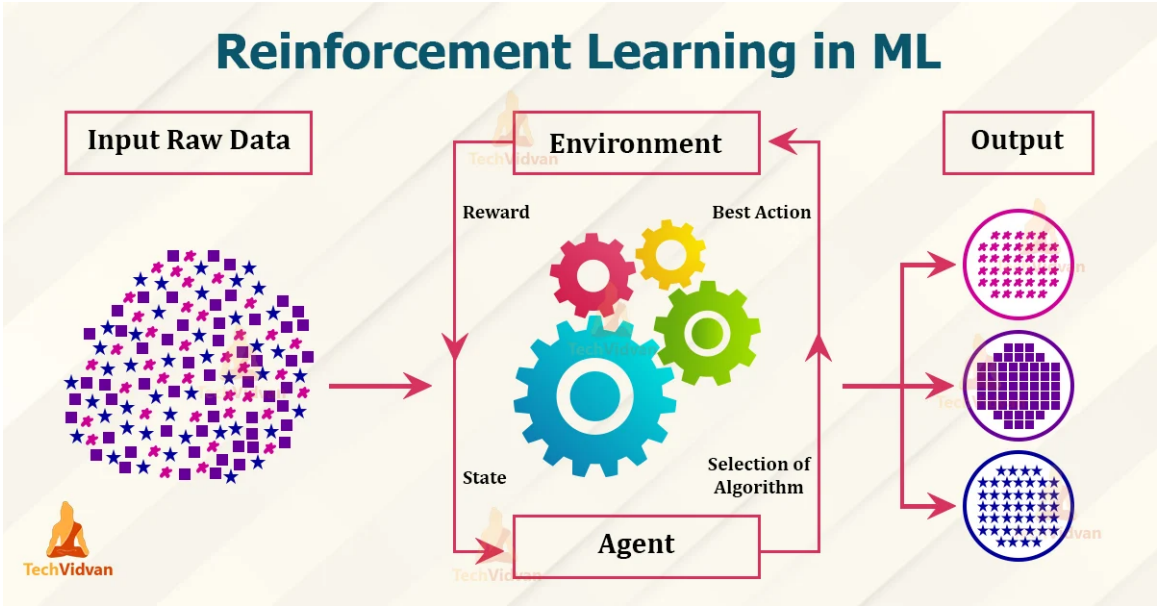


Figure 2.11: Diagram of reinforcement learning

There are many examples of reinforcement learning algorithm, including: SARSA and Q learning for control.

2.3 Concepts of deep learning

Recent advancements in sensor networks and communication technology have made it possible to collect vast amounts of data. This data, known as big data, presents many challenges for data mining and information processing due to its large volume, variety, velocity, and veracity [63].

Deep learning has become a critical tool of big data analytic because it can accurately extract knowledge from complex systems. Since its introduction in 2006, deep learning has rapidly growing as one of the machine learning community's most active research areas [64]. The increasing availability of processing power and data storage capacity has also contributed to the adoption of deep learning [65].

Deep learning is a branch of machine learning and artificial intelligence that uses artificial neural networks as its foundation. The term "deep" in deep learning refers to the number of layers in a neural network. Deep learning involves algorithms inspired by the structure and function of the human brain and is particularly effective for object classification, pattern recognition, and predictive analysis. In order to depict intricate relationships between the data, it can learn many levels of representation [66]. Deep learning is designed to handle huge amounts of data by stacking layers. Through several processing layers and linear and nonlinear transformations, a deep learning model may extract features from raw data and learn these characteristics incrementally through each layer with little to no human involvement [67]. This fundamental aspect of deep learning architectures enables advances toward the main goal of artificial intelligence, which is to make sense of the world around us independently of expert knowledge and interference [68].

Deep learning was partly developed due to traditional machine learning algorithms' failure to perform well in particular AI applications. Deep learning's full potential was not realized until a significant amount of data became available. Deep learning adapts better when more data are provided, which is one of the critical distinctions between it and traditional machine learning systems. The amount of data that many typical machine learning algorithms can process has a cap, or what is known as a "performance plateau," but deep learning models have no such limitations. The feature extraction phase is another difference between conventional machine learning and deep learning methods. Feature extraction in conventional machine learning algorithms is carried out manually; it is a laborious and challenging process that calls for a qualified individual. In deep learning, the algorithm can carry out this phase automatically [68].

Deep learning is a powerful machine learning technique that can be used to solve a variety of complex problems, including natural language processing (NLP), image recognition, voice recognition, and product recommendation systems. Deep learning models are typically trained on large datasets and can learn complex patterns and relationships that would be difficult or impossible to identify using traditional machine learning techniques.

2.3.1 Definition of deep learning

Deep learning is a sub-field of machine learning that uses a mathematical model inspired by the brain's neural networks. It mimics how our brains work and how our nervous systems are organized, where each neuron communicates with the others and passes information [69]. A deep learning must have at least two hidden layers to qualify as a deep learning model (at least ten hidden layers are required for very deep learning) [70]. Deep learning is used to solve problems involving classification and regression.

2.3.2 Deep learning models

There are many different deep learning models, including convolutional neural networks, stacked auto-encoders, deep belief networks, and recurrent neural networks.

1. **Convolutional neural network (CNN):** Convolutional neural networks (CNNs) are a popular deep learning method for image data and other use cases such as audio and NLP (Natural Language Processing) [70]. A production-quality CNN can be highly complex with many layers. In this part, we will examine a CNN that uses only the convolutional, pooling, and fully connected layers described in the items below [71] and in the Figure 2.12 [72]:

    - **Convolutional Layer (CONV):** Convolutional layers are used to generate a feature map from the input data. A collection of filters, which are small square matrices with typical dimensions of 2x2 but can also be 3x3, 5x5, or even 7x7 , is used. Some of the values in a feature map may be negative after it has been constructed. The ReLU activation function's goal is to replace any negative values with zero, is defined as:

$$ReLU(x) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \tag{2.5}$$

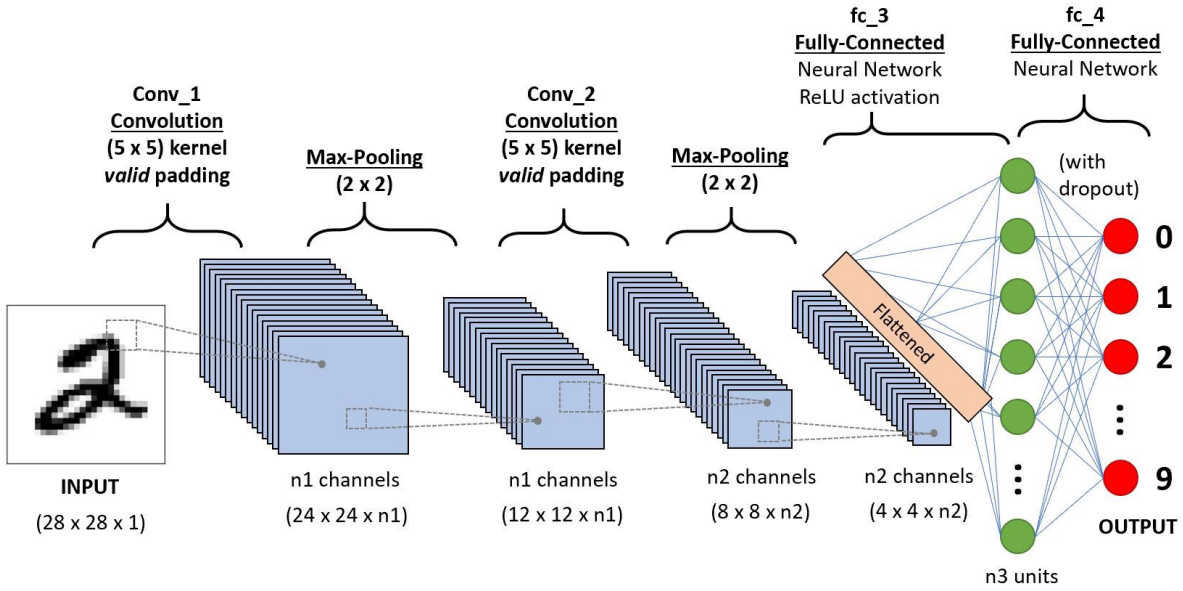


Figure 2.12: A simple convolutional neural network architecture

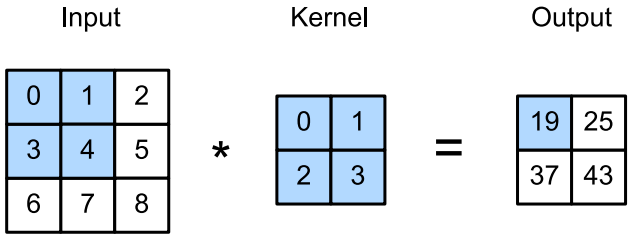


Figure 2.13: Performing a convolution with a kernel window shape of 2x2

- **Pooling Layer (POOL):** The pooling layer is used to reduce the feature map's dimension for a simpler output. An average pooling operation or a maximum pooling operation can be used. The CNN's Max Pooling outcome is shown in Figure 2.14.

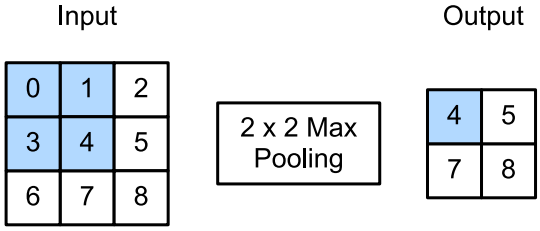


Figure 2.14: An Example of max pooling with a pooling window shape of 2x2

- **Fully Connected Layer (FC):**Fully connected layers, also known as linear layers,

connect every input neuron and every output neuron. They are used to find a probability score for each label in the training dataset using an input of a flattened image vector (1-D image tensor), as shown in picture 2.15.

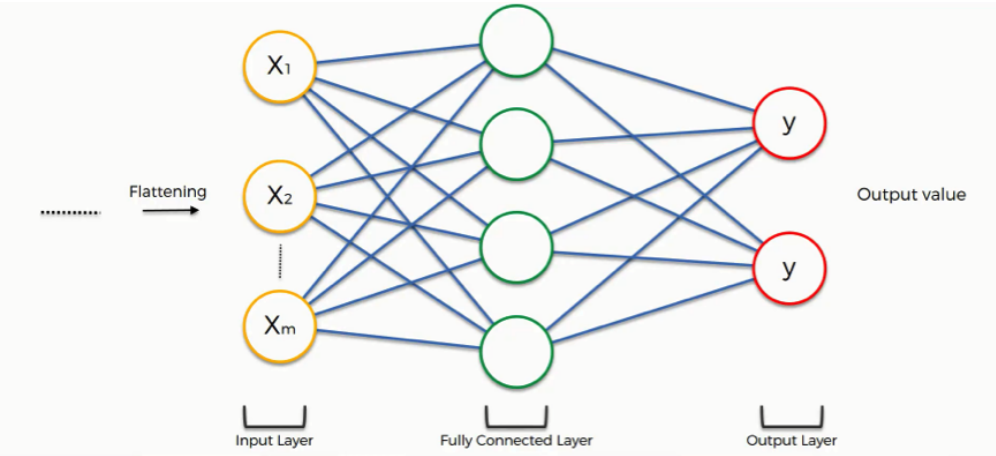


Figure 2.15: An example of fully connected Layer

- **Activation Function:** An activation function is performed by a node placed at the conclusion or in the middle of two neural networks. They determine whether or not the neuron will fire. As seen in figure 2.16, there are various types of activation functions.

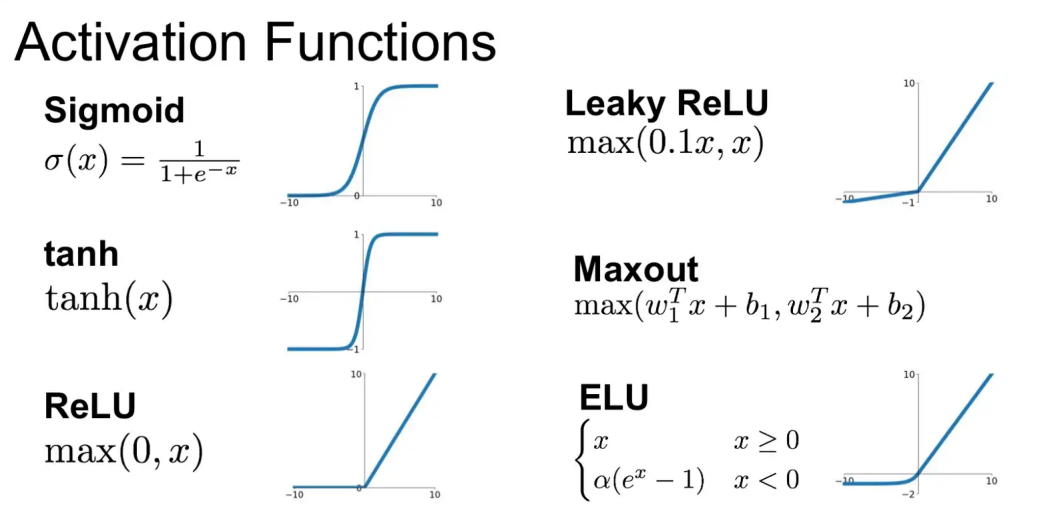


Figure 2.16: Plots of the activation functions

VGGNet, GoogLeNet, ResNet, Inception-V3, and the most recent CNN architecture EfficientNet are just a few examples of CNN architectures that have been developed in the literature as shown in figure 2.17 [73]. These models were evaluated using ImageNet data, which contains over a million images and a thousand different classes.

2. **Recursive neural networks (RNN):** Recurrent neural networks (RNNs) are deep learning models that use recurrent connections to capture the dynamics of sequences when pro-

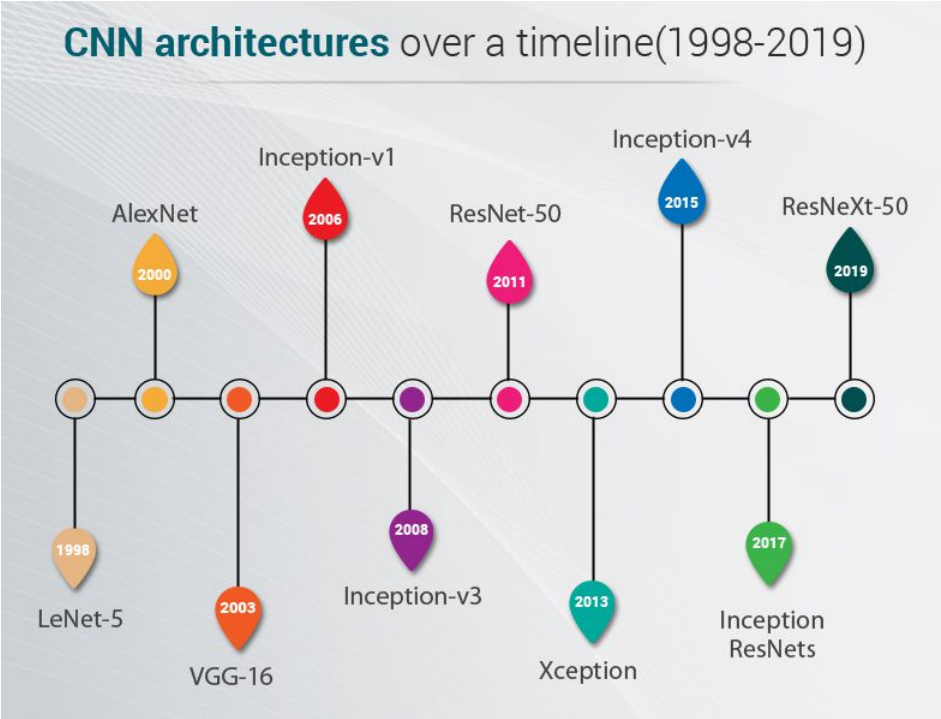


Figure 2.17: CNN architecture over a timeline (1998-2019)

cessing sequential data for NLP tasks like language modeling, text generation, or sentence auto-completion. By using a memory of prior inputs that are kept in the internal state of the neural network, it learns features for the sequential data. The network is able to keep track of what it has already seen in the sequence in order to assist it in deciding what to do with the current input [63]. Figure 2.18 shows the contents of a simple RNN. There are more advanced models available besides simple RNNs, such as LSTMs [74] and GRUs [75]. The recurrent neural network and its derivatives have been successful in a variety of applications, including speech recognition, machine translation, and natural language processing [76, 77].

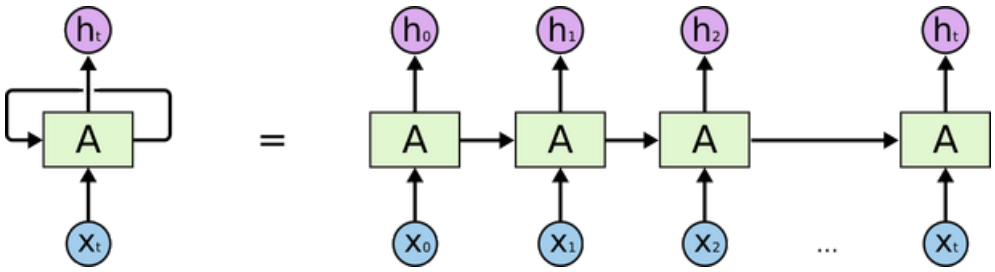


Figure 2.18: Recurrent neural network architecture

3. **Auto-encoder (AE):** The autoencoder is an unsupervised learning algorithm that can be broken down into encoder, code, and decoder blocks to learn effective encodings. The autoencoder is taught to effectively encode the input data, and from that encoding, the

inputs are subsequently reconstructed. The autoencoder's output is identical to its input [78]. Figure 2.19 depicts an autoencoder's overall operation. The learned feature is the appropriate code, and during the process, the autoencoder is optimized by decreasing the reconstruction error. According to [79], a single layer typically cannot extract the discriminative and representative features of raw data. There is a more complex design and training process known as stacked autoencoder (SAE) that can be used to increase the expressive capability of classical autoencoders [80].

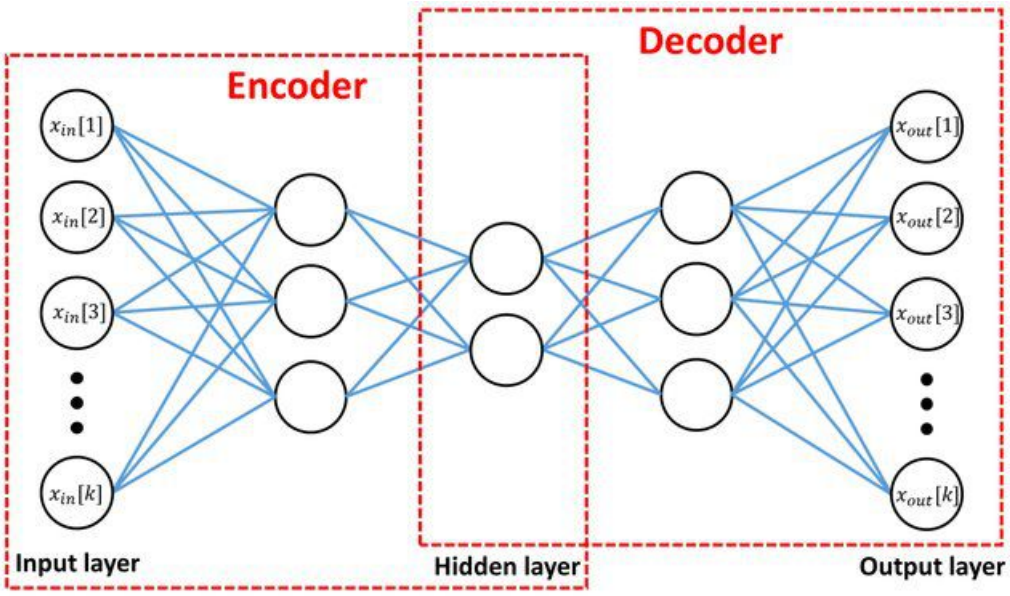


Figure 2.19: The basic representation of auto-encoder

Autoencoders are frequently used to minimize the dimensions of data when a nonlinear function, such as principal component analysis (PCA), explains the connection between dependent and independent variables. It is one of the most promising feature extraction technologies utilized in a variety of applications, including facial alignment and gesture detection, self-driving automobiles, and voice recognition.

2.4 Concepts of transfer learning

Transfer learning is a technique that allows you to use a model that has already been trained as the foundation for a new assignment. It is now particularly well-liked in deep learning due to its ability to train deep neural networks with fairly minimal data [3]. Instead of creating models from scratch, this method uses pre-trained models as a foundation for computer vision. This gives us the opportunity to overcome the challenge of creating Deep Learning models, which

[3] https://builtin.com/data-science/transfer-learning

requires a sizable amount of processing and storage resources. It's crucial to remember that transfer learning in deep learning is only successful if the features discovered during the initial job are universal [81]. Due to the enormous amount of CPU power needed, transfer learning is mostly employed in computer vision and natural language processing tasks like sentiment analysis [3]. For example, you can use a model that was taught to identify dogs and cats to identify other animals. Transfer learning can save time and resources by utilizing previously acquired knowledge instead of beginning from scratch.

2.4.1 Definition of transfer learning

Transfer learning is a machine learning technique that uses a model created for one task as the foundation for a model on another. It is a deep learning and machine learning approach in which knowledge is transmitted from one model to another as shown in figure 2.20 [82].

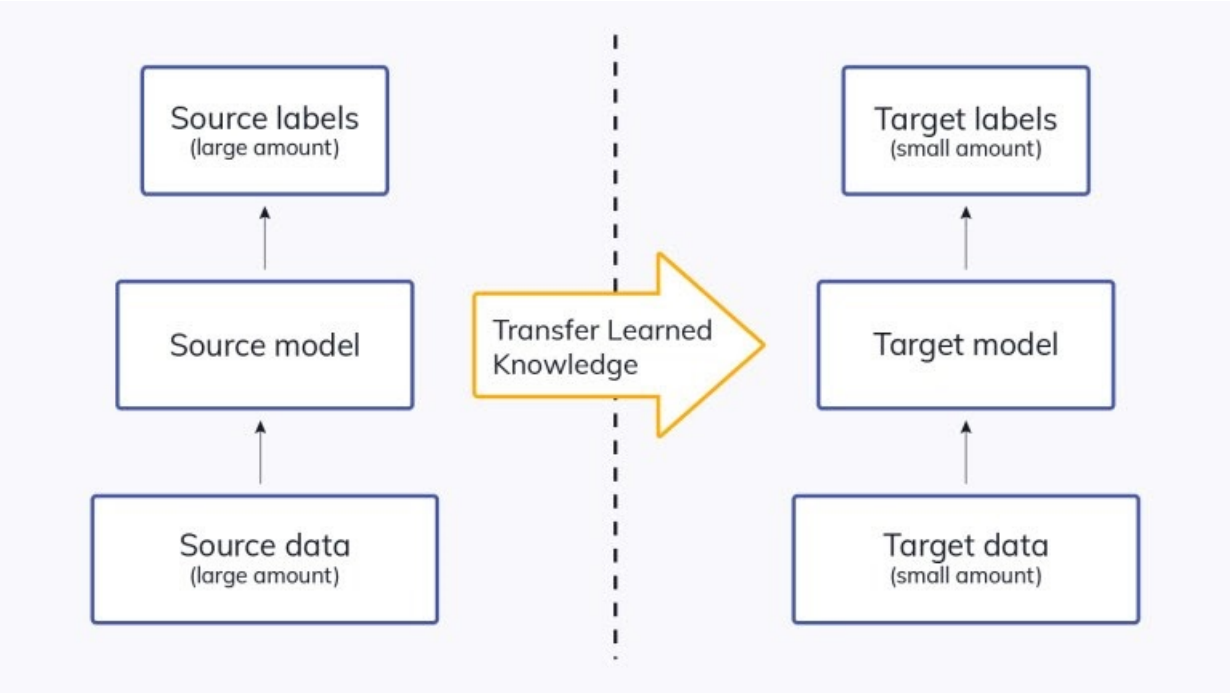


Figure 2.20: The basic representation of transfer learning

2.4.2 Transfer learning strategies

In transfer learning, there are two kinds of methods that can be used (depending on the problem statement). They are as follows:

1. **Fixed feature extractor:** As seen in figure 2.21, the pre-trained model is used as a feature extractor in which the weights in the feature extraction layer are frozen while the fully connected layer is removed. This is known as a fixed feature extractor.

