TÉCNICO LISBOA

Movie Summarization for the automatic generation of Movie Tributes

Ricardo Manuel Mendes Bota Martins Espadinha

Thesis to obtain the Master of Science Degree in

Electrical Engineering and Computer Science

Supervisor: Doctor David Manuel Martins de Matos
Co-supervisor: Doctor Ricardo Daniel Santos Faro Marques Ribeiro

Examination Committee

Chairperson: Doctor João Manuel de Freitas Xavier
Supervisor: Doctor David Manuel Martins de Matos
Member of the Committee: Doctor Bruno Emanuel Da Graça Martins

June 2023

Acknowledgements

First, I would like to express my gratitude to my supervisors, David Martins de Matos and Ricardo Ribeiro, for their guidance, expertise, and support throughout this work.

I would also like to extend my most profound appreciation to my family for their endless love and encouragement. Their constant motivation and understanding have been a source of strength during the challenging phases of my academic journey.

Lastly, I would like to thank everyone who has ever crossed my path in recent years. Thank you for all the laughs, cries, hugs, frustrations, moments and feelings that I will keep in the form of memories and stories forever.

This journey was an incredible chapter in my life, and I cannot wait for the next one!

Lisboa, July 9, 2023
Ricardo Espadinha

That day, for no particular reason, I decided to go for a little run. So I ran to the end of the road. And when I got there, I thought maybe I'd run to the end of town. And when I got there, I thought maybe I'd just run across Greenbow County. And I figured, since I run this far, maybe I'd just run across the great state of Alabama. And that's what I did. I ran clear across Alabama. For no particular reason, I just kept on going. I ran clear to the ocean. And when I got there, I figured, since I'd gone this far, I might as well turn around, just keep on going. When I got to another ocean, I figured, since I'd gone this far, I might as well just turn back, keep right on going.

- Forrest Gump

Sumário

O rápido crescimento do conteúdo de vídeo em plataformas como o YouTube levou a um influxo de dados multimedia, apresentando desafios computacionais para o seu armazenamento, navegação, indexação, recuperação e compartilhamento. Além disso, o estudo das emoções em conteúdo multimedia tem aplicações promissoras em vários setores, incluindo publicidade e apresentação de conteúdo personalizado a um utilizador. O estado da arte também tem vindo a avançar cada vez mais nos processos criativos orientados por IA, principalmente na indústria cinematográfica.

Ao combinar o poder expressivo de filmes e música, a abordagem proposta visa capturar a essência de um filme, evocando sentimentos nostálgicos, através da geração automática de um tributo ao filme. O processo tradicional de criação destas homenagens é demorado e manual, exigindo meticulosa edição e seleção de cenas, músicas e efeitos visuais. Automatizar a geração de tributos a filmes democratiza a capacidade de homenagear obras-primas cinematográficas, promove uma apreciação mais profunda da cultura cinematográfica e contribui para a preservação e divulgação de filmes influentes para as gerações futuras.

Este trabalho combina algoritmos de segmentação de áudio e vídeo, vários modelos de aprendizagem profunda e processamento de linguagem natural para propor uma ferramenta capaz de gerar tributos de filmes emocionalmente coerentes automaticamente.

Abstract

The rapid growth of video content on platforms like YouTube has led to an influx of multimedia data, posing computational challenges for storage, browsing, indexing, retrieval, and sharing. Beyond that, the study of emotions in multimedia content has promising applications in various industries, including advertising and personalized multimedia content. The research also advances AI-driven creative processes, particularly in the film industry.

By combining the expressive power of movies and music, the proposed approach aims to capture the essence of a movie and evoke nostalgic sentiments through the automatic generation of a movie tribute. The traditional process of creating these tributes is time-consuming and manual, requiring meticulous editing and selection of scenes, music, and visual effects. Automating the generation of movie tributes democratizes the ability to pay homage to cinematic masterpieces, fosters a deeper appreciation of film culture, and facilitates the preservation and celebration of influential movies for future generations.

This work combines audio and video segmentation algorithms, various deep learning models, and natural language processing to propose a tool to generate emotionally resonant movie tributes automatically.

Palavras Chave
Keywords

Palavras Chave

Criatividade computacional

Segmentação de Áudio

Segmentação de Video

Geração de Tributos de Filmes

Sumarização de Video

BERT para Sumarização Extrativa de texto

Aprendizagem por correspondencia Afetiva entre Áudio e Video

Segmentação de Música

Processamento de Áudio e Video

Transformadores

Mecanismos de Atenção

Keywords

Computational Creativity

Audio Segmentation

Video Segmentation

Generation of Movie Tributes

Video Summarization

BERT for Text Extractive Summarization

Affective Audio-visual Correspondence Learning

Music Segmentation

Audio and Video data processing

Transformers

Attention Mechanisms

Contents

List of Figures

List of Tables

1 Introduction

There should be no boundaries to human endeavor. We are all different. However bad life may seem, there is always something you can do, and succeed at. While there's life, there is hope.
– Stephen Hawking, "The Theory of Everything"

In filmmaking, tributes play a significant role in honoring the legacy of influential movies and their creators. A tribute is a heartfelt homage, allowing audiences to revisit beloved films' remarkable moments, themes, and artistic achievements. Traditionally, creating such tributes required extensive manual effort, often involving meticulous editing and selection of scenes, music, and visual effects. This work aims to propose a pipeline for the automatic generation of compelling and emotionally resonant movie tributes.

Over the years, through the advancements of the Internet and social networking sites, Humankind has been experiencing a massive influx of multimedia, particularly video content. Considering YouTube as an example, over 500 hours of video content are currently uploaded to the platform every minute. Entertainment, education, sports, news, and general users or consumer videos are some of the different domains contributing to the fast-growing video content. Nevertheless, while big video data is an excellent source of information discovery, the computational challenges are becoming more demanding and unparalleled. This type of information is typically large, and storing, browsing, indexing, retrieving, and sharing large amounts of data are not trivial tasks. Furthermore, processing such large amounts of data requires significant time and hardware storage. As a result, efficient techniques are needed to deliver the content in a compact format while maintaining the context and the most critical aspects. Intelligent algorithms for different kinds of video processing, e.g., summarization, retrieval, recognition, and others, (re-)emerge as a pressing need.

Being a movie tribute a summary of a movie, we take advantage of the recent discoveries to present an approach that combines computer vision, audio analysis, natural language processing, and deep learning techniques to automatically generate tributes that capture the essence of a movie and evoke nostalgic sentiments.

1.1 Motivation

Usually, the way humans naturally communicate and express emotions is multimodal (Morency, Mihalcea, & Doshi, 2011). That means we can express and apprehend emotions in various ways (verbally, visually, and others). What is inspiring is that Humankind developed the capability of creating tools that help us perform or improve the performance of a given task. Because of our need and thrive on connecting and expressing ourselves amongst each other, we used this proficiency to create tools to communicate more efficiently at a larger scale (bigger audiences) through creating pieces of art, music, movies, and others. This phenomenon has been the concern and the research focus of many psychologists (Fernández-Aguilar, Navarro-Bravo, Ricarte, Ros, & Latorre, 2019). Also, more broadly, over the last three decades, interest in the study of emotions has increased notably, focusing both on the construct itself and its interaction with other concepts such as cognition, behavior, personality, and physiology (Kreibig, Samson, & Gross, 2013; Kuo, Neacsiu, Fitzpatrick, & MacDonald, 2014; Vianna, Weinstock, Elliott, Summers, & Tranel, 2006).

The research on emotions in multimedia content is also very promising to the industry. For instance, recognizing the continuous dynamic emotion evoked by movies can be used to build better multimedia intelligent applications, such as computational affective video-in-video advertising and personalized multimedia content (Yadati, Katti, & Kankanhalli, 2013; Aditya, Manvitha, Samyak, & Shamitha, 2021), to create automatic summaries and adaptive playback speed adjustment for long videos.

This thesis explores these concepts, resorting to the conjunction of movies and music, specifically through the automatic generation of movie tributes.

1.2 Problem Formulation & Objectives

As described in a previous iteration of this topic by Aparício (2015), *a movie tribute consists of a short music video containing essential parts of the movie playing along with the specified song*. The length of the video corresponds to the song's length. By coordinating two very powerful tools for human expression (music and movies), the purpose of a movie tribute consists of *reliving emotions from the movie quickly and effectively. There are videos of this kind on YouTube, made manually by people who long to gather their most meaningful scenes to remember later a movie that they have seen and enjoyed.*

This work will focus on creating a framework to generate a movie tribute taking advantage of recent developments. The primary objective is to develop a robust framework to automatically select and assemble scenes from a given movie into a given musical composition, generating a cohesive and emotionally engaging movie tribute.

In order to achieve this, a set of milestones were defined as follows:

- **1) Exploration of the recent advancements in general video summarization techniques:** Movies can be described as long videos; therefore, to better comprehend the semantics and internal interactions of this kind of structure, we must first look at how the problem of summarization is being addressed in the corresponding shorter format (videos). Through this step, we will learn the intuition behind the current methodology for selecting what is essential to keep in a summary and what is currently used for feature extraction and some video segmentation techniques.
- **2) Exploration of the recent advancements in extractive text summarization techniques:** As done by Aparício (2015), in order to reduce the processing time related to the movie, we will also focus on how to retrieve the movie highlights by applying a text summarization technique to the subtitles corpus of the movie.
- **3) Exploration of the recent advancements in affective music-video retrieval techniques:** The produced movie tribute should carry emotional coherence between the movie and music segments.
- **4) Implementation.**

Ultimately, this research aims to contribute to advancing AI-driven creative processes, specifically in the film industry. By automating the generation of movie tributes, we strive to democratize the ability to pay homage to cinematic masterpieces, fostering a deeper appreciation of film culture and facilitating the preservation and celebration of influential movies for future generations. Unlocking an intuitive and efficient way to summarize a movie accordingly to a specific soundtrack would have an enormous impact on the creation of movie trailers, for example, which are even more impactful in the industry.

1.3 Outline

The thesis is organized as follows:

- **Chapter 2** presents a *Conceptual Approach on Video Summarization* for a better understanding of the critical concepts related to general video summarization. These concepts involve the *Hierarchical Structure of a video*, the *Structure of a Video Summary*, the *Main Challenges* related to this task, and a *General Framework for Video Summarization*. We also present the existing research on the fields used in the proposed approach for this work: general video summarization, extractive text summarization, and affective music-video retrieval;
- **Chapter 3** presents our proposed solution, encompassing a detailed explanation of the movie and music streams' data processing, content selection, emotional coherence, and post-production concerns;
- **Chapter 4** provides an overview of the dataset utilized in our experiments, as well as the respective results and further discussion;
- **Chapter 5** presents our overall conclusions and directions for further research and development.

2 Background and Related Work

Our lives are defined by opportunities, even the ones we miss.

– Benjamin Button, "The strange case of Benjamin Button"

This chapter presents a *Conceptual Approach on Video Summarization* for a better understanding of the critical concepts related to general video summarization. These concepts involve the *Hierarchical Structure of a video*, the *Structure of a Video Summary*, the *Main Challenges* related to this task, and a *General Framework for Video Summarization*.

Further, we present the existing research on the fields used in the proposed approach for this work (chapter 1.2): general video summarization, extractive text summarization, and affective music-video retrieval.

2.1 Conceptual Approach on Video Summarization

The evolution of technology and the way it blends into our day-to-day basis led to extreme ease in creating video content. However, with the growth of the generation and availability of this media, there is a direct correlation between the need for new and more efficient ways to handle this type of data. That is why research on this topic has seen exponential growth over the years. Efficient video summarization techniques can facilitate efficient storage, quick browsing, indexing, fast retrieval, and quick content sharing (Tiwari & Bhatnagar, 2021).

For this section, we inspired ourselves on the structure and research highlights presented in the recent survey regarding video summarization techniques by Tiwari and Bhatnagar (2021). This survey outlines the existing video summarization frameworks and a comprehensive view of the existing approaches and techniques.

2.1.1 Hierarchical Structure of a Video

A video stores spatiotemporal information. In order to be able to manipulate and rearrange this information, we need to define some key concepts in the video structure. We followed the concepts presented by Tiwari and Bhatnagar (2021):

- **Frame:** The elementary unit of a video that is displayed in sequential order (individual images of a video stream).
- **Shot:** Collection of frames captured in an uninterrupted time interval with a single camera.
- **Scene:** Several shots representing a part of an event's complete sequence.

The final video comprises an appropriate combination of the scenes in a timeline, producing a story with its context.

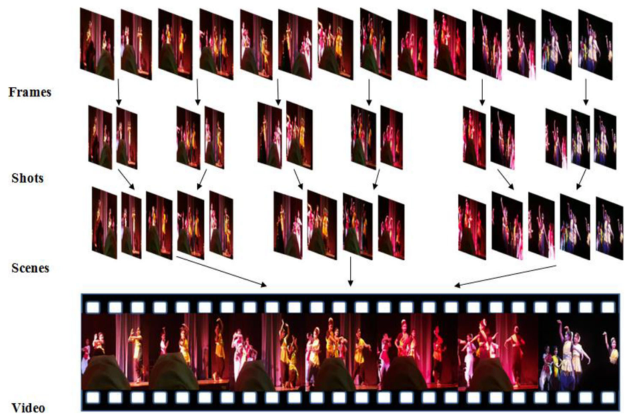


Figure 2.1: Hierarchical structure of a video. Tiwari and Bhatnagar (2021)

2.1.2 Structure of a Video Summary

Video summarization techniques aim to produce a compact representation of a given video, keeping the most relevant information intact. How this compact information is presented to the user can vary in different ways depending on the purpose of the summarization task. For example, we could summarize a Youtube video's content in text format to add to its description or in the format of isolated keyframes to present to the user while browsing. The different applications of video summarization lead to the absence of a standard definition for what kind of structures should be included or excluded from a video summary. Money and Agius (2008) identified the audio-visual cues as follows. **Keyframe cues** are the most representative frames extracted from a video sequence and must be presented in temporal order to preserve context. **Video Segment cues** are the essential continuous parts of the original video, being considered an extension of keyframe cues that generally preserve both the motion and audio elements. **Graphical Cues** use visual elements and syntax as a supplement to other cues, presenting an additional level of detail (e.g., the *in-video* text identification of the action that is being presented at that specific moment). **Textual Cues** summarize the content of the video via textual descriptors.

As mentioned before, there are multiple types of video summaries. Nonetheless, if the generated summary maintains the video format, a general mathematical representation of the problem can be formulated as follows:

Let V be the video with n frames in sequence. Then, the summarization video, S, is a collection of m keyframes, not necessarily consecutive, but in temporal order.

$$V = \{F_1, F_2, F_3, ..., F_n\} \tag{2.1}$$

$$S = \{F_{x_1}, F_{x_2}, ..., F_{x_m} \mid 1 \leq x_i \leq n \text{ and } m \ll n\} \tag{2.2}$$

It is essential to mention here that there is no specified need to present the video segments in the temporal order for our specific problem of generating movie tributes. Music/video synchronization and coherence must be the top priority.

2.1.3 Main Challenges

Summarizing a video stream presents several challenges. The sequential nature of a video makes it a more complex type of content than singular images. The spatiotemporal dependencies of this type of data must be taken into account to produce a summary that properly maintains the context of the original video.

A video agglomerates a wide range of semantics in various ways, such as still and moving images, sound, music, and text. This multimodal nature makes this task much more complex than analyzing text documents or single images (Money & Agius, 2008).

Video summarization is a subjective task. Different users may have different opinions and preferences over the summaries, all being valid (Otani, Nakashima, Rahtu, & Heikkilä, 2019). This hinders the comparison of a generated summary with well-defined ground truth.

In order to identify which frames or parts of a video will be present in the summary, an importance score must be generated. This score depends on the type of summary, user requirements, and genre of the videos. Since what defines importance may vary for different persons, determining what is essential is also a highly subjective task (Otani et al., 2019).

2.1.4 General Framework for Video Summarization

B. T. Truong and Venkatesh (2007) analyzed the concept of video abstraction as *a mechanism for generating a summary of a video, which can either be a sequence of stationary images (keyframes) or moving images (video skims)*, corresponding to Static and Dynamic Summaries. Apart from those, other types of video summaries can be generated depending on the purpose of the task they are aimed for. Although they differ in structure, all of them must be able to carry the semantic content and maintain the overall context of the original video. Therefore, different proposed techniques are addressed in the literature according to different kinds of summaries.

A **Static Summary** is a collection of singular frames (*keyframes*) extracted from an original video depending upon the summarization criteria (Kanehira, Van Gool, Ushiku, & Harada, 2018; Khan, Shao, Ali, & Tumrani, 2020; T. Liu & Kender, 2002b; Money & Agius, 2008; Zhou et al., 2018). This summary is structured as a set of images; therefore, it does not contain audio cues and may lack continuity. However, its generation reveals efficiency in computational time and memory (Sreeja & Kovoor, 2019).

A **Dynamic Summary** is a concatenation of different segments of the original video (Potapov, Douze, Harchaoui, & Schmid, 2014; Sreeja & Kovoor, 2019; X. Zhu, Loy, & Gong, 2016). This kind of summary may contain audio cues, provide the user with more information, and exhibit continuity.

An **Image Summary** extracts a single frame or a combination of frames from the original video and gathers them all into a single image (Chen, Lu, & Hu, 2012). These summaries represent the flow of events or create what is referred to as a schematic storyboard by annotating the image with arrows and text describing the respective motion.

A **Text Summary** is a textual description of the original video sequence (Sah et al., 2017; Dilawari & Khan, 2019). These types of summaries are generated with the use of Natural Language Processing (NLP) techniques. Even though they may be efficient in terms of storage and computational cost, they do not include audio and visual cues, risking not expressing the complete information derived from these components.

A **Hierarchical Summary** is a collection of summaries from the same original video distributed on different levels of abstraction. These levels increase in detail (number of selected frames) from the highest to the lowest one (Herranz & Martinez, 2010; X. Zhu, Wu, Fan, Elmagarmid, & Aref, 2004; X. Zhu, Elmagarmid, Xue, Wu, & Catlin, 2005). The advantage of this kind of summary is that it can assist the users in determining what is appropriate by giving the user various levels of summary (Tiwari & Bhatnagar, 2021).

A **Multi-view Summary** refers to summaries generated out of a set of different videos simultaneously recorded or captured by more than one camera for the same purpose (Hussain et al., 2021; Panda & Roy-Chowdhury, 2017; Fu et al., 2010). These summaries are beneficial for surveillance and sports videos.

The type of summary we aim to produce with this work is a Dynamic Summary. The general framework to generate this type of summary out of the existing video sequence is shown in Fig. 2.2.

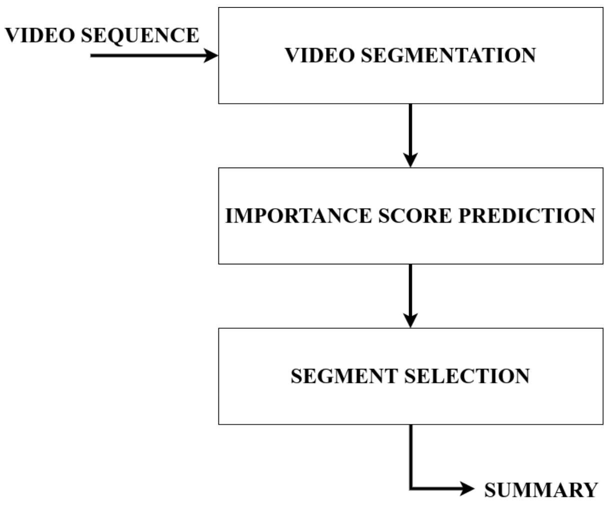


Figure 2.2: General framework for Dynamic Summary generation. Tiwari and Bhatnagar (2021)

The process consists of three phases, as follows. **1) Video Segmentation** divides the original video stream into smaller parts that can be comprehended and processed independently. Each of these parts is a composition of sequential frames defining a specific activity or moment and correctly carrying its meaning. For structured videos, where the different segments are easy to locate, this process identifies well-defined shots and scenes from the video stream. **2) Importance Score Prediction** is considered a crucial step as it aims to attribute a score to each segmented unit of the video that defines what will be present in the summary. The challenge of this step is due to the subjectivity related to defining what is essential, as this criterion is not always the same. The formulation of importance may vary depending on the application domain, user preferences, or specific requirements. Some previous works have focused on visual "interestingness", compactness, and diversity to produce this score (Gygli, Grabner, Riemenschneider, & Van Gool, 2014; Gygli, Grabner, & Van Gool, 2015; K. Zhang, Chao, Sha, & Grauman, 2016a; B. Zhao & Xing, 2014; Zhou et al., 2018; Atencio, German, Branch, & Delrieux, 2019). **3) Segment Selection** removes the redundant frames and selects the segments in the intended summary based on the computed importance scores.

2.2 Related Work

By looking at the research community, we can find significant contributions related to the fields we need to consider to generate a movie tribute. In this section, we compiled an overview of the current state-of-the-art related to the topics of *Video Summarization Techniques*, *Extractive Text Summarization Techniques*, and *Music - Video Retrival*.

2.2.1 Video Summarization Techniques

The majority of the techniques for automatic video summarization fall into two broad categories. **Unsupervised** approaches are based on techniques that rely on manually designed criteria to prioritize and select frames or sub-shots from videos (H. J. Zhang, Wu, Zhong, & Smoliar, 1997; Mundur, Rao, & Yesha, 2006; Lee, Ghosh, & Grauman, 2012; Ngo, Ma, & Zhang, 2003; Lu & Grauman, 2013; Hong et al., 2009; Khosla, Hamid, Lin, & Sundaresan, 2013; T. Liu & Kender, 2002a; Ma, Lu, Zhang, & Li, 2002; De Avila, Lopes, da Luz Jr, & de Albuquerque Araújo, 2011; Furini, Geraci, Montangero, & Pellegrini, 2010; Y. Li & Merialdo, 2010; Potapov et al., 2014; Morere, Goh, Veillard, Chandrasekhar, & Lin, 2015; G. Kim & Xing, 2014;

B. Zhao & Xing, 2014; Song, Vallmitjana, Stent, & Jaimes, 2015; Chu, Song, & Jaimes, 2015). **Supervised** methods make use of techniques that leverage human-edited summary examples (or frame importance ratings) to learn how to summarize novel videos (B. Gong, Chao, Grauman, & Sha, 2014; Gygli et al., 2015; K. Zhang et al., 2016a; Gygli et al., 2014; Chao, Gong, Grauman, & Sha, 2015).