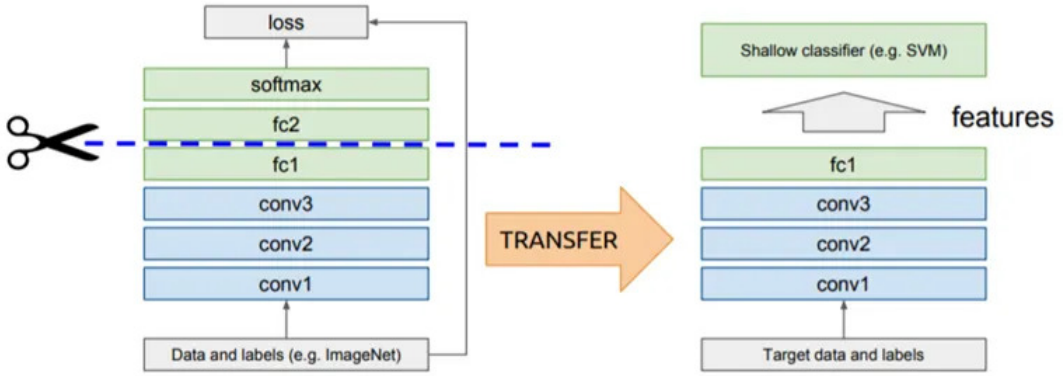


Figure 2.21: Pre-trained models as feature extractor

2. **Fine-tuning:** As seen in figure 2.22 [83], transfer learning can be applied through multiple distinct strategies:

   - **Train the entire model:** In this instance, we train the model using our dataset and the architecture of the previously trained model. Since we are learning the model from scratch, we will require a big dataset and powerful computing resources.
   - **Train some layers while freezing the others:** is a technique used in neural networks to prevent overfitting. Overfitting occurs when a model becomes too specific to the training data and is not generalizable to new data. This can happen for a variety of reasons, including having too many parameters or having a model that is too complex. By freezing some layers, we prevent them from being trained and their weights from being changed during training. This is useful when we have a limited dataset and a lot of parameters because it helps prevent over-fitting. On the other hand, if the dataset is big and there are few parameters, we can enhance the model by training more layers for the new task because over-fitting is not a problem.
   - **Freeze the convolutional base:** is a technique used in neural networks to prevent overfitting. Overfitting occurs when a model becomes too specific to the training data and is not generalizable to new data. This can happen for a variety of reasons, including having too many parameters or having a model that is too complex. By freezing the convolutional base, we prevent the weights in the convolutional layers from being updated during training. This is useful when we don't have enough computational capacity, our dataset is tiny, or the pre-trained model already has a solution to a problem that is extremely similar to the one we are trying to solve.

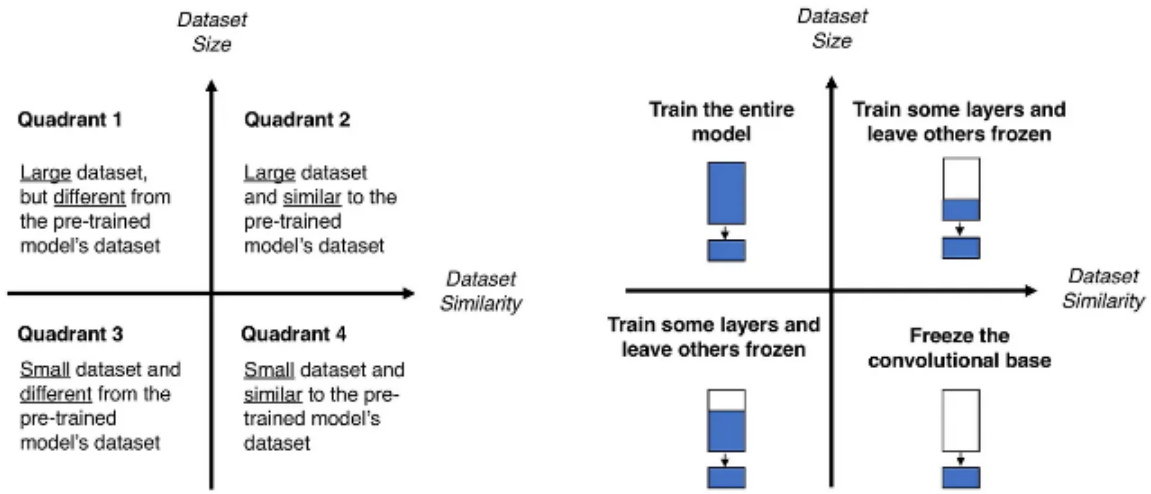


Figure 2.22: Fine-tuning strategies

2.5 Conclusion

This chapter provides an overview of the basic concepts related to machine learning. We have discussed the different definitions of machine learning and how it involves the development of algorithms that can learn from data to make predictions or decisions. We have also outlined the series of steps typically followed to build a machine learning model and have introduced the different categories of machine learning: supervised learning, unsupervised learning, and reinforcement learning. Additionally, we have provided a theoretical background on deep learning and transfer learning with a detailed study of different models such as Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and AutoEncoders (AE). This foundation will serve as a basis for further exploration of the field of machine learning.

In conclusion, machine learning is a powerful tool that allows us to make predictions and decisions based on data. By following a series of steps, we can build machine learning models that improve their performance as they are exposed to more data. As we continue to develop and refine machine learning algorithms, the potential for this technology to transform our world is truly limitless.

Chapter 3

Video shot boundary detection

Contents

3.1 Introduction

The big data revolution in multimedia is driven by the enormous and rapidly increasing amount of multimedia data available every day, made possible by the vast development of multimedia technology and the availability of multimedia sources.

Video is the most frequently accessed data type on the internet. Video content has grown rapidly due to platforms like YouTube, Vimeo, Dailymotion, and social media sites such as Facebook, Twitter, and Instagram. This has created challenges for managing video content.

Manually browsing and uploading large videos to determine their relevance is a difficult and time-consuming task [84]. Automated video analysis applications, such as Content-Based Video Indexing and Retrieval (CBVIR) systems, are necessary to manage large multimedia data sets. These systems include video folder browsing, news event analysis, intelligent video management, video surveillance [41], key-frame extraction [85].

Video summarization can be used to convert large videos into structured, more concise information that are easier to understand and share. The process involves segmenting the video into shots and extracting key frames that represent the entire video [2]. The first stage of video summarization is video shot boundary detection (SBD), which has a significant impact on subsequent processes. The main idea behind SBD is to extract features from video frames and detect shot types based on feature differences. There are two types of SBD: Cut Transition (CT) and Gradual Transition (GT) [3].

The effectiveness of an SBD algorithm is determined by its ability to identify transitions in a video sequence. The accuracy of detection is influenced by the features extracted from the video frames and their ability to represent visual information, as well as the computational cost of the algorithm [86]. Factors such as flashlights, light variations, object or camera motion, camera operations like zooming, panning, and tilting, and similar backgrounds can affect SBD. Despite increased attention to SBD in recent decades, there is still no complete solution to these challenges due to the randomness and size of raw video data. A robust and efficient automated SBD method is necessary [87].

Previous assessments of shot boundary detection (SBD) methods have not often covered recent developments, such as the application of deep learning. This chapter discusses and evaluates several SBD methods that are applied in the uncompressed domain, taking into account their accuracy, computational load, feature extraction method, benefits, and drawbacks. The chapter also discusses future research directions

3.2 Basic Concepts of Shot Boundary Detection

The initial stage in summarizing a video is dividing it into shots. A shot is a sequence of related frames captured continuously by one camera, showing an uninterrupted action in time and space. The point where one shot transitions to another is called a shot boundary. This section discusses the key ideas behind detecting shot boundaries in videos. [88].

3.2.1 Video Definition

A video is made up of a series of image frames displayed in a specific order over time. The number of frames in a video depends on its size and can take up a significant amount of memory. Typically, videos have a frame rate of 20 to 30 frames per second [89].

3.2.2 Video Hierarchically

A video can be divided into scenes, shots, and frames. A scene is a group of shots that form a meaningful unit. A shot consists of a series of frames recorded by one camera in an uninterrupted action. The frames within a shot have similar visual features and information that change over time. A frame is the smallest element that makes up a shot. [89]. (see Fig 3.1 [90])

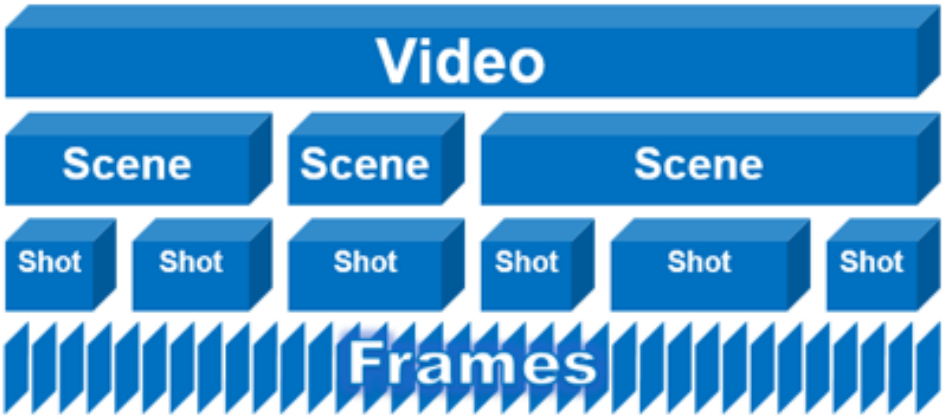


Figure 3.1: Hierarchical video

3.2.3 Shot transition types

The transition from one shot to the next can be either abrupt or gradual. An abrupt transition, also known as a cut or hard transition, happens when two shots are joined together without any special effects. This results in an immediate change from one shot to another. On the other hand, a gradual transition involves combining two shots using special effects during the editing process. This type of transition can take place over several frames that are visually connected and contain overlapping information [91]. There are various types of gradual transitions, such as fade in/fade out, dissolve, and wipe [92]. (see Fig 3.2 [90])

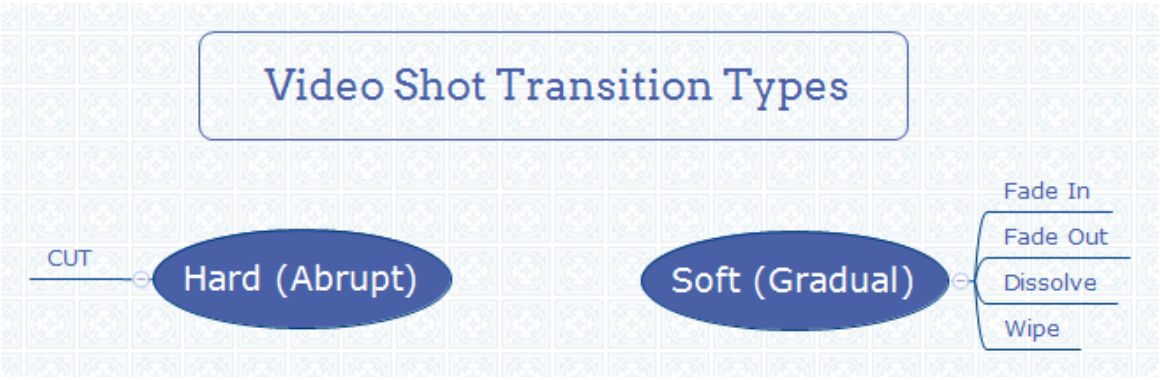


Figure 3.2: Video shot transition types

1. **A Cut :** Is the abrupt transition from video shot to another. [3]. (see Fig 3.3 [90])

Figure 3.3: Cut Transition

2. **A fade out :** Is a shot gradually fade away to a single monochrome frame, typically dark [3]. (see Fig 3.4 [90])

Figure 3.4: Fade out

3. **A fade in :** Occurs when a scene gradually appears on the screen [3]. (see Fig 3.5 [90])

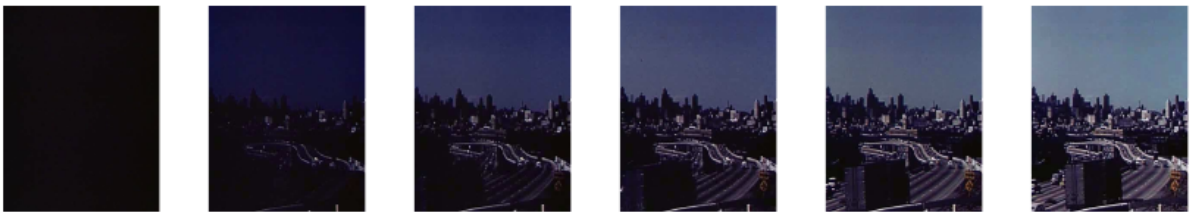
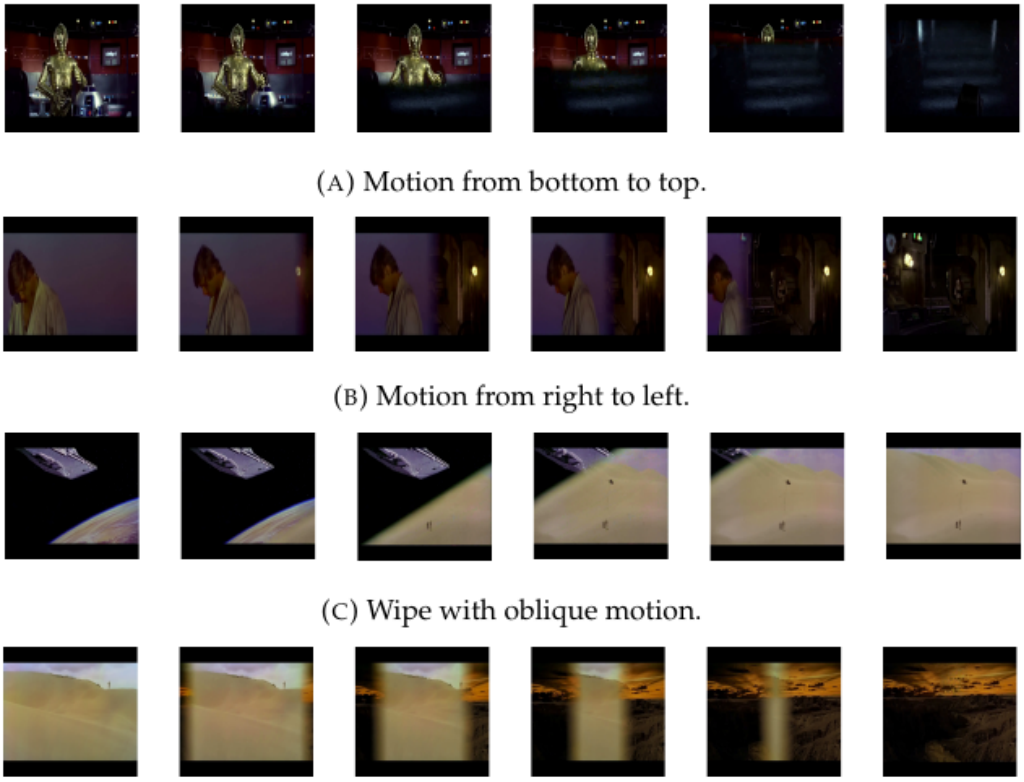


Figure 3.5: Fade in

4. **A dissolve :** Is a transition between two shots in which the first shot gradually fades out while the second shot gradually fades in. This creates the effect of the two shots overlapping for a brief moment.[3]. (see Fig 3.6 [90])

Figure 3.6: Dissolve

5. **The wipe :** Is the most difficult to predict and detect since they are the most dynamic. They occur when one shot knocks the other off the screen. In this instance, two neighboring shots are separated only by space and not by time.[3]. (see Fig 3.7 [90])



(A) Motion from bottom to top.

(B) Motion from right to left.

(C) Wipe with oblique motion.

(D) The movement is going to center.

Figure 3.7: Wipe transitions types

3.2.4 Feature extraction

Feature extraction is a process that takes raw images and extracts low-level, mid-level, or high-level information from them. This information can then be used to detect patterns, classify objects, or identify people [93].

1. **Low-level features :** Low-level features are the most basic features of an image, such as edges, corners, and textures. These features include RGB values/histograms, intensity values ...etc [89].

2. **Mid-level features :** Mid-level features are combinations of low-level features, such as shapes and objects [89].

3. **High-level features :** High-level features are the most complex features of an image, such as faces and scenes [89].

Due to the significance of SBD, Numerous researchers have developed methods to improve the accuracy of SBD for both cut transitions (CT) and gradual transitions (GT). Below, we provide an overview of various SBD techniques.

3.3 Shot boundary detection evaluation metrics

When assessing the performance of Shot Boundary Detection (SBD) methods, it is important to consider both their accuracy and computational cost. However, improving one often means sacrificing the other. For instance, increasing accuracy may require additional computations, while real-time applications may prioritize low computational costs over accuracy. To accurately compare different techniques, evaluations must be conducted under similar conditions and using comparable data sets. In this section, we will discuss popular evaluation criteria for SBD methods, such as recall, precision, and F1-score. [94].

1. **Precision :** Is the fraction of correctly detected positives out of all detected positives (correct and false).

$$P = \frac{N_c}{N_c + N_f} \tag{3.1}$$

2. **Recall :** Is the fraction of correctly detected positives out of all actual positives (correct and missed).

$$R = \frac{N_c}{N_c + N_m} \tag{3.2}$$

3. **F1-score :** Is a measure of a model's accuracy that combines precision and recall. It is a single number that ranges from 0 to 1, where 1 indicates the best possible performance.

$$F1\_score = \frac{2 \times recall \times precision}{recall + precision} \tag{3.3}$$

Where, N_c is number of transitions detected correctly, N_m is number of transitions missed detection, and N_f is number of transitions detected falsely.

Precision and recall are opposing measures for a system, as one often decreases when the other increases. A well-designed system should aim to maximize both. The F1 score combines

both recall and precision and is calculated as the harmonic mean of the two scores. It provides a consistent way to measure the overall effectiveness of an information retrieval system.

3.4 Shot boundary detection methods

Many researchers are working on developing more reliable and precise algorithms for identifying shot boundaries. There are a number of different approaches to this problem, including: Transitions (CT) and Gradual Transitions (GT).

3.4.1 Pixel-Based Methods

This method evaluates the intensity of pixels by comparing two consecutive video frames pixel by pixel or by calculating the percentage of pixels that have changed between the two frames. If the pixel intensity exceeds a particular threshold, it is considered a shot change [3]. There are a number of different pixel-based approaches, one of which is to compare the total of all pixels' absolute differences [95]. These methods require manual threshold setting and are susceptible to rapid object and camera movement, camera panning, and zooming.

3.4.2 Histogram-Based Methods

The difference between the histograms of two successive frames is a common statistic used to detect cut transitions. A histogram is a graphical representation of the distribution of pixel intensities in an image. It can be used to compare the similarity of two images by ignoring their spatial information. To use this technique, the histograms of video frames must be extracted and their distances between them must be calculated. A shot change is considered to have occurred if the distance exceeds a certain threshold. There are several ways to determine the distance between two histograms, including the Euclidean distance, the chi-square distance, and the Manhattan distance. Swanberg et al. In [96], gray level histogram differences in regions were used and weighted by how likely the region was to change in the video sequence. Lu et al. In [97], Singular Value Decomposition (SVD) and Hue Saturation Value (HSV) histograms are used to develop a SBD with low computational complexity. Candidate segments are selected using an adaptive threshold. Color histograms are extracted from all frames in each candidate segment in the HSV color space to form a frame feature matrix. SVD is then applied to the frame feature matrices of all candidate segments to reduce the dimensionality of the features. Bendraou et al. In [98] developed a new method to detect both CT and GT transitions in videos.

Their approach analyzes the video in segments and has two main components: verifying static segments and identifying shot transitions. Features are extracted using Concatenated Block Based Histograms (CBBH) and an economic SVD is performed for each non-static segment. Hard cuts are detected using an adaptive double thresholding process while gradual transitions are detected using the folding-in technique.

The study found that using histogram differences is less affected by object movement than pairwise comparison because it accounts for spatial changes within a frame. However, this approach can still miss shots when two frames have similar histograms but different contents.

3.4.3 Edge-Based Methods

Edge information can also be used to characterize an image. An edge marks the boundary between an object and the background or between overlapping objects. In edge-based methods, a transition is detected when there is a significant difference in the location of edges between the current and previous frames, indicating that edges have disappeared. For example, Zabih et al. In [99], an edge change ratio (ECR) method was proposed to perform SBD. A shot transition is indicated when the disappearance of old edge pixels and appearance of new edge pixels are far from each other. In [100], a technique based on edges was developed. The authors defined the concept of an object's edge by focusing on the pixels that are close to the edge. The edges of an object were matched between two consecutive frames. The percentage of the object's edge that remained constant over time was then used to determine when a transition had occurred. In [101] a method for identifying fade-in and fade-out transitions was proposed. The authors first located the edges in a frame by comparing gradients to a predetermined threshold. Next, they counted the number of edges that were visible. A fade-in or fade-out was signaled when a frame without edges occurred.

This feature has the benefit of being relatively unaffected by changes in lighting and various types of movement, and it corresponds to how humans visually perceive a scene. However, its main drawbacks are that it is computationally expensive and sensitive to noise.

3.4.4 Motion-Based Methods

Motion is an essential aspect of videos. Camera movement in shots can sometimes be misidentified as gradual transitions. By detecting zooms and pans, the accuracy of shot boundary detection algorithms can be improved. Zhang et al. In [102] used motion vectors determined from block matching to detect whether or not a shot was a zoom or a pan. A method for

predicting linear motion using wavelet coefficients that were obtained directly from two subsequent frames was proposed in [103]. In [104], a block matching method was used to match a block in the current frame with every other block in the following frame. The motion vectors were then extracted as part of a region-based pixel difference computation to detect whether there is a lot of camera or object motion in a shot.

To accurately estimate motion, each block must be compared with all blocks in the following frame. However, this results in a high and impractical computational cost.

3.4.5 Deep Learning-Based Methods

Deep learning algorithms have recently gained a lot of attention from researchers for use in computer vision. Convolutional neural networks (CNNs) are one of the most important deep learning techniques because they can extract high-level features from images and video frames [105].

In [106], a CNN model was used to extract high-level, comprehensible features from frames. The model was able to detect both CT and GT boundaries. Candidate segments were selected using an adaptive threshold technique. The network outputs a probability distribution across 1000 classes when given a single frame as input. The frame's TAGs are the top five classes with the highest probabilities, and they serve as the frame's high-level features. However, if the changes in GT are minimal and the background is similar, the semantics may not change, and the detection accuracy may be reduced.

In [107], used CNNs to extract characteristic features from frames. They employed a candidate segment selection method that uses adaptive thresholds to roughly locate shot boundaries and eliminate most non-boundary frames. A new pattern-matching method based on a novel similarity strategy, partially inspired by [97], can be used to detect both CT and GT.

In [108], the DeepSBD network was proposed as a shot boundary detection (SBD) technique for large-scale video datasets. The network categorizes the input of fixed-length segments into three categories: cut, gradual, and no transition. An SVM classifier receives the output and outputs a preliminary labeling estimate. Consecutive segments with the same label are combined using a histogram-driven temporal differential measurement and then put through a post-processing stage that reduces false alarms about slow transitions. However, DeepSBD is not designed for multi-scale detection and is less computationally complex than C3D ConvNet which requires more computing power than a 2D ConvNet.

In [109], researchers proposed a new technique for learning shot detection from pixels to

final shot boundaries using a fully convolutional neural network. They created a dataset of one million frames with automatically generated transitions, including cuts, dissolves, and fades. The model was trained to determine if a frame was part of the same shot as the preceding frame, and it achieved cutting-edge results on the RAI dataset. However, the model has three significant flaws: it misses long dissolves, it only changes scenes partially, and it fails to detect fast scenes with motion blur.

In [110], presented the TransNet neural network that achieved state-of-the-art results on the RAI dataset. The training process employed randomly generated transitions using selected shots from the TRECVID IACC.3 dataset. The architecture is trained on just two common types of transitions (cuts and dissolves).

In [111], the Euclidean distance between two successive frames is used to detect whether they are from the same shot or separate shots. The deep features extracted from the frames are normalized between 0 and 1, and an ideal threshold of 0.7 is chosen after conducting numerous experiments. The features are extracted from the fully connected layer (FC7) of a CNN model that was trained on the ImageNet dataset using the MobileNet architecture. The framework preserves the interest of the generated summary by using image memorability and entropy. The keyframe for each shot is the one with the best memorability and entropy scores.

The two-stage shot boundary detection (TSSBD) approach described in [112]. First detects abrupt shot changes by fusing color histograms (HSV) and deep features (CNN). The video is then divided into segments containing gradual transitions. A 3D-convolutional neural network is then used to classify each segment into one of several categories of gradual shot changes. Finally, gap filling is used to efficiently separate the frames into different shot types and to locate the shot boundaries.

A video shot change detection framework that combines local feature matching and convolutional neural networks (CNNs) is proposed in [113]. The framework first uses ResNet to train on video images and learn to categorize video frames into different groups. In the second identification step, the framework uses the image matching method to determine whether or not the shot has changed by a threshold value after extracting feature keypoints from continuous frames using SIFT. This helps to achieve a low failure detection rate.

In [114], proposes a new approach for video shot boundary detection. The approach uses the CNN model to extract features of video sequences parallelly on the GPU. Local frame similarity and dual-threshold sliding window similarity are taken into consideration to increase recall and precision of shot detection. The experimental result shows that the proposed method can achieve a high F1 score and excellent detection speed.

A method for detecting shot boundaries was proposed in [115] using gradual shot detection based on spatial-temporal convolutional neural networks and shot filtering based on histograms. The cut shots are taken from the entire video using histogram-based shot filtering. After that, a C3D deep model is built to extract frame attributes and separate different shot kinds, including dissolve, swipe, fade in and fade out, and normal. A frame level merging approach is created for untrimmed videos to assist in locating the border of images from adjacent frames.

Previous methods extracted features from videos using convolutional neural networks (CNNs) and then detected changes in the scene using conventional classifiers. However, recent advances in deep learning have led to the development of more effective networks that can be used for a variety of tasks, including shot change detection. For example, ResNet-based networks have been shown to achieve excellent accuracy in object and image detection on large image datasets. However, one drawback of this method is the requirement for large, annotated datasets. Additionally, real-world data may contain cuts between images of the same scene that are not present in synthetic datasets.

3.4.6 Others approaches

In [116] proposed a shot detection method that uses genetic algorithms (GA) and fuzzy logic. The method uses fuzzy logic to classify video frames into two types of transitions: cuts and gradual transitions. The color histogram difference is used to find the differences between two successive frames in a video and to extract features. GA is used as an optimizer to determine the optimal range of values for the fuzzy membership functions. The results show that this feature combination is effective and that accuracy increases as the number of GA iterations or generations increases.

In [117] proposed a novel shot boundary detection technique that combines the gravitational search algorithm (GSA) and particle swarm optimization (PSO) to improve the weights and biases of the feed-forward neural network (FFNN) for classifying frames as normal or probable transition frames. The histogram difference, CIEDE2000 color difference, and normalized 3D Euclidean standard deviation methods are used for feature extraction. The outlier value and continuity matrix are used to extract a potential set of transition frames. A set of thresholds is chosen to identify CT and GT.

A new cut detection technique was proposed in [118], that uses information theory and support vector machines (SVMs). The technique first computes the dissimilarity between frames using information theory. Then, it creates a discriminative feature vector based on mutual

Table 3.1: A comparison of the most advanced shot boundary detection (SBD) algorithms

Ref	Methods	Dataset	CT			GT		
			P	R	F1	P	R	F1
[97]	Histogram-Based	TRECVID-2001	0.91	0.85	0.88	0.83	0.81	0.81
[98]	Histogram-Based	TRECVID-2001	0.97	0.95	0.96	0.87	0.93	0.90
[106]	DL-Based-CNN	TRECVID-2001	0.99	0.87	0.92	0.87	0.83	0.87
[107]	DL-Based-CNN	TRECVID-2001	1.00	0.98	0.99	0.99	0.95	0.97
[108]	DL-Based-CNN	UCF101	0.98	1.00	0.99	0.99	0.99	0.99
[123]	DL-Based-CNN	TRECVID-2007	0.98	1.00	0.99	0.84	0.84	0.84
[114]	DL-Based-CNN	Other	0.95	0.97	0.96	0.86	0.91	0.87
[116]	GA and Fuzzy Logic based	TRECVID-2001	0.88	0.92	0.90	0.86	0.78	0.82
[118]	TI and SVM based	TRECVID-2002	0.98	0.97	0.98	-	-	-
[120]	SURF matching score based	Golf-Video	1.00	0.98	0.99	0.89	0.81	0.85

information. Finally, it uses a trained SVM to classify each frame as a cut or non-cut frame. This technique does not require a conventional global or adaptive threshold, which makes it more robust to noise and variations in lighting.

The proposed method in [119], uses frame entropy and SURF descriptors to retrieve shot boundaries from videos. Cut boundaries are identified by comparing the entropy of the grayscale intensity in adjacent frames. Fade boundaries are automatically detected based on temporal variations in the entropy of pixel intensity throughout each image. False detections are then efficiently eliminated using local descriptors like SURF.

A multi-modal visual features-based shot boundary detection (SBD) framework was developed in [120]. The framework uses a candidate segment selection method that does not require a threshold. The discontinuity signal is computed based on the SURF matching score and the RGB histogram cosine distance value.

A new cut detection technique based on the derived texture feature of local binary pattern (LBP) is proposed in [121]. The technique, called Absolute Sum Local Binary Pattern Histogram Difference (ASLBPHD), computes the difference between the sum of the LBP histograms of two adjacent frames. This difference is used to identify cut boundaries.

A novel approach for cut detection using multi-fractal characteristics was proposed in [122]. The color and texture multi-fractal features are used to identify cut changes between shots. The HSV color space is used to create the color histograms, and the discrete wavelet transform is used to compute the texture features. The proposed method uses Hölder exponents to accurately detect singularities in content changes between frames and describe cut shot changes.

Table 5.2 compares various shot boundary detection (SBD) algorithms based on the features

used, frame skipping, dataset, and accuracy (measured by precision, recall, and F1 score). As Lu et al. in [97], showed algorithms that use frame skipping have lower computational costs while still achieving respectable accuracy. However, some algorithms that use frame skipping still have a moderate computational cost, such as the SURF algorithm in [120] due to the complexity of the features used. CNN-based SBD algorithms, such as those in [27, 28, 29, 32, 36], have a high computational cost but outperform other algorithms in terms of accuracy.

3.5 Open Challenges

Despite significant progress in shot boundary detection, many challenges remain and warrant further research. A good video shot detection method depends heavily on the features, similarity measures, and thresholds used. Changes in lighting and object and camera motion can make shot boundary detection difficult. Color histograms can distinguish between frames taken at different times in the same scene and are robust to small camera motion, but they are sensitive to large camera motion. Motion features can handle object and camera motion well, but edge features are more robust to changes in lighting and motion than color histograms. Shot boundary detection (SBD) relies on the effective use of motion information. Therefore, it is important to conduct research in the areas of separating background from foreground motion, recognizing moving objects and events, merging static and motion features, and creating motion-based indices. While using multiple types of features can slow down the process, using only one type of feature may not yield adequate results. Finding an automatic threshold based on the qualities of the video is a significant task. Machine learning efforts to replace thresholding have just recently started, and they could lead to additional advancements in SBD.

3.6 Conclusion

Video shot boundary detection is the first and most important step in video processing. This work provides a comprehensive survey of SBD algorithms and discusses video definitions, transition types, and hierarchies. Various techniques for detecting shot boundaries based on video content and changes in content are discussed. Despite extensive research on SBD methods, there is still a need to address certain practical issues in many video contexts, such as sudden changes in illumination, dimly lit frames, similar background frames, object and camera motion, and changes in small regions. Addressing these issues will improve the effectiveness of SBD algorithms.

Machine learning techniques have become popular in computer vision applications, but efforts to replace thresholding in SBD with machine learning have only recently started. Machine learning for SBD has been relatively under-researched, so examining the benefits of cutting-edge tools like deep learning could provide new insights for future studies.

While comparing frames and spotting shot boundaries may seem straightforward in the sequential instance, processing large multimedia datasets can be time-consuming. Performance for long video data is still a research gap. The use of deep learning techniques for SBD and the analysis of massive multimedia data will be the main topics of our future work.

Chapter 4

Video summarization

Contents

4.1 Introduction

The exponential growth of daily video data generation due to the fast evolution of multimedia technologies has led to voluminous, redundant, and unstructured data streams. Although video frames convey real-world scenes most vividly, it is always a painful task to find either the appropriate video sequence or the desired portion of the video from a large video data collection. To turn unstructured, voluminous video frames into exciting, valuable information resources, analyzing and summarizing tools that would allow the user to quickly get an idea of the overall content of video footage have become indispensable [1].

Video analysis is the process of analyzing video content to extract meaningful information. This can include identifying objects or people in the video, tracking their movements, and analyzing their behavior. Currently, most video analysis tools use a set of keyframes to provide a content summary of a video sequence. One of the key components of video analysis is creating a video summary.

A video summary is a short version of a longer video that captures the most important parts of the video through the selection of the most important and pertinent semantic content within the video. There are two types of video summaries: static and dynamic. Static video summaries are created by selecting keyframes from the video and arranging them in a sequence. Dynamic video summaries are created by selecting keyframes and then adding transitions between them to create a more fluid summary [4].

Creating a video summary can be challenging because it requires identifying the most important parts of the video and deciding how to present them in a way that makes sense. Some of the challenges include dealing with different types of videos (e.g., instructional videos vs. entertainment videos), dealing with different languages, and dealing with different cultural contexts. An important missing component in the existing video summarization tools is the mechanism to estimate how many keyframes would be suffcient to provide a good, nonredundant representation of the video frames.

To conclude this chapter, we will discuss hierarchical video summarization, which can be created using both static keyframe extraction and dynamic video skimming. This is done after describing each kind independently and going over sub-categories.

4.2 Definition of video summary

A video summary is a short version of a video that is created by selecting the most important parts of the video. It allows you to quickly understand the contents of very large collections of videos without having to watch the entire video. The process of creating a video summary involves extracting a set of still images called keyframes, which are put together to form the video summary [124]. Basic steps involved in creating a summary of video are presented in Figure 4.1.

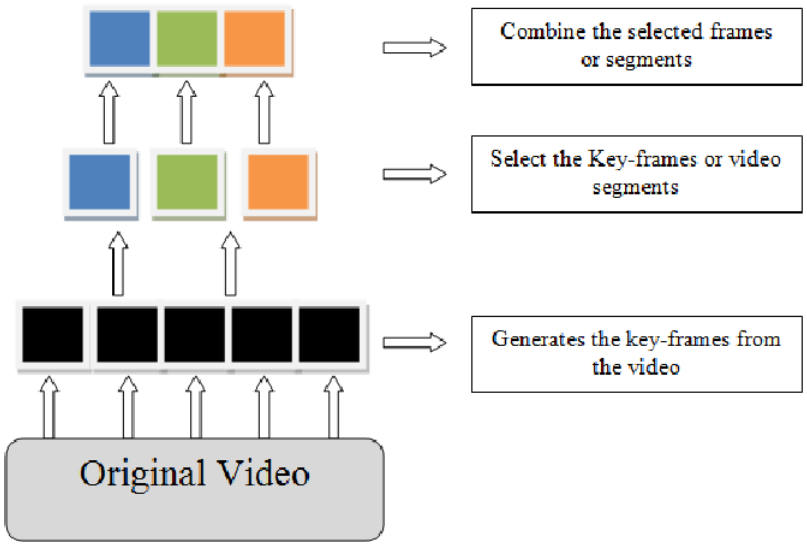


Figure 4.1: Basic procedure for video summarization.

A video summary can be created in two ways: static or dynamic. Static summaries are a collection of keyframes, also known as still images, static storyboards, or skimmed videos. Dynamic summaries are a sequence of video frames, also known as moving images, moving storyboards, or skimmed videos.

4.3 Static video summarization

Static video summary is a technique that reduces repetition in videos by selecting a group of sample frames. It has been the subject of substantial research due to its importance in various video-related applications such as video analysis and retrieval. This section outlines some of the current techniques for summarizing static videos and describes how they produce a set of key frames that represent the content of a video [125].

4.3.1 Cluster and shot based methods

This approach groups together related frames or shots and then selects a few frames (often one frame) from each cluster to serve as key frames. The main idea is to represent video frames as points in space. The characteristics (such as color histogram, brightness, and motion vector) and clustering techniques (such as k-means, hierarchical) used in these methods vary [124].

In [126], a shot detection algorithm based on color histograms is proposed. First, the color distribution of a video frame is described using an RGB histogram. Then, the PCA (principal component analysis) is used to decrease the dimension of the feature vector to a one-dimensional vector reflecting the frequency of the color histogram. Fuzzy-ART and Fuzzy

C-Means algorithms are used to automatically determine the number of clusters in the video and extract the shots from the original video to detect various shots. Once all keyframes have been identified in the video, the storyboard is created by extracting the 10-frame neighborhood surrounding each keyframe.

In [127], a summary method called VGRAPH is suggested that uses both color and texture features. The video is partitioned into shots using the color features extracted from the color histogram computed from the HSV. The video key frames are extracted using a nearest neighbor graph built from the texture features extracted from the shots' representative frames using Discrete Haar Wavelet Transforms.

In [128], the fuzzy c-means clustering algorithm was used. The method began by breaking down the original footage into simple, controllable components such as shots or frames. Then, using a sample of frames, color features were extracted to produce a color histogram in the HSV color space. The fuzzy c-means clustering algorithm was then used to group these frames, and one frame from each cluster was picked. To make the output easier to understand, the chosen keyframes were put in the correct temporal order. FCM clustering has an advantage over competing techniques because it generates a membership matrix that directly determines the most representative frames of each cluster.

In [129], shot boundary detection is not necessary due to the implementation of the content-based adaptive clustering (CBAC) algorithm. This algorithm examines each frame of the video for content changes without segmenting the movie into shots. To decide which portions of the movie are crucial for description, the content changes are compared worldwide in progressively rising units. This method has been used to extract as few as 5% of the video's sample frames for histogram analysis. It can be used in applications that browse and summarize video information.

In [130], an improved video summary technique called Density-based Surveillance video abstraction (DbSva) is described to create a condensed version of the original video. The unique feature of DbSva is its use of the DENsity-based CLUstEring algorithm (DENCLUE) to apply the benefits of both local and global video content features, considerably enhancing the quality of summary videos.

In [131], an unsupervised static video summarizing technique that extracts keyframes representing the entire video is proposed. The technique uses a two-stream approach to collect motion and visual elements from the video. Multi-level feature extraction and fusion are used for the visual stream to take into account features from many levels of abstraction. By concentrating on both the coarse- and fine-grained aspects of the frames, the use of features from

several layers enables better frame representation. The final keyframes representing the video summary are then created by using neighborhood peak identification and redundancy removal techniques to the fused features.

In [132], the authors propose a novel generative adversarial framework consisting of an autoencoder long short-term memory network (LSTM) that selects video frames and then decodes the resultant summarization to reconstruct the input video. Another LSTM, the discriminator, is used to differentiate between the original video and its reconstruction from the summarized version.

In [133], a Temporal- and Spatial-Driven Video Summarization Using Optimum-Path Forest is suggested. The method for summarizing videos is based on clustering using the Optimum-Path Forest. The method considers the temporal data for video summarization. The clustering procedure is the focus of this work's key contribution.

In [134], a brand-new static video summarizing technique is proposed that formulates the issue as a clustering problem and creates a brand-new clustering algorithm. The method's four steps are pre-sampling, video frame representation, clustering, and result creation for video summarization.

4.3.2 Other methods

The authors in [135] proposes a video summarization technique that uses a temporal collaborative representation (TCR) model to select frames for video summarization. The TCR model takes into consideration the visual similarity of adjacent frames to avoid selecting transitional frames.

In [136] a new method for video summarization is proposed that uses a nonlinear sparse dictionary selection model. The model projects the video into a high-dimensional feature space using a kernel function, which induces nonlinearity between video frames and converts it to linearity. The suggested technique outperforms several state-of-the-art algorithms on two distinct benchmark video datasets.

In [137] a video summarization method is proposed that uses a mixture of color information collected from video frame patches instead of using an entire video to identify shot boundaries. The frames are further broken down into subshots by evaluating their structural similarities, and a keyframe is extracted from the subshot that best represents the unique video shot. Finally, redundant frames are removed by comparing each of the keyframes that were recovered from the subshot of each video shot.

In [138] a new video summarization technique called VISCOM is introduced. VISCOM uses color co-occurrence matrices to describe video frames and produce a summary with the most representative frames. A quantitative criterion that requires little human input and strikes a good balance between computation speed and result quality is used to evaluate the method. VISCOM can be used by frameworks that deal with video indexing, retrieval, and browsing.

In [139], the authors propose a novel subset selection method that uses human-written summaries as supervision to automatically carry out keyframe-based video summarization. The key concept is the nonparametric transfer of summary structures from annotated videos to test videos that have not yet been viewed. To direct the transfer process using training videos that are semantically consistent with the test input, the authors demonstrate how to adapt their method to leverage semantic information about the category or genre of the video. Additionally, they demonstrate how to expand their approach to subshot-based summarization, which offers more flexible ways of defining visual similarity across subshots spanning many frames and minimizes computing costs.

4.4 Dynamic video summarization

Dynamic video is the process of creating a condensed version of a video without losing its meaning. This is useful for industries that produce large amounts of video data, such as education, entertainment, surveillance, and information archiving. Video skimming can make video watching, navigation, and storage easier by reducing the amount of time it takes to consume video data. This section discusses some of the most recent techniques for summarizing dynamic videos and explains how they can be used without losing the original video's meaning.

4.4.1 Visual-textual and audio based methods

Dynamic video summarization is more engaging than static video summarization because it retains the video's original movement and audio. This makes the summarization more realistic, which can improve the viewer's understanding of the video. When subtitles or spoken text are available in a video, a written summary can be combined with the visual summary to create a more comprehensive and informative summary.

The authors of [140] propose a bottom-up video summarization method that identifies perceptually relevant video events by combining the saliency patterns of audio, visual, and textual information in a video. The method creates a single attention curve by combining multiple

modality curves, which represent the presence of an event in one or more domains. The algorithm that improves the output of monomodal or audiovisual skimming is based on the multimodal saliency curve. The algorithm successfully summarizes the information and entertainment value of a video.

Dynamic summaries of video and audio data are created in [141]. Automatic speech recognition (ASR) is used to extract speech transcripts from the audio. The video is then segmented into n-gram-based sections using this textual data. The significance of each section is assessed using a modified Maximal Marginal Relevance (MMR) method. The informational value of the sections is used to calculate the MMR.

In [142], we propose a method for summarizing videos based on visual and segment-level diversity using pre-trained deep visual and category models. The method uses a pre-trained deep convolutional network (DCN) and a pre-trained word integration matrix to extract visual and category information. The method then estimates the diversity of a video using visual and categorical data and uses that estimate as a significance score to select the input video segments that best fit that description. The method also allows for queries to be performed throughout the search process in order to tailor the resulting video summaries to the specific goals of the user.

The authors in [143], propose a method for automatically creating video summaries from texts retrieved by automatic speech recognition (ASR). The authors first use pause detection to partition the entire program into segments. They then calculate a score for each segment based on the frequency of words and capital letters in that section. Finally, a summary is produced by selecting the segments with the highest duration percentage scores and expanding summary coverage across the entire program.

This research [144] introduces a novel unsupervised framework for simultaneously learning from independently collected visual and non-visual data sources to identify significant latent structure in surveillance video data. The proposed multi-source learning system can accurately infer missing invisible semantics from previously viewed video, in addition to producing better video content synthesis than current methods. The effectiveness of the final video produced using the proposed multi-source approach was evaluated through a comprehensive user study.

In [145] proposes a Deep Summary Network (DSN) for summarizing films by conceptualizing video summarization as a sequential decision process. DSN predicts the probability of selecting a frame for each video frame and then decides which frames to choose to create a video summary based on the probability distributions. The authors propose an end-to-end reinforcement learning architecture for training DSN, which uses a novel reward function that

takes into account both the diversity and representativeness of the generated summaries. The reward function is completely independent of labels or user interactions, and it controls the diversity and representativeness of the generated summaries during training. DSN aims to increase rewards by producing more diverse and representative summaries.

The authors in [146], use supporting text in a vlog post to create a video summary that accurately reflects the author's goals. This is because the supporting text often highlights the key points of the video, such as specific events or objects. The technique first creates shorter snippets of the video and annotates them with object tags. These object tags and the words in the supporting text are then combined to represent the video and text, respectively. The technique then selects a subset of the videos that contain interesting objects by determining the relevance of the objects in the videos to the text.

4.4.2 Event or object based methods

The authors in [147], present a method for creating a video summary by identifying the most instructive segments of a video. The method, called the Contextual Video Abstraction Framework (CAVS), uses an extended sparse lasso set and a sparse coding methodology to learn two dictionaries: a dictionary of video features and a dictionary of spatial and temporal association graphs. The feature dictionary captures the global relationships between areas of motion in the video, while the association graph dictionary captures the local relationships between features. Both dictionaries are updated online each time CAVS analyzes a new video.

In [148], describes a novel Bayesian network-based approach for automatically recognizing and summarizing events in soccer videos. The approach uses effective methods for extracting mid-level visual features, classifying snapshot widths, detecting snapshot borders, and creating associated Bayesian networks. The approach has three main stages: snapshot boundary detection, feature extraction, and Bayesian network creation.

The authors in [149], propose a new method for identifying and creating cricket highlights. The method first extracts exciting audio snippets using audio features. Then, each clip is analyzed to extract significant events, such as replays, players, umpires, spectators, and player gatherings. A hybrid deep neural network with emperor penguin optimization (HDNN-EPO) is proposed to automatically categorize excitement cues in a cricket video based on observable events. Finally, the selected, chronologically-ordered concepts from the importance ranking are grouped to create highlights.

The researchers in [150], propose a model for automatically summarizing videos of anoma-

lous events. This is important for the organized storage, quick navigation, and retrieval of a variety of video data without losing important details, as it is portable. The proposed method uses two steps: a machine learning algorithm to identify significant events and a deep learning algorithm to remove unnecessary frames. This results in a summary video that highlights anomalous activity.

This research [151] proposes a dashboard detection-based technique for automatically extracting a major event and following summary from sports videos. The You Only Look Once (YOLO) object recognition algorithm was trained using a database of 1,300 images. YOLO was used to identify the dashboard in each frame of the video. The dashboard was then removed from the image. The trimmed dashboard was then processed using image processing techniques to reduce noise and false positives. The processed image was then analyzed by an Optical Character Recognition (OCR) tool to extract the text. The OCR output was then processed by a rule-based system to create timestamps for major game events.

4.5 challenges and future scope

Videos are made up of a hierarchy of scenes, snapshots, and images. The final video is a compilation of these key parts. As a result, both static and dynamic summaries can be used to provide users with a brief overview of a longer video. Static summaries are most beneficial to users who want to quickly scan a video for key information. Dynamic summaries, on the other hand, are more adaptable because they can be customized to meet user requirements. The most significant accomplishments of the presented study is the identification of several challenges and potential directions for future research in the field of video summarization. The identified gaps and future recommendations will assist researchers in making progress and improving in this field.

Video summarization presents the following challenges:

- Adopting deep learning for event recognition in video summarization is challenging because existing deep learning methods are less reliable than building new event detection systems.
- Most current pre-trained convolutional neural networks (CNNs) are designed for other computer vision tasks and trained on other datasets. They limit the range of features available for summarization when used to represent frames in video summarization, which can affect the quality of the summaries generated. An adaptive deep neural network module is required to capture spatio-temporal and semantic information.

- Videos are multidimensional, containing data, such as motion, sound, and images. Each of these data is important for understanding the video, so multi-model processing is required. However, most recent studies on video summarization only use visual data, ignoring the other modalities. This is a problem because visual data alone cannot fully capture the complex concepts present in videos. Audio, text, and motion data also play an important role in selecting critical images. Multi-model approaches are still in their early stages of development because current systems lack an effective way of integrating the multiple data required for video summarization. Additionally, there is no publicly available multi-model video summarization dataset that can be used to train and evaluate existing multi-model approaches.