The most recent studies show that machine learning techniques yield better results than traditional methods (presented in Section 2.2.1.1). At the time of the writing of this document, Deep Learning models are also producing efficient results, and the current technology is being explored extensively by researchers to produce better results (Tiwari & Bhatnagar, 2021). This is why our focus in this Section will be more on exploring deep learning approaches to video summarization. Nevertheless, an overview of the evolution of video summarization techniques before introducing deep learning-based architectures is presented below.

2.2.1.1 Paradigm shift in Video Summarization Techniques

Over the years, the video summarization process and techniques have seen many advancements and a broad paradigm shift (Sharghi, Gong, & Shah, 2016). As previously said, selecting keyframes is regarded as the most crucial stage in any approach, with the goal of obtaining a summary based on the video's semantic information. The earlier techniques mainly used low-level appearance cues, motion cues, and graph modeling to identify the key frames from a video sequence such that the identified key frames are important, diverse, and representative (Tiwari & Bhatnagar, 2021). It is up to the programmer to measure the importance and diversity of the key frames through the cues.

Rav-Acha, Pritch, and Peleg (2006) presents a dynamic video synopsis, where most of the activity in the video is condensed by simultaneously showing several actions, even when they initially occurred at different times. To achieve this type of video synopsis, the authors present two approaches. The first approach uses optimizations on Markov Random Fields (Kolmogorov & Zabin, 2004) and a 2D graph where each node corresponds to a spatial location in the synopsis movie. The second approach starts by detecting the moving objects and then the optimization is performed. There are other object-based video summary methods in the literature (C. Kim & Hwang, 2000; Ferman & Tekalp, 1997; Stefanidis, Partsinevelos, Agouris, & Doucette, 2000), and they all use the detected objects to select significant frames. Goldman, Curless, Salesin, and Seitz (2006) also uses these earlier techniques by presenting a method for visualizing short video clips in a single static image using the visual language of storyboards. These schematic storyboards are assembled from multiple input frames and annotated using outlines, arrows, and text describing the motion in the scene.

Later on, high-level supervised information, such as rich web images, was addressed, and weakly supervised priors were utilized to capture the user-oriented value of a video's visual content. Khosla et al. (2013) had the idea of using web images as a prior to facilitating the creation of summaries of user-generated videos. Their intuition was that people tend to take pictures of objects to capture them in a maximally informative way. As a result, such photos with specific objects could help identify the same set of objects in a video to produce a dedicated summary. The authors aimed to collect *cannonical viewpoints* of various objects and connect them to an object class (e.g. automobiles). Using these discovered viewpoints, they developed a discriminative model that can recognize similar frames in a video that each capture a different instance of the object class. These keyframes can be used to represent the video and summarize its content. To do this, the authors proposed a variant of a multi-class Support Vector Machine (SVM) (Crammer & Singer, 2001) framework that jointly discovers the canonical viewpoints as well as learns their discriminative decision boundaries.

Another work on video summarization involving web content focused on generating summaries from different large datasets of online images and videos. G. Kim, Sigal, and Xing (2014) addressed this problem using Flickr images and YouTube videos. They started from the idea that the characteristics of the two media types are different yet complementary to develop a fast and easily-parallelizable approach for creating high-quality video summaries and novel structural summaries of online images as storyline graphs. The video summarization is achieved by applying the diversity ranking algorithm proposed by G. Kim, Xing, Fei-Fei, and Kanade (2011) on the similarity graphs between images and video frames.

Other techniques for video summarization have included Category and Domain-specific summarization. For example, Potapov et al. (2014) proposed a category-specific summarization approach that performs the segmentation of the given video by considering general change points in addition to shot boundaries, resulting in semantically-consistent segments, in contrast to video-specific importance (Y. Liu, Zhou, Liu, De la Torre, & Liu, 2010; De Avila et al., 2011; B. T. Truong & Venkatesh, 2007). After the segmentation, an SVM classifier that was trained on videos for the category at hand assigns importance scores to each segment and finally assembles the segments with the highest scores.

Tiwari and Bhatnagar (2021) establish that the summarization process based on user preferences is more prevalent and that Machine Learning and, more specifically, Deep Learning techniques can be used to accomplish it. Machine Learning has proved to be beneficial for the summarization process to yield better results compared to the traditional methods. Therefore, a more detailed analysis of the advancements in video summarization using Deep Learning techniques is presented in the next section.

2.2.1.2 Deep Learning in Video Summarization

Deep Learning concepts and architectures must be analyzed to understand the literature further.

Movies and music are sequences. The context in each segment of these sequences depends on the previously gathered knowledge. For example, it is unclear how a traditional neural network could reason about previous events in the film to classify what is happening at a given time. Because of this, the deep learning architectures that show better results in video summarization are the ones dedicated to processing sequential data.

Long Short-Term Memory

Deep Neural Network (DNN) is a highly expressive model that can learn highly complex vector-to-vector mappings. The Recurrent Neural Networks (RNN) is a DNN adapted to sequence data, and as a result, the RNN is also highly expressive (Jozefowicz, Zaremba, & Sutskever, 2015). RNNs are networks with loops in them, allowing information to persist. An RNN can be thought of as multiple copies of the same network, each passing a message to a successor (Olah, 2015).

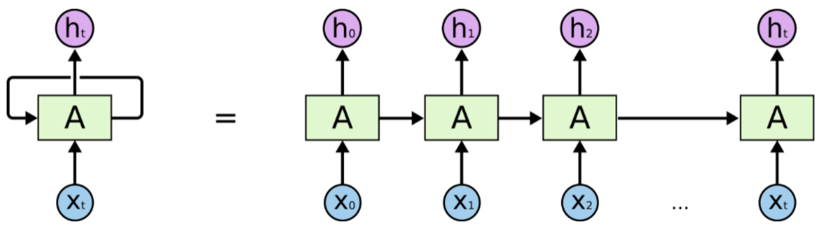


Figure 2.3: An unrolled RNN. Olah (2015)

This chain-like nature in Fig. 2.3 reveals that RNNs are intimately related to sequences and lists. Therefore, RNNs have been used in various problems such as speech recognition, language modeling, translation, image captioning, and others.

The concern regarding this architecture is that it is entirely possible for the gap between the relevant information and the point where it is needed to become large enough that RNNs become unable to learn to connect the information. This issue with the "default" RNN architecture, known as short-term memory, is caused by the well-known vanishing gradient problem, which also occurs in other neural network architectures. As Phi (2018) explains, the training process of an RNN uses an application of back-propagation called back-propagation through time. By doing this, each node computes its gradient with respect to the effects of the gradients in the node before it. Therefore, if the adjustment to the node before it is small, the adjustment to the current node will be even more minor. As a result, the gradient values will exponentially shrink as they propagate through each step. That causes the early nodes not to learn.

Hochreiter (1998) made a theoretical analysis of the problem of the vanishing gradients, and some alternative methods were briefly discussed. Two specialized recurrent neural networks appeared: Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU).

LSTM is a particular type of RNN capable of learning long-term dependencies. The model was introduced by Hochreiter and Schmidhuber (1997). Since then, many people have refined and popularized it in the following work. The mechanisms behind this architecture are the cell state and gates.

The cell state transfers relative information down the sequence chain, allowing for information from the earlier time steps to make its way to the following time steps, reducing the effects of short-term memory. Each LSTM cell adds or removes information to the cell state via three different gates:

- **Forget Gate:** Decides what information from the previous hidden state and the current input should be thrown away or kept;
- **Input Gate:** Responsible for updating the cell state, deciding what information is relevant to add from the current step;
- **Output Gate:** Decides what the next hidden state, containing information on previous inputs, should be.

The conceptual diagram of an LSTM unit (Fig. 2.4) and its algebraic definitions (Eq. 2.3) are presented below.

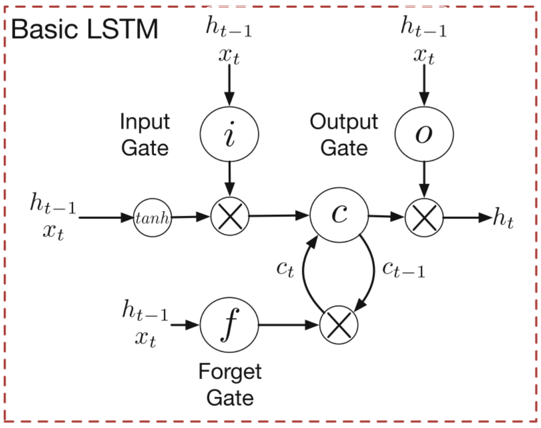


Figure 2.4: The LSTM unit. K. Zhang et al. (2016b)

$$\begin{aligned}
\mathbf{i_t} &= sigmoid(\mathbf{W_i}\left[x_t^T, h_{t-1}^T\right]^T) \\
\mathbf{f_t} &= sigmoid(\mathbf{W_f}\left[x_t^T, h_{t-1}^T\right]^T) \\
\mathbf{o_t} &= sigmoid(\mathbf{W_o}\left[x_t^T, h_{t-1}^T\right]^T) \\
\mathbf{c_t} &= \mathbf{i_t} \odot \tanh(\mathbf{W_c}\left[x_t^T, h_{t-1}^T\right]^T) \\
&\quad + \mathbf{f_t} \odot \mathbf{c_{t-1}} \\
\mathbf{h_t} &= \mathbf{o_t} \odot tanh\left(\mathbf{c_t}\right)
\end{aligned} \tag{2.3}$$

LSTMs can model dependencies with a data-dependent on/off switch, which is extremely powerful for modeling sequential data (Graves, Mohamed, & Hinton, 2013). This architecture has also been used in various image/video-related tasks (Xu et al., 2015; Jin, Fu, Cui, Sha, & Zhang, 2015; Yao et al., 2015; Venugopalan et al., 2015, 2014).

Inspired by the success of applying LSTMs to structured prediction problems such as speech recognition (L. Deng, Hinton, & Kingsbury, 2013; Graves et al., 2013; Graves & Jaitly, 2014) and image and video captioning (Donahue et al., 2015; Yao et al., 2015; Venugopalan et al., 2015, 2014; Karpathy & Fei-Fei, 2015), K. Zhang et al. (2016b) proposed an LSTM-based model for video summarization focused on *keyframe* and *key subshot selection* (Fig. 2.5). They demonstrated that the sequential modeling aspect of LSTM is essential for video summarization. The model's memory cells are used to understand how storylines evolve, allowing it to know when to ignore or incorporate prior occurrences while making decisions.

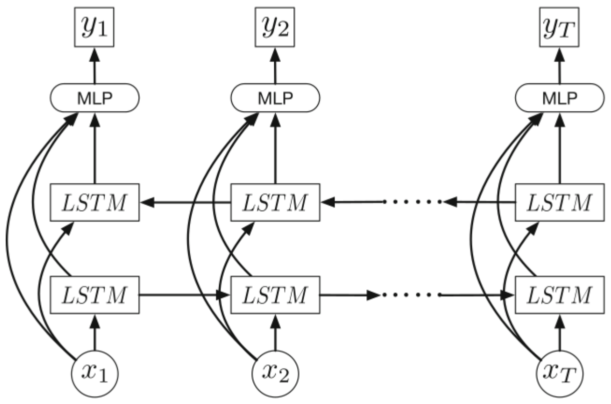


Figure 2.5: The vsLSTM model for video summarization. Each LSTM block is an LSTM unit, shown in Fig. 2.4. K. Zhang et al. (2016b)

This model takes as input a sequence of extracted features (x_i). It can model the temporal interdependencies between the past and the future by using a Bidirectional Long Short-term Memory (BiLSTM). A BiLSTM is composed of two LSTM layers that consume the input forward and backward, respectively. The combination of hidden states and visual features passes through an Multilayer Perceptron (MLP), obtaining a binary indicator vector (being selected or not) or frame-level importance scores (y_j).

To enhance the summary's diversity, the authors further introduce the Determinantal Point Process (DPP) algorithm to vsLSTM, called dppLSTM. This combination takes advantage of LSTMs to measure which frames are essential to the summary and from the DPP to ensure diversity, which can only be measured "collectively", not as independent or sequential frames.

The modeling advantage provided by the DPP has been exploited in other DPP-based summarization methods (Sharghi et al., 2016; Sharghi, Laurel, & Gong, 2017; Kulesza & Taskar, 2011; Chao et al., 2015; K. Zhang et al., 2016a; B. Gong et al., 2014).

In another work that makes direct use of the LSTM architecture, B. Zhao et al. (2018) developed a structure adaptive video summarization approach that integrates shot segmentation and video summarization into a Hierarchical Structure-Adaptive RNN, denoted as HSA-RNN.

As presented in Fig. 2.6, HSA-RNN has two layers composed of a BiLSTM.

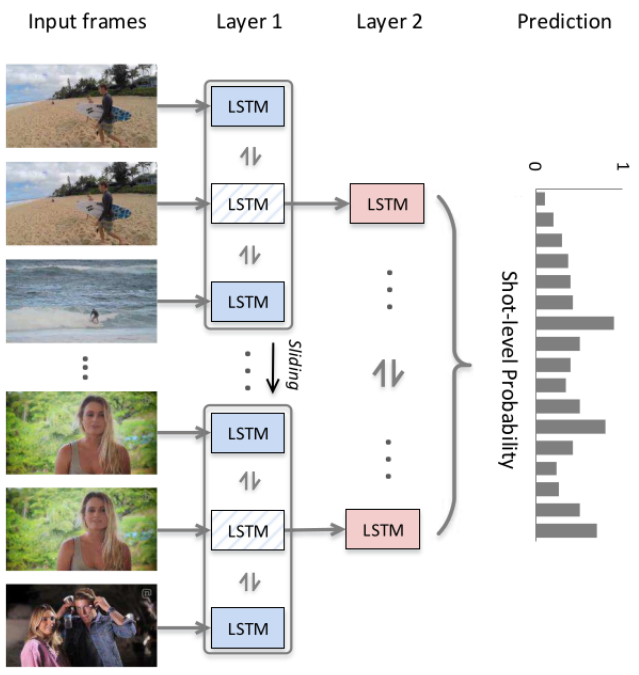


Figure 2.6: The diagram of the proposed HSA-RNN. B. Zhao et al. (2018)

The first layer was explicitly implemented to take advantage of the video structure. It corresponds to a fixed-length sliding BiLSTM capable of moving along the video frames, attempting to detect shot boundaries step by step. Once the shot boundaries are detected, the hidden states corresponding to those locations are taken as the encoded shot features and input to the upper layer. As stated by the authors, the intuition lying behind the sliding BiLSTM is that: 1) The sliding operation enables short LSTM to process long videos. It avoids long temporal dependency exploitation among thousands of frames, mitigating the vanishing gradient problem. 2) The BiLSTM jointly captures the forward and backward information in frame sequence, which can detect the shot boundary effectively. 3) The sliding BiLSTM processes only the local frames at each step, which reduces the interference of irrelevant global information.

The second layer is designed to capture the forward-backward temporal dependencies among shots and predict the probability of each shot being selected in the summary.

Generative Networks

Over the years, many techniques aiming to generate data similar to the input samples have emerged. The primary principle behind generative networks is to capture the underlying distribution of the data. This distribution can not be observed directly and must be approximately inferred from the training data. Overall, these networks can uncover underlying latent variables in a dataset.

Early deep generative approaches used Autoencoders (Hinton & Salakhutdinov, 2006). These architectures are composed of two neural networks (encoder and decoder) that learn the best encoding-decoding scheme using an iterative optimization process. At each step of the training process, we feed the encoder with some data and compare the encoded-decoded output with the original data, back-propagating the error through the architecture to update the networks' weights. The main goal of these networks is to compress the underlying distribution into a lower-dimensional latent space by reducing the layer sizes continuously. This process of "compressing" the data is called dimensionality reduction. If the encoder and decoder architectures have only one layer without non-linearity (linear autoencoder), then both parts are simple linear transformations that can be expressed in matrices. In this case, we are looking for the best linear subspace to project data on, with as little information loss as possible, just like Principal Component Analysis (PCA) (Hotelling, 1933) does. The use of linear autoencoders allows the newly generated features to be dependent (no orthogonality constraints), contrarily to PCA. Indeed, several bases can be chosen to describe the same optimal subspace, implying that several encoder-decoder pairs can give the optimal reconstruction error.

The objective is to find the pair encoder-decoder that keeps the maximum of information when encoding and has the minimum of reconstruction error when decoding. Denoting respectively E and D the families of encoders and decoders we are considering, then the dimensionality reduction problem can be written as $(e^*, d^*) = arg\, min_{(e,d) \in E \times D} \epsilon(x, d(e(x)))$ where $\epsilon(x, d(e(x)))$ defines the reconstruction error between the input data, x, and the encoded-decoded data $d(e(x))$.

The original autoencoder architecture presented above does not consider the organization of the latent space. This property turned out to be a significant problem if we want to use the decoder of our autoencoder for generative purposes that require the latent space to be regular enough. Kingma and Welling (2013) proposed a solution that avoids overfeating and ensures that the latent space has suitable properties for the generative process. The Variational Autoencoder (VAE) (Fig. 2.7) adds a regularization term to the loss function. Instead of encoding an input as a single point, it encodes it as a distribution over the latent space.

In VAEs, the loss function comprises a reconstruction term (which makes the encoding-decoding scheme efficient) and a regularization term (responsible for making the latent space

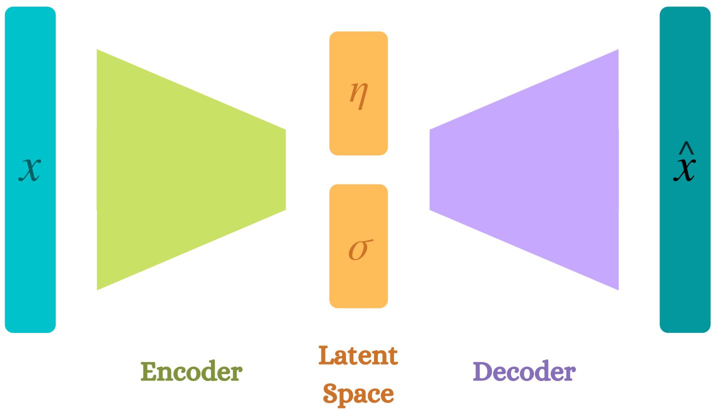


Figure 2.7: Schematic representation of VAE.

regular).

To specifically generate new instances based on some data, a new framework for estimating generative models via an adversarial process was proposed by Goodfellow et al. (2014): Generative Adversarial Networks (GAN). GANs are a way to make a generative model by having two neural networks compete with each other (Fig. 2.8). The **Generator** network turns noise into an imitation of the data to try to trick the **Discriminator** and the **Discriminator** network tries to identify real data from fakes created by the **Generator**. The main goal is for the **Generator** to learn a function that transforms a simple distribution (white noise) into a complex distribution, representing the desired data.

After training, it is possible to use the Generator network to create new data that's never been seen before.

Inspired by the research on these new technologies and the use of LSTMs in the HSA-RNN (Fig. 2.6), Mahasseni et al. (2017) proposed a new architecture (Fig. 2.9) that uses the unsupervised generative-adversarial learning to train the LSTMs.

This architecture starts by extracting deep features from a given frame using a Convolutional Neural Network (CNN), specifically, GoogleNet (Szegedy et al., 2015). Then, the sequence of outputs of the CNN goes through the sLSTM for capturing long-range dependencies among the frames and selecting a subset of these frames (attributing an importance score to each frame), producing a new deep feature representing the input sequence.

The input sequence of frames' features, x, is weighted with these importance scores s_t.

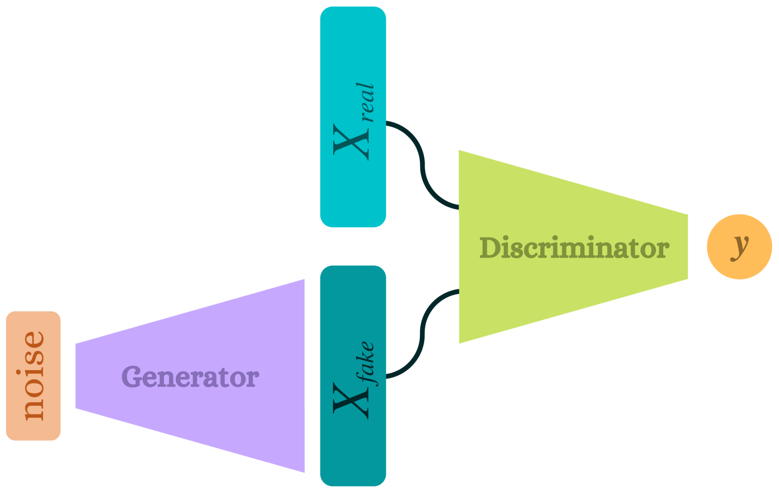


Figure 2.8: Schematic representation of GAN.

Then the subset of frames for which $s_t = 1$ is forwarded to the eLSTM to encode the selected frames to a deep feature e. The last component of the summarizer is the dLSTM, which takes e as input, and reconstructs a sequence of features corresponding to the input video, $\hat{x} = \hat{x}_1, \hat{x}_2, ..., \hat{x}_M$.

The discriminator's (cLSTM) task is to distinguish between x and $\hat{x}$ belonging to two classes: 'original' and 'summary'. The discriminator serves to estimate a representation error between the original video and the video summary. This way, the dLSTM and the cLSTM form the GAN. The summarizer and discriminator networks are trained adversarially until the discriminator cannot discriminate between the reconstructed videos from summaries and the original videos.

Zhou et al. (2018) argued that the DPP-LSTM (the second iteration of the architecture presented in Fig. 2.5 proposed by K. Zhang et al. (2016b)) *supervised learning cannot fully explore the potential of deep networks for video summarization because there does not exist a single ground truth summary for a video* and that the adversarial nature of the architecture presented in Fig. 2.9 (Mahasseni et al., 2017) makes the training unstable, which may result in model collapse. This way, the authors Zhou et al. (2018) proposed an approach (Fig. 2.10) that employs unsupervised video summarization using a diversity-representativeness reward function to simulate how people summarize films. The summaries are generated by predicting the probabilities

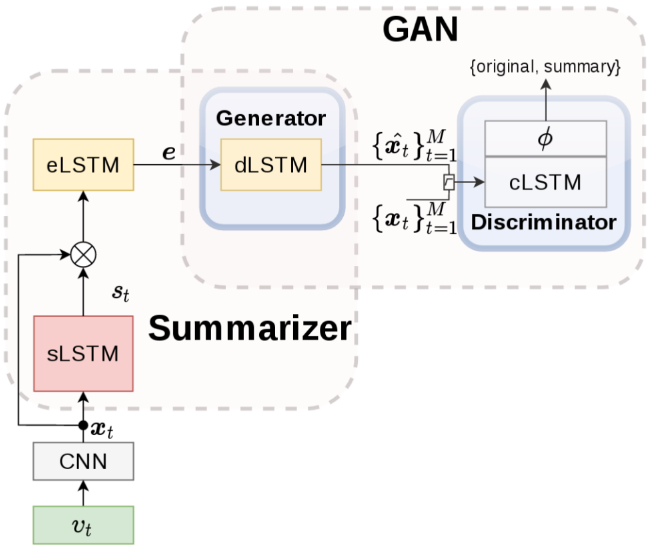


Figure 2.9: Adversarial LSTM Networks proposed. Mahasseni et al. (2017)

that a given frame is a key-frame and then selecting summary frames based on this probability. This methodology applies Reinforcement Learning (RL) to unsupervised video summarization. RL is the technique applied to an agent facing a problem by learning behavior through trial-and-error interactions with a dynamic environment (Kaelbling, Littman, & Moore, 1996).

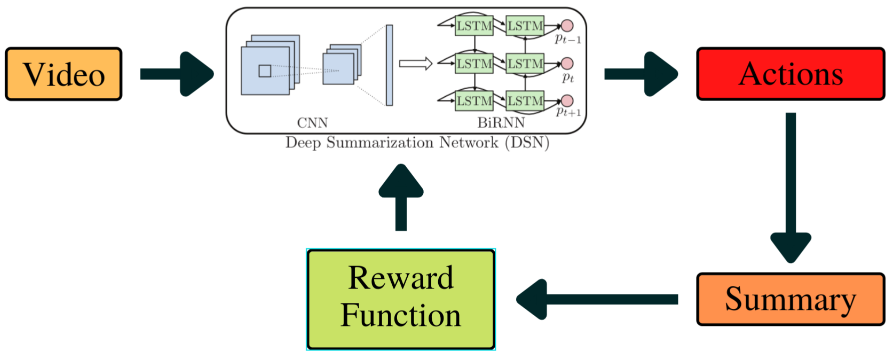


Figure 2.10: Scheme of the training process of Deep Summarization Network (DSN) reinforcement learning. Zhou et al. (2018)

DSN has an encoder-decoder architecture. The encoder performs feature extraction from the video frames using a CNN. The decoder is a BiLSTM. DSN receives a video as input and passes it to the encoder-decoder model that outputs a sequence of binary variables, considered the Actions, representing which parts of the video are selected as the Summary.

During training, DSN will receive a reward $R(S)$ that evaluates the quality of generated summaries. The objective of DSN is to maximize the expected rewards over time by producing high-quality summaries.

$$R(S) = R_{div} + R_{rep} \tag{2.4}$$

The reward function is composed by two other that aim to measure Diversity (R_{div}) and Representativeness (R_{rep}). Let the indices of the selected frames be $\gamma = y_i | a_{y_i} = 1, i = 1, ..., |\gamma|$:

$$\begin{aligned} R_{div} &= \frac{1}{|\gamma|(|\gamma| - 1)} \sum_{t \in \gamma} \sum_{\substack{t' \in \gamma \\ t' \neq t}} d(x_t, x_{t'}) \\ \text{where } d(x_t, x_{t'}) &= 1 - \frac{x_t^T x_{t'}}{\|x_t\|_2 \|x_{t'}\|_2} \end{aligned} \tag{2.5}$$

$$R_{rep} = \exp(-\frac{1}{T} \sum_{t=1}^{T} \min_{t' \in \gamma} \|x_t - x_{t'}\|_2) \tag{2.6}$$

Attention Mechanism

Up to this point, we have seen that LSTMs (and GRUs) provide a way to carry only relevant information from one step to the next through gates. Although the longer the input sequence length, the more difficult it is for the hidden vector to capture the context (Cho, Van Merriënboer, Bahdanau, & Bengio, 2014; Koehn & Knowles, 2017).

In the book *The Principles of Psychology* by James (2007), the author wrote that *Attention is the taking possession by the mind, in clear and vivid form, of one out of what seem several simultaneously possible objects or trains of thought. Focalization, concentration, of consciousness are of its essence.* Attention is a behavioral and cognitive process of selectively concentrating on specific parts of information while ignoring others. For instance, if we want to guess which person is the oldest in an image of a group of people, intuitively, we will not analyze all the aspects of the image. Instead, perhaps we would look at the faces in the image for some specific features. It was based on this notion that Bahdanau, Cho, and Bengio (2014) introduced the fundamental concept of attention mechanism for neural networks. His work focused on NLP; therefore, the attention mechanism appeared as an improvement over the encoder-decoder architecture for neural machine translation systems.

The use of Attention provides a way to build an architecture that consumes all hidden states of the LSTM instead of using just the last hidden state as a proxy for the entire sentence. This way, the context will not weaken through the sequence.

There are two main distinct attention algorithms, hard and soft. Xu et al. (2015) describe both as follows. Soft Attention is when the context vector is computed as a weighted sum of the encoder's hidden states. Opposingly, Hard Attention uses attention scores to select a single hidden state for the context vector. The challenge with this latest algorithm is that the function that selects the hidden state is not differentiable (e.g., argmax), leading to more complex techniques than back-propagation to update the model's weights.

Ji et al. (2019) utilized this mechanism to treat video summarization as a sequential encoder-decoder problem and formulate it with an attention-based LSTM framework, the Attentive encoder-decoder networks for Video Summarization (AVS).

Inspired by the success of the Attention mechanisms in machine translation approaches (Bahdanau et al., 2014; Luong, Pham, & Manning, 2015), the AVS is able to assign importance weights to different shots/frames of the input instead of treating all the input ones equally.

The AVS employs an Attention mechanism in the encoder-decoder framework (Fig. 2.11).

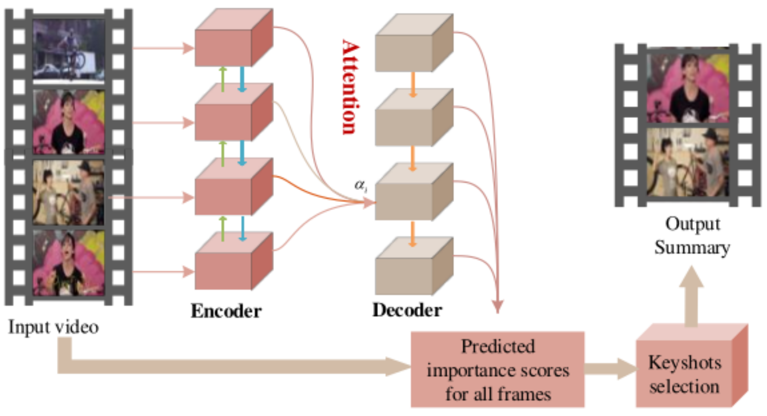


Figure 2.11: Scheme of the AVS framework. Ji et al. (2019)

The encoder is a BiLSTM to encode the necessary information in sequence, considering the temporal relation of video frames. The decoder is an LSTM with an attention mechanism that allows the decoder to selectively focus on only a subset of inputs by increasing their attention weights. Once obtained the predicted importance scores for all the frames,the shot boundaries are defined by applying the Kernel Temporal Segmentation (KTS), proposed by Potapov et al.

(2014), and the key segments selected.

Fajtl et al. (2019) argued that the use of encoder-decoder architectures has a significant problem because of the fixed size of the latent space, completely independent from a possible variation in the input's length. This "bottleneck" implies a higher information loss for longer sequences. To address this problem, the authors proposed a supervised keyshot-based architecture (Fig. 2.12) that completely replaces the LSTM encoder-decoder network with soft self-attention and a two-layer, fully connected network for regression of the frame importance score.

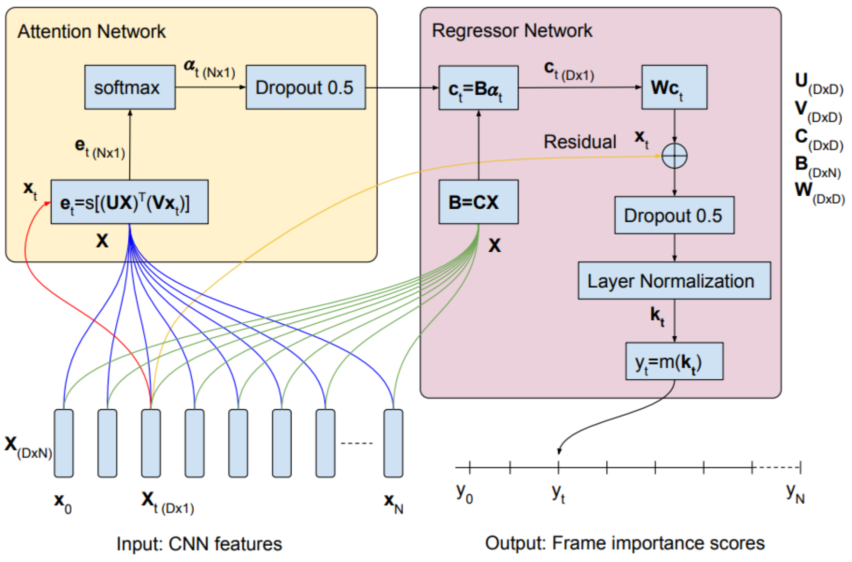


Figure 2.12: Diagram of VASNet network attending sample x_t. Fajtl et al. (2019)

The input to this architecture is a sequence of CNN feature vectors, extracted for each video frame. The self-attention weight $e_{t,i}$ is computed according to Luong et al. (2015):

$$e_{t,i} = s[(Ux_i)^T(Vx_t)] \qquad t = [0, N), \quad t = [0, N) \tag{2.7}$$

The authors justify using Equation 2.7 of the multiplicative attention, opposing to additive because it is easier to parallelize, and both formulas showed similar performance. The attention vector e_t is then converted to actual probabilities through a *softmax* function representing the importance of input features concerning the desired frame-level score at the time t. In the end, a two-layer neural network performs the frame score regression.

An interesting factor of this approach is that, even if the objective is keyshot summarization, the importance scores are attributed at a frame level. After that, the authors maximize the total frame score within each keyshot for selecting the ones that will end up in the summary. The keyshot boundaries are detected by the KTS method (Potapov et al., 2014).

Transformers

The Transformer architecture was introduced by Vaswani et al. (2017) in a paper called *Attention is all you need* to improve the performance of deep learning NLP translation models using attention mechanisms.

At its core, the Transformer is composed by a stack of encoders and decoders (Fig. 2.13). The original architecture uses six of each.

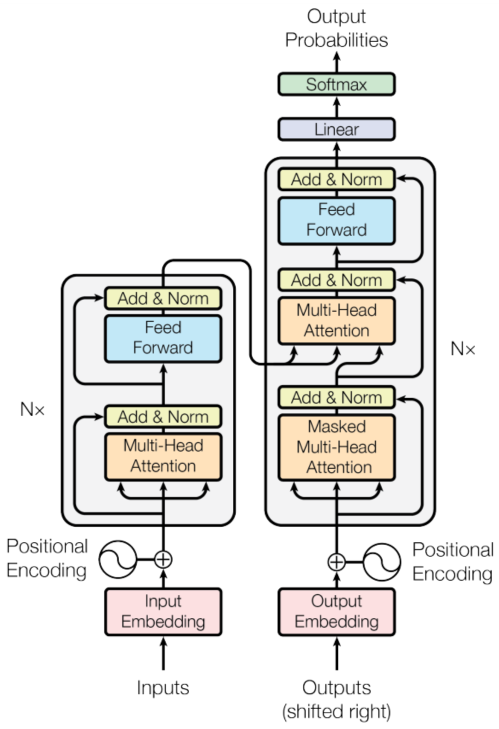


Figure 2.13: The Transformer - model architecture. Vaswani et al. (2017)

In training, the input enters the Encoder consisting of a self-attention layer to compute the relationship between each word of the input sequence and a Feed-forward layer. By stacking multiple encoders on top of each other, the model can further encode the information where each layer has the opportunity of learning different attention representations, potentially improving the predictive power of the Transformer. The sequence that we are trying to obtain enters the Decoder consisting of a first self-attention layer to compute the relationship between each word of the output sequence, a second encoder-decoder attention layer that aims to compute the relation between the output of the self-attention layer below it and the encoded input sequence, and a Feed-forward layer as well. Each stacked Decoder takes as inputs the multiple outputs from the layers of encoders, allowing the model to focus on different combinations of attention, once again, potentially improving the predictive power.

The Multi-Head Attention is a way to compute the Attention with a greater power of discrimination by combining several similar Attention calculations performed in parallel.

Both inputs of the encoder and decoder stacks pass through a Positional Embedding layer to obtain the position information of the inputs since the sequence is processed simultaneously, contrarily to RNN architectures (LSTM and GRU).

The way these models are trained is as follows. The input sequence passes through a positional embedding layer and enters the Encoder's stack, which outputs an encoded input representation. The target sequence also passes through a positional embedding layer and enters the Decoder stack alongside the encoded representations of the input. The output will be a sequence to be compared, via a Loss function, to the target one, generating gradients to train the Transformer during back-propagation. As stated before, the complete target sequence is fed to the Decoder, allowing it to access all the single inputs (past and future ones). However, the goal is for the Decoder to predict the word based only on the past ones. Therefore, it uses an attention mask in the self-attention layer to prevent the Decoder from "peaking" ahead at the rest of the target sentence when predicting the next word.

The steps are identical to the training process during inference, but we do not have a complete sequence for the decoder. Therefore, at the start of the prediction, we use an empty sequence with only a start-of-sentence token; the decoder will predict a word at a time with the concatenation of the previously predicted words until it predicts an end-of-sentence token.

Narasimhan et al. (2021) have proposed a multimodal summarization model, shown in Fig. 2.14, which takes a video and a natural language text as inputs to generate a summary of the video conditioned by the text.

Note that, for generic video summarization, the text input is a system-generated video description.

The authors formulate the video summarization task as a per-frame binary classification

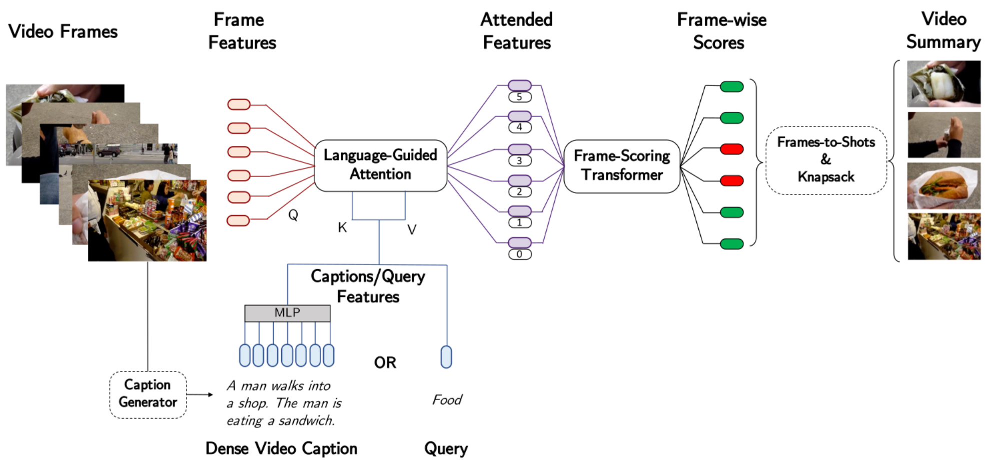


Figure 2.14: Overview of CLIP-It. Narasimhan et al. (2021)

problem. The image and text embeddings are extracted using pre-trained networks for each purpose, respectively.

They modified the Multi-Head Attention described by Vaswani et al. (2017) to a Language-Guided Multi-Head Attention to fuse information across the video and language modalities efficiently and infer long-term dependencies across both. Besides, because the objective is for all sentences of the text description to attend to all frames in the video, a single attention layer is not sufficient.

After that, a Frame-Scoring Transformer uses the fused image-text representations as input and outputs scores to individual frames in the video. The image-text embeddings are fed to the bottom of both the encoder and decoder stacks. Similar to Vaswani et al. (2017), the authors add positional encoding to the input embeddings at the bottom of the encoder and decoder stacks to insert information about the relative positions of the tokens in the sequence.

Finally, the frame-level scores are converted into shot-level scores, and these shots are selected and arranged to produce the final summary.

2.2.2 Extractive Text Summarization Techniques

2.2.2.1 Classical Approaches

In the work of Aparício (2015), they considered five text-based summarization approaches: LexRank (Erkan & Radev, 2004) and Support Sets (Ribeiro & de Matos, 2011), which are centrality-based, MMR (Carbonell & Goldstein, 1998), and GRASSHOPPER (X. Zhu, Goldberg, Van Gael, & Andrzejewski, 2007), which are diversity-based, and LSA (Y. Gong & Liu, 2001), which is a mathematical technique based on Singular Value Decomposition (SVD). Centrality-based algorithms consider that the most important content of an input is the most central, considering its representation as a graph, spatial, etc. On the other hand, Diversity-based algorithms focus on maintaining diversity in the summary.

Aparício (2015) summarized the movie's subtitles using the centrality-based LexRank (Erkan & Radev, 2004) algorithm to determine the film's most important content and avoid diversity (to guarantee coherence). The LexRank algorithm is a graph-based algorithm in which the text is converted into a graph representation, where sentences, represented by TF-IDF score vectors, are nodes, and edges between sentences are weighted based on their cosine similarity.

An adapted version of Google's PageRank algorithm for ranking web pages (Brin & Page, 1998) is applied to determine the centrality of each sentence. It measures the importance of a sentence based on its connections to other sentences, and its computation follows Equation 2.8.

$$S(V_i) = \frac{1-d}{N} + d \times \sum_{V_j \in adj[V_i]} \frac{Sim(V_i, V_j)}{\sum_{V_k \in adj[V_j]} Sim(V_j, V_k)} S(V_j) \tag{2.8}$$

where d is a damping factor that ensures the convergence of the method, N is the total number of vertexes, and $S(V_i)$ is the score of the ith vertex.

2.2.2.2 BERT for Extractive Text Summarization

Bidirectional Encoder Representations from Transformers (BERT) was introduced by Devlin et al. (2018) to improve language understanding by pre-training the bidirectional transformer architecture on a large text corpus.

When it was presented, BERT generated significant excitement within the Machine Learning community due to its ability to achieve state-of-the-art performance in various NLP tasks. These tasks include Question Answering (SQuAD v1.1), Natural Language Inference (MNLI), and several others.

The primary technical breakthrough of BERT involves utilizing bidirectional training from the Transformer (explained previously in section 2.2.1.2). Unlike previous approaches focused on sequential or combined left-to-right and right-to-left training, BERT trains the model bidirectionally. Devlin et al. (2018) findings also demonstrate that bidirectional training enables the model to better understand language context and coherence compared to unidirectional models. In BERT's case, as it aims to build a language model, only the encoder part of the Transformer is needed.

The input to BERT consists of a token sequence. These tokens are initially transformed into vector embeddings and subsequently processed within the neural network. The network's output is a sequence of vectors, each representing an input token with a corresponding index, and all vectors have a size of n. The main objective of implementing bidirectional training is to capture contextual relationships between words or sub-words in text. This way, to be able to leverage this quality, the authors defined two training strategies:

- **Masked-Language Modeling (MLM):** This pre-training objective is inspired by the *Cloze* task in Taylor (1953). A percentage of input tokens in a given text sequence are randomly replaced with a special [MASK] token. The objective is for the BERT model to predict the original value of these masked tokens based on the surrounding context. The BERT loss function considers only the masked values' prediction and ignores the non-masked words' prediction. As presented in Fig. 2.15, to obtain the prediction of the output words, the final hidden vectors of the Transformer Encoder are fed to a classification layer which outputs are transformed into the vocabulary dimension, computing the probability of each word in the vocabulary with *softmax*. In the case of BERT, during training, the data generator selects 15% of the token positions randomly. For each iteration, the token in each selected position has an 80% chance of being replaced with the special [MASK] token, a 10% chance of being replaced with a random token, and 10% of being unchanged. This is done to mitigate the mismatch between pre-training and fine-tuning since the [MASK] token does not appear during fine-tuning.

- **Next Sentence Prediction (NSP):** During BERT training, the model is presented with sentence pairs and trained to determine whether each pair's second sentence follows the original document's subsequent sentence (Fig. 2.16). In the training data, 50% of the inputs consist of sentence pairs where the second sentence is indeed the subsequent one in

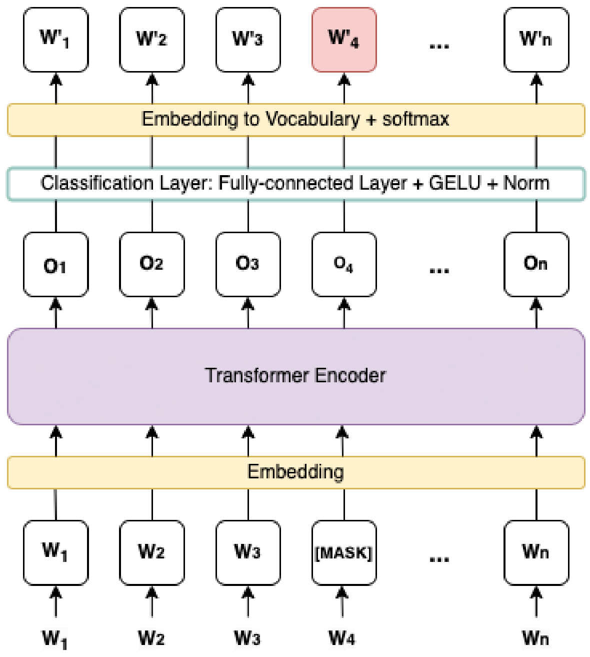

W'1
W'2
W'3
W'4
...
W'n
Embedding to Vocabulary + softmax
Classification Layer: Fully-connected Layer + GELU + Norm
O1
O2
O3
O4
On
Transformer Encoder
Embedding
W1
W2
W3
[MASK]
Wn
W4

Figure 2.15: MLM - Masked Language Modeling.

the document. Conversely, the remaining 50% of inputs involve a random sentence selected from the corpus as the second sentence. The intention is for the random sentence to be unrelated or disconnected from the first sentence, allowing the model to learn to distinguish between coherent and incoherent sentence pairs. This is done by inserting tokens indicating the beginning and end of each sentence and adding to each token a positional embedding and sentence embedding differentiating between sentence A and sentence B. In BERTs training, the entire input sequence goes through the Transformer model, and the probability of being the following sentence is computed with *softmax* (Jernite, Bowman, & Sontag, 2017; Logeswaran & Lee, 2018).