4.6 Conclusion

In this chapter, we focus on video summarization. We found that there are two main types of summaries: static and dynamic. We discussed several state-of-the-art methods for summarizing videos. Based on this, we were able to evaluate a number of challenges and limitations. We also considered the advantages and disadvantages of different approaches.

In the following chapter, we present our first contribution to this thesis. We aim to develop a model for SBD that learns a deep network to discriminate input frames through a similarity measurement. This work is an image similarity model used to detect cut transitions from video frames.

Part II

Proposed Approaches and Results Validation

Chapter 5

Shot boundary detection approach using deep learning

Contents

5.1 Introduction

The rapid growth of multimedia technologies has resulted in a significant increase in multimedia data, particularly digital video. This has created a demand for innovative technologies and automatic tools that can efficiently retrieve, summarize, browse, index, and manage video sequences.

Shot boundary detection (SBD) is the first and most crucial step in video processing. It involves extracting features from video frames and detecting shots based on the differences be-

tween these features. There are two types of VSBD: cut transition (CT) and gradual transition (GT). While CT may seem easy to detect, there are many challenges such as sudden changes in lighting, object/camera motion, camera operation, and similar backgrounds that can result in high false-positive rates [90]. The accuracy of VSBD depends on the effectiveness of the extracted features in representing the visual content of video frames and the computational cost of the algorithm used [86].

This work aims to develop a novel deep neural network model for video shot boundary detection (SBD). The model uses a Long Short-Term Memory (LSTM) network to classify video frames as similar or not similar, and it only considers high-level features extracted from a pre-trained convolutional neural network (CNN) model (InceptionV3). The model also addresses the problem of determining the cut detection threshold using machine learning, and it aggregates information from distances between CNN values and LSTM-extracted features to improve performance. Finally, the shot boundary detection algorithm is applied based on the proposed similarity model [152].

The rest of the chapter is structured as follows: The proposed methodology is detailed in Sections 5.2 and 5.3. Experimental results and analysis are presented in Section 5.4, which also discusses the performance of the proposed method. Finally, concluding remarks are provided in Section 5.5.

5.2 Learning image similarity model

Traditional image similarity metrics are limited by their dependence on thresholds and inability to learn from data. Recent studies have shown that learned similarity metrics can significantly improve performance in classification, clustering, and retrieval tasks. Measuring similarity between two images involves two components: the first extracts features from an input image and represents it as a multi-dimensional feature vector, while the second is a similarity function measure that can vary depending on the types of features used. These two components are combined in a Siamese network, which is a method for learning feature similarity by optimizing the feature distance of image pairs [153]. The network consists of two parallel branches of convolutional neural networks (CNN) with identical architecture and weights. (See Figure 5.1).

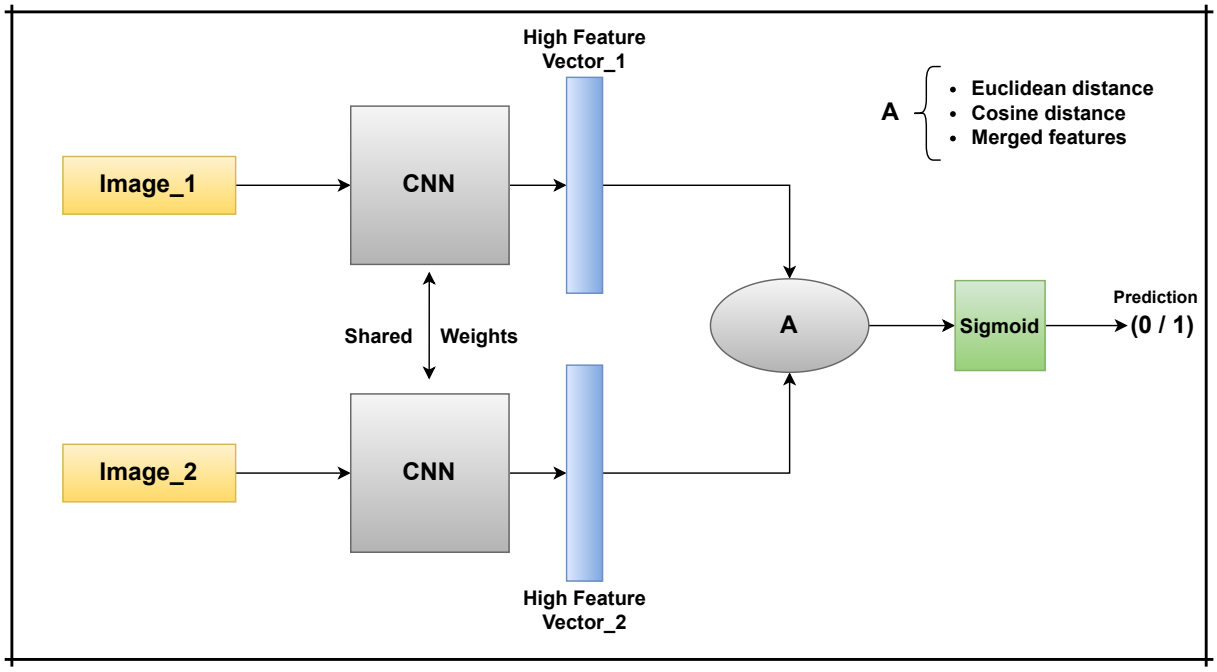


Figure 5.1: Siamese network architecture

5.2.1 Features Extraction Process

Deep CNN models are used for feature extraction with the help of transfer learning because they effectively capture the semantics of images and are stable against deformations such as translation, zooming, and tilting. These models have achieved excellent performance in image classification, object detection, and feature extraction tasks, making their features suitable for measuring similarity in an embedding way. The InceptionV3 model is a widely used neural network for both image classification and feature extraction. It has over 78% accuracy on the ImageNet dataset and is known for its speed and precision. The processing produces high-level features obtained from the last fully connected layer of InceptionV3, known as Block10 [154].(See Figure 5.2).

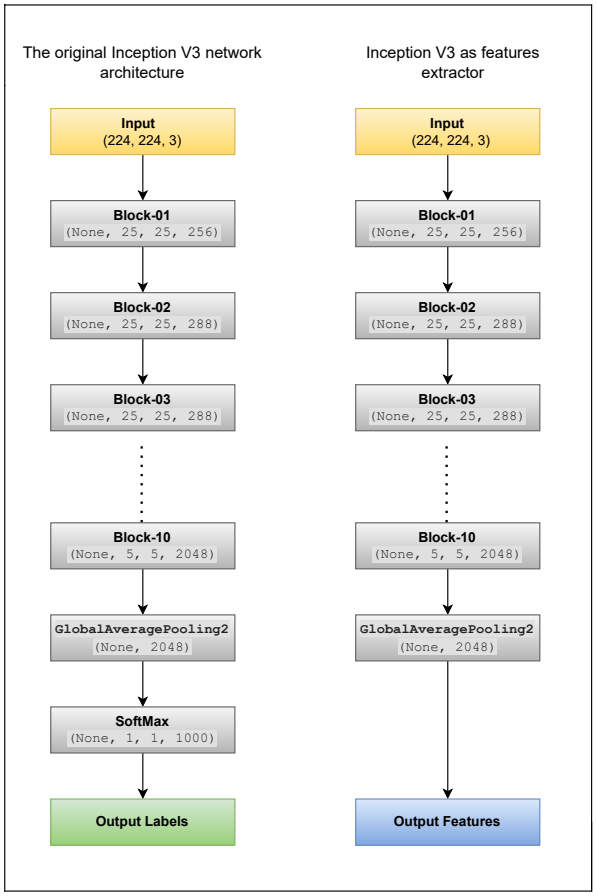


Figure 5.2: InceptionV3 model as feature extractor

Figure 5.3 shows our model workflow. We use two pre-trained deep convolutional neural networks (inceptionV3) to extract features from two input frames, f_1 and f_2. The extracted features $h(f_1)$ and $h(f_2)$ are then combined together and used as input to both the Euclidean distance layer and LSTM layers. Through the euclidean distance layer, we obtain a vector Z_d expressed as follows:

$$Z_d = \|h(f_1), h(f_2)\|_2 \tag{5.1}$$

On the other hand, using the LSTM model equations produces the vector Z_l . After processing the concatenated vector through a number of dense layers (FC), the output of the merged layer is used as input to the classification layer. The prediction is formulated more accurately as follows:

$$y_{pred} = \phi(W^{[l]}.h^{[l]}(Z_d + Z_l) + b^{[l]}) \tag{5.2}$$

The last layer l of the network uses the Sigmoid activation function ϕ to provide a metric for the learned feature space of the hidden layers. When two input frames are categorized into the same category, their similarity should be close to zero. Otherwise, their similarity should be

close to one. Formally, we have::

$$y_{sim} = \begin{cases} 0 & \text{if } y(f_1) = y(f_2) \\ 1 & \text{otherwise} \end{cases}$$

Where $y(f_1)$ and $y(f_2)$ denote the class labels associated to the input frames f_1 and f_2

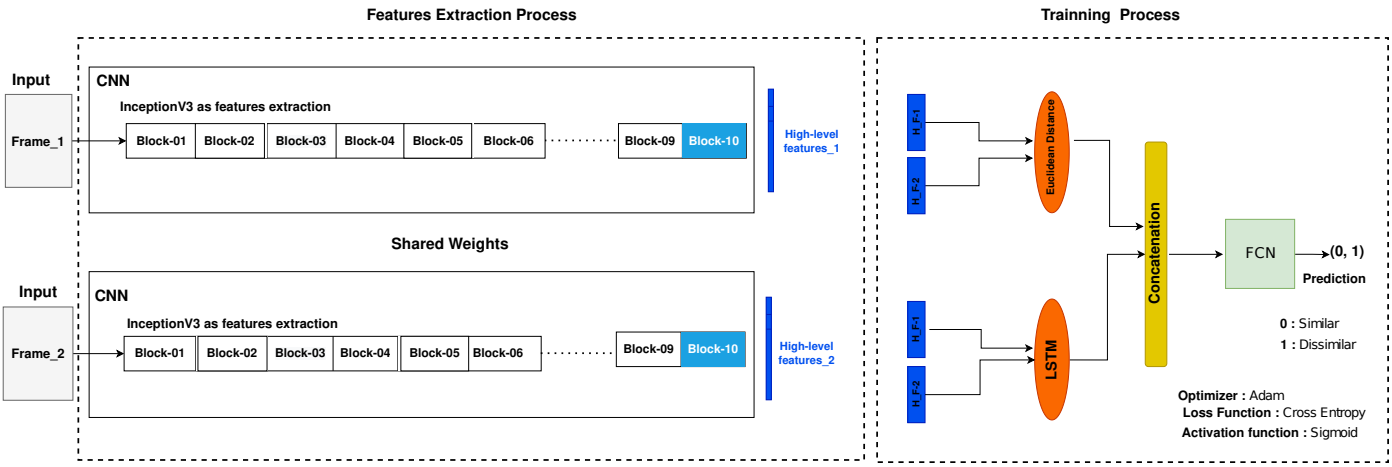


Figure 5.3: Learning image similarity model

5.2.2 Training Process

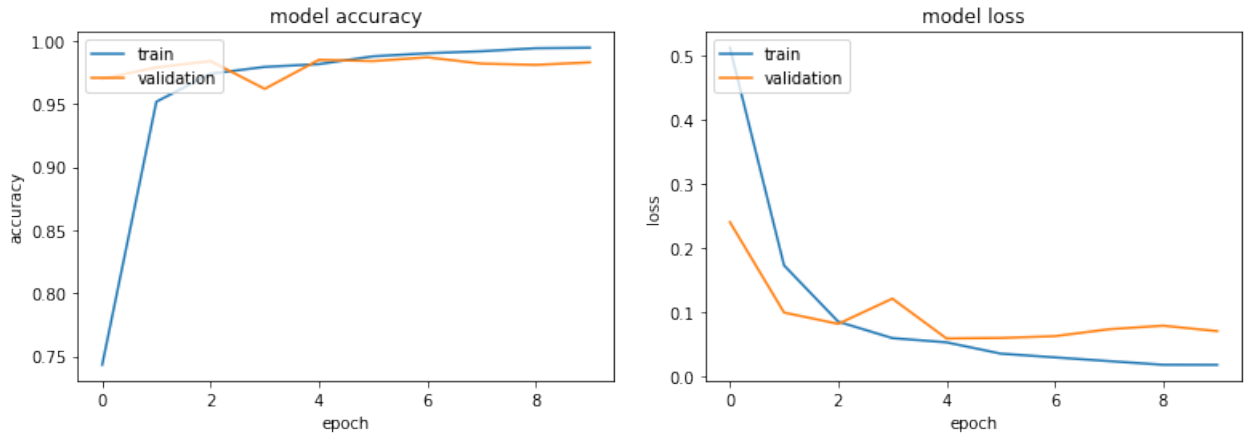
The model is trained using a new database containing 5000 frame pairs with two types of labels (similar and dissimilar), and 1000 frame pairs for testing were generated from online videos. Only the LSTM and fully connected layers are trained, while the CNN blocks are not. This reduces resource consumption, the number of parameters, and computational time. After a feed-forward pass where data flows through all the blocks according to the model's structure, the computed error is gradually minimized by applying the binary-cross-entropy formula using the following loss function:

$$Loss(f_1, f_2) = y_{sim}(f_1, f_2).log(y_{pred}) + (1 - y_{sim}(f_1, f_2)).log(1 - y_{pred}) \tag{5.3}$$

The training and validation loss of the model converge to a point of stability with a minimal gap between the final values, indicating a good fit as shown in Figure 5.4. The model achieves 98.3% accuracy, 98% precision, 98% recall, and 98% f1 score. (see Tab 5.1)

Table 5.1: Report on classification

	Precision	Recall	f1_score	Support
Class 0	0.98	0.98	0.98	507
Class 1	0.98	0.98	0.98	493



(a) (b)

Figure 5.4: Accuracy and loss for our model

5.3 A new shot boundary detection algorithm

The model introduced in the previous section demonstrates that the similarity function can be learned directly from image pairs. Using this model, it is possible to determine whether a pair of frames represents a cut transition. The overall method is illustrated in the block diagram shown in Figure 5.5.

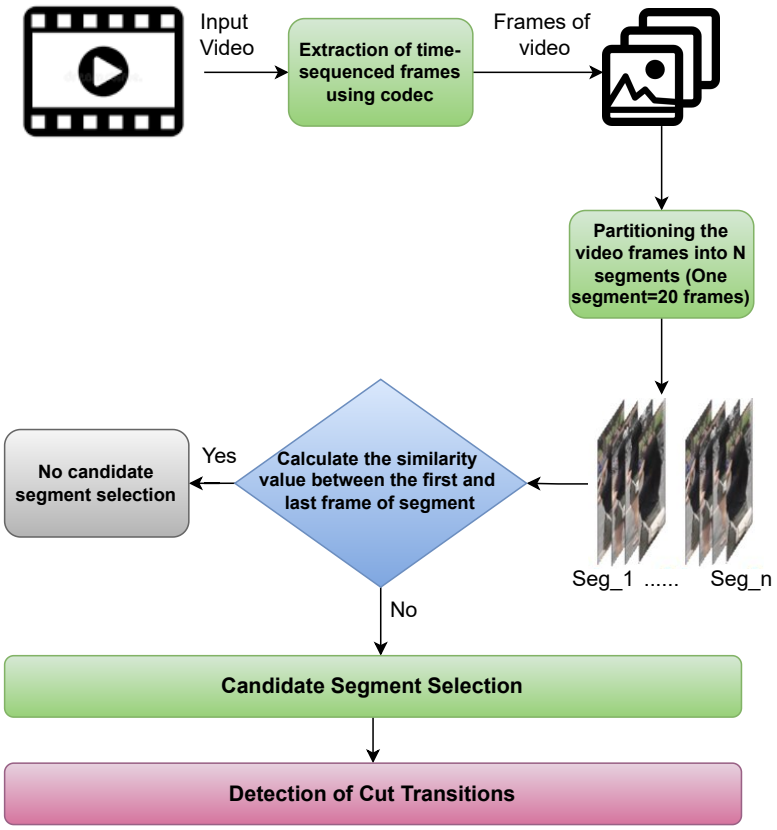


Figure 5.5: Diagram of the proposed method

The method was created using two main steps:

5.3.1 Candidate segment selection

The proposed model scans the video frame sequence to identify segments that may contain shot boundaries. It then skips segments that do not contain shot boundaries, which improves processing time. This concept was previously proposed in [97], [155], and [120] with an adaptive threshold mechanism for obtaining video frame segments. The proposed model does not require a threshold calculation, which makes it more efficient. Video sequences are divides into N segments of 20 frames each. For each segment, it calculates the similarity between the first and last frame. If the output value is 1, the segment is a candidate for shot boundary detection (SBD).

5.3.2 Shot transition detection

The proposed model attempts to detect cut transitions in each candidate segment by calculating the similarity of each successive frame pair. A cut transition is detected if only one different frame pair is found within the segment. Otherwise, no transition is detected. This method is

outlined in Algorithm 1. Figure 5.6 shows three segments where cut transitions are detected only in the first and second segments.

Figure 5.6: Cut and no cut transition detection

Algorithm 1 Proposed SBD algorithm

F: Set of frames in video v; $F = \{f_1, f_2, ..., f_m\}$, with m: number of frames;
S: Set of segments in video v; $S = \{s_1, s_2, ..., s_n\}$, with n: number of segments;
p: Length of segment with $p = 20$;
$y_{sim}(f_a, f_b)$: Similarity between frames f_a and f_b (our proposed similarity metric);
tab_sim: Set of similarity values between frames;
function CUT_TRANSITION_DETECTION(S)
    for $i = 1 \rightarrow n$ **do**
        if $y_{sim}(f_{p(i-1)+1}, f_{p(i-1)+p})$=False **then**
            $tab\_sim \leftarrow []$;
            for $j = 1 \rightarrow (p-1)$ **do**
                $tab\_sim[j] \leftarrow y_{sim}(f_{p(i-1)+j}, f_{p(i-1)+j+1})$;
            end for
            if only one False **in** tab_sim **then**
                Cut transition detected;
            end if
        else
            No transition;
        end if
    end for
end function

5.4 Experimental results and discussion

5.4.1 TRECVid 2001, 2007 datasets

The proposed system was tested using the TRECVid 2001 and TRECVid 2007 video datasets. The videos provided by these datasets are MPEG compressed and were downloaded from the Open Video Project. Table 5.2 provides a detailed description of selected videos from the TRECVid database.

Table 5.2: Description of TRECVid 2001 and TRECVid 2007 datasets

Video sources	**Videos**	**Frames**	**Types**	
			CT	GT
TRECVid 2001	D2	16586	42	31
	D3	12304	39	64
	D4	31389	98	55
	D5	12510	45	26
	D6	13648	40	45
TRECVid 2007	BG-3027	49815	126	1
	BG-3097	44987	91	-
	BG-16336	2462	20	-
	BG-28476	23238	176	1
	BG-36136	29426	88	12
	BG-37309	9639	11	8
	BG-37770	15836	8	27

5.4.2 RAI database

The proposed model was also tested on the publicly available RAI database to assess its generality in the context of shot boundary detection. This database consists of ten randomly selected broadcasting videos from the Rai Scuola video archive, including documentaries and talk shows [156]. It contains 724 cuts and 263 gradual transitions, with shots manually annotated by a set of human experts. The proposed method achieves competitive results compared to other methods [110], [109] and [108].

5.4.3 Performance evaluation

This paper uses standard metrics for evaluating different SBD methods, including precision, recall, and F1-score. Table 5.3 shows the overall performance of the proposed system on the TRECVid 2001 and 2007 datasets. Figures 5.7 and 5.8 show the visual results of the proposed technique correctly detecting cut transitions on the TRECVid 2001 dataset and RAI database, respectively. Additionally, we used the RAI database to test the performance of our system. Table 5.4 shows the obtained results.

5.4.4 Experiments on TRECVid 2001, 2007 datasets

The proposed approach for cut transition detection was tested on the TRECVid 2001 and TRECVid 2007 datasets, with results shown in Table 5.3. The system's effectiveness and superiority were evaluated by comparing it to state-of-the-art SBD techniques. Tables 5.5 and

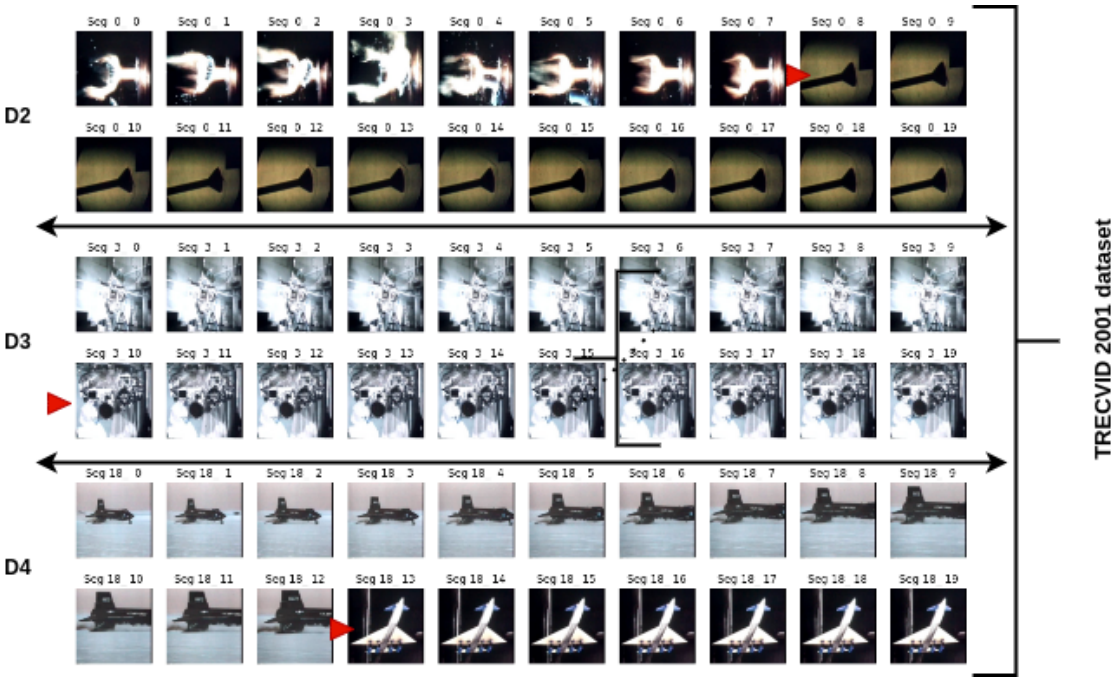


Figure 5.7: The visual results on TRECVid 2001 datasets

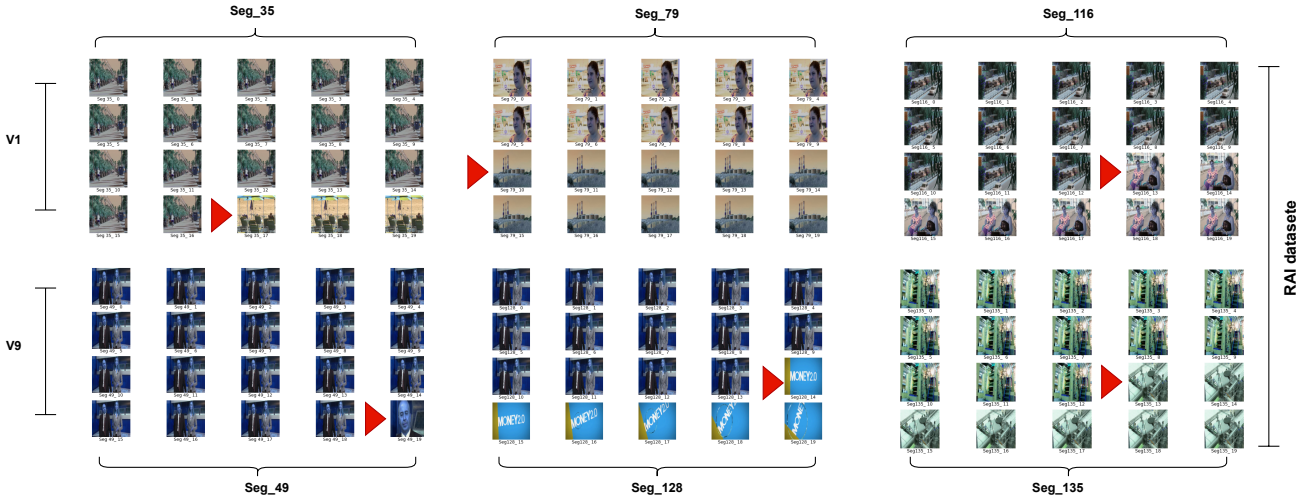


Figure 5.8: The visual results on RAI datasets

Table 5.3: Proposed system results for TRECVid 2001 and 2007 datasets

Video sources	**Videos**	**Our method**		
		P	R	F1
TRECVid 2001	D2	97.6	97.6	97.6
	D3	100	94.8	97.3
	D4	98.9	96.9	97.8
	D6	97.5	100	98.7
	Average	98.5	97.3	97.8
TRECVid 2007	BG-3027	97.6	99.2	98.4
	BG-3097	91.3	92.3	91.8
	BG-16336	100	95.0	97.4
	BG-28476	96.1	98.3	97.2
	BG-36136	95.5	97.7	96.6
	BG-37309	100	100	100
	BG-37770	100	100	100
	Average	97.2	97.5	97.3

Table 5.4: Proposed system results for RAI dataset

Video sources	**Videos**	**Our method**		
		P	R	F1
RAI	V1	0.87	0.85	0.86
	V2	0.90	0.95	0.92
	V3	0.96	0.98	0.97
	V4	0.98	0.98	0.98
	V5	0.96	0.94	0.95
	V6	0.91	0.96	0.93
	V7	0.96	0.98	0.97
	V8	0.95	0.94	0.94
	V9	0.97	0.97	0.97
	V10	0.95	0.95	0.95
	Average	0.941	0.95	0.944

5.6 show the results of these comparisons on the TRECVid 2001 and TRECVid 2007 datasets, respectively. The overall performance is measured by the F1 score, which combines recall and precision values. For cut transition detection, the proposed method's average F-score is higher than that of other methods, although some methods perform better in terms of precision or recall but not both.

In previous works [117] and [157], a threshold parameter was used to experiment with cut transition detection. However, the proposed system effectively handles this without imposing such a constraint. A closer examination of Table 5.6 reveals that the F1 score for all video sequences is improved due to the increased precision of the proposed system. An average F1

score of 97.3%, with a precision of 97.2% and a recall of 97.5%, was achieved using the proposed framework for cut transition detection. Visual results shown in Figure 5.7 demonstrate the robustness of the proposed technique, as the system correctly identifies cut transition positions despite various disturbances.

5.4.5 Experiments on RAI dataset

The performance of the proposed shot boundary detection method was evaluated on a sequence of videos from the RAI database, with results shown in Table 5.4. The proposed method achieved better results than other methods for shot boundary detection in some papers in terms of recall and F1 score, except for precision. (See Table 5.7.)

The tables 5.5, 5.6 and 5.7 show that the proposed method achieved the highest overall F1 score on all selected videos from the TRECVid 2001, 2007, and RAI databases. The performance is illustrated graphically in Figures 5.9 and 5.10.

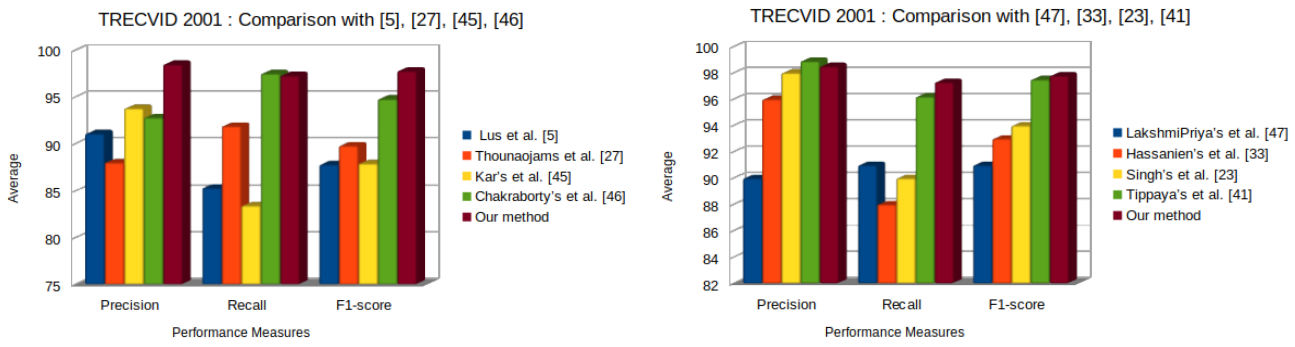


Figure 5.9: Comparative results on TRECVid 2001 dataset

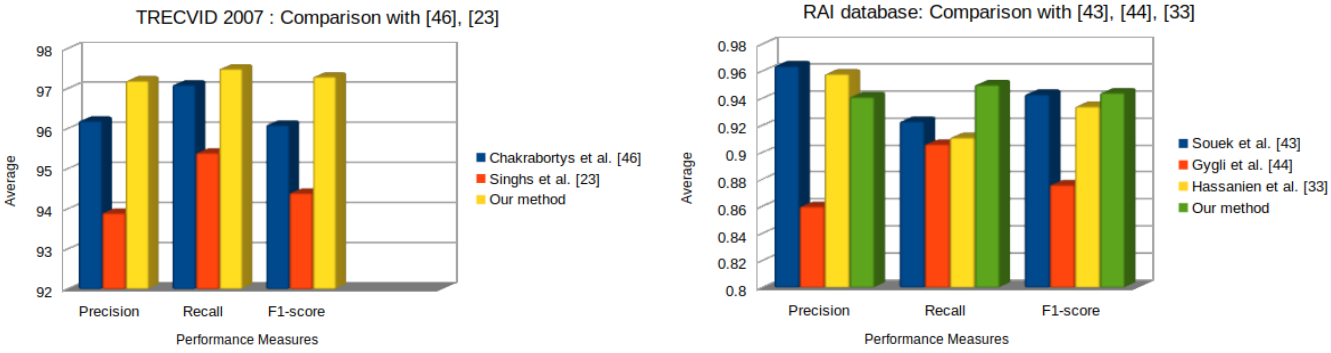


Figure 5.10: Comparative results of proposed method on TRECVid 2007 and RAI datasets

5.5 Conclusion

This paper presents a new shot boundary detection algorithm called SBD that uses image similarity model proposed. The similarity model is learned with a Siamese recurrent architecture and uses a pre-trained InceptionV3 model and an LSTM model to extract features. These features are then fused and classified by the final layer of the model. The segment selection process is used to predict shot boundaries. The SBD method was validated on several databases and compared to other SBD methods. The proposed method outperformed other methods in terms of detection rate.

Table 5.5: Comparison of the proposed method with other systems on the TRECVid 2001 video dataset.

Methods	**Videos**	**CT**		
		P	R	F1
Lu et al.[97]	D2	90.5	90.5	90.5
	D3	86.7	66.7	75.4
	D4	89.7	88.8	89.2
	D6	97.4	95	96.2
	Average	91.1	85.3	87.8
Thounaojam et al.[116]	D2	88.9	95.2	91.9
	D3	80.5	84.6	82.5
	D4	93.5	87.8	90.6
	D6	88.9	100	94.1
	Average	88.0	91.9	89.8
Kar et al.[158]	D2	94.4	81.0	87.2
	D3	100	82.1	90.1
	D4	96.5	78.1	86.3
	D6	84.1	92.5	88.1
	Average	93.8	83.4	87.9
Chakraborty et al.[117]	D2	82.0	97.6	85.1
	D3	100	92.3	96.0
	D4	89.1	100	94.2
	D6	100	100	100
	Average	92.8	**97.5**	94.8
LakshmiPriya et al[159]	D2	85.0	97.0	91.0
	D3	86.0	82.0	84.0
	D4	90.0	88.0	89.0
	D6	97.0	95.0	96.0
	Average	90.0	91.0	91.0
Hassanien et al.[108]	D2	87.0	89.0	88.0
	D3	100	92.0	96.0
	D4	100	85.0	92.0
	D6	100	87.0	93.0
	Average	96.0	88.0	93.0
Singh et al.[157]	D2	100	90.0	95.0
	D3	100	89.0	94.0
	D4	94.0	89.0	92.0
	D6	97.0	92.0	94.0
	Average	98.0	90.0	94.0
Tippaya et al[120]	D2	97.5	92.9	95.1
	D3	100	94.9	97.4
	D4	97.9	96.9	97.4
	D6	100	100	100
	Average	**98.9**	96.2	97.5
Our method	D2	97.6	97.6	97.6
	D3	100	94.8	97.3
	D4	98.9	96.9	97.8
	D6	97.5	100	98.7
	Average	98.5	97.3	**97.8**

Table 5.6: Comparison of the proposed method with other systems on the TRECVid 2007 dataset

Methods	**Videos**	**CT**		
		P	R	F1
Chakraborty et al.[117]	BG-3027	96.9	100	98.4
	BG-3097	89.5	93.4	91.4
	BG-16336	100	90	94.7
	BG-28476	98.3	100	99.2
	BG-36136	100	96.6	98.3
	BG-37309	100	100	100
	BG-37770	88.9	100	94.1
	Average	96.2	97.1	96.6
Singh et al.[157]	BG-3027	95	95	95
	BG-3097	95	91	93
	BG-16336	100	90	94
	BG-28476	85	92	89
	BG-36136	98	100	99
	BG-37309	84	100	91
	BG-37770	100	100	100
	Average	93.9	95.4	94.4
Our method	BG-3027	97.6	99.2	98.4
	BG-3097	91.3	92.3	91.8
	BG-16336	100	95.0	97.4
	BG-28476	96.1	98.3	97.2
	BG-36136	95.5	97.7	96.6
	BG-37309	100	100	100
	BG-37770	100	100	100
	Average	**97.2**	**97.5**	**97.3**

Table 5.7: Comparison of the proposed method with other systems on the RAI dataset.

Videos	Method [110]			Method [109]			Method [108]			Our Method		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
V1	0.96	0.71	0.82	0.89	0.84	0.86	0.84	0.82	0.83	0.87	0.85	0.86
V2	0.96	0.90	0.93	0.95	0.99	0.97	0.93	0.84	0.88	0.90	0.95	0.92
V3	0.96	0.99	0.97	0.91	0.97	0.94	0.99	0.90	0.94	0.96	0.98	0.97
V4	0.92	0.98	0.95	0.92	0.99	0.95	0.96	0.98	0.97	0.98	0.98	0.98
V5	0.92	0.97	0.94	0.94	0.94	0.94	0.97	0.97	0.97	0.96	0.94	0.95
V6	0.94	0.98	0.96	0.97	0.96	0.96	0.92	0.94	0.93	0.91	0.96	0.93
V7	0.99	0.94	0.96	0.93	0.94	0.93	0.97	0.96	0.96	0.96	0.98	0.97
V8	0.97	0.92	0.95	0.81	0.64	0.72	0.97	0.87	0.92	0.95	0.94	0.94
V9	0.96	0.90	0.93	0.62	0.90	0.73	0.98	0.98	0.98	0.97	0.97	0.97
V10	1.00	0.90	0.95	0.66	0.89	0.76	1.00	0.93	0.96	0.95	0.95	0.95
Avg	**0.96**	0.92	0.94	0.86	0.90	0.87	0.95	0.91	0.93	0.94	**0.95**	**0.94**

Chapter 6

Static video summarization based on clustering and shot boundary detection

Contents

6.1 Introduction

In this chapter, we present our system for summarizing a static videos by extracting representative key-frames. This chapter begins with a detailed description of the proposed video summarization process. This process relies on feature extraction with a pre-trained model (Inception-V3) and the DBSCAN clustering algorithm to identify the best key-frames. To improve the efficiency of this system, we present a genetic algorithm that optimizes the DBSCAN hyper-parameters.

6.2 Key-frame extraction method for static video summarization

Our system is composed of three main steps: 1) preprocessing, 2) feature extraction and selection, and 3) clustering.

First, we use our shot boundary detection method (described in Chapter 4) to extract shots from the video. Next, we select candidate frames for each shot and apply Inception-V3 to obtain a new feature representation. We then reduce these features using principal component analysis (PCA). Finally, we apply the DBSCAN method to extract keyframes and make the process more efficient by proposing a genetic algorithm for optimizing the DBSCAN hyperparameters. Algorithm 2 describes the steps of this process and Figure 6.1 summarizes the proposed static video summarization system.

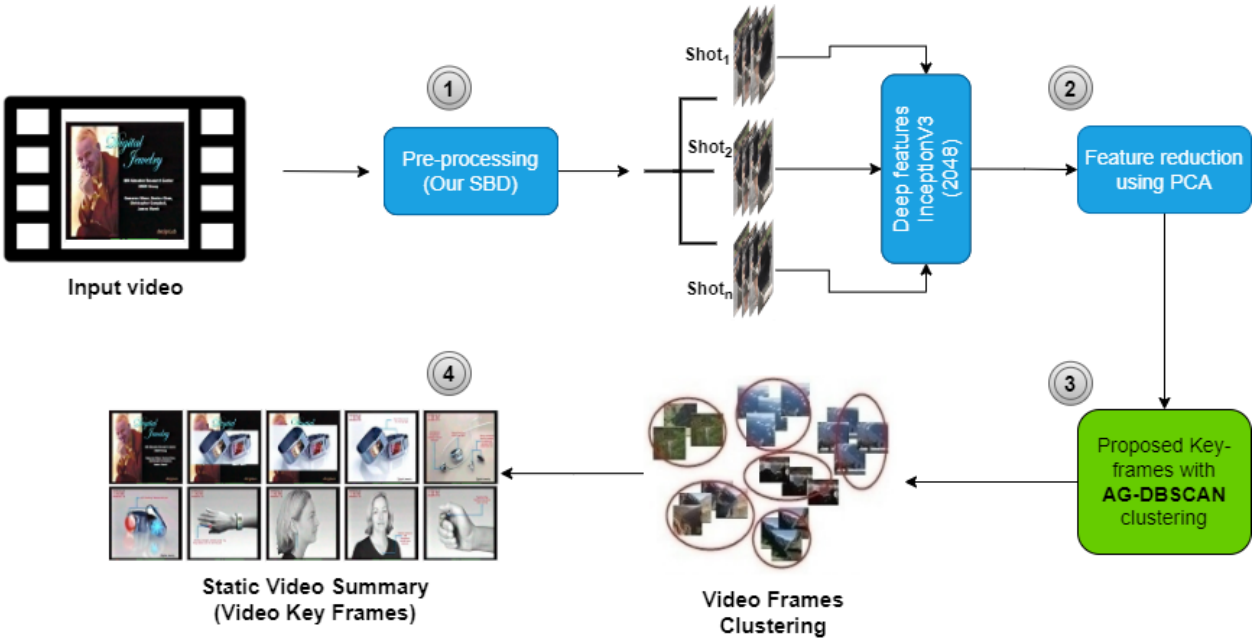


Figure 6.1: Proposed method for static video summarization

6.2.1 Pre-processing using Shot boundary detection

The scanning of the video frames sequence aims to locate segments that may include shot boundaries and skip those that do not. This reduces the processing time by only processing a few candidate segments. We use our SBD method, which is described in Chapter 4, to identify these segments. Our SBD method is based on frame similarity detection. To decide whether two frames are similar, we extract spatial and temporal features from each frame using

Algorithm 2 Proposed static video summarization

Input: Set of frames in video v; $F = \{f_1, f_2, ..., f_m\}$, with m: number of frames;
Output: Set of keyframes in video v; $KF = \{kf_1, kf_2, ..., kf_n\}$, with n: number of keyframes;
$Shots \leftarrow SBD(F)$ // $Shots = \{s_1, s_2, ..., s_z\}$ with 20 frames for each shot s
$Shots' \leftarrow rep\_frames(Shots)$ // $Shots' = \{s'_1, s'_2, ..., s'_z\}$ with 3 representative frames for each shot s'
$F' \leftarrow concat\_frames(Shots')$ // $F' = \{f'_1, f'_2, ..., f'_y\}$
foreach f **in** F' **do**
    $feature\_vect \leftarrow InceptionV3(f)$ // Feature extraction
    $feature\_pca \leftarrow PCA(feature\_vect)$ // Dimensionality reduction
end for
$P \leftarrow initial\_population()$
while true **do**
    $scores = []$
    foreach chromosome ch **in** population P **do**
        $Eps, minP \leftarrow get\_phenotype(ch)$ // get values of DBSCAN hyperparams from ch
        $clusters \leftarrow DBSCAN(Eps, minP, feature\_pca)$
        $scores[ch] \leftarrow FitnessSi\_eval(clusters)$
    end for
    $list\_fitness \leftarrow rank\_based(scores)$ // fitness evaluation of population
    $Eps_{op}, minP_{op} \leftarrow get\_op\_hparams(list\_fitness)$ // optimal hyperparams of best ch
    if $counter < 20$ **then** // stopping criterion after 20 generations
        while $|P| < |P_{new}|$ **do**
            $ch_1, ch_2 \leftarrow RW\_selection(P)$
            $ofs_1, ofs_2, ofs_3, ofs_4 \leftarrow Uniform\_crossover(ch_1, ch_2)$ // ofs means offspring
            $Mofs_1, Mofs_2, Mofs_3, Mofs_4 \leftarrow Mutation(ofs_1, ofs_2, ofs_3, ofs_4)$ // Mofs means offspring Mutated
            $P_{new} \leftarrow Mofs_1, Mofs_2, Mofs_3, Mofs_4$
        end while
        $P \leftarrow P_{new}$ // Generate new population P
        $P_{new} \leftarrow []$
    else
        $clusters_{op} = DBSCAN(Eps_{op}, minP_{op}, feature\_pca)$
        foreach $cluster$ **in** $clusters_{op}$ **do**
            $KF[cluster] \leftarrow get\_middle\_frame(cluster)$
        end for
        return KF
    end if
end while

a pre-trained model (Inception-V3) and an LSTM model, respectively. We then use a segment selection process to predict the shots.

Using all the frames of each segment to extract keyframes increases the computation time and affects the system performance. Therefore, it is important to select candidate frames from each segment. Since the frames of each segment are pseudo-similar, it is sufficient to select three frames that cover the whole segment. In this case, we choose the first, the middle, and

the last frames, as illustrated in Figure 6.2.

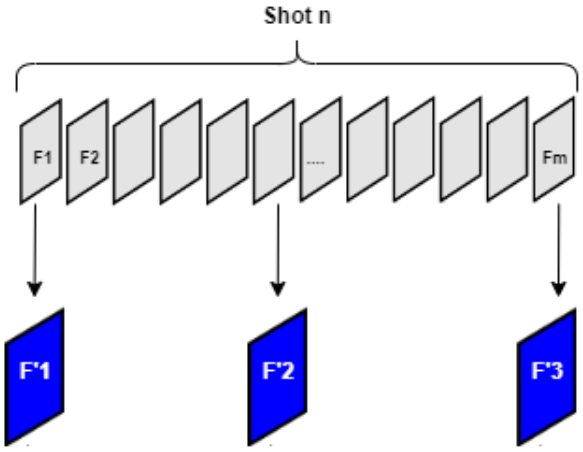


Figure 6.2: Selection of the frames representatives of the shot

6.2.2 Feature extraction and selection

Due to the large number of features (2048 for a frame video extracted by pre-trained model InceptionV3) and after video frames normalization, we reduce the number of features with principal component analysis (PCA). Our PCA model allows to keep 100% of information. For the videos frames dataset , the PCA model will reduce the feature vector size for each video frames from data. Exemple from OPV dataset where V69(92, 2048) after apply pca will be V69(92, 42) and V70 (36, 2048) after apply pca will be V70(36, 26). To predict the number of clusters of the video frames from this vector, we train an DBSCAN clustering. DBSCANs are well known for their efficiency in high dimensional spaces. The optimal DBSCAN hyperparameter values are obtained with proposed genetic algorithm.

6.2.3 Key-frame extraction with DBSCAN clustering

The main idea behind static video summarization is to group similar frames together. Clustering is one of the effective methods used to identify key-frames.

After extracting all candidate frames using our SBD method, the next step in our proposed method is to separate them into clusters. We used Density-Based Spatial Clustering of Applications with Noise (DBSCAN) based on Genetic Algorithm (GA) for this purpose. Unlike partitioning algorithms like k-means, DBSCAN doesn't require the number of clusters to be specified beforehand and has a notion of noise. This DBSCAN clustering has three types of data points namely core point, border point and outlier. Initially all the points are considered

as one cluster and based on the euclidean distance the clusters are formed [160]. This algorithm requires two input parameters: Epsilon (*Eps*) and minimum Points (*minPts*). *Eps* is the radius of the circle created around each data point to check the density, and *minPts* is the minimum number of data points required inside that circle for that data point to be classified as a core point [161].

Our work proposes a method for finding optimal hyperparameters for DBSCAN using a genetic algorithm to make the algorithm adaptive to the video content and help achieve good performance. After clustering the video frames, the final step is selecting key frames from the video clusters. In this step, noise frames are accepted because they represent unique frames. For each cluster, we select the middle core frame in the ordered frames sequence to construct the video summary. According to our experiments, we found that this middle core frame usually is the best representative of the cluster to which it belongs. The number of optimal clusters obtained from GA-DBSCAN are illustrated graphicall in Figure 6.3

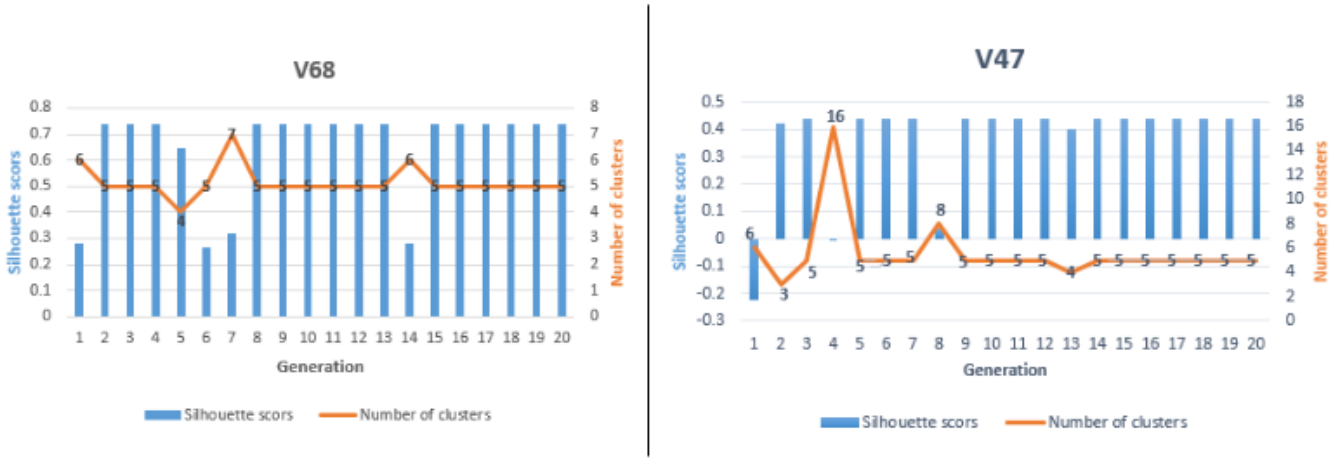


Figure 6.3: The optimal number of clusters extraction from V68 and V47 of OVP dataset

In section 6.2.4, we will describe our genetic algorithm named AG-DBSCAN for optimizing DBSCAN hyperparameters to maximize clustering accuracy.

6.2.4 GA-DBSCAN hyperparameter optimization with genetic algorithm

Evaluating all possible combinations of the hyperparameters *Eps* and *minPts* would be computationally expensive, as there are $2^{n_{minPts}+n_{Eps}}$ possible combinations. This is why we need a more efficient way to optimize the hyperparameters of DBSCAN. The genetic algorithm (GA) is a metaheuristic that can be used to explore large search spaces quickly, making it an ideal approach for problems such as DBSCAN hyperparameter optimization.