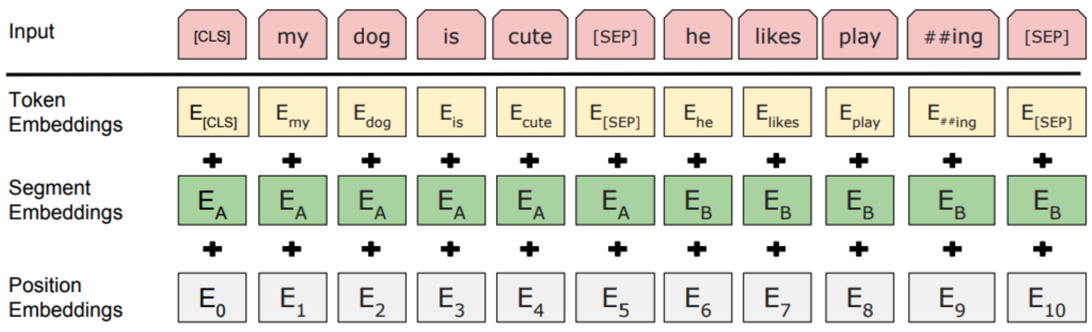

Input
[CLS]
my
dog
is
cute
[SEP]
he
likes
play
##ing
[SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$ E_{he} E_{likes} E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}

Figure 2.16: NSP - Next Sentence Prediction. Devlin et al. (2018)

During the training of the BERT model, both MLM and NSP strategies are trained simultaneously. The objective is to minimize the combined loss function that incorporates both training objectives.

The pre-training process in BERT aligns with the established practices in language model pre-training. The authors utilized the BooksCorpus (800M words) (Y. Zhu et al., 2015) and English Wikipedia (2,500M words) as the pre-training corpus. In the case of Wikipedia, only the text passages were extracted (disregarding lists, tables, and headers). To extract longer contiguous text sequences, utilizing a corpus at the document level rather than a shuffled corpus at the sentence level, like the Billion Word Benchmark (Chelba et al., 2014), is crucial.

The pre-trained BERT model can be used to obtain sentence embeddings to be applied in many different ways. Miller (2019) leverage BERT for Extractive Text Summarization in Lectures. By visual examinations of clusters, the authors determined that the second to last averaged layer produced the best embeddings for representations of words for their use-case. One hypothesis for this, which the authors also mention, is that the *final layer was biased by the classification tasks in the original training of the model*.

After extracting the embeddings for each sentence of the complete text corpus, the authors selected the K-Means algorithm for clustering the embeddings. They defined, as k, the final desired number of sentences in the produced summary. The sentences closest to the clusters' centroids were selected for the final summary.

The K-Means algorithm gained significant popularity in MacQueen (1967), where the author expanded on it, formalizing its principles and demonstrating its effectiveness for clustering analysis. The algorithm is an unsupervised machine-learning technique for partitioning a dataset into distinct groups or clusters. The algorithm aims to group similar data points based on their embeddings similarity. It starts by randomly initializing k cluster centroids in the embeddings space and assigning each data point to the nearest centroid based on their distance (usually Euclidean distance). From here, in each iteration, a new centroid position for each cluster is computed by taking the mean of all data points assigned to it and reassigning all of them to the new centroids' positions until the centroids stabilize and there is minimal change in the data point assignments. This procedure results in obtaining k clusters, with each data point belonging to the cluster defined by its nearest centroid.

2.2.3 Music - Video Retrival

2.2.3.1 Emotion Representations

Numerous studies have demonstrated the impact of multimedia, such as music and videos, on human emotions (Cowen & Keltner, 2017; Juslin & Laukka, 2004; Jaquet, Danuser, & Gomez, 2014). The recent work of Thao, Roig, and Herremans (2023) shows the vast majority of ways emotions can be represented. They usually fall into two categories: using discrete categories or continuous dimensions.

The categorical approach represents emotions by describing them in terms such as happy, fearful, sad, and others. This approach encompasses various emotional categories to capture the different facets of human emotions. The drawback of this representation is that there is no commonly defined taxonomy. For instance, in the Geneva Emotional Music Scales (GEMS) model (Zentner, Grandjean, & Scherer, 2008), there are nine different categories with a total number of up to 45 emotion terms (in GEMS-45), but Ekman, Sorenson, and Friesen (1969) ”only” have six basic emotions, and Cowen and Keltner (2017) have 26 categories.

In the continuous approach, emotions are represented by mapping them into a dimensional space (Baveye, Dellandrea, Chamaret, & Chen, 2015; Zlatintsi et al., 2017; Bradley, Greenwald, Petry, & Lang, 1992; Watson & Tellegen, 1985; Watson, Wiese, Vaidya, & Tellegen, 1999). This approach offers the advantage of effectively modeling the diversity and complexity inherent in human emotions. However, this representation has difficulty in accurately distinguishing and representing certain emotions, such as nostalgia, within the framework of continuous dimensions (Van Tilburg, Wildschut, & Sedikides, 2018; B. Li & Kumar, 2019).

2.2.3.2 Available Datasets

There are limited datasets for *affective* audio-visual correspondence learning, and they are derived from combinations of existing ones.

Some other studies involving action recognition assignments or audio signal categorization produced the datasets used in most audio-visual correspondence learning tasks (Gemmeke et al., 2017; Aytar, Vondrick, & Torralba, 2016; Kay et al., 2017; Chung, Senior, Vinyals, & Zisserman, 2017; Parkhi, Vedaldi, & Zisserman, 2015; Nagrani, Chung, & Zisserman, 2017). The

IMEMNet (S. Zhao et al., 2020) and IMAC (Verma, Dhekane, & Guha, 2019) datasets are created from music and images. The two datasets in B. Li and Kumar (2019) are both dedicated to affective correspondence learning between music and video. However, they are not released. The authors constructed music-video pairs involving a crowd-sourcing approach, where annotators provided feedback on the prevalence of emotions in both streams. The dataset comprises 3,000 music-video pairs, evenly divided into matched and mismatched pairs, covering 140 emotions.

The video streams for the first dataset were gathered from *Cowen's* dataset (Cowen & Keltner, 2017), and the music segments were chosen at random from the Unbalanced Train set of the *Music Mood* dataset, which is part of the *AudioSet* ontology (Gemmeke et al., 2017).

For the second dataset, the music was collected from Spotify and the videos from Instagram and from *Moments in Time* dataset (Monfort et al., 2019).

2.2.3.3 EmoMV - Datasets and Proposed Model

The limitation on available datasets with affective audio-visual correspondence learning benchmarks motivated Thao et al. (2023) to create a collection of three datasets (EmoMV) for affective correspondence learning between music and video modalities. Furthermore, alongside creating three novel datasets, a benchmark deep neural network model is introduced for binary classification of affective music-video correspondence. Subsequently, this model undergoes modifications to accommodate affective music-video retrieval.

EmoMV Datasets

The authors utilize emotion categories to construct matched and mismatched music-video pairs, as discrete representations of emotions are generally more comprehensible for non-experts. Three datasets were produced:

- **EmoMV-A Dataset:** This dataset made use of the 30 seconds music video segments from the MVED dataset (Pandeya, Bhattarai, & Lee, 2021) without the ones corresponding to the ”neutral” category. Table 2.1 presents the quantity of the collected music video segments associated with each emotion label. Each music video segment has an emotional label based on visual and audio clues. With a vote of confidence in the annotation process for the MVED dataset, the authors considered the music and video streams from each music video segment taken from the MVED dataset as matched in terms of emotions. Taking this into consideration, the final result of this dataset is $2,208/1,902$ matched/mismatched music/video pairs in the training set, $310/246$ matched/mismatched pairs in the validation set, and $124/124$ matched/mismatched pairs in the test set. The mismatched pairs correspond to eight pairs of (mismatched) emotion labels: exciting – fearful, exciting – tense, exciting – sad, exciting - relaxing, fearful - sad, fearful – relaxing, tense – sad, and tense – relaxing. As stated by Pandeya et al. (2021), the MVED dataset includes music video segments annotated with fearful or tense labels but exhibits distinct visual elements while sharing similar audio components such as high pitch and rhythmic variation. Similarly, segments labeled with sad or relaxing emotions share common audio characteristics like slow tempo and soft music. Due to this overlap, the authors excluded fearful-tense and sad-relaxing emotion pairs.

Table 2.1: Number of music video segments corresponding to each emotion label in the reduced MVED dataset.

Reduced MVED dataset	**Exciting**	**Fearful**	**Tense**	**Sad**	**Relaxing**	**Total**
Train set	843	828	652	730	1,057	**4,110**
Validation set	102	111	84	111	148	**556**
Test set	50	50	50	50	50	**250**

- **EmoMV-B Dataset:** This dataset used the Music Mood dataset (Olah, 2015) of the AudioSet ontology. This dataset consists of music segments derived from music clips on YouTube and, because of this reason, to create this dataset, the authors started by manually validating every single music video segment in the Music Mood dataset, filtering out the segments that only had footage of video games or some unrelated images in the video stream, as well as the ones corresponding to low quality of the music stream or that mainly contain speech. Adding the unavailability of many given YouTube links, the authors ended this process with 4,487 music videos. In the original Music Mood dataset, only the music stream was considered for the emotion annotation. Therefore the authors trained a model to tackle the emotion classification task on the MVED dataset, whereby music video segments are annotated with one of six emotions (including exciting, fearful, tense, sad, relaxing, and neutral). This was done by modifying the Feature AttendAffectNet model (Thao, Balamurali, Roig, & Herremans, 2021) by changing its last fully-connected layer from one neuron to six neurons, followed by a softmax layer. This model was introduced to predict arousal/valence values. Following the authors of the original model, the authors of EmoMV used the ResNet-50 (He, Zhang, Ren, & Sun, 2016) (pre-trained on the ImageNet dataset (J. Deng et al., 2009)), FlowNetS (Dosovitskiy et al., 2015) (pre-trained on the Flying Chairs dataset (Dosovitskiy et al., 2015)) and RGB-stream I3D networks (Carreira & Zisserman, 2017) (pre-trained on the Kinetics dataset (Kay et al., 2017)) as feature extractors to obtain the visual features from the video stream. As for the audio features, the VGGish network (Hershey et al., 2017) pre-trained on the AudioSet dataset (Gemmeke et al., 2017) was used. The authors reported that this model reaches the highest classification accuracy of 86.67% in the test set of the MVED dataset when using both visual and audio features. The reached F1-score (Goutte & Gaussier, 2005) was 0.866 and Area Under the ROC Curve (AUC) (Brown & Davis, 2006) was 0.982. After training the model, the authors obtained emotion labels for music video segments in the filtered Music Mood dataset. They only considered music video segments for which the same emotion label was predicted for their video stream, music stream, and both, ending up with 832 labeled music video segments. Discarding the ”neutral” label and running the same process implemented for EmoMV-A to get match/mismatch labels music/video pairs, the EmoMV-B dataset ended up consisting of 496 music video segments for the train set and 120 for the validation set, both balanced in the number of matched and mismatched pairs.

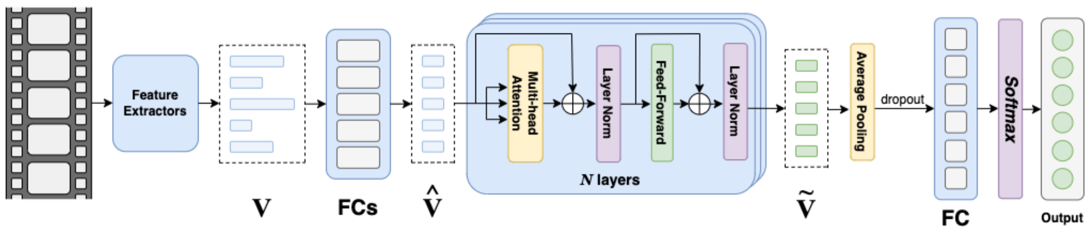


Figure 2.17: The Modified Feature AttendAffectNet employs dimension reduction by passing the feature vectors V through fully connected layers with eight neurons each, obtaining a set of dimension-reduced feature vectors $\hat{V}$. These vectors then go through N identical layers, each containing a multi-head self-attention mechanism and a feed-forward layer. The resulting stack outputs encoded feature vectors $\tilde{V}$. These vectors are subsequently processed through an average pooling layer, dropout, a fully connected layer with six neurons, and a *softmax* layer to predict the probability of each emotion category.

- **EmoMV-C Dataset:** This dataset comprises self-collected music videos of songs featured in movies (soundtracks) using Google and YouTube. Many of them contain movie scenes. The collected ”raw” data were split into $2,688$ music video segments for 30 seconds each. After this step, to get the emotion labels of all streams and the consequent match/ labels music/video pairs, the same process described for EmoMV-B was applied (Fig. 2.17). This resulted in a collection of 360 music video segments for the train set and 96 for the validation set, both balanced in the number of matched and mismatched pairs.

A summary of the three datasets in the EmoMV collection is presented in Table 2.2.

Table 2.2: The EmoMV Dataset Collection.

Dataset	Music Videos	Train Set	Validation Set	Test Set
EmoMV-A	Matched	2208	310	124
	Mismatched	1902	246	124
EmoMV-B	Matched	248	60	-
	Mismatched	248	60	-
EmoMV-C	Matched	180	48	-
	Mismatched	180	48	-

EmoMV Model for Affective Music-video Retrieval

In addition to the dataset Collection, the EmoMV authors also proposed two models for distinct tasks: binary affective music-video correspondence classification and affective music-video retrieval. In the scope of this work, we are presenting the affective music-video retrieval model, although they are both very similar.

The proposed model utilizes pre-trained deep neural networks, initially designed for action recognition and audio classification, to extract visual and audio features from video and music streams. To achieve a shared representation, video and music projection heads are employed to embed the visual and audio features into a common representation space where the distance between the visual and audio embeddings is computed. Following the multi-task learning approach, the authors appended the music and video branches (for music and video emotion classification) to the music and video subnetworks, respectively. A highlighted representation of the proposed model is presented in Fig. 2.18.

Video subnetwork

To extract the embeddings of the video stream, the authors of Thao et al. (2023) made use

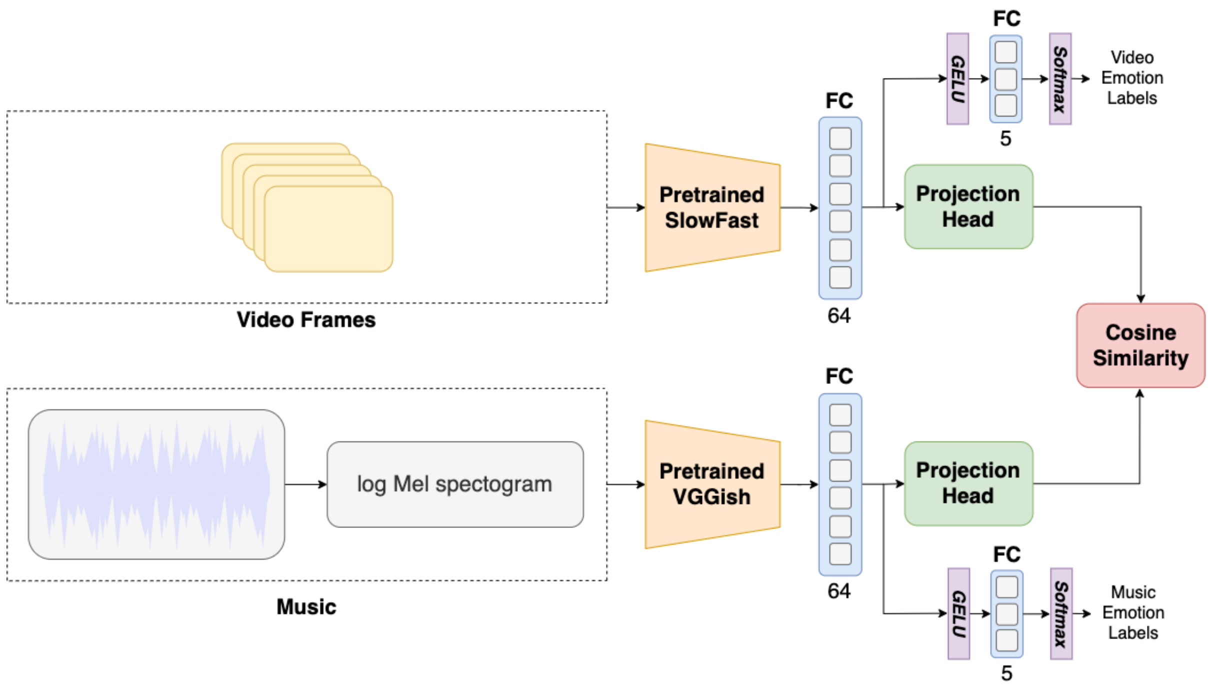


Figure 2.18: EmoMV Model for Affective Music-video Retrieval.

of the SlowFast (Feichtenhofer, Fan, Malik, & He, 2019) network pre-trained on the Kinetics human action video dataset (Kay et al., 2017).

The SlowFast (Feichtenhofer et al., 2019) network is a popular video classification model that leverages spatial and temporal information for better performance. It is typically pre-trained on large-scale video datasets like Kinetics, which contains various human action videos.

The SlowFast network consists of two pathways: a slow pathway and a fast pathway. The slow pathway captures the spatial information and processes the frames at a slower rate. In comparison, the fast pathway captures the temporal information and processes the frames faster. This combination allows the model to effectively capture fine-grained details and motion dynamics in videos.

During pretraining on the Kinetics human action video dataset (Kay et al., 2017), the Slow-Fast network is trained to predict the correct action labels for many video clips. This process enables the model to learn meaningful video representations that can generalize to other video-related tasks.

After pretraining, the SlowFast network can be used as a feature extractor for video streams by removing its last classification layer. Given a video, the network processes it frame by frame,

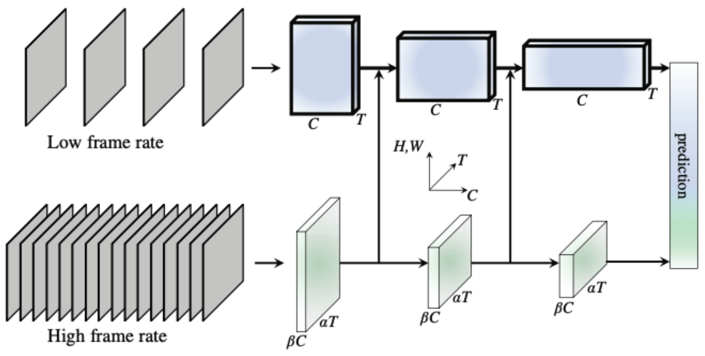


Figure 2.19: A **SlowFast network** has a low frame rate, low temporal resolution *Slow* pathway and a high frame rate, $\alpha\times$ higher temporal resolution *Fast* pathway. The *Fast* pathway is lightweight by using a fraction (β, e.g., $1/8$) of channels. Lateral connections fuse them.

and the output embeddings capture high-level semantic information about the video content.

As a result, the authors of Thao et al. (2023) end up with a $2,304$ dimensional feature vector from each video stream, which goes through a fully-connected layer with 64 neurons for dimensionality reduction and passes to the video projection head.

Music subnetwork

To extract the embeddings of the music stream, the authors of Thao et al. (2023) made use of the VGGish (Hershey et al., 2017) network pre-trained on the AudioSet ontology (Gemmeke et al., 2017) to extract a 128-dimensional feature vector from the log-mel spectrogram computed with each 0.98-second music segment (which is at a sampling rate of 16 kHz with signed 16-bit PCM encoding and a mono channel). The resulting log-mel spectrogram is a two-dimensional matrix where the x-axis represents time (corresponding to the audio frames), and the y-axis represents frequency (corresponding to the Mel scale). Each matrix element represents the logarithmically compressed energy in a specific frequency bin at a particular time frame. The Mel scale is a perceptual scale of pitches or frequencies that approximates the human auditory system's frequency perception. It was developed to reflect better how humans perceive and distinguish different sound frequencies.

The feature vectors extracted from all 0.98-second (*hop size*) music segments were averaged element-wise to obtain a condensed representation of the music stream within each segment within each music segment. This averaging process resulted in a 128-dimensional vector that captures the essence of the music in the segment. Similar to the video subnetwork, this 128-dimensional vector is then passed through a fully-connected layer with 64 neurons to reduce its dimensionality further and passed to the music projection head.

The AudioSet ontology (Gemmeke et al., 2017) is a large-scale dataset that contains a wide range of audio clips, each labeled with a specific sound event. The VGGish (Hershey et al., 2017) network is trained on this dataset to learn discriminative audio representations.

The architecture of the VGGish network is inspired by the VGG network architecture commonly used for image recognition tasks. However, it is adapted for audio analysis. The network consists of several convolutional layers followed by fully connected layers.

During pretraining, the VGGish network learns to extract low-level audio features, such as spectrograms, from raw audio waveforms. It captures both temporal and spectral information from audio signals, enabling it to represent audio content meaningfully.

Emotion classification branches

To enable multi-task learning, Thao et al. (2023) extend the video and music subnetworks by adding video and music branches. As shown in Fig. 2.17, the dimensionally reduced visual and audio feature vectors, obtained by the fully-connected layers of 64 neurons in their respective subnetworks, are directed to the newly introduced video and music emotion classification branches.