Figure 6.4 shows the proposed AG-DBSCAN architecture for optimizing the hyperparameters of DBSCAN. Genetic algorithms work by iteratively creating a population of solutions, evaluating each solution's fitness, and then selecting the best solutions to create the next generation of solutions. We detail the steps of our genetic algorithm and how we configured it.

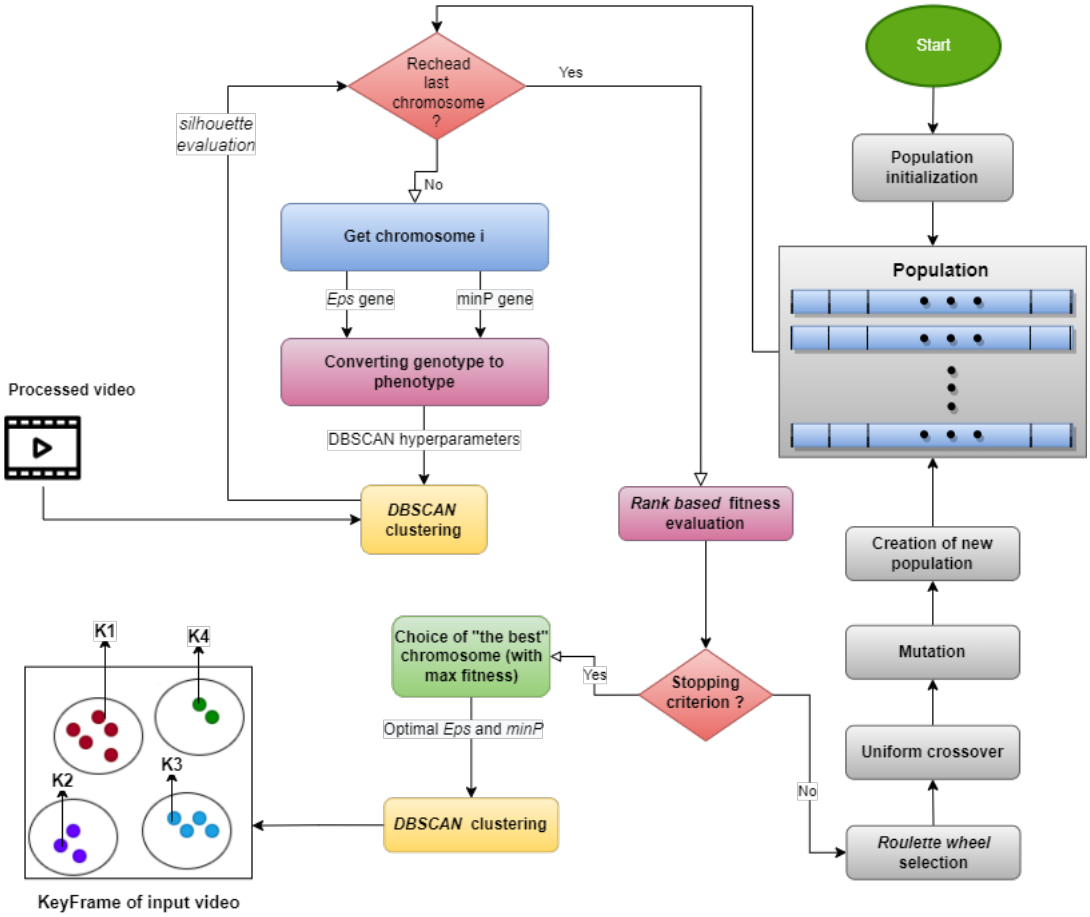


Figure 6.4: Proposed genetic algorithm (AG-DBSCAN)

- **Chromosome coding and population initialization:** The first step in genetic algorithm (GA) is to create and initialize a population of chromosomes randomly. Each chromosome consists of the Eps and $minPts$. Figure 6.5 shows the representation of a binary chromosome in our model, where n and m are the number of bits representing Eps and $minPts$, respectively. The binary strings that constitute the genotype of Eps and $minPts$ hyperparameters should be transformed into phenotype using Equation 6.1 [162].

$$p = \lfloor min_p + \frac{max_p - min_p}{2^l - 1} \times d \rfloor \tag{6.1}$$

where min_p and max_p are the minimum and maximum values of the hyper-parameters, d is the decimal value of the bit string, p is the phenotype of the bit string, and l is the bit string length.

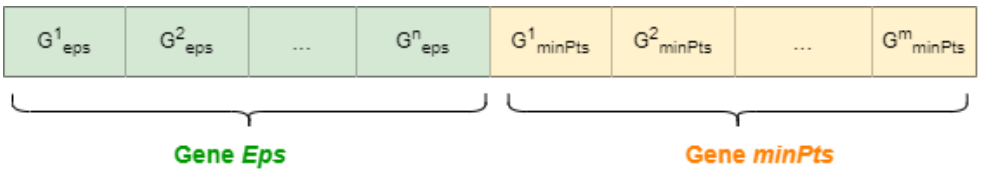


Figure 6.5: Binary coding of hyperparameters *Eps* et *minP*

The minimum and maximum min_p and max_p of each hyperparameter are described as follows:

- Minimum radius value (min_{Eps}): We set it with the minimum distance between two images of the input video.

$$min_{Eps} = argminDist(\{F_1, F_2\}, \{F_1, F_3\}, ..., \{F_i, F_j\}), i, j = n \tag{6.2}$$

- Maximum radius value (max_{Eps}): We set it with the maximum distance between two images of the input video.

$$max_{Eps} = argmaxDist(\{F_1, F_2\}, \{F_1, F_3\}, ..., \{F_i, F_j\}), i, j = n \tag{6.3}$$

- min_{minPts}: After several experiments, we decided to set this hyperparameter to 2 ($min_{minPts} = 2$).
- max_{minPts}: It is fixed with the quarter of the candidate frames of video. That is $max_{minPts} = \frac{F_{condidats}}{4}$

For the size of each gene, we fix n the size of *Eps* and m the size of $minPts$ by $\lfloor log_2(max_{Eps}) \rfloor$ and $\lfloor log_2(max_{minPts}) \rfloor$ respectively. Concerning the population size, it is equal to 10 times the chromosome size.

- **Fitness Assignment:** The next step is to calculate the fitness of each solution. The fitness of a solution is a measure of how good it is at solving the problem. To assign fitness values to chromosomes, we calculate the silhouette values of the DBSCAN model for each chromosome. The silhouette value is a metric that measures the quality of clustering by measuring both the separation between clusters and the cohesion within the clusters.

$$\mathcal{L}_{FitnessSi} = \frac{1}{N} \sum_{i=1}^{N} \frac{b_i - a_i}{max(a_i, b_i)}, -1 \leq \mathcal{L}_{FitnessSi} \leq 1 \tag{6.4}$$

where a_i is the average distance between frame i and all other frames in the same cluster,

and b_i is the minimum average distance between frame i and all other frames in different clusters, N is the number of clusters.

We use the Rank Based method [163] to assign fitness values, where each fitness value depends only on its position in the chromosome ranking. The fitness value assigned to each chromosome by the Rank Based method is:

$$\Phi(i) = \omega * R(i), i = 1..N \tag{6.5}$$

where ω is a constant called *selective pressure* and its value is set between 1 and 2 [163]. When the *selective pressure* is higher, the fittest chromosomes are more likely to cross over. The rank of chromosome i is denoted by $R(i)$. The value of ω is chosen to be 1.5 to calculate the *fitness* value.

- **Selection:** The third step in the genetic algorithm is to select the fittest chromosomes to create the next generation. The selection operator chooses chromosomes based on their fitness level, with the fittest chromosomes being more likely to be selected. Half of the population (N/2) is selected, where N is the population size. This is done by using the roulette wheel selection method, which assigns areas proportional to the fitness level of each chromosome. The corresponding chromosome is chosen for the crossing step [164].

- **Crossover:** The fourth step is to crossover the best solutions to create new solutions, the crossover operator joins the chosen chromosomes. Uniform crossover is a method of crossover where each bit of the offspring is chosen from either parent with equal probability [165]. Figure 6.6 shows how uniform crossover can be used to generate four offspring from two parents.

- **Mutation:** The fifth step is to mutate the new solutions. To obtain a diverse population, the mutation operator randomly changes the value of some bits in the offspring. To decide if a bit will be mutated, we generate a random number between 0 and 1. If the number is lower than the mutation rate, the bit is inverted. The mutation rate is chosen to be $1/sizeofthechromosome$ [166]. Figure 6.7 shows an example of mutation.

6.3 Experimental results

The section provides a brief overview of the various relevant datasets available and diferent evaluation methods for video summaries. Some of the most commonly used video summaries

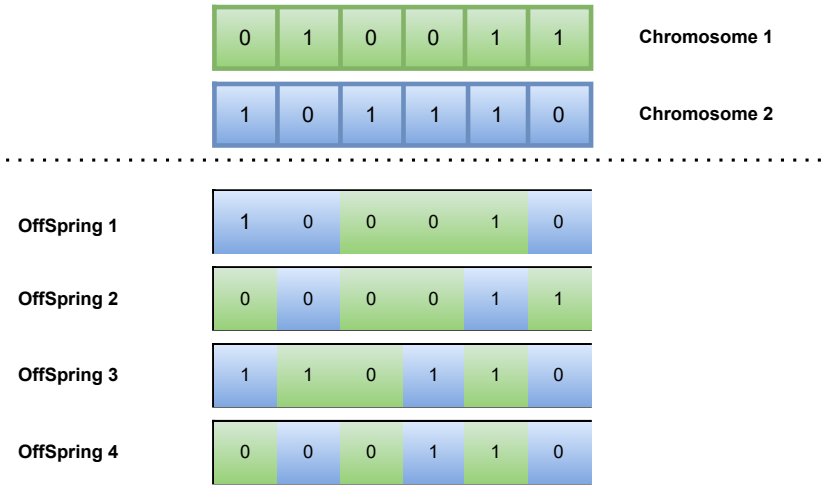


Figure 6.6: Example of uniform crossing of two chromosomes

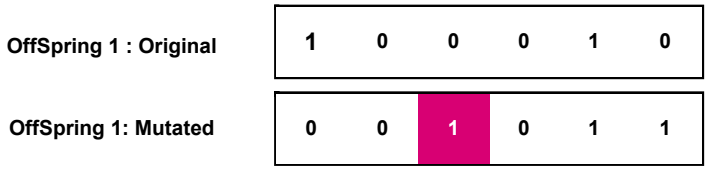


Figure 6.7: Mutation Example

datasets are Open Video Project (OVP) and YouTube dataset (YT).

6.3.1 Datasets used in video summaries

This work presents and evaluates the results of GA-DBSCAN on two datasets: The Open Video Project (OVP) [167] and YouTube dataset (YT). Both datasets are available at the VSUMM database [168], which was created by the authors of the approach described in [169]. Open video project (OVP) contains 50 videos distributed among several genre (Educational, transitory, scientifc, comics, media, events, advertising, TV shows, and home movies.) annotated with five diferent user keyframe sets meaning that each video has 5 summaries. So, the total number of video summaries created by the users is 250 summaries and each user may create different summary. The total duration of all the video sequences is approximately 75 minutes. Each video lasts between 1 and 4 minutes, and the frame rate is 30 fps. This generates 150,000 frames, each of which is 352 $\times$ 240 pixels in size. YouTube dataset (YT) contains 50 video from websites like YouTube. These videos are distributed among several genres (cartoons, news, sports, commercials, tv-shows and home videos) and their duration varies from 1 to 10 minutes annotated with five different user key-frame sets meaning that each video has 5 summaries. So, the total number of video summaries created by the users is 250 summaries and each user may create different summary Table 6.1gives the short description of the datasets used.

Table 6.1: Short description of the datasets OVP and YT

Dataset	Videos	Video format	Duration of each video (approx)	Total duration (approx)
OVP	50	MPEG-1	1-4 minutes	75 minutes
YT	50	AVI	1-10 minutes	2 hours 30 minutes

6.3.2 Performance evaluation

In this study, a modified version of the metric of Comparison of User Summaries (CUS) described in [169] was used to evaluate the quality of video summaries. The CUS metric compares automatic summaries of videos to manual summaries produced by five different users (i.e. ground truth). If two frames (one from an automatic summary and the other from a user summary) are considered to be similar, they are eliminated from the following CUS iteration [169].

Another metric was used to evaluate the automatic video summary is a F1-score. The F1-score consolidates both Precision and Recall values into one value using the harmonic mean. It is varies in the range [0, 1] where a score of 1 indicates the best efficacy of a system [170], and their are defined as:

$$Precision = \frac{n_{MF}}{n_{AS}} \tag{6.6}$$

$$Recall = \frac{n_{MF}}{n_{US}} \tag{6.7}$$

where : n_{MF} is a number of similar key-frames, n_{AS} is a number of key-frames in the automatic summaries, n_{US} is a number of key-frames in the user summaries.

$$F1\_score = \frac{2 \times recall \times precision}{recall + precision} \tag{6.8}$$

Table 6.2 shows the average performance of the proposed system on the OVP and YT datasets, measured in terms of CUS(A), CUS(E), precision, recall, and F-score. Figure 6.8 and 6.9 show the visual results of the key-frames extracted by the proposed system respectively.

Table 6.2: Proposed system results for OVP and YT databases

Data set	CUS(A)	CUS (E)	Precision (P)	Recall (R)	F1_score
OVP	0.768	0.208	0.792	0.769	0.772
YT	0.690	0.301	0.716	0.696	0.695

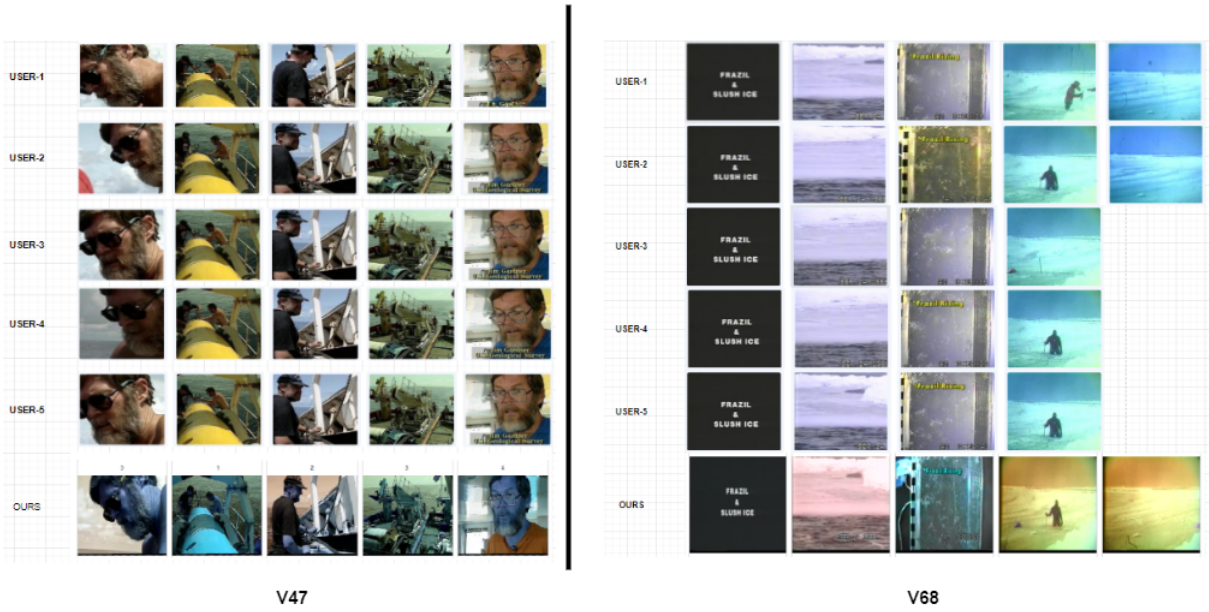


Figure 6.8: Outputs of automated summary using proposed GA-DBSCAN and user summaries of video V47 and V68 from OVP dataset

6.3.3 Experiments on OVP database

We tested our approach for static video summarization on the OVP dataset. Tables 6.3 and 6.4 display the results that were achieved. Table 6.4 shows the accuracy and error rates for each strategy. Our results indicate that we were successful in achieving high accuracy because our mistake rate was minimal compared to other methods.

The performance of both supervised and unsupervised approaches is shown in detail in Figures 6.10 and 6.11 in terms of Precision, Recall, F1-score, and CUS(A) and CUS(E).

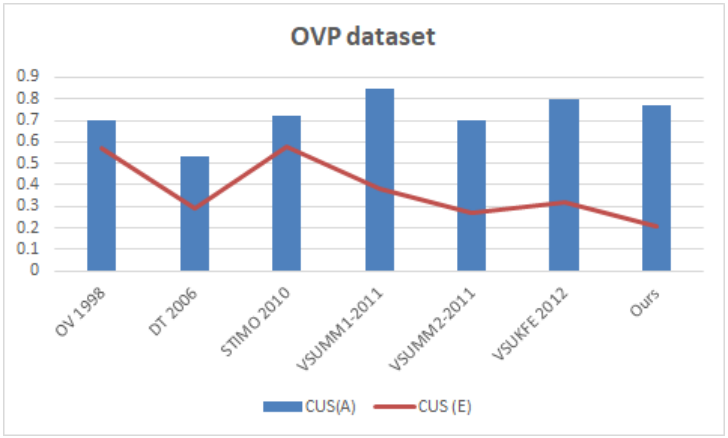


Figure 6.11: Displays comparisons between the proposed method and other systems assessed using the OVP dataset

The results indicate that the suggested strategy performs better than alternative unsupervised methods for the OVP dataset. Supervised approaches are typically more effective than unsupervised ones. The suggested method produces results that are comparable to those of

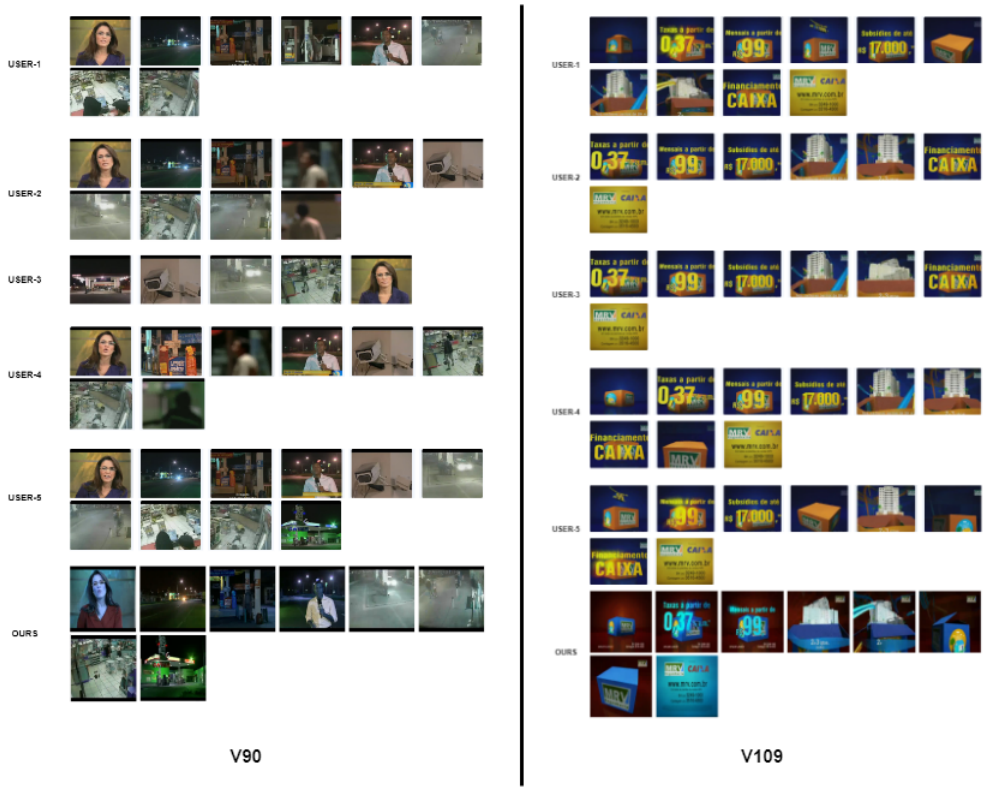


Figure 6.9: Outputs of automated summary using proposed GA-DBSCAN and user summaries of video V90 and V109 from YT dataset

Table 6.3: Displays the average precision, recall, and F1_score of the summaries generated by each approach on the OVP dataset.

Data set	Methods	Unsupervised ?	P	R	F1_score
OVP	TCRa2017[135]		44.94	56.44	48.28
	DT2006[171]	×	54.7	43.3	48.3
	RKSDS2019[136]		43.14	69.93	51.18
	STIMO2010[172]	×	51.9	62.1	56.5
	OV1998 [173]	×	58.4	65.7	61.8
	VISON2012[174]	×	59.5	67.5	63.2
	VSQUAL2014[175]		55.7	74.3	63.6
	VRCVS2017[134]	×	68	63	65.4
	VSCF2018[137]				67
	VUSMM2011[169]		72.1	64.1	67.9
	OPF2016[133]	×			68.2
	VSCAN2013[176]		62.5	**83.1**	71.3
	VISCOM2018[138]		64.9	81.1	72.1
	OPFS2016[133]				72.8
	SMGdpp2017[132]	×			72.8
	TSMVS2023[131]	×			73.14
	SumTr2016[139]				76.5
	Ours	×	**79.2**	76.9	**77.2**

supervised approaches. Figures 6.12 and 6.13 visualize the output of key-frame extraction for videos V23 and V34 from the OVP dataset using various approaches.

Table 6.4: Displays the average CUS(A) and CUS(E) of the summaries generated by each approach on the OVP dataset.

Data set	Methodes	CUS(A)	CUS (E)
OVP	OV1998[173]	0.7	0.57
	DT2006[171]	0.53	0.29
	STIMO2010[172]	0.72	0.58
	VSUMM12011[169]	**0.85**	0.38
	VSUMM22011[169]	0.7	0.27
	VSUKFE2012[177]	0.8	0.32
	Ours	0.768	**0.208**

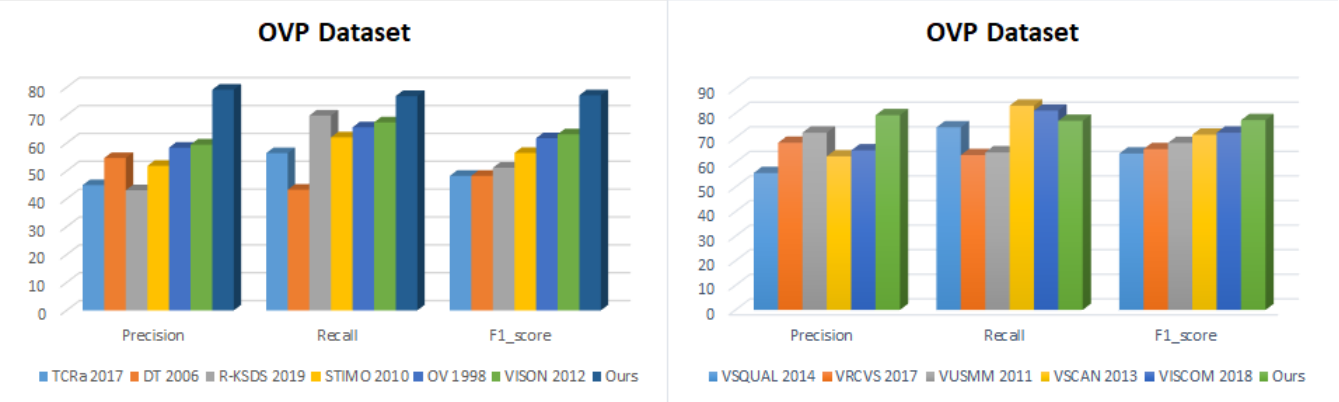


Figure 6.10: Displays comparisons between the proposed method and other systems assessed using the OVP dataset

(a) OV. $CUS_A = 0.78$, $CUS_E = 0.58$.

(b) DT. $CUS_A = 0.57$, $CUS_E = 0.30$.

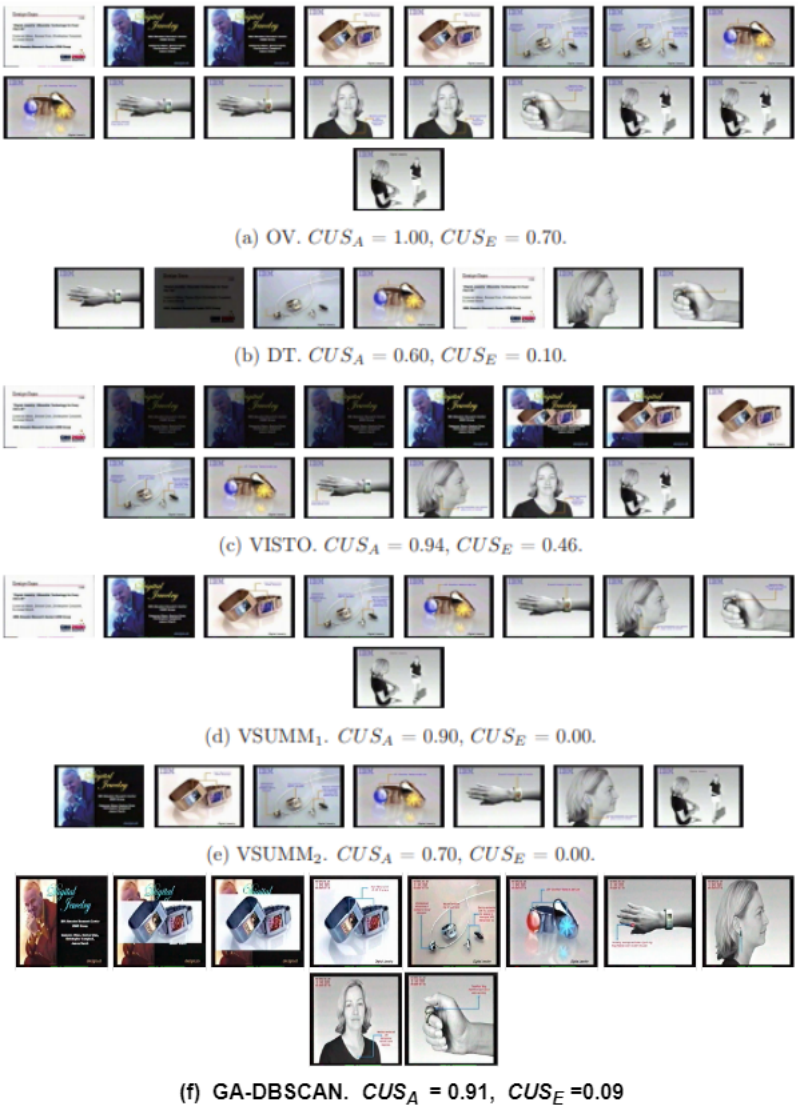
(c) VISTO. $CUS_A = 0.52$, $CUS_E = 0.35$.

(d) $VSUMM_1$. $CUS_A = 0.98$, $CUS_E = 0.88$.

(e) $VSUMM_2$. $CUS_A = 0.93$, $CUS_E = 0.68$.

(f) GA-DBSCAN. $CUS_A = 0.92$, $CUS_E = 0.25$

Figure 6.12: Results of key-frame extraction for V23 from the OVP dataset using several techniques and the suggested method (GA-DBSCAN)



(a) OV. $CUS_A = 1.00$, $CUS_E = 0.70$.

(b) DT. $CUS_A = 0.60$, $CUS_E = 0.10$.

(c) VISTO. $CUS_A = 0.94$, $CUS_E = 0.46$.

(d) VSUMM$_1$. $CUS_A = 0.90$, $CUS_E = 0.00$.

(e) VSUMM$_2$. $CUS_A = 0.70$, $CUS_E = 0.00$.

(f) GA-DBSCAN. $CUS_A = 0.91$, $CUS_E = 0.09$

Figure 6.13: Results of key-frame extraction for V34 from the OVP dataset using several techniques and the suggested method (GA-DBSCAN)

It's worth noting that while the suggested method is entirely unsupervised, OPF [133] and VISON [174] strategies require user intervention. The OV [173] approach fails to identify some critical frames. The DT [171] approach produces a limited number of redundant frames and substantially smaller summaries than those created by users, resulting in reduced accuracy. VSUMM2 [169] also produces smaller summaries and frequently omits some critical frames. The suggested SBD method for creating a static video summary selects a variety of frames.

The proposed approach is more effective than SMGdpp [132], an unsupervised approach that uses deep features. We believe that SMGdpp's use of features from the previous layer is the primary reason for this. We reduce the features from the last layer using the PCA approach. Based on our observations, we can conclude that the suggested method produces concise summaries that accurately reflect events in the video.

6.3.4 Experiments on YT database

The GA-DBSCAN proposed method achieved an average precision, recall, and F-score of 0.716, 0.696, and 0.695, respectively, on the YT dataset as shown in Table 6.5. This is due to the optimization of DBSCAN clustering parameters, which resulted in the generation of more accurate keyframes with less repetition. As a result, the GA-DBSCAN method outperforms current techniques in generating keyframe. A thorough analysis of the F-scores video in Table 6.5 and the bar chart diagrams in Figures 6.14 demonstrate the effectiveness of each strategy on the YT dataset.

Table 6.5: Comparison of the proposed system with state of the art on the entire YT dataset

Data set	Methods	Unsupervised?	CUS(A)	CUS(E)	P	R	F1-score
YT	OPF2016[133]	×					41.9
	OPFS2016[133]						45.1
	VSUMM12011[169]	×	0.84	0.4			58.7
	VUSMM22011[169]	×	0.81	0.31			59.9
	SMGdpp2017[132]	×					60.1
	TSMVS2023[131]	×					41.9
	seqDPP2014[178]						45.1
	NPDPP2016[139]						58.7
	SumTr2016[139]						59.9
	VSDL2018[179]						63.2
	PCDLs2019[180]						63.4
	MAVS2019[181]						66.2
	VRCVS2017[134]	×			77	61	68
	ptrNet2019[182]						69.7
	GAN2019[182]						41.9
	Ours	×	0.69	0.30	71.6	69.6	69.5

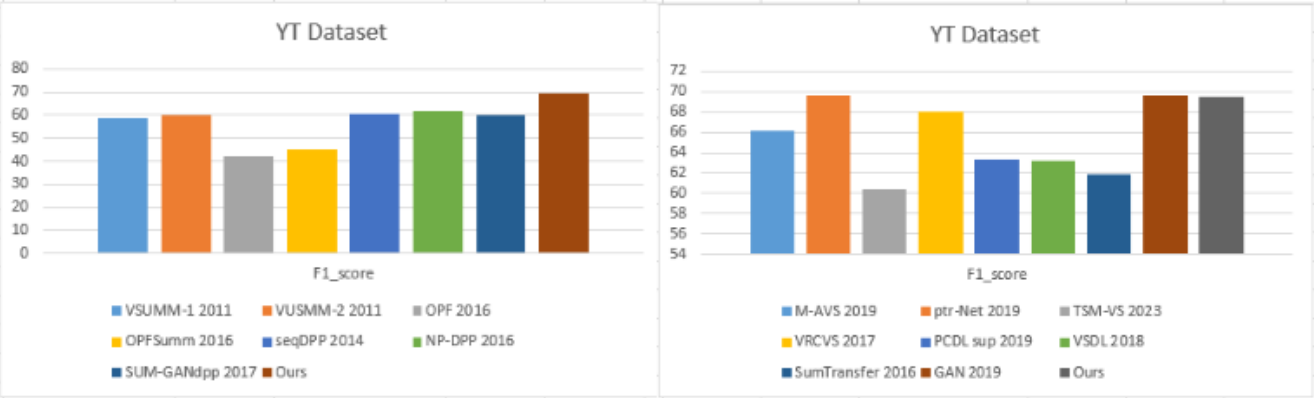


Figure 6.14: Displays comparisons between the proposed method and other systems assessed using the YT dataset

Figure 6.15 shows an example of summaries generated by different methods. The ground truth summary of V91 is shown on the left side. The first two lines on the right side show

summaries generated by VSUMM, and the third and fourth lines show summaries generated by our GA-DBSCAN method. The comparison shows that our method can capture more representative frames with higher accuracy and with less error.

Figure 6.15: key-frame extraction results of V91 from YT dataset by user summaries and VSUMM approach and the proposed method (GA-DBSCAN)

The proposed system was compared to some state-of-the-art SBD techniques to determine its effectiveness. As shown in Tables 6.3 and 6.5, the suggested method performed better than the other strategies, with a much higher F-score on the OVP dataset than the YT dataset. This is so because there is a stronger correlation between video category associations in the OVP dataset than there is in the YT dataset. As a result, the OVP dataset is more suited for the attention mechanism, which aids in focusing attention on the common essential components of a movie. As a result, the performance improves, proving the superiority of the attention mechanism to alternative strategies.

Experimental results show that the suggested method outperforms and is more effective than a number of innovative clustering-based video summarizing techniques as well as some traditional clustering algorithms.

6.4 Conclusion

In this chapter, we proposed a static video summarization approach that uses the unsupervised learning technique DBSCAN and the genetic algorithm for model optimization. We first

extract shots using our SBD method and then extract features from candidate frames using InceptionV3. We then apply PCA for dimension reduction. In the second part, we use the genetic algorithm to find the best DBSCAN hyperparameters to extract key-frames that represent the video summary.

We validated our system on OVP and YT datasets and achieved good performance compared to other methods. However, some techniques that mainly use supervised learning methods are more effective in terms of f1-score.

General conclusion

In this thesis, we focused on video summarization and discovered that there are two types of summaries: static and dynamic. After reviewing the literature on various classifications and methodologies, we began by examining static techniques, which are further divided into two categories: (1) Video digest using local approaches, often referred to as shot boundary detection. (2) Global approaches, or abstracts without revealing the necessary transformations. The shot boundary detection region was the next step that caught our attention. This prompted us to develop a new method for static video summarization as well as suggest a shot detection technique. After this conclusion, we provided an overview of the problems dealt with in this thesis and suggested some points of view.

In the first part of our thesis, we provided an overview of the domain. We talked about big data analytics and machine learning in the first and second chapters and noted a symbiotic relationship between big data and machine learning. Machine learning algorithms are trained on large data sets to make more accurate predictions. However, big data can provide the large training data needed for a machine learning algorithm. Furthermore, in chapters 3 and 4 we took an in-depth research on recent methods used in static video summarization. Through this research, we gained an idea about the field of video summarization, the problems related to it, and the suggested methods for video summarization to identify the advantages and disadvantages of each technique. Furthermore, we looked at recent improvements proposed to overcome problems with shot bounds detection techniques.

Contributions

The second part of the thesis contains our contributions. The first main contribution of this work was the proposal of a novel shot boundary detection (SBD) method that focuses on cut transition detection and leverages the success of deep learning techniques in image similarity comparison tasks. The performance of the VSBD approach is validated on several databases and compared to other SBD methods. The second contribution of this work was the proposal of a

static video summarization system by extracting representative key-frames based on SBD. This system relies on feature extraction with a pre-trained model (Inception-V3), then applies PCA for dimension reduction and the DBSCAN clustering algorithm to identify the best key-frames. To improve the efficiency of this system, we present a genetic algorithm that optimizes the DBSCAN hyper-parameters to extract optimal key-frames that represent the video summary. Different experiments have shown the good performance of our system

Perspectives

As prospects for future research, they can be divided into several parts: We will expand the technique for detecting gradual transitions in videos. This will help us identify the most important parts of the video and generate video skims (dynamic key frames, e.g. movie trailers) from the extracted key frames. We plan to combine several sources of video summary to improve its performance, focusing on text and audio, which are more feasible in the video summary context. This is made possible by multi-modal artificial intelligence, a new paradigm in artificial intelligence that combines various types of data (image, text, audio).

Bibliography

[1] Yihong Gong and Xin Liu. Video summarization using singular value decomposition. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No.PR00662)*. IEEE Comput. Soc.

[2] Shayok Chakraborty, Omesh Tickoo, and Ravi Iyer. Adaptive keyframe selection for video summarization. In *2015 IEEE Winter Conference on Applications of Computer Vision*. IEEE, jan 2015.

[3] Youssef Bendraou. *Video shot boundary detection and key-frame extraction using mathematical models*. Theses, Université du Littoral Côte d'Opale, November 2017.

[4] Youssef Bendraou, Fedwa Essannouni, and Ahmed Salam. From local to global key-frame extraction based on important scenes using SVD of centrist features. *Multimedia Tools and Applications*, 78(2):1441–1456, jun 2018.

[5] Steve Mills, Steve Lucas, Leo Irakliotis, Michael Rappa, Teresa Carlson, and Bill Perlowitz. Demystifying big data: a practical guide to transforming the business of government. *TechAmerica Foundation, Washington*, 2012.

[6] Wo L Chang, Nancy Grady, and N. NIST. Nist big data interoperability framework: volume 1, big data definitions. 2015.

[7] Jonathan Stuart Ward and Adam Barker. Undefined by data: a survey of big data definitions. *arXiv preprint arXiv:1309.5821*, 2013.

[8] Gang-Hoon Kim, Silvana Trimi, and Ji-Hyong Chung. Big-data applications in the government sector. *Communications of the ACM*, 57(3):78–85, mar 2014.

[9] Min Chen, Shiwen Mao, and Yunhao Liu. Big data: A survey. *Mobile Networks and Applications*, 19(2):171–209, jan 2014.

[10] slideteam.net. Big data sources technologies ppt powerpoint presentation clipart. `https://www.slideteam.net/big-data-sources-technologies-ppt-powerpoint-presentation-clipart.html`, 2016. Online; accessed 29 November 2022.

[11] Fernando Iafrate. *Du Big Data au Smart Data: Au service d'un monde connecté*, volume 1. ISTE Group, 2015.

[12] Judith Hurwitz, Alan Nugent, Fern Halper, Marcia Kaufman, et al. *Big data for dummies*, volume 336. John Wiley & Sons Hoboken, NJ, 2013.

[13] K.V. Kanimozhi and Dr.M. Venkatesan. Unstructured data analysis-a survey. *IJARCCE*, pages 223–225, mar 2015.

[14] Paul Zikopoulos, Chris Eaton, Dirk DeRoos, Tom Deutsch, and George Lapis. *Understanding big data: Analytics for enterprise class hadoop and streaming data*. McGraw-Hill Osborne Media, 2011.

[15] Beki Grinter. A big data confession. *Interactions*, 20(4):10–11, jul 2013.

[16] Kiranjit Pattnaik and Bhabani Shankar Prasad Mishra. Introduction to big data analysis. In *Techniques and Environments for Big Data Analysis*, pages 1–20. Springer, 2016.

[17] Daniel J Power. Using 'big data'for analytics and decision support. *Journal of Decision Systems*, 23(2):222–228, 2014.

[18] Tyler Akidau, Robert Bradshaw, Craig Chambers, Slava Chernyak, Rafael J. Fernández-Moctezuma, Reuven Lax, Sam McVeety, Daniel Mills, Frances Perry, Eric Schmidt, and Sam Whittle. The dataflow model: A practical approach to balancing correctness, latency, and cost in massive-scale, unbounded, out-of-order data processing. *Proceedings of the VLDB Endowment*, 8:1792–1803, 2015.

[19] Ticiana L Coelho da Silva, Regis Pires Magalhães, Igo Ramalho Brilhante, José AF de Macêdo, David Araújo, Paulo AL Rego, and Aloisio Vieira Lira Neto. Big data analytics technologies and platforms: A brief review. *LADaS@ VLDB*, pages 25–32, 2018.

[20] Nawsher Khan, Ibrar Yaqoob, Ibrahim Abaker Targio Hashem, Zakira Inayat, Waleed Kamaleldin Mahmoud Ali, Muhammad Alam, Muhammad Shiraz, and Abdullah Gani. Big data: Survey, technologies, opportunities, and challenges. *The Scientific World Journal*, 2014:1–18, 2014.

[21] M. Janaki Meena and S. P. Syed Ibrahim. Statistical and evolutionary feature selection techniques parallelized using MapReduce programming model. In *Studies in Big Data*, pages 159–180. Springer International Publishing, 2016.

[22] Harshali Patel. What are some good graphics of Apache Hadoop architecture. `https://www.quora.com/What-are-some-good-graphics-of-Apache-Hadoop-architecture`, 2019. Online; accessed 29 November 2022.

[23] Shan Suthaharan. Distributed file system. In *Machine Learning Models and Algorithms for Big Data Classification*, pages 79–97. Springer US, 2016.

[24] Dhruba Borthakur. The hadoop distributed file system: Architecture and design. *Hadoop Project Website*, 11(2007):21, 2007.

[25] Nada Elgendy and Ahmed Elragal. Big data analytics: A literature review paper. In *Advances in Data Mining. Applications and Theoretical Aspects*, pages 214–227. Springer International Publishing, 2014.

[26] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

[27] Aisling O'Driscoll, Jurate Daugelaite, and Roy D Sleator. 'big data', hadoop and cloud computing in genomics. *Journal of biomedical informatics*, 46(5):774–781, 2013.

[28] Matei Zaharia, Mosharaf Chowdhury, Michael J Franklin, Scott Shenker, and Ion Stoica. Spark: Cluster computing with working sets. In *2nd USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 10)*, 2010.

[29] Holden Karau, Andy Konwinski, Patrick Wendell, and Matei Zaharia. *Learning spark: lightning-fast big data analysis*. " O'Reilly Media, Inc.", 2015.

[30] por Diego Calvo. Arquitectura Spark. `https://www.diegocalvo.es/arquitectura-spark`, 2018. Online; accessed 29 November 2022.

[31] Rabhi Loubna, Falih Noureddine, Afraites Lekbir, and Bouikhalene Belaid. Big data analytics, reshaping the new trends of healthcare: Literature review. In Mohamed Elhoseny, Xiaohui Yuan, and Salah-ddine Krit, editors, *Distributed Sensing and Intelligent Systems*, pages 737–746, Cham, 2022. Springer International Publishing.

[32] Zhihan Lv, Houbing Song, Pablo Basanta-Val, Anthony Steed, and Minho Jo. Next-generation big data analytics: State of the art, challenges, and future research topics. *IEEE Transactions on Industrial Informatics*, 13(4):1891–1899, aug 2017.

[33] Loubna Rabhi, Noureddine Falih, Abdlekbir Afraites, and Belaid Bouikhalene. Big data approach and its applications in various fields: Review. *Procedia Computer Science*, 155:599–605, 2019.

[34] Jacob A Benfield and William J Szlemko. Internet-based data collection: Promises and realities. *Journal of Research Practice*, 2(2):D1–D1, 2006.

[35] Darshan Barapatre and A Vijayalakshmi. Data preparation on large datasets for data science. *Asian Journal of Pharmaceutical and Clinical Research*, 10(13), 2017.

[36] Francesco Guerra, Giovanni Simonini, and Maurizio Vincini. Supporting image search with tag clouds: a preliminary approach. *Advances in Multimedia*, 2015, 2015.

[37] Ekaterina Olshannikova, Aleksandr Ometov, Yevgeni Koucheryavy, and Thomas Olsson. Visualizing big data with augmented and virtual reality: challenges and research agenda. *Journal of Big Data*, 2(1):1–27, 2015.

[38] Reza Zafarani, Mohammad Ali Abbasi, and Huan Liu. *Social media mining: an introduction*. Cambridge University Press, 2014.

[39] Samira Pouyanfar, Yimin Yang, Shu-Ching Chen, Mei-Ling Shyu, and S. S. Iyengar. Multimedia big data analytics. *ACM Computing Surveys*, 51(1):1–34, jan 2019.

[40] P Ushapreethi and GG Lakshmipriya. Survey on video big data: Analysis methods and applications. *International Journal of Applied Engineering Research*, 12(10):2221–2231, 2017.

[41] Weiming Hu, Nianhua Xie, Li Li, Xianglin Zeng, and S. Maybank. A survey on visual content-based video indexing and retrieval. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 41(6):797–819, nov 2011.

[42] Yalin Başţanlar and Mustafa Özuysal. Introduction to machine learning. *miRNomics: MicroRNA biology and computational analysis*, pages 105–128, 2014.

[43] Ethem Alpaydin. *Introduction to machine learning*. MIT press, 2020.

[44] Kajaree Das and Rabi Narayan Behera. A survey on machine learning: concept, algorithms and applications. *International Journal of Innovative Research in Computer and Communication Engineering*, 5(2):1301–1309, 2017.

[45] Himanshu Singh and Yunis Ahmad Lone. Introduction to machine learning. In *Deep Neuro-Fuzzy Systems with Python*, pages 129–156. Apress, dec 2019.

[46] T Mitchell. Machine learning. 1997. isbn: 0070428077, mcgraw-hill.

[47] Sonoo Jaiswal. Difference between Machine Learning and Deep Learning. `https://www.javatpoint.com/machine-learning-vs-deep-learning`, 2021. Online; accessed 04 December 2022.

[48] Shaomin Wu. A review on coarse warranty data and analysis. *Reliability Engineering &amp System Safety*, 114:1–11, jun 2013.

[49] Tlamelo Emmanuel, Thabiso Maupong, Dimane Mpoeleng, Thabo Semong, Banyatsang Mphago, and Oteng Tabona. A survey on missing data in machine learning. *Journal of Big Data*, 8(1), oct 2021.

[50] Peshawa Jamal Muhammad Ali, Rezhna Hassan Faraj, Erbil Koya, Peshawa J Muhammad Ali, and Rezhna H Faraj. Data normalization and standardization: a technical report. *Mach Learn Tech Rep*, 1(1):1–6, 2014.

[51] Harsurinder Kaur, Husanbir Singh Pannu, and Avleen Kaur Malhi. A systematic review on imbalanced data challenges in machine learning. *ACM Computing Surveys*, 52(4):1–36, aug 2019.

[52] RAFAEL ALENCAR. Resampling strategies for imbalanced datasets. `https://www.kaggle.com/code/rafjaa/resampling-strategies-for-imbalanced-datasets/notebook`, 2017. Online; accessed 04 December 2022.

[53] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, jun 2002.

[54] Suhang Wang, Jiliang Tang, and Huan Liu. *Feature Selection*, pages 503–511. Springer US, Boston, MA, 2017.

[55] David Selasi Koblah, Rabin Yu Acharya, Daniel Capecci, Olivia P. Dizon-Paradis, Shahin Tajik, Fatemeh Ganji, Damon L. Woodard, and Domenic Forte. A survey and perspective on artificial intelligence for security-aware electronic design automation. *ACM Transactions on Design Automation of Electronic Systems*, sep 2022.

[56] David Koblah, Rabin Acharya, Daniel Capecci, Olivia Dizon-Paradis, Shahin Tajik, Fatemeh Ganji, Damon Woodard, and Domenic Forte. A survey and perspective on artificial intelligence for security-aware electronic design automation. *ACM Transactions on Design Automation of Electronic Systems*, 28(2):1–57, 2023.

[57] Alvira Swalin. Choosing the Right Metric for Evaluating Machine Learning Models Part 2. `https://www.kdnuggets.com/2018/06/right-metric-evaluating-machine-learning-models-2.html`, 2018. Online; accessed 05 December 2022.

[58] Sonoo Jaiswal. Machine Learning Tutorial. `https://www.javatpoint.com/machine-learning`, 2021. Online; accessed 05 December 2022.

[59] A.K. Jain, P.W. Duin, and Jianchang Mao. Statistical pattern recognition: a review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):4–37, 2000.

[60] Horace B Barlow. Unsupervised learning. *Neural computation*, 1(3):295–311, 1989.

[61] Khadija El Bouchefry and Rafael S. de Souza. Learning in big data: Introduction to machine learning. In *Knowledge Discovery in Big Data from Astronomy and Earth Observation*, pages 225–249. Elsevier, 2020.

[62] TechVidvan. Reinforcement Learning Algorithms and Applications. `https://techvidvan.com/tutorials/reinforcement-learning/`, 2021. Online; accessed 06 December 2022.

[63] Qingchen Zhang, Laurence T. Yang, Zhikui Chen, and Peng Li. A survey on deep learning for big data. *Information Fusion*, 42:146–157, jul 2018.

[64] Xue-Wen Chen and Xiaotong Lin. Big data deep learning: Challenges and perspectives. *IEEE Access*, 2:514–525, 2014.

[65] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, jan 2015.