Both the video and music emotion classification branches share a common structure. They consist of a Gaussian Error Linear Unit (GELU) (Hendrycks & Gimpel, 2016) activation function, followed by a fully-connected layer comprising five neurons (corresponding to the number of emotion categories). A *softmax* layer is then applied to obtain the emotion classification output for each modality.

These emotion classification branches are jointly trained with the main branch during training. The training process involves using three cross-entropy loss functions, where each loss function carries equal weight.

By incorporating these multi-task branches and jointly training them, the authors aimed to leverage the shared information between video and music modalities, enhancing the model's ability to classify emotions accurately.

Projection Heads

Inspired by the projection heads originally applied to the textual and visual features in the

Contrastive Language-Image Pretraining (CLIP) model (Radford et al., 2021), Thao et al. (2023) also applied this technique for music and visual features instead, with the objective of embed the visual and audio features into a common representation space.

Both video and music projection heads follow the same structure: fully-connected layers of 64 neurons each, the GELU (Hendrycks & Gimpel, 2016), a dropout ratio of 0.5, a residual connection, and L2-normalization, as described in Fig. 2.20.

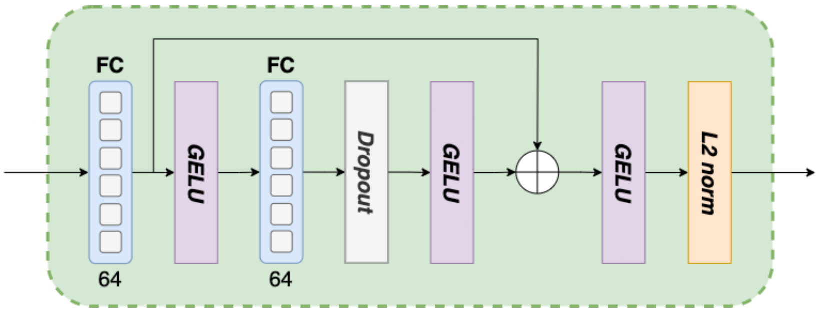


Figure 2.20: Structure of the Projection Heads used to project visual and audio features into a common representation space.

Cosine Similarity

Similar to the works of S. Zhao et al. (2020); B. Li and Kumar (2019); Wang, Yang, Jhuo, Lin, and Wang (2012), to perform the affective music-video retrieval task, the main branch of the model computes the cosine distance $d_{\cos}(f_v, f_m)$ between visual (f_v) and audio embeddings (f_m) as follows:

$$d_{\cos}(f_v, f_m) = 1 - S_{\cos}(f_v, f_m) \tag{2.9}$$

Where $S_{\cos}(f_v, f_m)$ is the cosine similarity between the visual and audio embeddings:

$$S_{\cos}(f_v, f_m) = \frac{f_v . f_m}{\|f_v\| \times \|f_m\|} \tag{2.10}$$

Where $\|f_v\|$ and $\|f_m\|$ are the Euclidean norm of the vectors f_v and f_m, respectively.

Experimental Setup

In this study, the Adam optimizer is used in the training phase, whereby the maximum number of epochs is $1,000$. The batch size was set to 256, and the learning rate to 0.0001. The early stopping is applied with the patience parameter of 20. The experiments were conducted using Python 3.6 on an NVIDIA GTX 1070 GPU. This setup is applied for both our proposed model and the baseline.

As mentioned previously, the three networks are trained simultaneously using three equal weighted loss functions, including two cross-entropy loss functions (Ackley, Hinton, & Sejnowski, 1985), as shown in Eq. 2.11, for the video and music subnetworks, and the contrastive loss (Hadsell, Chopra, & LeCun, 2006) on the cosine distance between the visual and audio embeddings.

$$L_{CE} = -\sum_{i=1}^{n} t_i \log(p_i), \textit{for n classes}, \tag{2.11}$$

where t_i is the truth label and p_i is the Softmax probability for the i^{th} class.

Contrastive loss functions compare pairs of samples and compute a loss based on their similarity. It aims to minimize the distance or maximize the similarity between their embeddings if they are similar and, contrarily, maximize the distance or minimize the similarity if they are dissimilar. This way, the loss encourages similar samples to have embeddings that are close together and dissimilar samples to have embeddings that are far apart. The intuition behind contrastive loss is to maximize the similarity of similar vectors, aiming for a value close to 1. This is because a similarity value of 1 results in an optimal loss of 0 ($-\log(1) = 0$). On the other hand, we want the similarity value to be close to 0 for different vectors. Any non-zero similarity values would reduce the loss for similar vectors, which is undesirable. Therefore, by pushing different vectors towards a similarity value of 0, we can effectively minimize their impact on the loss for similar vectors.

During the inference process, when a music query is provided, the model computes the cosine similarity score between the audio embedding of the queried music and the visual embeddings of all video segments in the provided database. This similarity score quantifies the similarity between the audio and visual features. The video segments are ranked by leveraging these similarity scores, and the top-ranked results are identified as the best matches to the music query. This ranking process enables us to find the video segments that align closely with the audio characteristics of the music query.

Results

To evaluate the performance of the proposed model on the affective music-video retrieval task, the authors compute the **Mean Average Precision (mAP) score** as used by Teufel (2007), and the **top-K retrieval accuracy**:

- **Mean Average Precision (mAP):** When calculating this metric, a retrieved result is considered relevant to the query if it shares the same label. Conversely, if the labels differ, the result is considered irrelevant. This implies that for each video query, multiple music segments can be considered relevant (according to the ground truth) and vice versa. The mAP score considers these relevance relationships to assess the overall performance of the retrieval system. According to the information retrieval theory (Teufel, 2007), the mAP corresponds to the mean of the Average Precision (AP) of all queries. This is presented in Eq. 2.12, where N is the number of queries and AP_i is the Average Precision for query i, which equation is presented in Eq. 2.13. R_i is the number of relevant documents for query i. $Precision_i(rel = j)$ (as presented in Eq. 2.14) is the precision at the j-th document that is rerelevant to query i.

$$mAP = \frac{1}{N} \sum Ni = 1AP_i, \tag{2.12}$$

$$AP_i = \frac{1}{R_i} \sum_{j=1}^{R_i} Precision_i(rel = j), \tag{2.13}$$

$$Precision_i(rel = j) = r_i(j)/j, \tag{2.14}$$

- **Top-K retrieval accuracy:** In music-video retrieval, this metric is defined as the percentage of music queries for which at least one relevant video segment is retrieved among the top K results. This metric quantifies the effectiveness of the retrieval system in returning relevant music segments for a given set of video queries.

The proposed model achieved the results presented in Table 2.3 for the aforementioned metrics.

Table 2.3: Given a music query, retrieve videos: Accuracy and the mAP on the EmoMV dataset collection.

Dataset	**Top-1 (%)**	**Top-3 (%)**	**Top-5 (%)**	**mAP (%)**
EmoMV-A	56.00	83.60	90.80	52.31
EmoMV-B	44.17	77.50	86.67	46.83
EmoMV-C	46.88	75.00	86.46	41.98

2.3 Summary

We verify that many recent advances in the Deep Learning field related to video summarization come from NLP architectures. This happens because of the similarities in the structure of both data types.

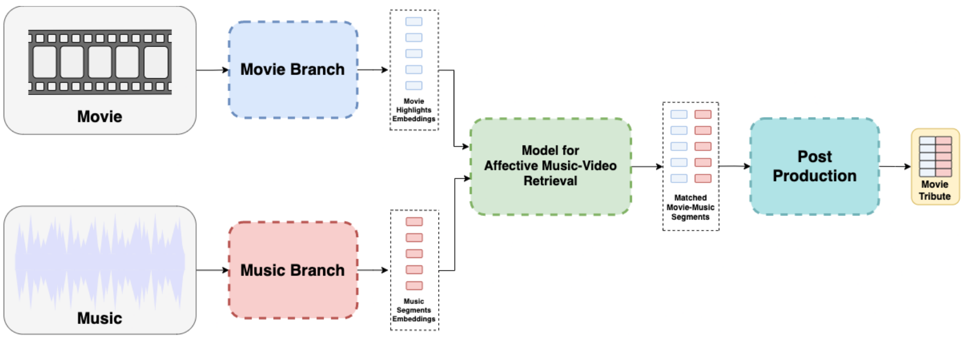
We chose to use BERT for extractive text summarization of subtitles and the EmoMV model for Affective music-video retrieval to generate a movie tribute automatically.

3 Generation Of A Movie Tribute

Do, or do not. There is no "try".

– Yoda, "Star Wars: Episode V - The Empire Strikes Back"

This chapter presents our approach to the generation of a movie tribute. An overview of the proposed pipeline is presented in Fig. 3.1.



Movie
Movie Branch
Movie Highlights Embeddings
Music
Music Branch
Music Segments Embeddings
Model for Affective Music-Video Retrieval
Matched Movie-Music Segments
Post Production
Movie Tribute

Figure 3.1: Proposed Architecture Overview.

In the Music Branch, we segment the music based on its tempo and extract audio features from each segment using the VGGish (Hershey et al., 2017) network pre-trained on the AudioSet ontology (Gemmeke et al., 2017).

For the Movie Branch, we run extractive summarization using BERT on the subtitles corpus to select highlights corresponding to movie scenes with subtitles. We also implemented K-means clustering to select movie highlights without subtitles. After selecting the movie highlights, the embeddings for each one are extracted using the SlowFast (Feichtenhofer et al., 2019) network pre-trained on the Kinetics human action video dataset (Kay et al., 2017).

We selected the EmoMV model presented before (section 2.2.3.3) to compute the similarity matrix between the selected movie highlights and the music segments. We also add a weight related to the movie scenes' temporal order before matching each music segment to the corresponding highlight.

The Post-Production corresponds to loudness adjustment for each soundtrack (music and collected movie segments) and an offset that is added to all the breaking points' values in case there is a need to adjust the movie scenes timestamps (in the majority of the cases, this is not needed).

On top of the model, we also created a simple UI to monitor the whole process and intuitively adjust the parameters to generate the movie tribute.

3.1 Music Branch

3.1.1 Music Breakingpoints

To load, manipulate and segment the sound data, we used the Librosa python package (McFee et al., 2015) alongside Ruptures (C. Truong, Oudre, & Vayatis, 2020) to provide us with search algorithms to determine the breaking points.

We follow the approach proposed in the Ruptures Documentation to select the music breaking points. This task is seen as a change point detection, consisting of identifying temporal boundaries that delineate meaningful Sections in a music stream.

We start by loading the desired music with Librosa and compute its tempogram. A tempogram is a representation of tempo information in a musical signal.

It is derived from the onset strength envelope, which measures the intensity of musical events or notes onsets over time. In this case, the onset strength envelope is computed with the spectral flux onset envelope method (Böck & Widmer, 2013). The spectral flux onset strength envelope is computed by transforming the audio signal into the frequency domain using techniques like the Short-Time Fourier Transform (STFT). This yields the magnitude spectrum for each time frame. The spectral flux is then determined by comparing the magnitude spectra of consecutive frames and measuring the magnitude difference between them. By analyzing the rate of change in the spectral content frame by frame, the spectral flux onset strength envelope captures energy fluctuations and variations in the audio signal, often indicating note onsets or other significant musical events, being useful for beat tracking, onset detection, and rhythm analysis in music signal processing.

The tempogram analyzes the patterns and periodicities in the onset strength envelope to estimate the underlying time-varying representation of tempo (measured in Beats Per Minute, BPM), allowing for the detection of tempo changes and variations throughout a musical piece. The tempo represents the fundamental rhythmic structure and provides a sense of timing and groove in music (Grosche, Müller, & Kurth, 2010).

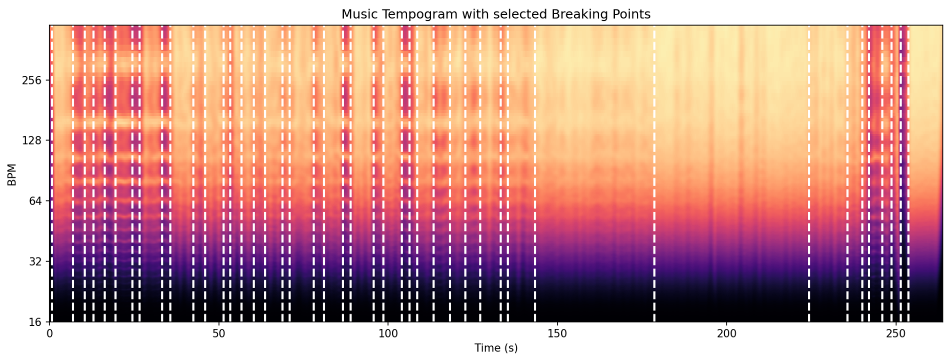


Figure 3.2: Computed tempogram and respective breaking points for the music ”Chevaliers De Sangreal” by Hans Zimmer.

To get the music breaking points, we detect the changes in the mean of the tempogram (a multivariate signal) using the Kernel change point detection (Celisse, Marot, Pierre-Jean, & Rigaill, 2018; Arlot, Celisse, & Harchaoui, 2019). A kernel function is chosen to capture the underlying structure or patterns in the data. The kernel function measures the similarity or dissimilarity between two data points based on their characteristics or features. We selected the CostL2 cost function, as suggested by Rupture’s authors.

The first user input we request is the *desired average music segment time*. We define the number of breaking points as presented in Eq. 3.1 with the provided parameter. This computed value is fed to Ruptures Kernel Change Point Detection, which outputs the values of the computed music breaking points. Fig. 3.2 presents an example music tempogram with the corresponding selected breaking points.

$$\textit{number of breakingpoints} = \frac{\textit{total music time}}{\textit{desired average music segment time}} \tag{3.1}$$

The second requested user input is the *desired minimum music segment time*. After computing the breaking points, we append consecutive music segments (by removing the breaking point that separates them), guaranteeing that all music segments have a duration higher than the provided user parameter.

3.1.2 Embedding extraction of the Segmented Music Clips

We are implementing the proposed EmoMV Model for Affective Music-video Retrieval, this way we extract the music embeddings using the VGGish (Hershey et al., 2017) network pre-trained on the AudioSet ontology (Gemmeke et al., 2017) in a similar manner as explained in Section 2.2.3.3. The main difference is that we let the user define the *hop size* (3^{rd} user input) instead of having it fixed at 0.98 seconds. For each music segment, a 128-dimensional feature vector is computed for each sub-segment defined by the *hop size*, which is averaged element-wise.

We obtain a 128 dimensional feature vector for each music segment from this process.

3.2 Movie Branch

3.2.1 Movie Scenes Breakingpoints

The first processing step of the movie is to compute the scene transitions (breaking points). To do this, we make use of the PySceneDetect (Castellano, 2023) python package. PySceneDetect is a Python package that provides a simple and efficient way to analyze videos and detect scene changes. It works by analyzing video frames and identifying significant changes between consecutive frames. These changes can include abrupt changes in brightness, color, motion, or other visual characteristics. PySceneDetect allows easy video segmentation into separate scenes or shots by detecting these scene changes.

We used the *detect()* function available in the PySceneDetect package with the *ContentDetector* detection algorithm (also available in the package) to extract the movie scenes breaking points. The *ContentDetector* compares the difference in content by considering pixel changes in the HSV colorspace (Smith, 1978) between adjacent frames against a set threshold/score, which, if exceeded, triggers a scene cut.

The components of the HSV colorspace are presented below:

- **Hue (H):** represents the pure color information without considering brightness or saturation (determines the type of color, such as red, green, or blue). It is measured in degrees on a circular scale, typically ranging from 0 to 360 degrees;

- **Saturation (S):** refers to the intensity or purity of a color (the amount of gray present in a color). It is expressed as a percentage;

- **Value/Brightness (V):** represents a color′s perceived lightness or darkness. A value of 0% corresponds to black, while 100% represents the brightest color.

After processing the movie, we get a list of all breaking points corresponding to the movie scenes′ transitions.

3.2.2 Highlights Selection

With the movie Scenes breaking points defined, we determine which scene contains dialogue and which does not by leveraging the information in the Movie′s *srt* (SubRip Text) file. In the process, we also group the subtitles according to the corresponding scene.

The (*srt*) file is a plain-text subtitle file format commonly used for displaying subtitles or captions in videos. This file contains information about the dialog in the Movie and the relative timestamps corresponding to the portion of the Movie where the respective dialogue occurs. We use the *pysrt* python package as the interface for this file type.

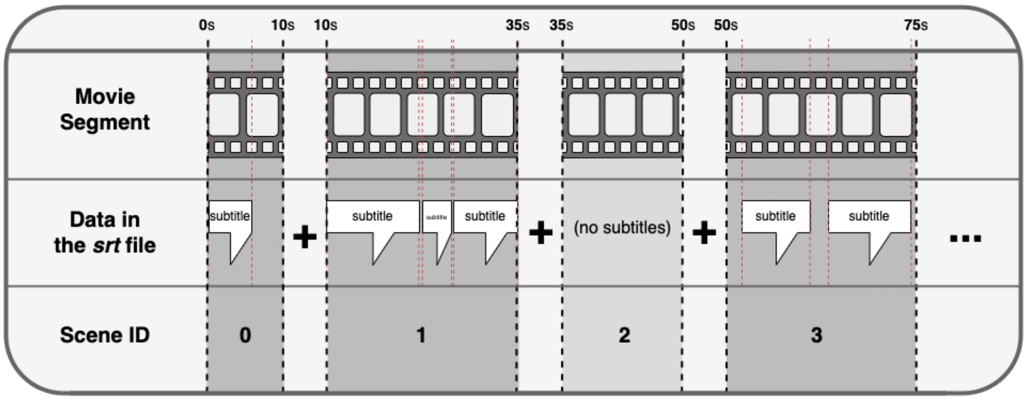


Figure 3.3: Process to categorize movie scenes as Subtitle or No-subtitled.

After separating the two different types of scenes, we select each group′s highlights by implementing two different methods.

3.2.2.1 Subtitled Scenes Selection

To define the highlights regarding the subtitled scenes, we make use of the *bert-extractive-summarizer* python package, which is a generalization of the lecture-summarizer repo from (Miller, 2019), which leverages BERTs embeddings to perform extractive summarization, as explained previously in Section 2.2.2.2.

To further be processed by the extractive summarization model, for each scene, all the corresponding subtitles are appended together into a single Python string and processed as follows:

- Remove extra white spaces;
- Convert all letters to lowercase;
- Remove all punctuations;
- Add a "." at the end of the string (composed of all subtitles of the same movie scene).

We need to define a summarization ratio to run the BERT model for extractive summarization. This ratio is computed as follows:

$$\begin{gathered} \textit{summarization ratio} = \frac{\textit{number of music segments}}{number of subtitled scenes} \times \alpha, \\ 0 < \textit{summarization ratio} <= 1 \end{gathered} \tag{3.2}$$

α is a constant directly multiplied by the default summarization ratio ($\alpha = 1$) defined as a 4^{th} user input. This allows the user to force select more subtitled scenes than the default.

3.2.2.2 No-Subtitled Scenes Selection

To select the highlights from no subtitled scenes, we first define the number of non-subtitled highlights to select as presented in Eq. 3.3.

$$\textit{number of no subtitled scenes} = \text{int}\left(\frac{\textit{number subtitled highlights}}{\frac{1}{\beta} - 1}\right), \qquad 0 <= \beta <= 1 \tag{3.3}$$

β being the *ratio of no subtitled highlights to consider*, the 5^{th} user input. Our approach to selecting the highlights in the no-subtitled scenes group is much more time-consuming than the extractive summarization method. For this reason, we allow the user to decide the weight that the movie scenes without subtitles may have in the tribute generation, which also may vary on the type of movie and music being used. We extract video embeddings (using the SlowFast (Feichtenhofer et al., 2019) network pre-trained on the Kinetics human action video dataset (Kay et al., 2017)) from randomly selected $1.2 \times \textit{number of no subtitled scenes}$ no subtitled movie scenes. We run the K-Means algorithm for clustering the obtained embeddings (in a similar way as it's done with the BERT embeddings for extractive summarization - Section 2.2.2.2) and select the scenes that are closer to each cluster centroid ($K = \textit{total number of no subtitled scenes to select}$).

3.2.3 Embedding extraction of the Selected Movie Clips

We are implementing the proposed EmoMV Model for Affective Music-video Retrieval, this way we extract the movie highlights embeddings using the SlowFast (Feichtenhofer et al., 2019) network pre-trained on the Kinetics human action video dataset (Kay et al., 2017) (explained in greater detail in Section 2.2.3.3).

We obtain a $2,304$ dimensional feature vector for each movie highlight from this process.

It is essential to notice that if the highlights contain scenes without subtitles, their embeddings were obtained beforehand, therefor we avoid computing them again during this step.

In our study, the training phase was done in a local machine using *Python 3.8.12* on a *2.3 GHz Quad-Core Intel Core i5 processor*. All the training parameters are the same as in the original work and as follows:

The **Optimizer** algorithm adjusts a model's parameters during training. The **Maximum number of epochs** is the maximum number of complete iterations through the training dataset.

Table 3.1: Model's parameters used in training.

Parameter	Value
Optimizer	Adam (Kingma & Ba, 2014)
Maximum number of epochs	1,000
Batch size	256
Learning Rate	0.0001
Early stopping patience	20

The **Batch size** is the number of feature vectors loaded per batch. The **Learning Rate** is the magnitude of the parameter update at each iteration. The **Early stopping patience** is how long to wait after the last time validation loss improved.

The model stopped training at **387 epochs during 86.07 seconds** of running.

The **mAP score** and the **top-K retrieval accuracy** metrics obtained are presented in Table 3.2 (detailed in Section 2.2.3.3).

Table 3.2: Model's metrics obtained in the EmoMV-C dataset.

Top-1(%)	Top-3(%)	Top-5(%)	mAP(%)
48.95	80.20	88.54	41.34

In inference, we feed the model one music segment at a time as the query to retrieve similarity scores from the set of embeddings from each selected movie highlight (Fig. 3.4).

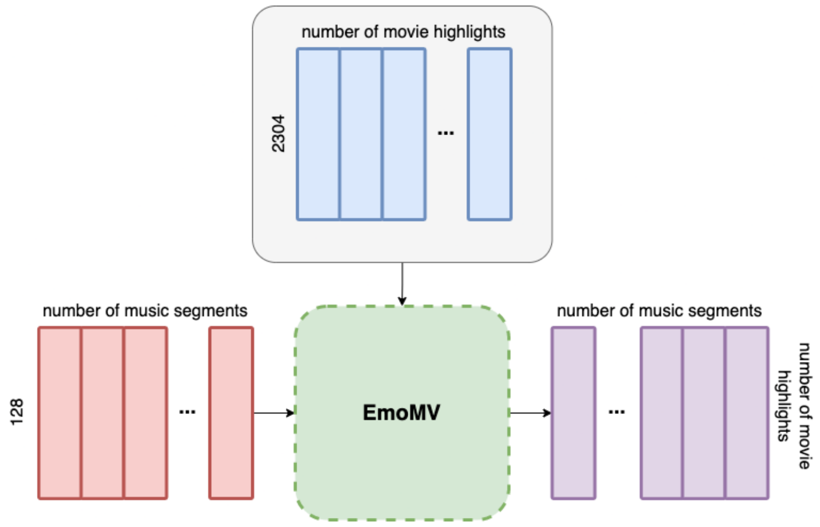


Figure 3.4: Dimention transformations between the audio and visual embeddings and the model's output.

3.2.4 Model Outputs Processing

The model outputs a similarity matrix (Fig. 3.5) representing the emotional coherence between the movie highlights and all the music segments.

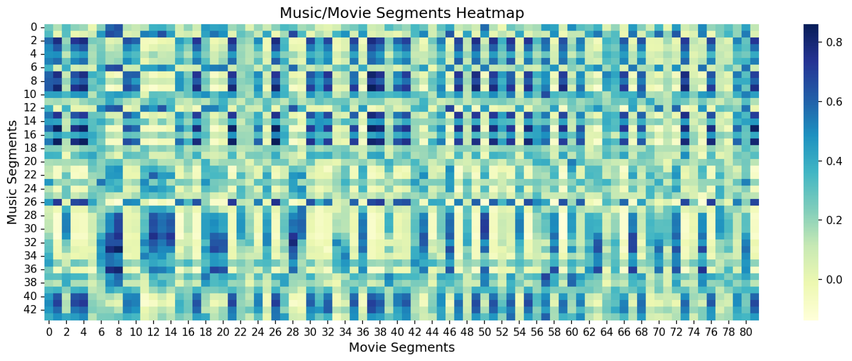


Figure 3.5: Example of a heatmap representation of the similarity matrix produced by the EmoMV model pre-trained on the EmoMV-C dataset.

A matrix of the same dimensions is computed regarding the chronological order of the movie. Since we guarantee that the scenes are ordered (relative to their indexes), each cell X of the temporal weight matrix is computed as presented in Eq. 3.4.

$$\begin{aligned} x_{ij} &= \left|\frac{i}{\textit{total number of music segments}} - \frac{j}{\textit{total number of highlights}}\right|, \\ X_{ij} &= 2 \times (0.5 - x_{ij}) \end{aligned} \tag{3.4}$$

The intuition to compute x_{ij} comes from the idea that music segments and movie highlights should attract accordingly to how much closer the relative positions to their total original signal are to each other (either for the music or the movie). Accordingly, we compute the distance between the two, with 0 representing the best temporal affinity and 1 the contrary.

To obtain X_{ij}, we make a domain transformation from $1 \rightarrow 0$ to $-1 \rightarrow 1$ to be able to match the emotion similarity scores. We retrieve the 6^{th} user input as the *temporal weight* and compute the final score matrix SM (Eq. 3.5) to be used to define the final matches between the music segments and movie highlights.

$$\begin{aligned} SM =& (1 - \textit{temporal weight}) \times \textit{emotional similarity matrix} \\ & + \textit{temporal weight} \times \textit{temporal weight matrix} \end{aligned} \tag{3.5}$$

The effect of the *temporal weight* on the score matrix is shown in Fig. 3.6.

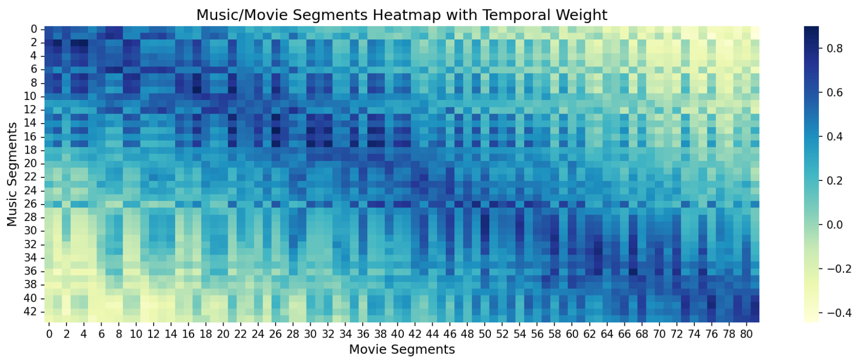


Figure 3.6: Example of a heatmap representation of the final score matrix to be used for the music segments/highlights matching.

Ultimately, we match the music and video segments, prioritizing the highest scores to the lowest.

3.3 Post-production

We take the breaking points of the selected movie highlights and use the video composition tools from the MoviePy (Zulko, 2017) python package to extract the needed movie segments, append them together with the music, and export the final tribute.

We provide a way for the user to add specific inputs that will only change the Tribute export process, avoiding making all the computations for the same configurations again. These specific inputs are:

- **Loudness adjustment for each soundtrack** (music and movie segments): By default, we adjust both soundtracks to have the same loudness in dBFS (dB relative to the maximum possible loudness). This input allows the user to manually force an increase or decrease in each soundtrack's volume;

- **Offset:** A constant positive or negative value (in seconds) that is added to all breaking points to manually adjust the movie scenes transition times (in the majority of the cases we tested, this was not necessary).

3.4 Architecture and User Interface

In the process of building the proposed pipeline, the following architectural designs were taken into consideration:

- The overall architecture is built with a clear distinction between different modules. This makes it easier to change models and algorithms being used for the multiple tasks in the pipeline: music/video feature extraction, emotional similarity scores, music/movie segmentation, and subtitle summarization, amongst others;
- Each module has its own internal *Jupyter Notebook*, where the related methods were tested;
- There is a module dedicated to storage management of the saved and temporary files generated in the process;
- In the project's root folder is a *Jupyter Notebook* with the "Main Pipeline" where it is possible to generate a movie tribute selecting from available music and movies and introducing the aforementioned *user inputs*;
- It is possible to run a *Straemlit* App that provides the user with an intuitive UI to generate movie tributes, monitor the whole process and adjust the post-production parameters (Section 3.3).

Some images of the *Straemlit* UI are provided below. *Straemlit* is a Python library that allows developers to create, as Python scripts, interactive web applications, and dashboards for data science and machine learning, accessible through a browser.

The homepage of the app (Fig. 3.7a) presents the user with some basic information regarding the difference between *Main Inputs* and *Export Inputs*:

Main Inputs:

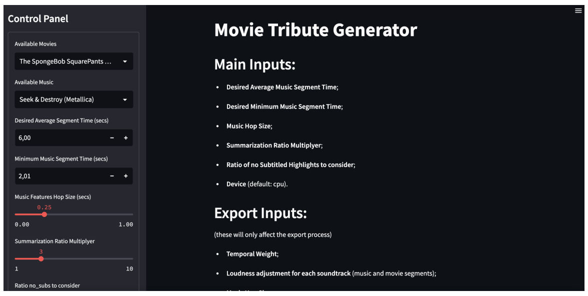
- **Movie:** Dropdown containing all the movies available in storage;
- **Music:** Dropdown containing all the music available in storage;
- **Desired Average Music Segment Time (seconds):** Number input limited by $[0.0, 20.0]$;
- **Desired Minimum Music Segment Time (seconds):** Number input limited by $[0.0, 20.0]$;
- **Music Hop Size (seconds):** slider with a 0.05 step value, lower-bounded by 0.0 and upper-bounded by 1.0. The default value is set as 0.98;
- **Summarization Ratio Multiplyer:** slider with a 1 step value, lower-bounded by 1 and upper-bounded by 10. The default value is set as 1;
- **Ratio of no Subtitled Highlights to consider:** slider with a 0.05 step value, lower-bounded by 0.0 and upper-bounded by 1.0. The default value is set as 0.5;
- **Device:** Dropdown containing two possible options: "cpu", "gpu".

Export Inputs:

- **Temporal Weight:** slider with a 0.05 step value, lower-bounded by 0.0 and upper-bounded by 1.0. The default value is set as 0.5;
- **Loudness adjustment for the music soundtrack:** slider with a 0.05 step value, lower-bounded by 0.0 and upper-bounded by 5.0. The default value is set as 1.0;
- **Loudness adjustment for the movie segments soundtrack:** slider with a 0.05 step value, lower-bounded by 0.0 and upper-bounded by 5.0. The default value is set as 1.0;
- **Movie Scenes Breakingpoints Offset (seconds):** slider with a 0.05 step value, lower-bounded by -3.0 and upper-bounded by 3.0. The default value is set as 1.0;

The *Export Inputs* are the user-defined parameters that will only affect the tribute's export process, not impacting all the remaining computation (movie scenes breaking points, music segmentation, movie highlights selection).

We do this separation because we prepared the system not to make all the computations involving the *Main Inputs* if the same configuration is already stored in the system (a tribute was



Available Movies: The SpongeBob SquarePants ...
Available Music: Seek & Destroy (Metallica)
Desired Average Segment Time (secs): 6,00
Minimum Music Segment Time (secs): 2,01
Music Features Hop Size (secs): 0.25 (0.00 – 1.00)
Summarization Ratio Multiplyer: 3 (1 – 10)
Ratio no_subs to consider

Movie Tribute Generator

Main Inputs:

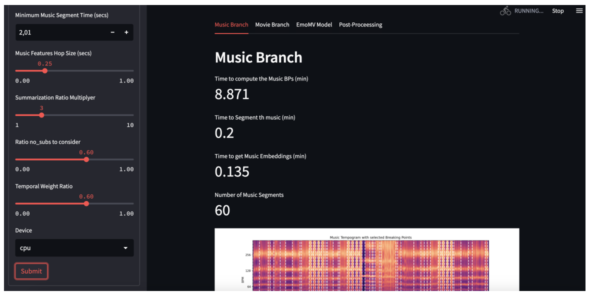
- **Desired Average Music Segment Time;**
- **Desired Minimum Music Segment Time;**
- **Music Hop Size;**
- **Summarization Ratio Multiplyer;**
- **Ratio of no Subtitled Highlights to consider;**
- **Device** (default: cpu).

Export Inputs:

(these will only affect the export process)

- **Temporal Weight;**
- **Loudness adjustment for each soundtrack** (music and movie segments);

(a) Home page of the *streamlit* UI.

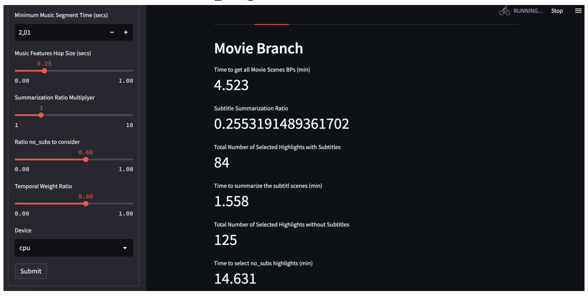


Minimum Music Segment Time (secs): 2,01
Music Features Hop Size (secs): 0.25 (0.00 – 1.00)
Summarization Ratio Multiplyer: 3 (1 – 10)
Ratio no_subs to consider: 0.60 (0.00 – 1.00)
Temporal Weight Ratio: 0.60 (0.00 – 1.00)
Device: cpu

Music Branch

Time to compute the Music BPs (min): 8.871
Time to Segment th music (min): 0.2
Time to get Music Embeddings (min): 0.135
Number of Music Segments: 60

(b) Music Branch page of the *streamlit* UI.

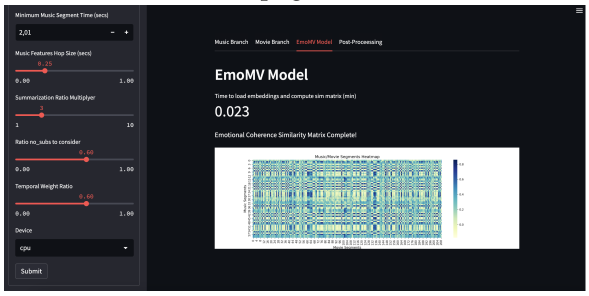


Minimum Music Segment Time (secs): 2,01
Music Features Hop Size (secs): 0.25 (0.00 – 1.00)
Summarization Ratio Multiplyer: 3 (1 – 10)
Ratio no_subs to consider: 0.60 (0.00 – 1.00)
Temporal Weight Ratio: 0.60 (0.00 – 1.00)
Device: cpu

Movie Branch

Time to get all Movie Scenes BPs (min): 4.523
Subtitle Summarization Ratio: 0.2553191489361702
Total Number of Selected Highlights with Subtitles: 84
Time to summarize the subtitl scenes (min): 1.558
Total Number of Selected Highlights without Subtitles: 125
Time to select no_subs highlights (min): 14.631

(c) Movie Branch page of the *streamlit* UI.



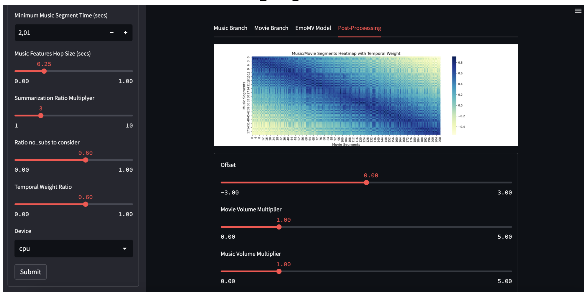
Minimum Music Segment Time (secs): 2,01
Music Features Hop Size (secs): 0.25 (0.00 – 1.00)
Summarization Ratio Multiplyer: 3 (1 – 10)
Ratio no_subs to consider: 0.60 (0.00 – 1.00)
Temporal Weight Ratio: 0.60 (0.00 – 1.00)
Device: cpu

EmoMV Model

Time to load embeddings and compute sim matrix (min): 0.023

Emotional Coherence Similarity Matrix Complete!

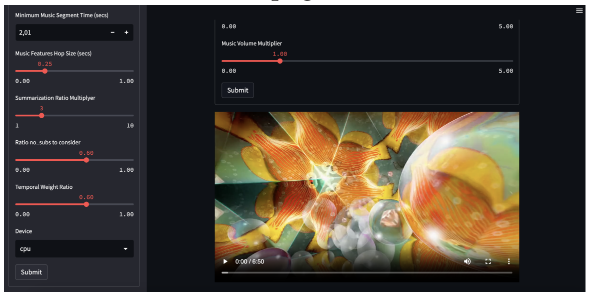
(d) EmoMV Model page of the *streamlit* UI.



Minimum Music Segment Time (secs): 2,01
Music Features Hop Size (secs): 0.25 (0.00 – 1.00)
Summarization Ratio Multiplyer: 3 (1 – 10)
Ratio no_subs to consider: 0.60 (0.00 – 1.00)
Temporal Weight Ratio: 0.60 (0.00 – 1.00)
Device: cpu

Offset: 0.00 (-3.00 – 3.00)
Movie Volume Multiplier: 1.00 (0.00 – 5.00)
Music Volume Multiplier: 1.00 (0.00 – 5.00)

(e) Post-Production page of the *streamlit* UI presenting the final score matrix.



Minimum Music Segment Time (secs): 2,01
Music Features Hop Size (secs): 0.25 (0.00 – 1.00)
Summarization Ratio Multiplyer: 3 (1 – 10)
Ratio no_subs to consider: 0.60 (0.00 – 1.00)
Temporal Weight Ratio: 0.60 (0.00 – 1.00)
Device: cpu

Music Volume Multiplier: 1.00 (0.00 – 5.00)

(f) Post-Production page of the *streamlit* UI presenting the Export User Inputs and the exported tribute.

Figure 3.7: Movie Tribute Generator UI, using *Streamlit*

previously done with the same *Main Inputs* values). We found this feature extremely relevant because the modifications in the final tribute caused by the *Export Inputs* require relatively fast visual feedback for a better experience (e.g., adjusting movie scenes' breaking points).

The *Main Inputs* are always accessible to the user through the "Control Panel" in the app's sidebar and the *Export Inputs* in the *Post-Processing* Section.

Once the *Main Inputs* are submitted, it triggers the whole process. The UI have a dedicated Section for each part of the Main Pipeline, where it presents metrics related to each part and a loading animation if the corresponding process is still running: Music Branch (Fig. 3.7b), Movie Branch (Fig. 3.7c) and EmoMV model (Fig. 3.7d).

The last Section is ”Post-Production”. In this Section, the user can introduce the *Export Inputs* (Fig. 3.7e) and visualize the latest exported movie tribute (Fig. 3.7f).

4 Experimental Setup

You need to learn how to select your thoughts just the same way you select your clothes every day. This is a power you can cultivate. If you want to control things in your life so bad, work on the mind. That's the only thing you should be trying to control.

– Richard, "Eat Pray Love"

This chapter provides an overview of the dataset utilized in our experiments, along with details about the individuals who assessed the final tributes. Subsequently, we delve into the evaluation process and engage in a comprehensive discussion of the obtained results.

4.1 Dataset

Using the *The Movie Tribute Generator* App, seven Movie Tributes were created using seven different movies and songs. Table 4.1 presents the metadata related to each tribute creation.

"To The Edge", "See You Again", "The Last Goodbye", and "About You" contain vocals, while the remaining do not.

All the genres of the movies according to IMDB are as follows:

- **"Atonement":** Drama, Mystery, Romance, War;
- **"300":** Action, Drama;
- **"Furious 7":** Action, Crime, Thriller;
- **"The Curious Case of Benjamin Button":** Drama, Fantasy, Romance;
- **"Interstellar":** Adventure, Drama, Sci-Fi
- **"Howls Moving Castle":** Animation, Adventure, Family, Fantasy;

- **"Before Sunset":** Drama, Romance.

"Howls Moving Castle" movie is the only animated movie in the dataset.

As expected, being the most processing-demanding branch, the Movie Branch is part of the architecture that takes longer to be completed.

4.2 Setup

Based on the same evaluation process described by the previous work regarding the "Automatic Generation of Movie Tributes" (Aparício, 2015), we collected opinions related to the generated movie tributes from 32 participants. Fig. 4.1 shows a characterization of the viewers, answering the following questions: "What is your age?"; "What is your gender?"; "What is your level of Education?"; "What is your area of training?"; "How often do you watch movies?"; "How often do you watch movie tributes?".

Most people were males (59.9%), had between 18 and 25 years (58.6%), had a Master's degree (53.1%), and were Computer Engineers (20.7%) or related to other areas of Engineering (20.7%) or related to Tourism (20.7%). Most people watch movies once a week (43.8%), and 7% didn't know what a movie tribute was, while 37.5% watches one once a year, at least.

4.3 Results

We present in Fig. 4.2 the results for content selection, emotional coherence criteria, and overall scores.

The tributes that had the best scores were "Atonement" and "Interstellar", with an average of 8 points in all three criteria (content selection, emotional coherence, and overall evaluation), except in the overall evaluation of "Atonement", reaching an average score of 7 points, on a scale from 1 to 10.

"Furious 7" obtained the worst scores, with 6.25 points on overall evaluation, 6.5 on content selection criteria, and 5.6 on emotional coherence criteria.

On average, our method led to average scores of 7.2, 6.8, and 6.9 on content selection, emotional coherence criteria, and overall evaluation, respectively (Fig. 4.3).

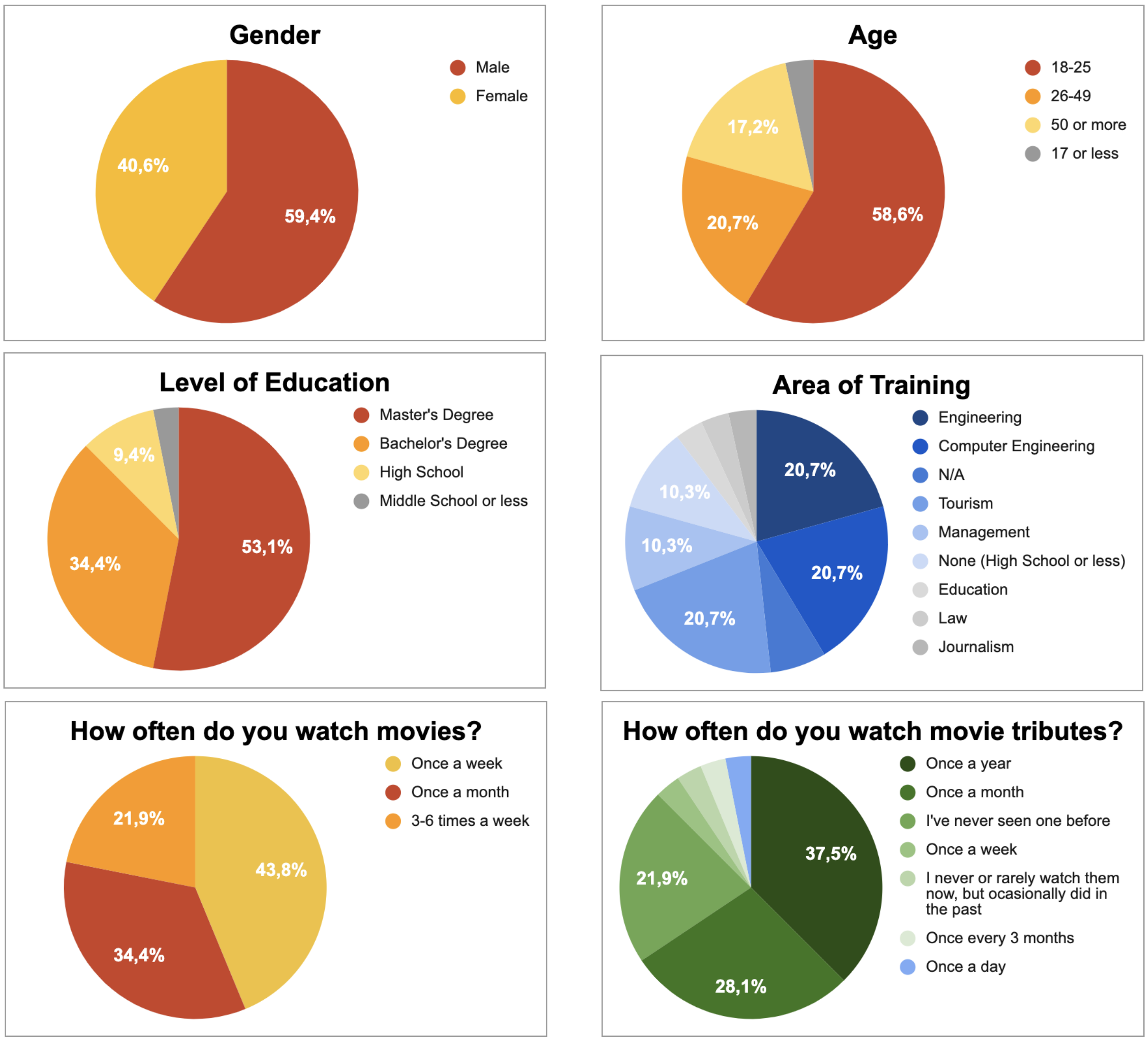
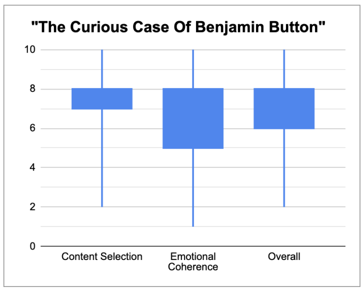


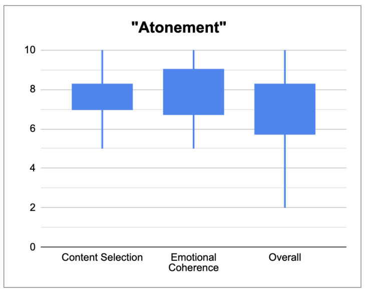
Figure 4.1: Subjects characterization.