[66] Roguia SIOUDA. *Pattern recognition using collaborative neural networks*. PhD thesis, 2022.

[67] Y. Bengio. Learning deep architectures for AI. *Foundations and Trends® in Machine Learning*, 2(1):1–127, 2009.

[68] Niall O'Mahony, Sean Campbell, Anderson Carvalho, Suman Harapanahalli, Gustavo Velasco Hernandez, Lenka Krpalkova, Daniel Riordan, and Joseph Walsh. Deep learning vs. traditional computer vision. In *Advances in Intelligent Systems and Computing*, pages 128–144. Springer International Publishing, apr 2019.

[69] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen A.W.M. van der Laak, Bram van Ginneken, and Clara I. Sánchez. A survey on deep learning in medical image analysis. *Medical Image Analysis*, 42:60–88, dec 2017.

[70] Oswald Campesato. *Artificial intelligence, machine learning, and deep learning*. Mercury Learning and Information, 2020.

[71] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2014.

[72] Phani Ratan. What is the Convolutional Neural Network Architecture. `https://www.analyticsvidhya.com/blog/2020/10/what-is-the-convolutional-neural-network-architecture/`, 2020. Online; accessed 08 December 2022.

[73] AISmartz. CNN Architectures Over a Timeline (1998-2019). `https://www.aismartz.com/blog/cnn-architectures/`, 2019. Online; accessed 10 December 2022.

[74] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, nov 1997.

[75] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling, 2014.

[76] X. Chen, X. Liu, Y. Qian, M. J. F. Gales, and P. C. Woodland. CUED-RNNLM — an open-source toolkit for efficient training and evaluation of recurrent neural network language models. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, mar 2016.

[77] Jen-Tzung Chien and Yuan-Chu Ku. Bayesian recurrent neural network for language modeling. *IEEE Transactions on Neural Networks and Learning Systems*, 27(2):361–374, feb 2016.

[78] Ons Aouedi, Kandaraj Piamrat, and Dhruvjyoti Bagadthey. A semi-supervised stacked autoencoder approach for network traffic classification. In *2020 IEEE 28th International Conference on Network Protocols (ICNP)*. IEEE, oct 2020.

[79] Yanming Guo, Yu Liu, Ard Oerlemans, Songyang Lao, Song Wu, and Michael S. Lew. Deep learning for visual understanding: A review. *Neurocomputing*, 187:27–48, apr 2016.

[80] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, jul 2006.

[81] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

[82] Lisa Torrey and Jude Shavlik. Transfer learning. In *Handbook of Research on Machine Learning Applications and Trends*, pages 242–264. IGI Global, 2010.

[83] Pedro Marcelino. Transfer learning from pre-trained models. `https://towardsdatascience.com/transfer-learning-from-pre-trained-models-f2393f124751`, 2018. Online; accessed 12 December 2022.

[84] Deepika Bajaj and Shanu Sharma. Video depiction of key frames- a review. In *Proceedings of the Sixth International Conference on Computer and Communication Technology 2015*, ICCCT '15, pages 183–187, New York, NY, USA, 2015. ACM.

[85] Tiecheng Liu and John R. Kender. Computational approaches to temporal sampling of video sequences. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 3(2):7–es, may 2007.

[86] Jaydeb Mondal, Malay Kumar Kundu, Sudeb Das, and Manish Chowdhury. Video shot boundary detection using multiscale geometric analysis of nsct and least squares support vector machine. *Multimedia Tools and Applications*, 77(7):8139–8161, apr 2017.

[87] Gautam Pal, Dwijen Rudrapaul, Suvojit Acharjee, Ruben Ray, Sayan Chakraborty, and Nilanjan Dey. Video shot boundary detection: A review. In *Advances in Intelligent Systems and Computing*, pages 119–127. Springer International Publishing, 2015.

[88] A. Hanjalic. Shot-boundary detection: unraveled and resolved? *IEEE Transactions on Circuits and Systems for Video Technology*, 12(2):90–105, 2002.

[89] Hrishikesh Bhaumik, Siddhartha Bhattacharyya, and Susanta Chakraborty. Content coverage and redundancy removal in video summarization. In *Intelligent Analysis of Multimedia Information*, pages 352–374. IGI Global.

[90] Benoughidene Abdel Halim and Titouna Faiza. Shot boundary detection: Fundamental concepts and survey. In Hamid Seridi, Muhammet Kurulay, Zineddine Kouahla, Brahim Farou, Mohamed Nadjib Kouahla, Mohamed Amine Ferrag, and Mohamed Amine Boudia,

editors, *The 1st International Conference on Innovative Trends in Computer Science, CITSC 2019, Guelma, Algeria, November 20-21, 2019*, volume 2589 of *CEUR Workshop Proceedings*, pages 34–40. CEUR-WS.org, 2019.

[91] Guangyu Gao and Huadong Ma. To accelerate shot boundary detection by reducing detection region and scope. *Multimedia Tools and Applications*, 71(3):1749–1770, Springer Science and Business Media, dec 2012.

[92] Zhonglan Wu and Pin Xu. Shot boundary detection in video retrieval. In *2013 IEEE 4th International Conference on Electronics Information and Emergency Communication*. IEEE, nov 2013.

[93] Heba Ahmed Elnemr, Nourhan Mohamed Zayed, and Mahmoud Abdelmoneim Fakhreldein. Feature extraction techniques. In *Handbook of Research on Emerging Perspectives in Intelligent Pattern Recognition, Analysis, and Image Processing*, pages 264–294. IGI Global, 2016.

[94] Amr Ahmed. Video representation and processing for multimedia data mining. In *Semantic Mining Technologies for Multimedia Databases*. IGI Global, 2009.

[95] S Kikukawa, T; Kawafuchi. Development of an automatic summary editing system for the audio-visual resources. *Trans. IEICE*, J75-A(2):204–212, 1992.

[96] Deborah Swanberg, Chiao-Fe Shu, and Ramesh C. Jain. Knowledge guided parsing in video databases. In Carlton W. Niblack, editor, *Storage and Retrieval for Image and Video Databases*. SPIE, apr 1993.

[97] Zhe-Ming Lu and Yong Shi. Fast video shot boundary detection based on SVD and pattern matching. *IEEE Transactions on Image Processing*, 22(12):5136–5145, dec 2013.

[98] Bendraou Youssef, Essannouni Fedwa, Aboutajdine Driss, and Salam Ahmed. Shot boundary detection via adaptive low rank and svd-updating. *Computer Vision and Image Understanding*, 161:20–28, aug 2017.

[99] J. Miller R. Zabih and K. Mai. A feature-based algorithm for detecting and classifying scene breaks,. In *In Proceedings of the Third ACM International Conference on Multimedia Multimedia 95*, pages 189–200, San Francisco, CA, USA, 1995. ACM.

[100] Wei Jyh Heng and King N. Ngan. An object-based shot boundary detection using edge tracing and tracking. *Journal of Visual Communication and Image Representation*, 12(3):217–239, sep 2001.

[101] Jie Zheng, Fengmei Zou, and Mandel Shi. An efficient algorithm for video shot boundary detection. In *Proceedings of 2004 International Symposium on Intelligent Multimedia, Video and Speech Processing, 2004.* IEEE.

[102] HongJiang Zhang, Chien Y. Low, Yihong Gong, and Stephen W. Smoliar. Video parsing using compressed data. In Sarah A. Rajala and Robert L. Stevenson, editors, *Image and Video Processing II*. SPIE, mar 1994.

[103] E. Bruno and D. Pellerin. Video shot detection based on linear prediction of motion. In *Proceedings. IEEE International Conference on Multimedia and Expo 2002*, volume 1, pages 289–292. IEEE, 2002.

[104] Behzad Shahraray. Scene change detection and content-based sampling of video sequences. In Arturo A. Rodriguez, Robert J. Safranek, and Edward J. Delp, editors, *Digital Video Compression: Algorithms and Technologies 1995*. SPIE, apr 1995.

[105] Eralda Nishani and Betim Cico. Computer vision approaches based on deep learning and neural networks: Deep neural networks for video analysis of human pose estimation. In *2017 6th Mediterranean Conference on Embedded Computing (MECO)*. IEEE, jun 2017.

[106] Wenjing Tong, Li Song, Xiaokang Yang, Hui Qu, and Rong Xie. Cnn-based shot boundary detection and video annotation. In *2015 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting*. IEEE, jun 2015.

[107] Jingwei Xu, Li Song, and Rong Xie. Shot boundary detection using convolutional neural networks. In *2016 Visual Communications and Image Processing (VCIP)*. IEEE, nov 2016.

[108] Ahmed Hassanien, Mohamed A. Elgharib, Ahmed Selim, Mohamed Hefeeda, and Wojciech Matusik. Large-scale, fast and accurate shot boundary detection through spatio-temporal convolutional neural networks. *CoRR*, abs/1705.03281, 2017.

[109] Michael Gygli. Ridiculously fast shot boundary detection with fully convolutional neural networks. In *2018 International Conference on Content-Based Multimedia Indexing (CBMI)*. IEEE, sep 2018.

[110] Tomás Soucek, Jaroslav Moravec, and Jakub Lokoc. Transnet: A deep network for fast detection of common shot transitions. *CoRR*, abs/1906.03363, 2019.

[111] Khan Muhammad, Tanveer Hussain, and Sung Wook Baik. Efficient CNN based summarization of surveillance videos for resource-constrained devices. *Pattern Recognition Letters*, aug 2018.

[112] Lifang Wu, Shuai Zhang, Meng Jian, Zhe Lu, and Dong Wang. Two stage shot boundary detection via feature fusion and spatial-temporal convolutional neural networks. *IEEE Access*, 7:77268–77276, 2019.

[113] Dayou Jiang and Jongweon Kim. A scene change detection framework based on deep learning and image matching. In *Lecture Notes in Electrical Engineering*, pages 623–629. Springer Singapore, nov 2018.

[114] Rui Liang, Qingxin Zhu, Honglei Wei, and Shujiao Liao. A video shot boundary detection approach based on CNN feature. In *2017 IEEE International Symposium on Multimedia (ISM)*. IEEE, dec 2017.

[115] Lifang Wu, Shuai Zhang, Meng Jian, Zhijia Zhao, and Dong Wang. Shot boundary detection with spatial-temporal convolutional neural networks. In *Pattern Recognition and Computer Vision*, pages 479–491. Springer International Publishing, 2018.

[116] Dalton Meitei Thounaojam, Thongam Khelchandra, Kh. Manglem Singh, and Sudipta Roy. A genetic algorithm and fuzzy logic approach for video shot boundary detection. *Computational Intelligence and Neuroscience*, 2016:1–11, 2016.

[117] Saptarshi Chakraborty and Dalton Meitei Thounaojam. A novel shot boundary detection system using hybrid optimization technique. *Applied Intelligence*, mar 2019.

[118] Jialei Bi, Xianglong Liu, and Bo Lang. A novel shot boundary detection based on information theory using SVM. In *2011 4th International Congress on Image and Signal Processing*. IEEE, oct 2011.

[119] Junaid Baber, Nitin Afzulpurkar, Matthew N. Dailey, and Maheen Bakhtyar. Shot boundary detection from videos using entropy and local descriptor. In *2011 17th International Conference on Digital Signal Processing (DSP)*. IEEE, jul 2011.

[120] Sawitchaya Tippaya, Suchada Sitjongsataporn, Tele Tan, Masood Mehmood Khan, and Kosin Chamnongthai. Multi-modal visual features-based video shot boundary detection. *IEEE Access*, 5:12563–12575, 2017.

[121] T. Kar and P. Kanungo. A texture based method for scene change detection. In *2015 IEEE Power, Communication and Information Technology Conference (PCITC)*. IEEE, oct 2015.

[122] Goran Zajic, Ana Gavrovska, Irini Reljin, and Branimir Reljin. A video hard cut detection using multifractal features. *Multimedia Tools and Applications: Springer Science and Business Media LLC*, 78(5):6233–6252, jul 2018.

[123] Shitao Tang, Litong Feng, Zhanghui Kuang, Yimin Chen, and Wei Zhang. Fast video shot transition localization with deep structured models. In *Computer Vision – ACCV 2018*, pages 577–592, Cham, 2019. Springer International Publishing.

[124] Mickaël Rouvier. *Structuration de contenus audio-visuel pour le résumé automatique*. Theses, Université d'Avignon, December 2011.

[125] Mickael Guironnet. *Méthodes de résumé de vidéo à partir d'informations bas niveau, du mouvement de caméra ou de l'attention visuelle*. PhD thesis, Université Joseph-Fourier-Grenoble I, 2006.

[126] EJY Cayllahua-Cahuina, G Cámara-Chávez, and D Menotti. A static video summarization approach with automatic shot detection using color histograms. In *Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV)*, page 1. The Steering Committee of The World Congress in Computer Science, Computer . . . , 2012.

[127] Karim Mahmoud, Nagia Ghanem, and Mohamed Ismail. Vgraph: an effective approach for generating static video summaries. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 811–818, 2013.

[128] Ebrahim Asadi and Nasrolla Moghadam Charkari. Video summarization using fuzzy c-means clustering. In *20th Iranian Conference on Electrical Engineering (ICEE2012)*, pages 690–694. IEEE, 2012.

[129] Xinding Sun and Mohan S. Kankanhalli. Video summarization using r-sequences. *Real-Time Imaging*, 6(6):449–459, dec 2000.

[130] Fereshteh Falah Chamasemani, Lilly Suriani Affendey, Norwati Mustapha, and Fatimah Khalid. Video abstraction using density-based clustering algorithm. *The Visual Computer*, 34:1299–1314, 2018.

[131] Khushboo Khurana and Umesh Deshpande. Two stream multi-layer convolutional network for keyframe-based video summarization. *Multimedia Tools and Applications*, pages 1–42, 2023.

[132] Behrooz Mahasseni, Michael Lam, and Sinisa Todorovic. Unsupervised video summarization with adversarial lstm networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 202–211, 2017.

[133] Guilherme B Martins, Joao P Papa, and Jurandy Almeida. Temporal-and spatial-driven video summarization using optimum-path forest. In *2016 29th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, pages 335–339. Ieee, 2016.

[134] Jiaxin Wu, Sheng-hua Zhong, Jianmin Jiang, and Yunyun Yang. A novel clustering method for static video summarization. *Multimedia Tools and Applications*, 76:9625–9641, 2017.

[135] Mingyang Ma, Shaohui Met, Junhui Hou, Shuai Wan, and Zhiyong Wang. Video summarization via temporal collaborative representation of adjacent frames. In *2017 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*, pages 164–169. IEEE, 2017.

[136] Mingyang Ma, Shaohui Mei, Shuai Wan, Zhiyong Wang, and Dagan Feng. Video summarization via nonlinear sparse dictionary selection. *IEEE Access*, 7:11763–11774, 2019.

[137] Muhammad Asim, Noor Almaadeed, Somaya Al-Máadeed, Ahmed Bouridane, and Azeddine Beghdadi. A key frame based video summarization using color features. In *2018 Colour and Visual Computing Symposium (CVCS)*, pages 1–6. IEEE, 2018.

[138] Marcos Vinicius Mussel Cirne and Helio Pedrini. Viscom: A robust video summarization approach using color co-occurrence matrices. *Multimedia Tools and Applications*, 77:857–875, 2018.

[139] Ke Zhang, Wei-Lun Chao, Fei Sha, and Kristen Grauman. Summary transfer: Exemplar-based subset selection for video summarization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1059–1067, 2016.

[140] Georgios Evangelopoulos, Athanasia Zlatintsi, Georgios Skoumas, Konstantinos Rapantzikos, Alexandros Potamianos, Petros Maragos, and Yannis Avrithis. Video event detection and summarization using audio, visual and text saliency. In *2009 IEEE international conference on acoustics, speech and signal processing*, pages 3553–3556. IEEE, 2009.

[141] Yingbo Li, Bernard Merialdo, Mickael Rouvier, and Georges Linares. Static and dynamic video summaries. In *Proceedings of the 19th ACM international conference on Multimedia*, pages 1573–1576, 2011.

[142] Pedro Atencio, Sánchez-Torres German, John William Branch, and Claudio Delrieux. Video summarisation by deep visual and categorical diversity. *IET Computer Vision*, 13(6):569–577, 2019.

[143] Cüneyt M Taskiran, Zygmunt Pizlo, Arnon Amir, Dulce Ponceleon, and Edward J Delp. Automated video program summarization using speech transcripts. *IEEE Transactions on Multimedia*, 8(4):775–791, 2006.

[144] Xiatian Zhu, Chen Change Loy, and Shaogang Gong. Learning from multiple sources for video summarisation. *International Journal of Computer Vision*, 117:247–268, 2016.

[145] Kaiyang Zhou, Yu Qiao, and Tao Xiang. Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[146] Mayu Otani, Yuta Nakashima, Tomokazu Sato, and Naokazu Yokoya. Video summarization using textual descriptions for authoring video blogs. *Multimedia Tools and Applications*, 76:12097–12115, 2017.

[147] Shu Zhang, Yingying Zhu, and Amit K Roy-Chowdhury. Context-aware surveillance video summarization. *IEEE Transactions on Image Processing*, 25(11):5469–5478, 2016.

[148] Mostafa Tavassolipour, Mahmood Karimian, and Shohreh Kasaei. Event detection and summarization in soccer videos using bayesian network and copula. *IEEE Transactions on circuits and systems for video technology*, 24(2):291–304, 2013.

[149] Hansa Shingrakhia and Hetal Patel. Emperor penguin optimized event recognition and summarization for cricket highlight generation. *Multimedia Systems*, 26(6):745–759, 2020.

[150] Rabbia Mahum, Aun Irtaza, Marriam Nawaz, Tahira Nazir, Momina Masood, Sarang Shaikh, and Emad Abouel Nasr. A robust framework to generate surveillance video summaries using combination of zernike moments and r-transform and deep neural network. *Multimedia Tools and Applications*, 82(9):13811–13835, 2023.

[151] Chakradhar Guntuboina, Aditya Porwal, Preet Jain, and Hansa Shingrakhia. Deep learning based automated sports video summarization using yolo. *ELCVIA Electronic Letters on Computer Vision and Image Analysis*, 20(1):99–116, 2021.

[152] Abdelhalim Benoughidene and Faiza Titouna. A novel method for video shot boundary detection using cnn-lstm approach. *International Journal of Multimedia Information Retrieval*, 11(4):653–667, 2022.

[153] Liangliang Wang and Deepu Rajan. An image similarity descriptor for classification tasks. *Journal of Visual Communication and Image Representation*, 71:102847, aug 2020.

[154] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, jan 2021.

[155] Sawitchaya Tippaya, Suchada Sitjongsataporn, Tele Tan, Kosin Chamnongthai, and Masood Khan. Video shot boundary detection based on candidate segment selection and transition pattern analysis. In *2015 IEEE International Conference on Digital Signal Processing (DSP)*. IEEE, jul 2015.

[156] Lorenzo Baraldi, Costantino Grana, and Rita Cucchiara. Shot and scene detection via hierarchical clustering for re-using broadcast video. In *Computer Analysis of Images and Patterns*, pages 801–811. Springer International Publishing, 2015.

[157] Alok Singh, Dalton Meitei Thounaojam, and Saptarshi Chakraborty. A novel automatic shot boundary detection algorithm: robust to illumination and motion effect. *Signal, Image and Video Processing*, 14(4):645–653, nov 2019.

[158] T. Kar and P. Kanungo. A motion and illumination resilient framework for automatic shot boundary detection. *Signal, Image and Video Processing*, 11(7):1237–1244, mar 2017.

[159] Lakshmi Priya G. G. and Domnic S. Walsh–hadamard transform kernel-based feature vector for shot boundary detection. *IEEE Transactions on Image Processing*, 23(12):5187–5197, dec 2014.

[160] M Dhanushree, R Priya, P Aruna, and R Bhavani. A keyframe extraction using hdbscan with particle swarm optimization. In *2023 10th International Conference on Signal Processing and Integrated Networks (SPIN)*, pages 445–450. IEEE, 2023.

[161] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.

[162] Cheng-Lung Huang and Chieh-Jen Wang. A ga-based feature selection and parameters optimization for support vector machines. *Expert Systems with applications*, 31(2):231–240, 2006.

[163] L Darrell Whitley et al. The genitor algorithm and selection pressure: why rank-based allocation of reproductive trials is best. In *Icga*, volume 89, pages 116–123. Fairfax, VA, 1989.

[164] Adam Lipowski and Dorota Lipowska. Roulette-wheel selection via stochastic acceptance. *Physica A: Statistical Mechanics and its Applications*, 391(6):2193–2196, 2012.

[165] William M Spears and Kenneth D De Jong. On the virtues of parameterized uniform crossover. Technical report, Naval Research Lab Washington DC, 1995.

[166] Ghodrat Moghadampour. Outperforming mutation operator with random building block operator in genetic algorithms. In *International Conference on Enterprise Information Systems*, pages 178–192. Springer, 2011.

[167] The Open Video Project. `http://www.open-video.org`, 2016.

[168] VSUMM (Video SUMMarization). `https://sites.google.com/site/vsummsite`, 2016.

[169] Sandra Eliza Fontes De Avila, Ana Paula Brandao Lopes, Antonio da Luz Jr, and Arnaldo de Albuquerque Araújo. Vsumm: A mechanism designed to produce static video summaries and a novel evaluation method. *Pattern recognition letters*, 32(1):56–68, 2011.

[170] Henk M Blanken, Arjen P de Vries, Henk Ernst Blok, and Ling Feng. *Multimedia retrieval*. Springer, 2007.

[171] Padmavathi Mundur, Yong Rao, and Yelena Yesha. Keyframe-based video summarization using delaunay clustering. *International Journal on Digital Libraries*, 6:219–232, 2006.

[172] Marco Furini, Filippo Geraci, Manuela Montangero, and Marco Pellegrini. Stimo: Still and moving video storyboard for the web scenario. *Multimedia Tools and Applications*, 46:47–69, 2010.

[173] Daniel DeMenthon, Vikrant Kobla, and David Doermann. Video summarization by curve simplification. In *Proceedings of the sixth ACM international conference on Multimedia*, pages 211–218, 1998.

[174] Jurandy Almeida, Neucimar J Leite, and Ricardo da S Torres. Vison: Video summarization for online applications. *Pattern Recognition Letters*, 33(4):397–409, 2012.

[175] Marcos Vinicius Mussel Cirne and Helio Pedrini. Summarization of videos by image quality assessment. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications: 19th Iberoamerican Congress, CIARP 2014, Puerto Vallarta, Mexico, November 2-5, 2014. Proceedings 19*, pages 901–908. Springer, 2014.

[176] Karim M Mahmoud, Mohamed A Ismail, and Nagia M Ghanem. Vscan: an enhanced video summarization using density-based spatial clustering. In *Image Analysis and Processing–ICIAP 2013: 17th International Conference, Naples, Italy, September 9-13, 2013. Proceedings, Part I 17*, pages 733–742. Springer, 2013.

[177] Naveed Ejaz, Tayyab Bin Tariq, and Sung Wook Baik. Adaptive key frame extraction for video summarization using an aggregation mechanism. *Journal of Visual Communication and Image Representation*, 23(7):1031–1040, 2012.

[178] Boqing Gong, Wei-Lun Chao, Kristen Grauman, and Fei Sha. Diverse sequential subset selection for supervised video summarization. *Advances in neural information processing systems*, 27, 2014.

[179] Didik Purwanto, Yie-Tarng Chen, Wen-Hsien Fang, and Wen-Chi Wu. Video summarization: How to use deep-learned features without a large-scale dataset. In *2018 9th International Conference on Awareness Science and Technology (iCAST)*, pages 220–225. IEEE, 2018.

[180] Bin Zhao, Xuelong Li, and Xiaoqiang Lu. Property-constrained dual learning for video summarization. *IEEE transactions on neural networks and learning systems*, 31(10):3989–4000, 2019.

[181] Zhong Ji, Kailin Xiong, Yanwei Pang, and Xuelong Li. Video summarization with attention-based encoder–decoder networks. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(6):1709–1717, 2019.

[182] Tsu-Jui Fu, Shao-Heng Tai, and Hwann-Tzong Chen. Attentive and adversarial learning for video summarization. In *2019 IEEE Winter Conference on applications of computer vision (WACV)*, pages 1579–1587. IEEE, 2019.

.