Concerning Content Selection, some evaluators mentioned the existence of some unnecessary short clips and cuts in the middle of the dialogue in the tributes corresponding to ”The Curious Case of Benjamin Button”, ”Atonement”, ”Furious 7”, and ”Before Sunset”.

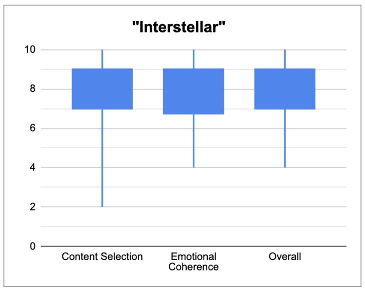
Regarding Post-Production, it was considered that some parts of the tribute have a big discrepancy between the background sound of the movie highlight and the correspondent music segment (shooting scenes, car sounds, low dialogue sound, movie background music playing at the same time as the tribute music). This was mainly identified in the following tributes: ”The Curious Case of Benjamin Button”, ”Furious 7”, ”300”, ”Interstellar”, and ”Before Sunset”



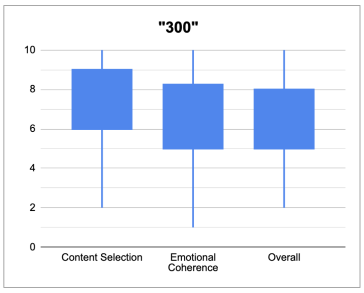
(a)



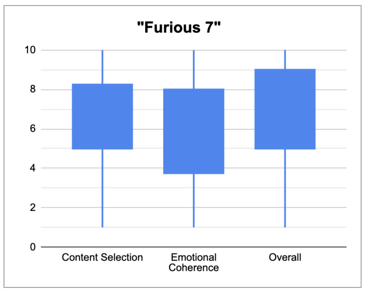
(b)



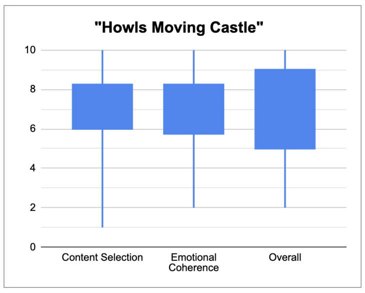
(c)



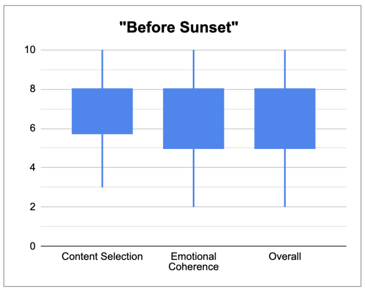
(d)



(e)



(f)



(g)

Figure 4.2: Each tribute's evaluation results.

In terms of emotional coherence, the majority of the negative critics evolve around a discrepancy between the overall tone of the song against the overall tone of the music. For instance, it was mentioned that in "Furious 7" the clips are majorly action-based and the song has more of an overall sad tone. In the case of "300", "most of the clips were pure action which felt right with the song", but there were a few clips where a bigger emotional dissonance was felt, majorly corresponding to more calm clips, with dialogues, for example.

Some highlighted comments related specifically to some tributes are the following:

- **Howls Moving Castle:** There were some comments regarding the solution's "difficulty identifying emotional intent in the movie clips or maybe does not take this into consideration at all, which makes it so the tribute doesn't transmit anything in particular";

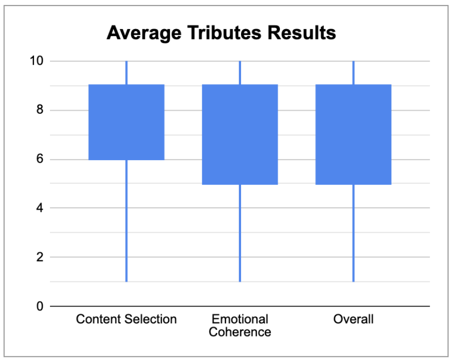


Figure 4.3: Average tributes' evaluation results.

Although the majority of this comments were from subjects that didn't watch the movie; The ones who have seen the movie, present an overall positive response (e.g.: "The nostalgia of this movie I simply love. The focus on the great characters");

- **The Curious Case of Benjamin Button:** "The main character should be more on the screen and the order of the movie should be kept since this is the whole point of the movie";
- **300:** "I liked the fact that many of the critical scenes were shown, and fantastic VFX shots were highlighted";
- **Atonement:** "Even without having watched the movie it was well shown and I feel like the important moments were there";
- **Furious 7:** "The clips don't go all the way to the end of the film. But select crucial moments of the story";
- **Interstellar:** "It's emotional and the scenes are very well linked with the music. Even the action scenes. I would just like to hear a little more of the dialogue";
- **Before Sunset:** There were a lot of complaints regarding the audio quality of the movie and the major quantity of dialogue scenes, that conflicted with the music vocals. "The music would probably fit the movie well if it was well edited"; "There is a lot of dialog and transitions while the characters are in the middle of a sentence".

Figure 4.4 shows that all metrics have a strong positive correlation with each other. The ones that appear to be more strongly related are Content Selection and Emotional Coherence, and Content Selection and Overall Evaluation.

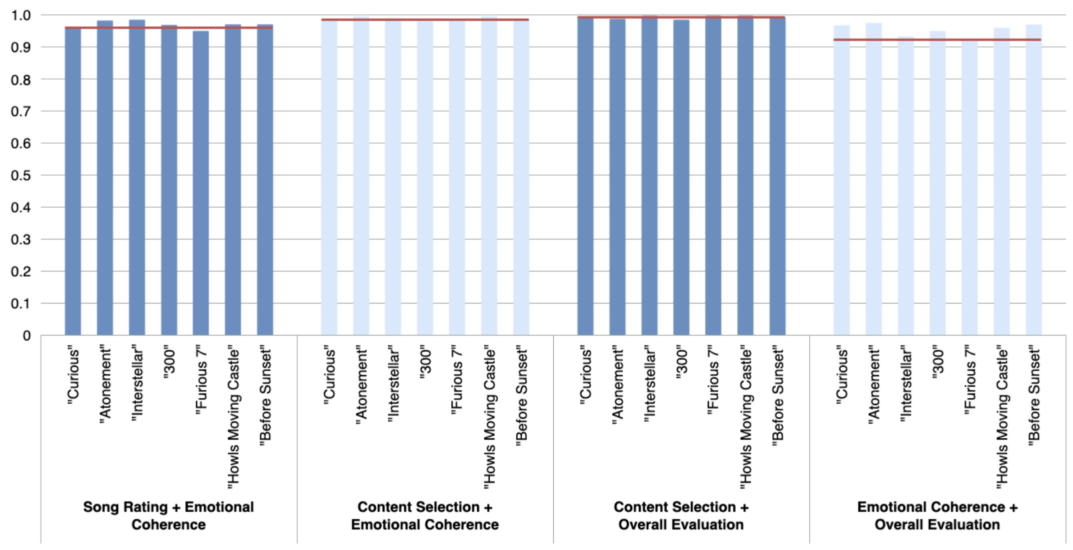


Figure 4.4: Spearman's ranks.

4.4 Discussion

In this work, content selection is driven by text streams to select the movie highlights containing subtitles and visual streams to select the movie highlights not containing subtitles. The text stream is obtained through the subtitles file (*.srt*) of the movie and the visual streams correspond to extracted embeddings of movie scenes without subtitles. We then match extracted video embeddings of the selected movie scenes with extracted audio embeddings from the music stream using the Emomv Model for Affective Music-video Retrieval and an auxiliary matrix containing weights corresponding to the chronological order of the selected highlights in the original movie.

The majority of the negative critics of the generated tributes, seem to be deeply related to the tribute's edition quality: bad quality and/or balance between the audio from the movie highlights and the music; scenes containing dialog being cut in the middle of it; unnecessary too short clips.

We present some possible solutions to tackle these issues in the Future Work (Section 5.2), although we are also able to tackle the majority of these problems by tweaking the input parameters of the architecture.

Taking the "Before Sunset" movie tribute as an example:

- **Bad audio quality of the movie:** To create this tribute we set *Loudness adjustment for* *the music soundtrack*$= 1.5$, leading to the screeching sound of the dialog; We can achieve a similar result by setting *Loudness adjustment for the movie segments soundtrack*$= 1.5$ instead, not damaging the movie's audio;

- **Transitions in the middle of dialogs:** This in specific can be characterized as a continuous conversation between two people That is why we defined the input parameters in order to process many more subtitled scenes than no subtitled scenes. Although, there are two parameters that we could tweak that would certainly increase the quality of the tribute: *Desired Average Music Segment Time* and *Desired Minimum Music Segment Time*. They were set to *00:04'00* and *00:02'06* respectively. By increasing both parameters, we can compute fewer music breaking points, resulting in longer music segments, which will lead to longer highlights, conveying more contextual information to the viewer about the context of the respective dialog.

Our proposed architecture makes the process of generating a tribute much more efficient by providing an intuitive UI where it is possible to tweak all the input parameters, returning live visual feedback of the changes. This facilitates the process of trial and error until the user is satisfied with the final result.

Tribute	**Movie**	**″Atonement″ (2007)**	**″300″ (2006)**	**″Furious 7″ (2015)**	**″The Curious Case of Benjamin Button″ (2008)**	**″Interstellar″ (2014)**	**″Howls Moving Castle″ (2004)**	**″Before Sunset″ (2004)**
	Music	**″La Plage″ by Yann Tiersen**	**″To The Edge″ by Lacuna Coil**	**″See You Again″ by Wiz Khalifa**	**″The Last Goodbye″ by Billy Boyd**	**″Chevallers De Sangreal″ by Hans Zimmer**	**″Merry-Go-Round of Life″ by Joe Hisaishi**	**″About You″ by The 1975**
	Duration	01:57	03:19	03:46	04:05	04:23	02:44	05:24
Main Inputs	**Desired Averaged Music Segment Time**	4	3	3	4	4	3	4
	Desired Minimum Music Segment Time	00:01′06	00:01′06	00:00′49	00:02′06	00:02′06	00:01′06	00:02′06
	Music Hop Size	00:00′15	00:00′15	00:00′12	00:00′12	00:00′12	00:00′15	00:00′12
	Summarization Ratio Multiplyer	2	2	2	2	3	2	4
	Ratio of no Subtitled Highlights to consider	0.25	0.75	0.7	0.5	0.3	0.5	0.2
	Device	cpu	cpu	cpu	cpu	cpu	cpu	cpu
Export Inputs	**Temporal Weight**	0.5	0.5	0.35	0.5	0.5	0.5	0.5
	Loudness adjustment for each the music soundtrack	1.0	1.0	1.0	1.0	1.0	1.0	1.0
	Loudness adjustment for each the movie segments soundtrack	1.0	1.0	1.0	1.0	1.0	1.0	1.5
	Movie Scenes breaking points Offset	0	0.3	0	0	0.5	0	0.2
Metadata	**Music Branch Processing Time**	00:51′00	02:20′00	02:54′00	03:41′00	03:47′00	01:35′00	06:21′00
	Number of Music Segments	30	53	71	48	44	46	64
	Movie Branch Processing Time	22:26′00	16:44′00	19:43′00	16:49′00	21:10′00	13:10′00	25:42′00
	Total Number of Selected Highlights with Subtitles	31	47	55	44	58	40	186
	Total Number of Selected Highlights without Subtitles	10	141	128	44	24	40	46
	Exporting Time	02:27′00	01:27′00	01:51′00	01:43′00	02:59′00	01:20′00	03:13′00

Table 4.1: Generated Movie Tributes′ data.

5 Conclusions and Future Work

Don't adventures ever have an end? I suppose not. Someone else always has to carry on the story.

– Bilbo Baggins

5.1 Conclusions

The focus of this work was on the creation of multimedia artifacts, particularly movie tributes. We implemented various methods for selecting content and ensuring emotional coherence in the generation process. As described in a previous iteration of this topic by Aparício (2015), *a movie tribute consists of a short music video containing essential parts of the movie playing along with the specified song*. In order to produce this artifact, we created a framework to: **1)** Segment the music based on its tempo and extract audio features from each segment using the VGGish (Hershey et al., 2017) network pre-trained on the AudioSet ontology; **2)** Run extractive summarization using BERT on the subtitles corpus to select highlights corresponding to movie scenes with subtitles; **3)** Implement K-means clustering to select movie highlights without subtitles; **4)** Extract video embeddings from the selected highlights using SlowFast (Feichtenhofer et al., 2019) network pre-trained on the Kinetics human action video dataset (Kay et al., 2017); **5)** Leverage the EmoMV Model for Affective Music Video Correspondence Learning (pre-trained on the EmoMV-C dataset) to retrieve similarity scores between the selected highlights and the music segments; **6)** Match each music segment to a corresponding movie highlight based on the similarity scores produced by the EmoMV model and a computed weight related to the chronological order of the scenes in the original movie.

All this process can be triggered and monitored in real-time through a simple UI created using *Streamlite*.

Seven tributes were generated, and their overall human evaluation was positive. On average, our method led to average scores of 7.2, 6.8, and 6.9 on content selection, emotional coherence, and overall evaluation on a scale from 1 to 10. The tributes that had the best scores were ”Atonement” and ”Interstellar,” with an average of 8 points in all three criteria (content selection, emotional coherence, and overall evaluation), except in the overall evaluation of ”Atonement,” reaching an average score of 7 points. ”Furious 7” obtained the worst scores, with 6.25 points on overall evaluation, 6.5 on content selection criteria, and 5.6 on emotional coherence criteria.

The more strongly correlated metrics are Content Selection and Emotional Coherence, and Content Selection and Overall Evaluation. However, all metrics present a strong correlation with each other.

5.2 Future Work

Regarding future work, we identified some actions in order to mitigate the majority of the identified issues:

- **Improving Emotional Coherence scores computation:** We could modify the EmoMV model, creating a third branch dedicated to extracting and processing the text embeddings of the subtitles if the scene has it, in a similar way that we do with the visual and audio streams, so we could leverage more contextual information for computing the emotional coherence scores;

- **Experiment pre-training the EmoMV model with EmoMV-A and EmoMV-B datasets (separately and all together):** In the proposed pipeline, we used the EmoMV model pre-trained on the EmoMV-C dataset, because, from the three, it is the one that related the better with our problem. Nevertheless, it would be interesting to test the performance of the architecture and the quality of the subsequently generated movie tributes by pre-training the model with the other available datasets;

- **Fine tune BERT with a dataset dedicated to the subtitle (movie dialog) format:** We gathered data comprising of complete *srt* files containing movie subtitles by contacting directly the *Open Subtitles Organization* (OpenSubtitles, 2023) and, separately, the subtitles of highlights to the collected movies. We obtained the latter by leveraging the YouTube

API to retrieve the subtitles and metadata related to the videos containing highlights from specific movies in the Movieclips Youtube channel (MovieclipsYoutubeChannel, 2023). We did not develop this further because most of the collected subtitles from youtube were automatically generated, possessing many discrepancies compared to the original movie subtitles. We tried to clean and correct the dataset, but we would need to dive deeper in order to be able to solve this issue completely, so we did not pursue it since it was taking a long time and it was leading us out of the scope of this thesis;

- **Improve Post-production:** This work shows the importance of having an intuitive User Interface to generate movie tributes. Being able to tweak some parameters with quick visual feedback of the consequent changes in the tribute gives the user a lot of power and control over the overall quality of the end product. That said, some improvements that could be made regarding this part of the pipeline are: By default, we normalize the audio of the movie highlights and the audio of the music before joining them. The problem with this implementation is that the audio stream from the movie usually comprises a much bigger spectrum of audio intensity (e.g., in "Furious 7," we have both scenes with loud car noises and calm dialog). By normalizing all the movie segments' audio together (as in a single audio stream), the discrepancy between the loudness of both scenes is maintained or even worsened. In order to solve this, we could normalize the audio from each movie highlight separately and only then join everything together. Another helpful feature would be to modify the *offset* and *loudness adjustment* parameters for each tribute segment separately instead of together.

Bibliography

Ackley, D. H., Hinton, G. E., & Sejnowski, T. J. (1985). A learning algorithm for boltzmann machines. *Cognitive science*, *9*(1), 147–169.

Aditya, D., Manvitha, R., Samyak, M., & Shamitha, B. (2021). Emotion based video player. *Global Transitions Proceedings*, *2*(2), 368–374.

Aparício, A. M. S. (2015). *Automated generation of movie tributes* (Unpublished master's thesis). ISCTE-IUL.

Arlot, S., Celisse, A., & Harchaoui, Z. (2019). A kernel multiple change-point algorithm via model selection. *Journal of machine learning research*, *20*(162).

Atencio, P., German, S.-T., Branch, J. W., & Delrieux, C. (2019). Video summarisation by deep visual and categorical diversity. *IET Computer Vision*, *13*(6), 569–577.

Aytar, Y., Vondrick, C., & Torralba, A. (2016). Soundnet: Learning sound representations from unlabeled video. *Advances in neural information processing systems*, *29*.

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Baveye, Y., Dellandrea, E., Chamaret, C., & Chen, L. (2015). Liris-accede: A video database for affective content analysis. *IEEE Transactions on Affective Computing*, *6*(1), 43–55.

Böck, S., & Widmer, G. (2013). Maximum filter vibrato suppression for onset detection. In *Proc. of the 16th int. conf. on digital audio effects (dafx). maynooth, ireland (sept 2013)* (Vol. 7, p. 4).

Bradley, M. M., Greenwald, M. K., Petry, M. C., & Lang, P. J. (1992). Remembering pictures: pleasure and arousal in memory. *Journal of experimental psychology: Learning, Memory, and Cognition*, *18*(2), 379.

Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, *30*(1-7), 107–117.

Brown, C. D., & Davis, H. T. (2006). Receiver operating characteristics curves and related decision measures: A tutorial. *Chemometrics and Intelligent Laboratory Systems*, *80*(1), 24–38.

Carbonell, J., & Goldstein, J. (1998). The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international acm sigir conference on research and development in information retrieval* (pp. 335–336).

Carreira, J., & Zisserman, A. (2017). Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the ieee conference on computer vision and pattern recognition* (pp. 6299–6308).

Castellano, B. (2023). PySceneDetect. (`https://scenedetect.com/en/latest/`)

Celisse, A., Marot, G., Pierre-Jean, M., & Rigaill, G. (2018). New efficient algorithms for multiple change-point detection with reproducing kernels. *Computational Statistics & Data Analysis*, *128*, 200–220.

Chao, W.-L., Gong, B., Grauman, K., & Sha, F. (2015). Large-margin determinantal point processes. In *Uai.*

Chelba, C., Mikolov, T., Schuster, M., Ge, Q., Brants, T., Koehn, P., & Robinson, T. (2014). *One billion word benchmark for measuring progress in statistical language modeling.*

Chen, T., Lu, A., & Hu, S.-M. (2012). Visual storylines: Semantic visualization of movie sequence. *Computers & Graphics*, *36*(4), 241–249.

Cho, K., Van Merriënboer, B., Bahdanau, D., & Bengio, Y. (2014). On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259.*

Chu, W.-S., Song, Y., & Jaimes, A. (2015). Video co-summarization: Video summarization by visual co-occurrence. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 3584–3592).

Chung, J. S., Senior, A., Vinyals, O., & Zisserman, A. (2017). Lip reading sentences in the wild. In *2017 ieee conference on computer vision and pattern recognition (cvpr)* (pp. 3444–3453).

Cowen, A. S., & Keltner, D. (2017). Self-report captures 27 distinct categories of emotion bridged by continuous gradients. *Proceedings of the national academy of sciences*, *114*(38), E7900–E7909.

Crammer, K., & Singer, Y. (2001). On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of machine learning research*, *2*(Dec), 265–292.

De Avila, S. E. F., Lopes, A. P. B., da Luz Jr, A., & de Albuquerque Araújo, A. (2011). Vsumm: A mechanism designed to produce static video summaries and a novel evaluation method. *Pattern Recognition Letters*, *32*(1), 56–68.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 ieee conference on computer vision and pattern recognition* (pp. 248–255).

Deng, L., Hinton, G., & Kingsbury, B. (2013). New types of deep neural network learning for speech recognition and related applications: An overview. In *2013 ieee international conference on acoustics, speech and signal processing* (pp. 8599–8603).

Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2018). BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, *abs/1810.04805*. Retrieved from http://arxiv.org/abs/1810.04805

Dilawari, A., & Khan, M. U. G. (2019). Asovs: abstractive summarization of video sequences. *IEEE Access*, *7*, 29253–29263.

Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., & Darrell, T. (2015). Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 2625–2634).

Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., ... Brox, T. (2015). Flownet: Learning optical flow with convolutional networks. In *Proceedings of the ieee international conference on computer vision* (pp. 2758–2766).

Ekman, P., Sorenson, E. R., & Friesen, W. V. (1969). Pan-cultural elements in facial displays of emotion. *Science*, *164*(3875), 86–88.

Erkan, G., & Radev, D. R. (2004). Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, *22*, 457–479.

Fajtl, J., Sokeh, H. S., Argyriou, V., Monekosso, D., & Remagnino, P. (2019). Summarizing videos with attention. In *Asian conference on computer vision* (pp. 39–54).

Feichtenhofer, C., Fan, H., Malik, J., & He, K. (2019). Slowfast networks for video recognition. In *Proceedings of the ieee/cvf international conference on computer vision* (pp. 6202–6211).

Ferman, A. M., & Tekalp, A. M. (1997). Multiscale content extraction and representation for video indexing. In *Multimedia storage and archiving systems ii* (Vol. 3229, pp. 23–31).

Fernández-Aguilar, L., Navarro-Bravo, B., Ricarte, J., Ros, L., & Latorre, J. M. (2019). How effective are films in inducing positive and negative emotional states? a meta-analysis. *PloS one*, *14*(11), e0225040.

Fu, Y., Guo, Y., Zhu, Y., Liu, F., Song, C., & Zhou, Z.-H. (2010). Multi-view video summarization. *IEEE Transactions on Multimedia*, *12*(7), 717–729.

Furini, M., Geraci, F., Montangero, M., & Pellegrini, M. (2010). Stimo: Still and moving video storyboard for the web scenario. *Multimedia Tools and Applications*, *46*(1), 47–69.

Gemmeke, J. F., Ellis, D. P., Freedman, D., Jansen, A., Lawrence, W., Moore, R. C., ... Ritter, M. (2017). Audio set: An ontology and human-labeled dataset for audio events. In *2017 ieee international conference on acoustics, speech and signal processing (icassp)* (pp. 776–780).

Goldman, D. B., Curless, B., Salesin, D., & Seitz, S. M. (2006). Schematic storyboarding for video visualization and editing. *Acm transactions on graphics (tog)*, *25*(3), 862–871.

Gong, B., Chao, W.-L., Grauman, K., & Sha, F. (2014). Diverse sequential subset selection for supervised video summarization. *Advances in neural information processing systems*, *27*, 2069–2077.

Gong, Y., & Liu, X. (2001). Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th annual international acm sigir conference on research and development in information retrieval* (pp. 19–25).

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... Bengio, Y. (2014). Generative adversarial nets. *Advances in neural information processing systems*, *27*.

Goutte, C., & Gaussier, E. (2005). A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In *Advances in information retrieval: 27th european conference on ir research, ecir 2005, santiago de compostela, spain, march 21-23, 2005. proceedings 27* (pp. 345–359).

Graves, A., & Jaitly, N. (2014). Towards end-to-end speech recognition with recurrent neural networks. In *International conference on machine learning* (pp. 1764–1772).

Graves, A., Mohamed, A.-r., & Hinton, G. (2013). Speech recognition with deep recurrent neural networks. In *2013 ieee international conference on acoustics, speech and signal processing* (pp. 6645–6649).

Grosche, P., Müller, M., & Kurth, F. (2010). Cyclic tempogram—a mid-level tempo representation for musicsignals. In *2010 ieee international conference on acoustics, speech and signal processing* (pp. 5522–5525).

Gygli, M., Grabner, H., Riemenschneider, H., & Van Gool, L. (2014). Creating summaries from user videos. In *European conference on computer vision* (pp. 505–520).

Gygli, M., Grabner, H., & Van Gool, L. (2015). Video summarization by learning submodular mixtures of objectives. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 3090–3098).

Hadsell, R., Chopra, S., & LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. In *2006 ieee computer society conference on computer vision and pattern recognition (cvpr'06)* (Vol. 2, pp. 1735–1742).

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 770–778).

Hendrycks, D., & Gimpel, K. (2016). Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

Herranz, L., & Martinez, J. M. (2010). A framework for scalable summarization of video. *IEEE Transactions on Circuits and Systems for Video Technology*, *20*(9), 1265–1270.

Hershey, S., Chaudhuri, S., Ellis, D. P., Gemmeke, J. F., Jansen, A., Moore, R. C., ... others (2017). Cnn architectures for large-scale audio classification. In *2017 ieee international conference on acoustics, speech and signal processing (icassp)* (pp. 131–135).

Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *science*, *313*(5786), 504–507.

Hochreiter, S. (1998). The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, *6*(02), 107–116.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Hong, R., Tang, J., Tan, H.-K., Yan, S., Ngo, C., & Chua, T.-S. (2009). Event driven summarization for web videos. In *Proceedings of the first sigmm workshop on social media* (pp. 43–48).

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, *24*(6), 417.

Hussain, T., Muhammad, K., Ding, W., Lloret, J., Baik, S. W., & de Albuquerque, V. H. C. (2021). A comprehensive survey of multi-view video summarization. *Pattern Recognition*, *109*, 107567.

James, W. (2007). *The principles of psychology* (Vol. 1). Cosimo, Inc.

Jaquet, L., Danuser, B., & Gomez, P. (2014). Music and felt emotions: How systematic pitch level variations affect the experience of pleasantness and arousal. *Psychology of Music*, *42*(1), 51–70.

Jernite, Y., Bowman, S. R., & Sontag, D. (2017). Discourse-based objectives for fast unsupervised sentence representation learning. *arXiv preprint arXiv:1705.00557*.

Ji, Z., Xiong, K., Pang, Y., & Li, X. (2019). Video summarization with attention-based encoder–decoder networks. *IEEE Transactions on Circuits and Systems for Video Technology*, *30*(6), 1709–1717.

Jin, J., Fu, K., Cui, R., Sha, F., & Zhang, C. (2015). Aligning where to see and what to tell: image caption with region-based attention and scene factorization. *arXiv preprint arXiv:1506.06272*.

Jozefowicz, R., Zaremba, W., & Sutskever, I. (2015). An empirical exploration of recurrent network architectures. In *International conference on machine learning* (pp. 2342–2350).

Juslin, P. N., & Laukka, P. (2004). Expression, perception, and induction of musical emotions: A review and a questionnaire study of everyday listening. *Journal of new music research*, *33*(3), 217–238.

Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of artificial intelligence research*, *4*, 237–285.

Kanehira, A., Van Gool, L., Ushiku, Y., & Harada, T. (2018). Aware video summarization. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 7435–7444).

Karpathy, A., & Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 3128–3137).

Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., . . . others (2017). The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*.

Khan, A. A., Shao, J., Ali, W., & Tumrani, S. (2020). Content-aware summarization of broadcast sports videos: An audio–visual feature extraction approach. *Neural Processing Letters*, *52*(3), 1945–1968.

Khosla, A., Hamid, R., Lin, C.-J., & Sundaresan, N. (2013). Large-scale video summarization using web-image priors. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 2698–2705).

Kim, C., & Hwang, J.-N. (2000). An integrated scheme for object-based video abstraction. In *Proceedings of the eighth acm international conference on multimedia* (pp. 303–311).

Kim, G., Sigal, L., & Xing, E. P. (2014). Joint summarization of large-scale collections of web images and videos for storyline reconstruction. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 4225–4232).

Kim, G., & Xing, E. P. (2014). Reconstructing storyline graphs for image recommendation from web community photos. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 3882–3889).

Kim, G., Xing, E. P., Fei-Fei, L., & Kanade, T. (2011). Distributed cosegmentation via submodular optimization on anisotropic diffusion. In *2011 international conference on computer vision* (pp. 169–176).

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Koehn, P., & Knowles, R. (2017). Six challenges for neural machine translation. *arXiv preprint arXiv:1706.03872*.

Kolmogorov, V., & Zabin, R. (2004). What energy functions can be minimized via graph cuts? *IEEE transactions on pattern analysis and machine intelligence*, *26*(2), 147–159.

Kreibig, S. D., Samson, A. C., & Gross, J. J. (2013). The psychophysiology of mixed emotional states. *Psychophysiology*, *50*(8), 799–811.

Kulesza, A., & Taskar, B. (2011). Learning determinantal point processes. *Computer Sciences Commons*.

Kuo, J. R., Neacsiu, A. D., Fitzpatrick, S., & MacDonald, D. E. (2014). A methodological examination of emotion inductions in borderline personality disorder: A comparison of standardized versus idiographic stimuli. *Journal of Psychopathology and Behavioral Assessment*, *36*(1), 155–164.

Lee, Y. J., Ghosh, J., & Grauman, K. (2012). Discovering important people and objects for egocentric video summarization. In *2012 ieee conference on computer vision and pattern recognition* (pp. 1346–1353).

Li, B., & Kumar, A. (2019). Query by video: Cross-modal music retrieval. In *Ismir* (pp. 604–611).

Li, Y., & Merialdo, B. (2010). Multi-video summarization based on video-mmr. In *11th international workshop on image analysis for multimedia interactive services wiamis 10* (pp. 1–4).

Liu, T., & Kender, J. R. (2002a). Optimization algorithms for the selection of key frame sequences of variable length. In *European conference on computer vision* (pp. 403–417).

Liu, T., & Kender, J. R. (2002b). Rule-based semantic summarization of instructional videos. In *Proceedings. international conference on image processing* (Vol. 1, pp. I–I).

Liu, Y., Zhou, F., Liu, W., De la Torre, F., & Liu, Y. (2010). Unsupervised summarization of rushes videos. In *Proceedings of the 18th acm international conference on multimedia* (pp. 751–754).

Logeswaran, L., & Lee, H. (2018). An efficient framework for learning sentence representations. *arXiv preprint arXiv:1803.02893*.

Lu, Z., & Grauman, K. (2013). Story-driven summarization for egocentric video. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 2714–2721).

Luong, M.-T., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Ma, Y.-F., Lu, L., Zhang, H.-J., & Li, M. (2002). A user attention model for video summarization. In *Proceedings of the tenth acm international conference on multimedia* (pp. 533–542).

MacQueen, J. (1967). Classification and analysis of multivariate observations. In *5th berkeley symp. math. statist. probability* (pp. 281–297).

Mahasseni, B., Lam, M., & Todorovic, S. (2017). Unsupervised video summarization with adversarial lstm networks. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 202–211).

McFee, B., Raffel, C., Liang, D., Ellis, D. P., McVicar, M., Battenberg, E., & Nieto, O. (2015). librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference* (Vol. 8, pp. 18–25).

Miller, D. (2019). *Leveraging bert for extractive text summarization on lectures.*

Money, A. G., & Agius, H. (2008). Video summarization: A conceptual framework and survey of the state of the art. *Journal of visual communication and image representation*, *19*(2), 121–143.

Monfort, M., Andonian, A., Zhou, B., Ramakrishnan, K., Bargal, S. A., Yan, T., ... others (2019). Moments in time dataset: one million videos for event understanding. *IEEE transactions on pattern analysis and machine intelligence*, *42*(2), 502–508.

Morency, L.-P., Mihalcea, R., & Doshi, P. (2011). Towards multimodal sentiment analysis: Harvesting opinions from the web. In *Proceedings of the 13th international conference on multimodal interfaces* (pp. 169–176).

Morere, O., Goh, H., Veillard, A., Chandrasekhar, V., & Lin, J. (2015). Co-regularized deep representations for video summarization. In *2015 ieee international conference on image processing (icip)* (pp. 3165–3169).

MovieclipsYoutubeChannel. (2023). Movieclips Youtube Channel. (https://www.youtube.com/@MOVIECLIPS)

Mundur, P., Rao, Y., & Yesha, Y. (2006). Keyframe-based video summarization using delaunay clustering. *International Journal on Digital Libraries*, *6*(2), 219–232.

Nagrani, A., Chung, J. S., & Zisserman, A. (2017). Voxceleb: a large-scale speaker identification dataset. *arXiv preprint arXiv:1706.08612*.

Narasimhan, M., Rohrbach, A., & Darrell, T. (2021). Clip-it! language-guided video summarization. *Advances in Neural Information Processing Systems*, *34*.

Ngo, C.-W., Ma, Y.-F., & Zhang, H.-J. (2003). Automatic video summarization by graph modeling. In *Proceedings ninth ieee international conference on computer vision* (pp. 104–109).

Olah, C. (2015). Understanding LSTM Networks. (https://colah.github.io/posts/2015-08-Understanding-LSTMs/)

OpenSubtitles. (2023). Open Subtitles. (https://www.opensubtitles.org/)

Otani, M., Nakashima, Y., Rahtu, E., & Heikkilä, J. (2019). Rethinking the evaluation of video summaries. In *2019 ieee/cvf conference on computer vision and pattern recognition (cvpr)* (p. 7588-7596). doi: 10.1109/CVPR.2019.00778

Panda, R., & Roy-Chowdhury, A. K. (2017). Multi-view surveillance video summarization via joint embedding and sparse optimization. *IEEE Transactions on Multimedia*, *19*(9), 2010–2021.

Pandeya, Y. R., Bhattarai, B., & Lee, J. (2021). Deep-learning-based multimodal emotion classification for music videos. *Sensors*, *21*(14), 4927.

Parkhi, O. M., Vedaldi, A., & Zisserman, A. (2015). Deep face recognition. *British Machine Vision Association*.

Phi, M. (2018). Illustrated Guide to Recurrent Neural Networks. (https://towardsdatascience.com/illustrated-guide-to-recurrent-neural-networks-79e5eb8049c9)

Potapov, D., Douze, M., Harchaoui, Z., & Schmid, C. (2014). Category-specific video summarization. In *European conference on computer vision* (pp. 540–555).

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., … others (2021). Learning transferable visual models from natural language supervision. In *International conference on machine learning* (pp. 8748–8763).

Rav-Acha, A., Pritch, Y., & Peleg, S. (2006). Making a long video short: Dynamic video synopsis. In *2006 ieee computer society conference on computer vision and pattern recognition (cvpr'06)* (Vol. 1, pp. 435–441).

Ribeiro, R., & de Matos, D. M. (2011). Centrality-as-relevance: support sets and similarity as geometric proximity. *Journal of Artificial Intelligence Research*, *42*, 275–308.

Sah, S., Kulhare, S., Gray, A., Venugopalan, S., Prud'Hommeaux, E., & Ptucha, R. (2017). Semantic text summarization of long videos. In *2017 ieee winter conference on applications of computer vision (wacv)* (pp. 989–997).

Sharghi, A., Gong, B., & Shah, M. (2016). Query-focused extractive video summarization. In *European conference on computer vision* (pp. 3–19).

Sharghi, A., Laurel, J. S., & Gong, B. (2017). Query-focused video summarization: Dataset, evaluation, and a memory network based approach. In *2017 ieee conference on computer vision and pattern recognition (cvpr)* (p. 2127-2136). doi: 10.1109/CVPR.2017.229

Smith, A. R. (1978). Color gamut transform pairs. *ACM Siggraph Computer Graphics*, *12*(3), 12–19.

Song, Y., Vallmitjana, J., Stent, A., & Jaimes, A. (2015). Tvsum: Summarizing web videos using titles. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 5179–5187).

Sreeja, M., & Kovoor, B. C. (2019). Towards genre-specific frameworks for video summarisation: A survey. *Journal of Visual Communication and Image Representation*, *62*, 340–358.

Stefanidis, A., Partsinevelos, P., Agouris, P., & Doucette, P. (2000). Summarizing video datasets in the spatiotemporal domain. In *Proceedings 11th international workshop on database and expert systems applications* (pp. 906–912).

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., … Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 1–9).

Taylor, W. L. (1953). "cloze procedure": A new tool for measuring readability. *Journalism quarterly*, *30*(4), 415–433.

Teufel, S. (2007). An overview of evaluation methods in trec ad hoc information retrieval and trec question answering. *Evaluation of text and speech systems*, 163–186.

Thao, H. T. P., Balamurali, B., Roig, G., & Herremans, D. (2021). Attendaffectnet–emotion prediction of movie viewers using multimodal fusion with self-attention. *Sensors*, *21*(24), 8356.

Thao, H. T. P., Roig, G., & Herremans, D. (2023). Emomv: Affective music-video correspondence learning datasets for classification and retrieval. *Information Fusion*, *91*, 64–79.

Tiwari, V., & Bhatnagar, C. (2021). A survey of recent work on video summarization: approaches and techniques. *Multimedia Tools and Applications*, 1–35.

Truong, B. T., & Venkatesh, S. (2007). Video abstraction: A systematic review and classification. *ACM transactions on multimedia computing, communications, and applications (TOMM)*, *3*(1), 3–es.

Truong, C., Oudre, L., & Vayatis, N. (2020). Selective review of offline change point detection methods. *Signal Processing*, *167*, 107299.

Van Tilburg, W. A., Wildschut, T., & Sedikides, C. (2018). Nostalgia's place among self-relevant emotions. *Cognition and Emotion*, *32*(4), 742–759.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., … Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998–6008).

Venugopalan, S., Rohrbach, M., Donahue, J., Mooney, R., Darrell, T., & Saenko, K. (2015). Sequence to sequence-video to text. In *Proceedings of the ieee international conference on computer vision* (pp. 4534–4542).

Venugopalan, S., Xu, H., Donahue, J., Rohrbach, M., Mooney, R., & Saenko, K. (2014). Translating videos to natural language using deep recurrent neural networks. *arXiv preprint arXiv:1412.4729*.

Verma, G., Dhekane, E. G., & Guha, T. (2019). Learning affective correspondence between music and image. In *Icassp 2019-2019 ieee international conference on acoustics, speech and signal processing (icassp)* (pp. 3975–3979).

Vianna, E. P., Weinstock, J., Elliott, D., Summers, R., & Tranel, D. (2006). Increased feelings with increased body signals. *Social cognitive and affective neuroscience*, *1*(1), 37–48.

Wang, J.-C., Yang, Y.-H., Jhuo, I.-H., Lin, Y.-Y., & Wang, H.-M. (2012). The acousticvisual emotion guassians model for automatic generation of music video. In *Proceedings of the 20th acm international conference on multimedia* (pp. 1379–1380).

Watson, D., & Tellegen, A. (1985). Toward a consensual structure of mood. *Psychological bulletin*, *98*(2), 219.

Watson, D., Wiese, D., Vaidya, J., & Tellegen, A. (1999). The two general activation systems of affect: Structural findings, evolutionary considerations, and psychobiological evidence. *Journal of personality and social psychology*, *76*(5), 820.

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., … Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning* (pp. 2048–2057).

Yadati, K., Katti, H., & Kankanhalli, M. (2013). Cavva: Computational affective video-in-video advertising. *IEEE Transactions on Multimedia*, *16*(1), 15–23.

Yao, L., Torabi, A., Cho, K., Ballas, N., Pal, C., Larochelle, H., & Courville, A. (2015). Describing videos by exploiting temporal structure. In *Proceedings of the ieee international conference on computer vision* (pp. 4507–4515).

Zentner, M., Grandjean, D., & Scherer, K. R. (2008). Emotions evoked by the sound of music: characterization, classification, and measurement. *Emotion*, *8*(4), 494.

Zhang, H. J., Wu, J., Zhong, D., & Smoliar, S. W. (1997). An integrated system for content-based video retrieval and browsing. *Pattern recognition*, *30*(4), 643–658.

Zhang, K., Chao, W.-L., Sha, F., & Grauman, K. (2016a). Summary transfer: Exemplar-based subset selection for video summarization. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 1059–1067).

Zhang, K., Chao, W.-L., Sha, F., & Grauman, K. (2016b). Video summarization with long short-term memory. In *European conference on computer vision* (pp. 766–782).

Zhao, B., Li, X., & Lu, X. (2018). Hsa-rnn: Hierarchical structure-adaptive rnn for video summarization. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 7405–7414).

Zhao, B., & Xing, E. P. (2014). Quasi real-time summarization for consumer videos. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 2513–2520).

Zhao, S., Li, Y., Yao, X., Nie, W., Xu, P., Yang, J., & Keutzer, K. (2020). Emotion-based end-to-end matching between image and music in valence-arousal space. In *Proceedings of the 28th acm international conference on multimedia* (pp. 2945–2954).

Zhou, K., Qiao, Y., & Xiang, T. (2018). Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward. In *Proceedings of the aaai conference on artificial intelligence* (Vol. 32).

Zhu, X., Elmagarmid, A. K., Xue, X., Wu, L., & Catlin, A. C. (2005). Insightvideo: toward hierarchical video content organization for efficient browsing, summarization and retrieval. *IEEE Transactions on Multimedia*, *7*(4), 648–666.

Zhu, X., Goldberg, A. B., Van Gael, J., & Andrzejewski, D. (2007). Improving diversity in ranking using absorbing random walks. In *Human language technologies 2007: The conference of the north american chapter of the association for computational linguistics; proceedings of the main conference* (pp. 97–104).

Zhu, X., Loy, C. C., & Gong, S. (2016). Learning from multiple sources for video summarisation. *International Journal of Computer Vision*, *117*(3), 247–268.

Zhu, X., Wu, X., Fan, J., Elmagarmid, A. K., & Aref, W. G. (2004). Exploring video content structure for hierarchical summarization. *Multimedia Systems*, *10*(2), 98–115.

Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., & Fidler, S. (2015). *Aligning books and movies: Towards story-like visual explanations by watching movies and reading books.*

Zlatintsi, A., Koutras, P., Evangelopoulos, G., Malandrakis, N., Efthymiou, N., Pastra, K., ... Maragos, P. (2017). Cognimuse: A multimodal video database annotated with saliency, events, semantics and emotion with application to summarization. *EURASIP Journal on Image and Video Processing*, *2017*(1), 1–24.

Zulko. (2017). MoviePy. (`https://zulko.github.io/moviepy/`)